Representing and Reasoning About Quality Using Enterprise Models

By

Henry M. Kim

Enterprise Integration Laboratory
Faculty of Applied Sciences and Engineering
University of Toronto
4 Taddle Creek Road
Toronto, Ontario M5S 3G9
Canada

A Thesis submitted in conformity with the requirements for the Degree of Doctor of Philosophy
Graduate Department of Mechanical and Industrial Engineering
University of Toronto

Abstract

There are few enterprise models that 1) explicitly describe the generic concepts of enterprise quality; 2) are constructed using formally defined terminology such that the model can be interpreted precisely; 3) are constructed using generic and re-useable terminology; and 4) prescribe improvements to enterprise quality by automating the task of evaluating ISO 9000 compliance of enterprises. These characteristics are very important in order to represent and reason about quality. Therefore, the following thesis is stated: There exist generic, formal, and re-useable representations of quality that describe concepts that underlie most quality applications, such that an enterprise model can be constructed from these representations; and ISO 9000 compliance of enterprises can be prescribed by reasoning about quality using this enterprise model. To support the thesis, generic quality concepts are represented as formally-defined terminology and axioms in the Ontologies for Quality Modelling. Then, the ISO 9000 Micro-Theory is constructed by building upon these representations. The micro-theory and the ontologies are encapsulated into the ISO 9000 Quality Advisor, a software tool for evaluating ISO 9000 compliance of organizations. The design, analysis, and prototypical implementations of Ontologies for Quality Modelling, ISO 9000 Micro-Theory, and the ISO 9000 Quality Advisor provide strong support of the thesis. It is concluded then that the ontologies, micro-theory, and advisor, collectively, comprise a unique model—with substantial practical application potential—that is formal, re-useable, and describes and prescribes enterprise quality.

Be not afraid.
I go before you always,
come follow Me,
and I will give you rest.

Acknowledgments

I would like to thank my mother and father: for waking up at 4AM to pick up their son from work, for taking pictures of their awkward son dropping a fly ball after he finally got to play in a 10-0 game, and for teaching their sons how to have faith in hope and to have hope in faith.

I would like to thank Frank and Christine: for just being there, and for giving to the world, Kyle and Ethan, my two favourite (my two only) nephews.

I would like to thank my friends for how they have enriched my life: Jeevan (for embodying in his life the philosophy that men and women can never be friends), Jeff (for his complete inability to find a date), Farhan (for his incredible humility and tactfulness), John (for his willingness to explore his emotional side), Tracy (for her amazing ability to remain calm in all situations especially as it relates to an emotional John), Paul (for his patience and his empathic nature), Francine (for her reserved and demure disposition), Anish (for choosing consulting as a long-term career), and Ush (for his uncanny diplomacy).

I would like to thank Debbie: reasons to the whys...though I may never know .. a faith in fate underlies.

I would like to thank Professor Mark Fox for giving me the opportunity to do my Ph.D. and to work in Australia. I also would like to thank my partner in the doctoral stream, Don: for misery loves company, and when I had no clue to what I was doing, he was great company. For all the grad students that have come and gone from RS315, thanks.

I would like to thank Dean Dave for giving me the chance to be a don: I was paid to do what I would have done for free. To the men of Wilson House: thanks, I learned so much from you guys.

Thank you, Lord.

Table of Contents

List of Figures

List of Tables

Primer on First-Order Logic

FOL.1 Introduction

In this dissertation, first-order predicate logic[1] is used as the mathematical language with which an enterprise is modelled. This chapter is provided to offer the dissertation reader a brief primer for understanding the terms and concepts represented in first-order logic. This primer is excerpted from [Cawsey 97].

Predicate logic allows us to represent fairly complex facts about the world, and to derive new facts in a way that guarantees that, if the initial facts were true then so are the conclusions. It is a well understood formal language, with well-defined syntax, semantics and rules of inference.

FOL.2 Propositional Logic

Predicate logic is a development of *propositional* logic. In proposition logic a fact such as "Alison likes waffles" would be represented as a simple atomic proposition. Lets call it P. We can build up more complex expressions (*sentences*) by combining atomic propositions with the *logical connectives* $\wedge$ (and), $\vee$ (or), $\neg$ (not), $\supset$ (if ... then), and $\equiv$ (if and only if). So, if we had the proposition Q representing the fact "Alison eats waffles," we could have the facts:

$P \wedge Q$ "Alison likes waffles and Alison eats waffles."

$P \vee Q$ "Alison likes waffles or Alison eats waffles."

$\neg Q$ "Alison does not eat waffles."

$P \supset Q$ "If Alison likes waffles then Alison eats waffles."

$P \equiv Q$ "Alison likes waffles if and only if Alison eats waffles." That is, "if Alison likes waffles then Alison eats waffles, and if Alison eats waffles then Alison likes waffles."

In general, if X and Y are sentences in propositional logic, then so are $X \wedge Y$, $X \vee Y$, $\neg X$, $X \supset Y$, and $X \equiv Y$. So the following are valid sentences in the logic:

$P \vee \neg Q$.

$P \wedge (P \supset Q)$.

1. In this dissertation, the terms "first-order logic," "first-order predicate logic," "predicate logic," and "predicate calculus" are used interchangeably.

$(Q \vee \neg R) \supset P$.

Propositions can be true or false in the world. An *intepretation* function assigns, to each proposition, a truth value (i.e., true or false). This interpretation function says what is true in the world. We can determine the truth value of arbitrary sentences using *truth tables* which define the truth values of sentences with logical connectives in terms of the truth values of their component sentences. The truth tables provide a simple *semantics* for expressions in propositional logic. As sentences can only be true or false, truth tables are very simple, for example:

X	**Y**	**X ∧ Y**
T	T	T
T	F	F
F	T	F
F	F	T

In order to infer new facts in a logic we need to apply *inference rules*. The semantics of the logic will define which inference rules are universally *valid*. One useful inference rule is the following (called modus ponens) but many others are possible:

$a, a \wedge b.$
$\therefore b.$

The above rule just says that if $a \wedge b$ is true, and a is true, then b is necessarily true. We could prove that this rule is valid using truth tables.

FOL.3 Predicate Logic: Syntax

The trouble with propositional logic is that it is not possible to write general statements in it, such as “Alison eats everything that she likes.” We'd have to have lots of rules, for every different thing that Alison liked. Predicate logic makes such general statements possible.

Sentences in predicate logic are built up from *atomic sentences*, which consist of a predicate name followed by a number of arguments. These arguments may be any *term*. Terms may be:

Constant symbols, such as ‘alison.’
Variable symbols, such as ‘X.’
Function expressions, such as ‘father(alison).’

So, atomic sentences in predicate logic include the following:

friends(alison, richard).
friends(father(fred), father(joe)).
likes(X, richard).

Sentences in predicate logic are constructed (much as in propositional logic) by combining atomic sentences with logical connectives, as in the following:

friends(alison, richard) ⊃ likes(alison, richard).
likes(alison, richard) ∨ likes(alison, waffles).
((likes(alison, richard) ∨ likes(alison, waffles)) ∧ ¬likes(alison, waffles)) ⊃ likes(alison, richard).

Sentences can also be formed using *quantifiers* to indicate how any variables in the sentence are to be treated. The two quantifiers in predicate logic are ∀ (for all) and ∃ (there exists), so the following are valid sentences:

∃X bird(X) ∧ ¬flies(X). “There exists some bird that doesn't fly.”
∀X (person(X) ⊃ ∃Yloves(X,Y)). “Every person has something they love.”

A sentence should have all its variables quantified. So strictly, an expression like “∀X loves(X, Y),” though a well formed *formula* of predicate logic, is not a sentence because Y is not quantified.

FOL.4 Predicate Logic: Semantics

The semantics of predicate logic is defined (as in propositional logic) in terms of the truth values of sentences. Like in propositional logic, we can determine the truth value of any sentence in predicate logic if we know the truth values of the basic components of that sentence. An *interpretation* function defines the basic meanings/truth values of the basic components, given some *domain* of objects that we are concerned with.

In propositional logic we saw that this interpretation function was very simple, just assigning truth values to propositions. However, in predicate calculus we have to deal with predicates, variables and quantifiers, so things get much more complex.

Predicates are dealt with in the following way. If we have, say, a predicate P with 2 arguments, then the meaning of that predicate is defined in terms of a mapping from all possible pairs of objects in the domain to a truth value. So, suppose we have a domain with just three objects in: 'fred,' 'jim,' and 'joe.' We can define the meaning of the predicate 'father' in terms of all the pairs of objects for which the 'father' relationship is true —say 'fred' and 'jim.'

The meanings of ∀ and ∃ are defined in terms of the set of objects in the domain. "∀X S." means that for every object X in the domain, S is true. "∃X S." means that for some object X in the domain, S is true. So, "∀X father(fred,X).", given our world (domain) of 3 objects ('fred,' 'jim,' 'joe'), would only be true if "father(fred,X)" was true for each object; that is, 'fred' is the 'father' of all objects in the domain. In our interpretation of the 'father' relation this only holds for "X=jim," so the whole quantified expression will be false in this interpretation.

The term "first-order" refers to the scope of quantification. In a first-order predicate logic sentence, only variables are quantified. In a "higher-order" predicate logic sentence, the predicates themselves are quantified.

FOL.5 Proving Things in Predicate Logic

To prove things in predicate logic we need two things. First we need to know what inference rules are valid—we can't keep going back to the formal semantics when trying to draw a simple inference! Second we need to know a good proof procedure that will allow us to prove things with the inference rules in an efficient manner.

When discussing propositional logic we noted that a much used inference rule was modus ponens. This rule is a *sound* rule of inference for predicate logic. Given the semantics of the logic, if the premises are true then the conclusions are guaranteed to be true. Other sound inference rules include *modus tollens* (if “A is true then B is true” is true and B is false, then conclude that A is false), *and-elimination* (if A∧B is true then conclude both A is true and B is true), and lots more.

In predicate logic we need to consider how to apply these rules if the expressions involved have variables. For example we would like to be able to use the facts “∀X (man(X) ⊃ mortal(X)).” and “man(socrates).” and conclude “mortal(socrates).” To do this we can use modus ponens, but allow universally quantified sentences to be *matched* with other sentences. So, if we have a sentence “∀X A ⊃ B.” and a sentence C then if A and C can be matched or unified (matching or unification means that A is equivalent to C for some X, as in if A=man(X) and C=mortal(socrates), then A=C for X=socrates) and then we can apply modus ponens.

The best known general proof procedure for predicate calculus is *resolution*. Resolution is a sound proof procedure for proving things by refutation—if you can derive a contradiction from ¬P then P must be true. Rules for resolution are applied to create a systematic proof procedure. So, under certain conditions, it is possible to automatically prove the truth of a given sentence. The Prolog programming language includes such a proof procedure.

FOL.6 Examples

These are some example first-order predicate logic sentences.

∃X tables(X) ∧ ¬numberoflegs(X,4). “There is some table that doesn’t have 4 legs.”

∀ X (macintosh(X) ⊃ ¬realcomputer(X)). “No macintosh is a real computer,” or “If something is a macintosh then it is not a real computer.”

∀ X glaswegian(X) ⊃ (supports(X,rangers) ∨ supports(X,celtic)). “All Glaswegians support either Celtic or Rangers.”

∃ X small(X) ∧ on(X,table). “There is something small on the table.”

¬∃X brusselsprouts(X) ∧ tasty(X). “There are no tasty brussel sprouts.”

List of Symbols and Conventions

Sym.1 List of Conventions

Figure Sym.1 Explanation of Agent Constraint Representations

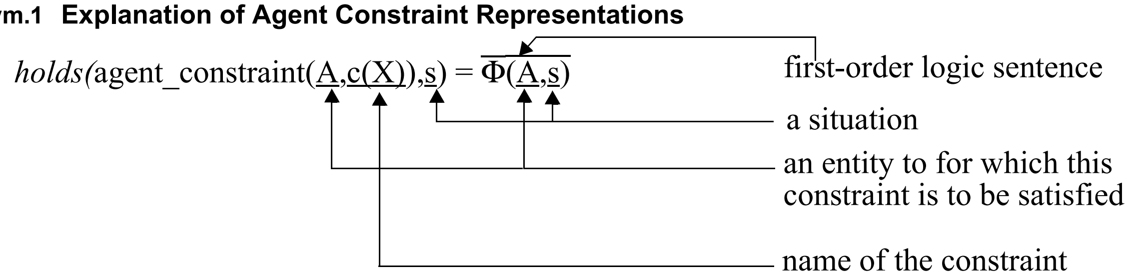
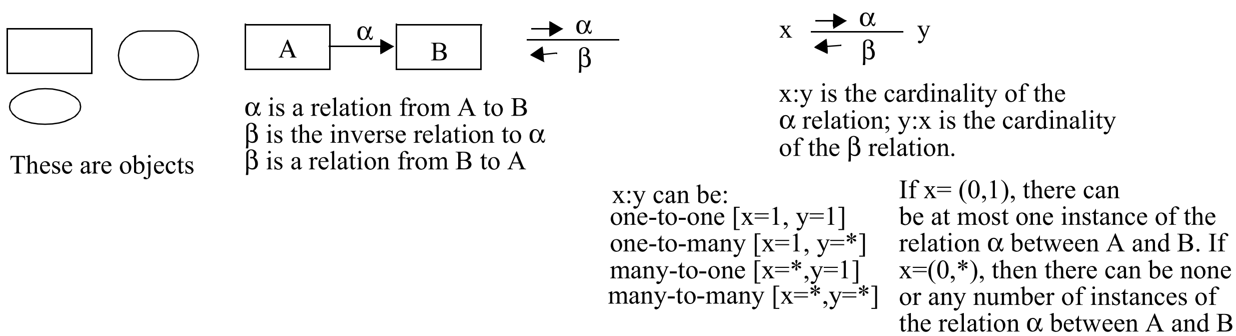


Figure Sym.2 Data Modelling Conventions used throughout this thesis



- unless explicitly stated otherwise. All terms defined in all ontology sections are fluents. So for example Axiom 3.12 activity(A) is actually defined as *holds*(activity(A),s).

- Please note the following conventions which hold throughout all ontology and micro-theory chapters: Words that appear in *italics* denote assumptions, terminology, or axioms for an ontology or micro-theory, where the ontology or micro-theory is the subject of that chapter. A term that is in *fuzzy italics* denotes a term for which its assumptions, definition and statements of constraint are provided in another chapter.

- Terminology and axioms of an ontology or micro-theory are annotated using the following convention:

  ONT TYPE: SUBTYPE-#.
  - ONT: denotes the ontology or micro-theory. These can be: Core (Core Ontologies), Meas (Measurement Ontology), Trace (Traceability Ontology), QMS (Quality Management System Ontology), and ISO 9K(ISO 9000 Micro-Theory).
  - TYPE: denotes type of representation. This is either Term (Terminology) or Axioms.

SUBTYPE: denotes the specific type of terminology or axioms. For terminology, this is Pred (predicate). For axioms, this is one of PT (Primitive Term), Defn (Definition), or Cons (Constraining Axiom).

#: This denotes the current number of representations with that ONT, TYPE, and SUBTYPE

- When terminology, not axioms, are introduced—e.g. Meas Term: Pred-3— it can be interpreted in two different ways:
  - The term and an informal definition for it is introduced, and will later be formally defined.
  - The term is introduced as a predicate and will not be formally defined; for brevity, and because the term's significance in the discussion of an ontology or micro-theory is not germane, the formal definition of the term is omitted.

Sym.2 List of Symbols

Sym.2.1 Parameters (typing) of Definitions

object-oriented modelling

Atr: any attribute of an object
V: object's value for Atr

situation calculus

s: extant or hypothetical situation
f: fluent

general ontology classes

X: entity— activity, resource, tru, or organization agent
I: information— goal, policy, or organizational constraint
St: state

from the measurement ontology

At: measured attribute; subclass of Atr

Mu: standard value for At
SL: set of values of conforming quality for At
Sp: sampling plan for At
Sz: sample sizing for At

Mp: measurement point
Q: conformance point

T: time period
Tp: time point
U: unit of measurement
UName: describer for the unit of measurement (e.g. capacity)

from the traceability ontology

A: activity

Rc: resource or a tru
R: resource
Rt, Rx, Ry:tru

Qu: resource point of Rt

from the quality management system ontology

Oa: organization agent
O: enterprise
M: employee

Ro: role
Y: policy
G: goal
C: organizational constraint
Qr: quality requirement; subclass of C
L: communication or authority link ID

E: objective evidence
D: document resource

1. Introduction

1.1 Executive Summary

In this world of global competition, importance of quality is well-acknowledged, and quality has even become a corporate cliché. As well, advances in information technology continue to enhance an enterprise's capability to manage quality. One such advance is the use of information systems-based enterprise models to represent and reason about quality. Enterprise models support making better, more informed decisions based upon information integrated from different parts of the enterprise.

An enterprise model of quality can be used to answer questions about the quality of the products and processes of an enterprise. It can also be used to identify quality improvement opportunities and suggest means to make improvements. An internationally accepted prescription for improving an organization's capability to consistently ensure the quality of its products, the ISO 9000 Standards for Quality Management, is certainly an important prescription that ought to be supported. So, in this thesis, a software tool that automatically evaluates ISO 9000 compliance of an organization and is used to analyze quality of products and processes is prototyped. This tool is called the *ISO 9000 Quality Advisor*. There are two enterprise models that underlie this software: A model that describes quality within an enterprise, called the *Ontologies for Quality Modelling*; and a model that prescribes ISO 9000 compliance of an organization, called the *ISO 9000 Micro-Theory*.

The *Ontologies for Quality Modelling* and the *ISO 9000 Micro-Theory* are constructed so that there is precision in understanding the state of a particular modelled enterprise. This precision is the result of carefully identifying and defining business terms (*terminology*) and rules (*axioms*). This leads to the capability to automatically deduce implicit truths and new insights, a powerful capability in analyzing an enterprise. This capability is called the *competency* of the model. The models are also constructed so that its parts are *re-useable*. Parts of the *Ontologies for Quality Modelling* are the building blocks for constructing the *ISO 9000 Micro-Theory*. As well, through a technique called *reducbility*, it is demonstrated that parts of the ontologies and micro-theory can be re-used to construct application software for quality that are similar to ones that exist in the market today.

Results of the literature survey show that there is a lack of enterprise models of quality that have all the useful characteristics of the *Ontologies for Quality Modelling* and the *ISO 9000 Micro-Theory*, and so construction of these models addresses a need in literature and practice.

A systematic approach called the Ontological Engineering Methodology is used to construct the ontologies and micro-theory. The methodology entails identifying the business issues of a specfic enterprise (*Motivating Scenario*); stating questions about an enterprise that an ontology or micro-theory must be used to answer (*Competency Questions*); identifying, classifying, and unambiguously defining terminology and axioms (*Ontology or Micro-Theory Representations*); and deducing answers and insights (*Answering Competency Questions*).

As well as the *Motivating Scenarios* of collaborating enterprises—BHP Steel and deHavilland Manufacturing—the need to construct the *ISO 9000 Micro-Theory*, using the *Ontologies for Quality Modelling* representations as building blocks, serves as an important motivation for the design of these models. The table below highlights key aspects of the design.

Figure 1.1 **Key Aspects of ISO 9000 Micro-Theory**

Micro-Theory represents:	**Rationale for Micro-Theory development**	**Micro-Theory represents the following as being key for ISO 9000 compliance:**	**Motivates development of:**
ISO 9000 inspection and testing requirements	Quality control through inspection is a basic form of ensuring product quality, so it should be represented in the micro-theory	Quality procedures and evidence must constrain inspection and testing activities.	Measurement Ontology

Figure 1.1 Key Aspects of ISO 9000 Micro-Theory

Micro-Theory represents:	Rationale for Micro-Theory development	Micro-Theory represents the following as being key for ISO 9000 compliance:	Motivates development of:
ISO 9000 traceability requirement	The requirement is straightforward, terse, and hence easier to prototype	Must demonstrate accurate material traceability.	Traceability Ontology
ISO 9000 management responsibility and documentation requirements	These requirements are considered to be most important, because they highlight organization's commitment to achieving ISO 9000 compliance.	Quality requirements, policy, procedures, and evidence must be rigorously defined and documented.	Quality Management System Ontology

Figure 1.2 Key Aspects of Ontologies for Quality Modelling

Ontologies for Quality Modelling	Rationale for ontology development	Ontology objective	Ontology key concepts
Measurement Ontology	First step in representing and reasoning about quality is to measure quality.	• measure quality • reason about measurement • support micro-theory construction	• There must be systems for describing what should be measured, what needs to be known before measurements take place, and how measurements should occur. • Quality is an evaluation of an entity as being of conforming or nonconforming quality. A quality evaluation is made by composing measurement values, according to rules for 'what is quality?' that are dictated by the enterprise.
Traceability Ontology	When measurement points to a problem, traceability is the primitive analysis capability required to solve the problem	• precisely represent material flow • reason about traceability • support micro-theory construction	• Special consideration must be given to splitting and merging of sets of resources. • Material flow trace is a composed by repeatedly tracing how one activity consumed one set of resources to produce another.
Quality Management System Ontology	In order to consistently ensure that quality problems are properly measured, traced, and analyzed, there must be a quality management system in place.	• represent and reason about the state of an enterprise's quality management system.	• Roles of an organization's quality management system are defined in order to satisfy customer quality requirements. • These roles must be rigourously documented; so must the evidence of whether these roles are properly performed.

The *ISO 9000 Quality Advisor* is used to prototypically demonstrate how the ontologies and micro-theory can be used to analyze quality within an enterprise, including use for evaluating ISO 9000 compliance. The advisor is also a prototypical tool for designing ontologies and micro-theories. It is useful to either an enterprise analyst or an ontology builder, and enables the peformance of each of the steps of the Ontological Engineering methodology.

Collaboration with industrial partners to develop the ontologies, micro-theory, and advisor yielded the following results:

Results	BHP Steel	deHavilland Manufacturing
Constructed model of partner enterprise using Ontologies for Quality Modelling	x	x
Reasoned about quality within partner enterprise model	x	x
Demonstrated proof-of-concept of engineering ontology-based enterprise models	x	
Tested ISO 9000 Micro-Theory by using partner enterprise model as testbed	x	x
Incorporated user requirements for prototyping ISO 9000 Quality Advisor	x	

It can be concluded then that the *Ontologies for Quality Modelling, ISO 9000 Micro-Theory*, and the *ISO 9000 Quality Advisor* collectively comprise a unique contribution. Not only can these models be used to represent and reason about quality using enterprise models, but they can also be used to improve the quality of the products of an enterprise.

1.2 Discussion

In this world of global competition, importance of quality is well-acknowledged, and *quality* has even become a corporate cliché. According to Godfrey [93], the capability to manage quality will continue to be enhanced, due to advances in information technology. Organizations that leverage information technology to access external information about quality, and share quality information internally, better than their competitors can then gain a substantial advantage. One way to effectively access and share information about quality of products, processes, and systems of an enterprise is to represent and reason about quality using information systems-based *enterprise models.*

To represent and reason about quality using an enterprise model, the model should be *descriptive*; that is, it should represent the key entities, structures, and concepts needed to describe quality within an enterprise such as an organization's activities, resources, organizational structures, information flows, and products.

The model should also be *prescriptive*. It should be possible to improve the quality of the products and processes of an enterprise using this model. The ISO 9000 Standards for Quality Management is a suitable quality prescription because:

- It prescribes quality management practices that give confidence to an organization's customers that they are receiving products that satisfy their quality expectations.
- It addresses quality for the whole enterprise.
- It is one of the most important, widely publicized, and globally applied quality initiatives.
- The conventional wisdom is that application of ISO 9000 to an enterprise is more of a craft than a systematic application of a repeatable methodology [Saarelainen 96].

As a result of the conventional wisdom, there are not many information systems models that specify a systematic prescription for achieving ISO 9000 compliance; the model developed in this thesis is intended to be such a prescription.

The model should be *re-useable*: it should be possible to re-use portions of the enterprise model to support different tasks than the tasks for which the model was initially developed. A model that re-uses portions of other models can be developed more quickly and cheaply than models developed from "first principles."

Finally, the model should be *formal*, where formal refers to a data model expressed using a logic language, not an analytical model expressed in mathematics. Like an equation written in mathematics, and unlike most English sentences, an expression written in a logic language can be interpreted unambiguously, as long as the terms that comprise the expression are precisely defined. In a mathematical model, the value of a dependent variable can be calculated from values of independent variables because the equation that relates the independent variables to the dependent variable has been defined. Similarly, in a formal model, additional facts can be deduced from existing facts because the expression that relates existing facts to a new fact has been defined. Inasmuch as calculators automate mathematical calculations, computers can automate this deduction.

The vocabulary of the formal model is called its *terminology*; the meanings of terms that comprise the terminology are the model's *semantics*; the grammar for composing expression is the model's *syntax*; and the expressions, written in the logic language, that precisely define and constrain these meanings are called *formalizations*. The formalizations, as well as the informal descriptions—expressed in English—of terminology and semantics constitute the *representations* of a model. In extending representations that model product, process, and system quality, a formal model is used to represent quality. By deducing new facts using the formalizations, a formal model is used to reason—that is, perform deduction—about quality. If the deductive capability of the logic language is a feature in a programming language, and the terminology and semantics of the model is composed using the syntax of the programming language—as a computer-based implementation of the formal model—then it is possible to use computers to automate deduction. If a computer-based enterprise model is constructed using a formal model of quality, and supports automatic deduction, then it is used to represent and reason about quality.

Precision in interpretation is inherent in a formal model. Because of this, it is especially appropriate to apply a formal model for ISO 9000 compliance prescription. Why? Since ISO 9000 requirements are interpreted differently by registrars, who audit and certify an organization for

ISO 9000 compliance, precise and unambiguous descriptions for applying the ISO 9000 to an organization are not provided. A formal model for ISO 9000 compliance could be a precise and unambiguous description. Moreover, this model can be implemented as an information system that automatically evaluates the ISO 9000 compliance of an organization which is represented as a computer-based enterprise model. ***The thesis of this dissertation***, then, is: ***Quality within an organization can be described by representing it in an enterprise model, and ISO 9000 compliance of an organization can be objectively prescribed by reasoning about quality using the model***.

The thesis is supported by the development of the following models:

- ***a formal, re-useable, descriptive model of enterprise quality, called an ontology of enterprise quality***
- ***a formal, re-useable, prescriptive model of ISO 9000 compliance, called a micro-theory of ISO 9000 compliance***

This thesis research is part of a broader enterprise modelling effort, the TOVE (TOronto Virtual Enterprise) project, whose goals are to ([Fox et. al. 93a], pg. 2):

- Provide a shared terminology.
- Define precise and unambiguous semantics for the enterprise in *first-order logic*—
  - First-order logic is used because it is a formal language with restrictive syntax and semantics. It is used for representing terminology and axioms, where axioms define and constrain the interpretation of the terminology. Given initial propositional truths, additional truths are deduced by applying axioms upon existing propositions.
- Implement the semantics in a set of Prolog *axioms* that will enable TOVE to automatically deduce the answers to many "*common sense*" questions about the enterprise—
  - Prolog is a programming language implementation of first-order logic and its deductive capability. A common-sense model represents fundamental, often-implicit concepts in a domain. For example, it is "common-sense" in organizations that if Al *works for* Bill, and Bill *works for* Carrie, then Al *works for* Carrie. By representing these relations as facts, and representing the transitivity of the *works-for* relation as an axiom, the proposition that Al *works for* Carrie can be automatically deduced. The questions that initiate the deductions are called *queries*.

The enterprise modelling approach of the TOVE project is adopted for this thesis.

The TOVE project's approach is to re-use and extend pre-existing models. Ontologies provide the means to construct specialized knowledge-based systems that are assembled from components of pre-existing systems [Patil et. al. 92]. "An *ontology* is a formal description of entities and their properties; it forms a shared terminology for objects of interest in the domain, along with definition for the meaning of each of the terms." ([Gruninger & Fox 95a], pg. 1). The TOVE ontologies at the Enterprise Integration Laboratory currently represent core knowledge about the enterprise such as activity [Gruninger & Pinto 95], time, causality, resource [Fadel et. al. 94], and organization [Fox et. al. 95]. These ontologies collectively are called the Core Ontologies. ***An objective of this thesis is to design, analyze, and construct a prototypical implementation of the Ontologies for Quality Modelling*** by re-using and extending representations from these core ontologies to describe generic, "common-sense" quality entities, structures, and concepts.

A second objective is to re-use these ontologies ***to design, analyze, and construct a prototypical implementation of the ISO 9000 Micro-Theory***, a formal model of ISO 9000 Compliance. A *micro-theory* is a a contextually-bounded formal model of knowledge that is often task-oriented. The ISO 9000 Micro-Theory is an ISO 9000-compliance perspective of quality within the enterprise, constructed from representations that describe a generic perspective of quality (the Ontologies for Quality Modelling). The micro-theory is detailed enough to solve the difficult task of automating ISO 9000 compliance evaluation.

The Ontologies for Quality Modelling and the ISO 9000 Micro-Theory have the potential to be used for practical applications. The information system implementation with which a decision-maker uses these representations to perform tasks is important to support the thesis; that is, the ontology and micro-theory representations should be integrated into a software tool that supports the *enterprise engineering* function of exploring alternative organization models [EIL 96]. ***So, the final objective of this thesis is to design, analyze, and construct a prototypical implementation of the ISO 9000 Quality Advisor*** software package that uses the Ontologies for Quality Modelling and the ISO 9000 Micro-Theory to analyze the re-engineering of the enterprise towards ISO 9000 compliance.

Two evaluation criteria of the TOVE project are used to evaluate the ontologies and micro-theory ([Fox & Gruninger 96], pg. 15-6).

- *Competency*
  - Is the model complete enough to perform the desired task? Can the formal model represent the information necessary for a function to perform its task?
- *Re-useability*
  - Is the model general? To what degree are the representations of the formal model shared between diverse activities such as design and troubleshooting, or design and marketing? Is the formal model specific to a sector, such as manufacturing, or applicable to other sectors, such as retailing, finance, etc.?

Queries answerable with a *populated model* (a formal model populated [or instantiated] with facts about a specific enterprise) determine a model's competency. If such queries, called competency questions, are sufficient to support a problem task, then the model is competent vis-a-vis that task. Using a model in different functions and in different sectors is one way of demonstrating re-useability. A stronger demonstration is *reducibility*: that competency questions of another model—where this model is competent in another function or sector than the evaluated model—can be reasonably translated (reduced) to competency questions answerable using representations of the evaluated model. This demonstrates that the evaluated model is general enough to be used as basis to construct models with different competencies. A general model is re-useable since components of the model can be re-used to construct other models.

1.3 Ontologies for Quality Modelling, ISO 9000 Micro-Theory, and ISO 9000 Quality Advisor: Precepts

In order to engineer a competent and re-useable, quality-related ontology or micro-theory, the main competency question to ask is: What is the quality of a product, process, or system of the enterprise? This question should be decomposed to competency questions that are narrower in scope. These questions in turn determine the competency of the ontology or micro-theory.

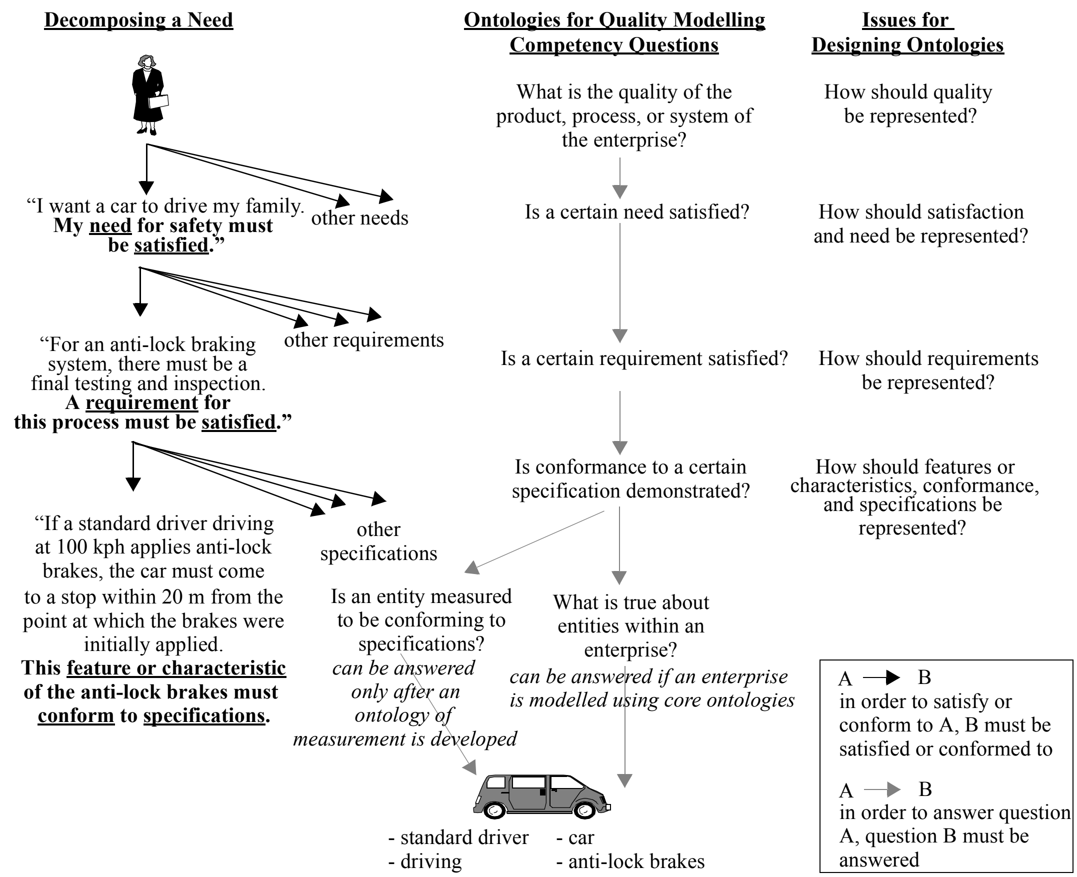
The official definition of quality, stated by the international standards body, is from the ISO 9000: *"Quality is the totality of features and characteristics of a product or service that bear on its ability to satisfy stated or implied needs.*" ([ISO 91], pg. 16). This vague definition can be augmented with a manufacturing-based definition: *"Quality means conformance to requirements."* ([Crosby 79], pg. 16). Combining these definitions, the basis of the Ontologies for Quality Modelling and the ISO 9000 Micro-Theory can be stated as the following[1].

- A need can be decomposed into a set of requirements upon features and characteristics of a product or service, and if all these requirements are conformed to, then that need has been satisfied.

The diagram below shows that as a quality need is decomposed, there are competency questions at each level of decomposition. A given question can be answered by answering the questions at a lower level. At the lowest level, questions are answered using the terms and axioms of existing ontologies or micro-theories, or answered after existing ontologies and micro-theories are augmented or new ones are constructed. The diagram shows that a competency question, "Is conformance to a certain specification demonstrated?" can only be answered if the following questions are answered: "What is true about entities within an enterprise?" and "Are these entities measured to be conforming to specifications?" The first question can be answered using terms and axioms from the Core Ontologies; the latter question requires that a new ontology in which the terms, conformance, specifications, and measurement, are defined be constructed. This new ontology is the Measurement Ontology.

1. Although this research can be applied to evaluate the quality of service, there is a bias towards use for products, and special considerations for quality of service or information is not explored. Wand and Wang [96] explore these considerations.

Figure 1.3 **Engineering the Ontologies for Quality Modelling: Decomposing a Need**



Measurement Ontology

Entities in the example such as standard driver, driving, car, and anti-lock brakes can be represented with the Core Ontologies. However, conformance to specifications cannot be represented just with these representations. In this example, it must be possible to *measure* the stopping distance. Thus, whether the customer need is satisfied cannot be determined unless measurements take place; in order to talk about quality, *measurement* and the measurement values must be represented. Measurement is at the root of quality management, as evidenced by Federal Express' quality philosophy of "measure, measure, measure."

Measurements link the features and characteristics specifications, decomposed from a quality need, to the representations of activities, states, time, resources, and organizations that exist in the enterprise model; hence, representations in an ontology of measurement are defined in terms of representations in the core ontologies. Measurement is an important sub-domain of quality, so, the Measurement Ontology ought to be, and is, one of the Ontologies for Quality Modelling.

Traceability Ontology

Measurements are taken because variability exists. Often this variability needs to be analyzed because it is a sign of a problem. A primitive analysis capability requires tracing back, for example, from a problematic assembly to its sub-assemblies in order to diagnose the root of the problem. Therefore, *traceability* is the basic form of quality analysis that identifies the relationship between a measured entity and other related entities. Hence, the Traceability Ontology is another of the Ontologies for Quality Modelling. This ontology is applicable only for enterprises for which it is reasonable to model that consumption and production occurs discretely. Where it is unreasonable or impossible to determine one distinct set of a resource from another distinct set of the same resource—e.g., continuous hydroelectric consumption—this ontology cannot be used.

Quality Management System Ontology

An organization can consistently satisfy its customers only if it has a well-designed internal system for managing the quality of its products and processes. It is because there exists a good *quality management system* that:

- The system of measurement is carefully designed, and policies and procedures are placed to facilitate problem-solving, so quality problems can be identified and corrected before they get to the customer.
- Proper measurements are taken, appropriate analyses using methods such as traceability capability are performed to address quality problems, and customers are satisfied with the products they receive.

So, an ontology of the Quality Management System, composed of representations of roles, policies, goals and documents of a quality management system, is included as one of the Ontologies for Quality Modelling.

ISO 9000 Micro-Theory

The micro-theory representations are constructed using the Core Ontologies and the Ontologies for Quality Modelling. Of the twenty ISO 9000 requirements, only those that can be fully expressed with these ontologies are formalized.

The following requirements are formalized because they are widely held to be the most important. The Quality Management System Ontology is predominantly used to model these requirements:

- ISO 9001 4.1 Management responsibility
- ISO 9001 4.2 Quality system
- ISO 9001 4.5 Document and data control
- ISO 9001 4.16 Control of quality records

The following requirement is formalized because it relates directly to the Traceability Ontology:

- ISO 9001 4.8 Product identification and traceability

Similarly, the following requirements relate directly to the Measurement Ontology and are formalized:

- ISO 9001 4.10 Inspection and testing
- ISO 9001 4.12 Inspection and test status

ISO 9000 Quality Advisor: Its Requirements

The advisor provides the capability to demonstrate competency of the Ontologies for Quality Modelling and the ISO 9000 Micro-Theory by:

- Implementing the representations of the ontologies and micro-theory as a set of Prolog axioms.
- Accessing the populated model of a specific enterprise, where the model is implemented as a set of Prolog facts.
- Applying axioms to facts, and deducing additional facts that answer the competency questions.

The advisor also supports the analysis of an enterprise for ISO 9000 compliance by providing:

- capability to modify a populated model to analyze what-if scenarios
- a user interface that shields the user of the advisor from its representational details—the user, for example, does not need to know first-order logic to use the advisor
- a data dictionary of terms that are familiar to the advisor user; these terms are not necessarily the terms from the ontologies

1.4 Outline of Chapters

The following is an outline of the steps taken to complete this thesis:

1) Existing data models of quality are reviewed, and a need in literature and practice for a descriptive, formal, re-useable model of quality for automatic evaluation of ISO 9000 compliance is identified. This is presented in chapter 2.

2) A methodology to develop the aforementioned data model of quality as an ontology-based model is used. The first step in this methodology is to examine parts of BHP Steel in Australia and deHavilland Manufacturing in Downsview to determine opportunities for applying an ontology-based information system as a decision-support tool. Some prototypical processes are modelled; these models constitute the specific enterprise models of the two companies. The application of this methodology is discussed in chapter 3.

3) The Ontologies for Quality Modelling are developed from the competency questions that were asked after examining the organizations. The measurement, traceability, and quality management system ontologies are presented in chapters 4, 5, and 6, respectively.

4) The ISO 9000 Micro-Theory is developed from the competency questions that need to be answered in order to evaluate ISO 9000 compliance. The micro-theory is presented in chapter 7.

5) The ISO 9000 Quality Advisor is developed to demonstrate competency of the ontologies and micro-theory, in particular to automate ISO 9000 compliance evaluation. It is also a prototypical decision-support tool for the two organizations. The use of the advisor is presented in the *demonstrations of competency* sections of chapters 4 through 7.

6) The re-useability of the representations of the ontologies and micro-theory is demonstrated in *demonstrations of reducibility* sections of chapters 4, 5 and 7.

7) The results of developing the ontologies, micro-theory, and advisor are presented. These results, as well as a listing of the contributions of the thesis and possible future works that extend the work are presented in chapter 8.

2. Literature Review

2.1 Précis

The literature in the three areas most pertinent to developing a formal model of quality—quality, enterprise modelling, and ontological engineering—is reviewed. The review reveals that there exists an opportunity in literature for an information systems model, based upon a formal and re-useable model of enterprise quality, that can be used to automatically evaluate an organization's ISO 9000 compliance. Then, different types of formal enterprise models with which such a model can be constructed are reviewed. Re-useable ontologies that support an ISO 9000 compliance perspective are identified as appropriate building blocks to construct these enterprise models. Certain models of quality that can be used for some aspect of ISO 9000 compliance evaluation are compared against each other using the following criteria expressed as questions about the capability of the models. *What type of quality processes does the model represent? For what type of tasks related to ISO 9000 compliance can the model be used? To what extent are these models formal?* And *to what extent are these models re-useable?* This examination of models plus the observations from surveying the literature offer evidence that formal, re-useable models of enterprise quality and ISO 9000 compliance address a need in literature and practice.

2.2 Introduction

The areas of literature that are examined can be parsed from a generalization of the thesis goal: to develop formal models of quality. First, different models of *quality* must be examined. Second, models that are formal, especially models of the enterprise, must be examined; hence the field of *enterprise modelling* is surveyed. Third, the different methodologies to develop a model of quality are examined, with specific emphasis on works that construct models of fundamental, "common-sense" concepts within a modelled domain: *ontological engineering*.

2.3 Quality

2.3.1 Gurus

According to Dr. Edwards Deming, after whom the Japanese have named their most prestigious quality award [Bush & Dooley 89], quality "creates constancy of purpose toward improvement of product and service, with the aim to become competitive and to stay in business, and to provide jobs." ([Deming 86], pg. 23-4). Deming's fourteen points for management constitute arguably the most famous quality doctrine:

- Create constancy of purpose for improvement of product and service.
- Adopt the new philosophy.
- Cease dependence on mass inspection.
- End the practice of awarding business on price tag alone.
- Constantly and forever improve the system of production and service.
- Institute modern methods of training on the job.
- Institute modern methods of supervision.
- Drive out fear.
- Break down barriers between staff areas.
- Eliminate numerical goals for the work force.
- Eliminate work standards and numerical quotas.
- Remove barriers that hinder the hourly worker.
- Institute a vigorous program of education and training.
- Create a situation in top management that will push every day on the above points.

For Dr. J. M. Juran [88], quality is managed through quality planning, quality control, and quality improvement. His chief contribution, the quality trilogy, provides a framework for an enterprise's quality system. His other ideas are about management's responsibility for quality such as management involvement, Pareto principle, need for widespread training in quality, definition of quality as fitness for use, and project-by-project approach to quality improvement. The Japanese, arguably the masters of building quality systems, draw the boundaries of the quality system wide enough to include societal needs: Ishikawa [85] states that a company must satisfy the society in which it exists, and Taguchi et. al. [89] define a mathematical quality loss function as the loss imparted by a defective product to society from the time that the product is shipped.

Garvin [84] has amalgamated other perspectives of quality:

1. *Transcendent approach of philosophy*: “Quality is neither mind nor matter, but a third entity independent of the two... even though it cannot be defined, you know what it is.” ([Pirsig 74], pg. 185,213)

2. *Product-based approach of economics*: In this view, the quality of the product is deemed concrete and measurable. Differences in quality reflect differences in attributes or in very measurable characteristics, such as product durability.

3. *User-based approach of economics, marketing, and operations management*: “Quality consists of capacity to satisfy wants” ([Edwards 68], pg. 37) and “Quality is fitness for use.” ([Juran 74], pg. 242)

4. *Manufacturing-based approach*: “Quality means conformance to requirements.” ([Crosby 79], pg. 15)

5. *Value-based approach of operations management*: In this approach a cost value is associated with the quality of that product. The cost of quality is assessed, internal to the organization, and price is considered as a factor in assessing product quality, external to the organization.

These quality gurus espouse different philosophies of what quality means to an organization. Their contribution to the field of quality management is the sharing of these philosophies and their vast experiences.

2.3.2 Information Systems for Quality

An enterprise-wide quality system is constructed upon a systematic approach to quality management. A systems perspective of quality—total quality management (TQM) is a commonly-used term for this approach—is key to achieving world-class behaviour and delighting the customer [Sullivan 91]. Feigenbaum [91], using the term “total quality control,” stresses the importance of taking quality management from the domain of manufacturing inspection and process control and applying it throughout the organization and across all industries. Since the potential competitive advantage gains of information technology are profound [Walton 89], an information system that supports the enterprise-wide quality system should be one of the first items discussed when establishing the quality system [Bersbach 92]. The implementation of the formal model of quality presented in this thesis is a form of a quality information system.

Sylla & Arinze [91] provide an example of a centralized quality information system framework, where the system serves as the information centre for all functions of the enterprise. They view the quality system as comprised of a centralized process that supports the processes of the organization. Dessousky et. al. [87] present a model for ensuring quality throughout the product lifecycle. So, this model is based upon a product-centred view of quality. Pohl & Jarke [92] present a model for representing flow of quality information such as policies and laws; they model process and product views. Bassie et. al. [95] present a reference model for classifying models of quality. Rather than proscribing a framework, [QIS/TC 92] surveys the desirable functions of a quality information system. The functions are statistical process/quality control (SPC/SQC), Taguchi Methods [Bisgaard 94], quality costing [Harrington 87], quality function deployment (QFD), quality documentation [Smith & Edge 90], and quality auditing [Bishara & Wyrick 94] are types of quality engineering techniques.

Within a quality system, Hancock et. al. [89] caution against the over-reliance upon verification methods. He says that such an approach is expensive, adds no value to the product or service, and may even create clerical and inconsistency errors. However, verification is necessarily part of a quality system. Since verification is rote and straightforward, some of the errors of such methods can be minimized by the use of an information system. Suzaki [87] states the importance of standardization when he states that standardized cycle times, work sequences, and standard amount of work-in-process are used to achieve maximum performance with minimum work. Ultimately a quality system can, and should, automate the dissemination and use of standards.

2.3.3 Quality Standards

A pervasive standard for verification of the quality of products and processes is ISO 9000. The ISO 9000 is actually the collective name for three international quality assurance standards, definitions for standardized quality vocabulary, and various accompanying documents. The globalization of business, and the accompanying need for standardization of terminology and quality expectations between suppliers and customers in different nations, have provided the impetus for the successful adoption of standards for quality [ISO 94a]. A successful verification of an enterprise leads to the designation of ISO 9000 certification. This means that an expert, independent auditor has found that the supplier's quality system complies with one of the ISO 9000 standards. The three standards that comprise the ISO 9000 are:

- ISO 9001: Quality Systems - Model for quality assurance in design/development, production, installation, and servicing.
- ISO 9002: Quality Systems - Model for quality assurance in production and installation.
- ISO 9003: Quality Systems - Model for quality assurance in final inspection and test.

A key assumption of ISO 9000 is that if an organization performs design, development and servicing, it performs production and installation. A concomitant assumption is that if an organization performs production and installation, then it performs final inspection and test.

The Malcolm Baldrige Award [94] perhaps has been the greatest source of quality awareness in the United States. This award has been given for organizational excellence in quality since 1987. Companies as varied as AT&T, IBM, Federal Express, and Florida Power and Light have won the Award. The award formally recognizes companies that attain preeminent quality leadership, and permits these companies to publicize the receipt of their awards. Other national quality awards that can be used to benchmark an organization's quality system are Japan's Deming Prize and the European Quality Award [Nakhai & Neves 94].

Within the specific field of software quality, there are many international guidelines for ensuring quality. The primary benchmark model is Carnegie-Mellon's Capability Maturity Model (CMM) [93b]. This model specifies five generic, progressively more difficult levels of maturity of a software development organization, where the classification is done as per the characterization of the organization's software development processes as ad hoc, repeatable, defined, managed, or optimized. A given organization is benchmarked against these levels, and classified as belonging to one level. Then, improvement opportunities are identified from this assessment. ISO 9000 [94d1] specifies guidelines for applying ISO 9001 to software producers, and ISO/IEC [92] presents very general models for evaluating software quality.

2.4 Enterprise Modelling

2.4.1 Integrating Quality for Enterprise Modelling

A quality model must be an integrated part of a model of the enterprise. An enterprise model is defined as: "a computational representation of the structure, activities, processes, information, resources, people, behaviour, goals, and constraints of a business, govenment, or other enterprise" ([Fox & Gruninger 98], pg.1). A model of the enterprise should help integrate diverse types of information [Fox - Globe 92]—such as information about quality, cost, and leadtime—and coordinate decision-making based upon this information. The virtual corporation [Byrne et. al. 93], for example, requires such enterprise integration to improve the performance of large complex processes by managing the interactions among the participants [Petrie 92]. An enterprise model serves an important function for enterprise integration: "Industries of the future must be responsive to their clients' needs and move beyond the old methods. This means that enterprise modelling tools will be vital to Canadian industries faced with the task of improving their efficiency and competitiveness in domestic and international markets." ([Mayman 93], pg. 4,38)

2.4.2 Enterprise Modelling Architectures

Underlying an enterprise model is the architecture upon which that model is constructed. Bernus [96] states that this architecture is comprised of the methods, models, and tools required to integrate the enterprise. The CIM-OSA (Computer Integrated Manufacturing - Open Systems Architecture) effort offers a detailed architectural framework for describing any manufacturing organization [CIM-OSA 90a]. The proper scope for modelling an organization is chosen by mapping the level of genericity or generality (e.g., instantiated models of real companies), the required view (e.g., a view focused primarily on organizational structure), and the appropriate stage of the lifecycle of the enterprise to which the model of the enterprise is applied (e.g., the model is used for strategic definition of requirements for a new business unit of the enterprise). Based upon the modelling scope, the appropriate set of modelling representations and computer tools is chosen to actually construct the model. In Enterprise-Wide Data Modelling [Scheer 94a], typical functions (departments) of enterprises—such as production, engineering, and accounting—are modelled. Scheer offers rather specific data structures of "typical" manufacturing departments, and hence offers reference by analogy. Creating a model of a specific enterprise then becomes one of customizing the reference model. The key difference between CIM-OSA and Scheer's model is that the former favours generality, whereas the latter favours rapid modelling. CIM-OSA reference models are generic enough to facilitate construction of enterprise models of various industries, but at the expense of a longer model construction time. Customizing Scheer's model to construct a specific enterprise model may take a shorter amount of time, but only if the enterprise to be modelled is a manufacturing organization with functional departments that can be easily modelled with the representations from his model.

One reference architecture for computer-integrated manufacturing that seeks to integrate the human factors in enterprise modelling is the Purdue Enterprise Reference Architecture (PERA). In PERA, representations are delineated into information systems tasks, manufacturing tasks, and human-based tasks [Williams 92]. Specific constructs provided within PERA are more sparse than previous two models. Yet another mostly conceptual model is the GIM-GRAI integrated methodology which classifies the systems of the enterprise into an overall information system and a decision system [Vallespir et. al. 91].

There are many enterprise modelling efforts that emphasize methods, models, and tools more than the architecture. These are examined next.

2.4.3 Enterprise Modelling Efforts

Although any model of an organization can be considered an enterprise model, if the definition of an enterprise model is restricted to “a computational representation of the structure, activities, processes, information, resources, people, behaviour, goals, and constraints of a business, government, or other enterprise” ([Fox & Gruninger 96], pg. 1), Macintosh [94] and AIAI [96] offer good reviews of enterprise modelling efforts around the world. Fox’s definition can also be used to classify these efforts: ICEIMT [92a] lists an enterprise model as being comprised of models of information flow, process flow, and resource use. Fox et. al. [95] put forth a model of organizational structure, and Finger et. al. [92] stress the importance of modelling products as design artifacts. Enterprise modelling efforts therefore can be classified according to their emphasis on models of: information, process, resource, organization, and product.

The CIM-OSA and Scheer efforts have all types of models except the product model. GIM-GRAI consists of process and information models, whereas PERA only has a process model. ADEPT [Jennings et. al. 96] views business processes as a collection of intelligent agents deciding upon allocation of resources; it omits only a product model. A similar perspective as applicable to information systems development is employed for the Regatta project [Swenson 93] and by Yu [93a]. FEND [96] is a process model that emphasizes control feedback between processes. MADE [Whitney 96] is a project that aims to build highly integrated CAD/CAM tools; therefore, it depends upon a product model. Some enterprise modelling efforts are noteworthy as they focus their efforts on the enterprise model as a tool. A traditional use of enterprise modelling tools has been for running parts of the operations of the organization. Information systems from traditional data processing programs, MRP software, and relational databases to state-of-the-art workflow systems [Bonner et. al. 96] require a computerized model of the enterprise. Another use of such enterprise modelling tools in the analysis and design of organizations is for business process re-engineering (BPR) [Hammer 90]. Spurr et. al. [94] and [BPR 96] offer reviews of the many reengineering tools. Specific efforts to create a workbench in which different configurations of the enterprise model can be tested include Savolainen, T., Beeckmann, D., Groumpos, P. and Jagdev, H., "Positioning of Modelling Approaches, Methods, and Tools", Computers in Industry, Vol 25, No. 3, March 1995, pp. 255-62. [Scacchi & Mi 93] and [Malone et. al. 93]. The BPR tools and enterprise engineering [Liles et. al. 96] workbenches emphasize that there must be a software workbench that provides an interface for an analyst to use the enterprise model without having to know the inner details of the model.

2.4.4 Models of Analysis

Although useful as a reference, the methods, models, and tools of these enterprise modelling efforts can at best only describe an enterprise. In order to use the enterprise model for analysis, a model also needs an accompanying prescriptive model. This prescription can be a set of queries asked of the enterprise model such that the answers to these queries offer valuable insights about the modelled enterprise. There are three types of such query models: factual, expert, and common-sense. Factual queries are direct questions upon explicit information in the enterprise model; an SQL query of a relational database is a common example. Expert systems provide a prescription that is applied to a description of an enterprise to solve a specific task. McDermott [82], for example, describes how an expert system was used to help technicians repair computer systems. Common-sense queries are answerable if the "common-sense" of the enterprise is explicitly represented. For example, the Cyc effort [Guha et. al. 90] tries to represent the common-sense reasoning power of a five-year old.

Traditional forward and backward chaining expert systems require a detailed model of the domain that may be unique to the specific enterprise. Such systems tend to be costly to build and maintain, and are narrow in scope [Fox 90]. If it is desired to apply a prescriptive model to different enterprises in different domains, an expert system is not an appropriate query model. Concomitantly, if the prescriptive query model captures fairly detailed information then a factual query model is also inappropriate, since explicitly representing all necessary information as facts makes the model too large and difficult to maintain. So in cases when a certain amount of generality in the prescriptive model is desired and the scope of the prescriptive model is neither too detailed nor trivial, use of a common-sense model may be warranted.

A common-sense model is a descriptive and prescriptive model of a domain in which the core, fundamental concepts of that domain are represented such that the model is able to answer queries about these fundamental concepts (i.e., the "common-sense" of that domain) by means of deduction. With respect to the common-sense of the real world, there exist common-sense models of: time [Allen 83], space [Kautz 85], materials [Hayes 85], causality [Reiger & Grinberg 77], activity [Sathi et. al. 85], and qualitative physics [Kuipers 86]. Ontologies from the field of information systems represent a means to organize concepts; an ontology of a domain can serve as the common-sense model of that domain.

2.5 Ontological Engineering

2.5.1 Ontologies

An ontology consists of a representational vocabulary with precise definitions of the meanings of the terms of this vocabulary plus a set of formal axioms that constrain interpretation and well-formed use of these terms [Campbell & Shapiro 95]. An important phase in constructing an ontology is representing knowledge in a formal language such as symbolic logic—a "symbolic encoding of propositions believed" in a particular domain [Levesque & Brachman 93]. Once an ontology is created to describe a domain, its representations can be shared and used by those that did not develop that ontology since the ontology is represented using a formalism in which ambiguous interpretations of syntax and semantics are minimized. In fact, the need to share and reuse knowledge bases is the main rationale for the development of formal ontologies Grosof, B.N., Morgenstern, L., Applications of Logicist Knowledge Representation in Enterprise Modelling, Technical Report, IBM T.J. Watson Research Center, P.O. Box 704, Yorktown Heights, NY, 10598. [Gruber 91].

There are three types of efforts related to ontology construction. There are efforts at creating sharable, re-useable ontologies for performing certain tasks. These are for real-world common-sense reasoning Lamprecht, James L., Implementing the ISO 9000 Series, Marcel Dekker Inc., 1993. [Lenat & Guha 90], mathematical modelling in engineering [Gruber 94], collaborative product design [Iwasaki et. al. 96] [Schreiber et. al.], planning complex logistics [Tate 95], and medical applications [Gennari et. al. 94]. Secondly, there are also efforts at creating standards for sharing heterogeneous sets of ontology representations, such that ontologies can be shared even if the languages in which the ontologies are represented differ. Ontolingua [Gruber 93a], Knowledge Query Manipulation Language (KQML) [Finin et. al. 94], and Knowledge Interchange Format (KIF) [Generseth & Fikes 92] constitute such efforts. Thirdly, there are large efforts at using ontologies for natural language processing, or knowledge-based machine translation: Ontologies represent natural language-independent concepts that can be used to translate between different types of natural languages. Some exemplar efforts are the μKosmos [Mahesh & Nirenberg 95] and Japangloss [Knight 96] projects. In order to use ontologies for enterprise modelling, ontologies for describing the business enterprise are needed; the Enterprise [Stader 96] and TOVE [Fox 92a] projects provide such ontologies.

2.5.2 TOVE Ontologies

For the TOVE project, an ontology is implemented as a deductive object-oriented database that consists of a data model of the objects in the database—implemented in ROCK™ or Prolog—and a logical model, expressed initially in first-order logic and translated to Prolog, which deduces answers to common-sense questions about the data model.

Within the TOVE Project, the ontologies considered fundamental to describe any enterprise are called the Core Ontologies. These are ontologies of activity, state, causality, and time, collectively called the activity-state ontology[Gruninger & Fox 94b]; resource [Fadel et. al. 94]; and organizational structure [Fox et. al. 95]. They are used as building blocks to construct additional ontologies that are peripheral to the core, such as ontologies of cost [Tham et. al. 94] and product

[Lin et. al. 96]. A rigourous ontological engineering methodology must be followed to construct, implement, and validate these ontologies.

2.5.3 Engineering an Ontology

There are two reasons why a systematic engineering methodology is important for ontologies. First, in order to affirm the scientific significance of these systems, it must be demonstrated that these systems are developed using a rigourous, repeatable methodology. A rigourous methodology ensures the systematic formalization of often-implicit concepts. Motivating and justifying explication of concepts give credibility that the ontology is rationally engineered, not whimsically crafted. Second, the construction of ontologies entails the use of classical systems engineering approach: classification of objects according to common characteristics and identification of hierarchical relationships of these classifications, identification of the nature of interrelationships and interdependencies between objects, and recognition of both cause and effect and synergism between objects [Turner et. al. 93]. The first stages of creating an ontology or micro-theory involve exactly this approach.

In assessing ontological engineering methodologies, one issue to consider is “What process is prescribed for building the ontology?” Some, like Sowa [95], prescribe a top-down approach of identifying general terminology and creating more specific and domain-dependent terms from general ones. Other efforts, like Plinius [van der Vet et. al. 94], start with the most specific terms and classify and generalize these. Uschold & Gruninger [96] prescribe a hybrid approach for starting from a middle layer of terminology and going top-down and bottom-up. METHONTOLOGY, “methodology to build ontologies from scratch” [Fernandez et. al. 97], is a more concrete specification for ontology construction. Some of its novel contributions are its premise that ontologies have a life cycle, and hence should be developed iteratively, and the importance it places upon documentation after each phase of the methodology. The domain of knowledge-based systems also has much to offer for an ontological engineering methodology such as know-how about requirements elicitation, knowledge acquisition and acceptance testing.

Another methodology is prescribed by Gruninger and Fox [95a]. The requirements of the ontology are expressed as queries, called competency questions, that should be answerable from an enterprise model constructed with ontology representations.

Gruninger and Fox prescribe a methodology to formally evaluate the completeness, consistency, and competency of the ontology by evaluating the capability of the ontology to answer these questions. In so doing, they address the second key issue for assessing an ontological engineering methodology: “What steps are prescribed for evaluating an ontology [Noy & Hafner 97]?” The choice of these evaluation factors dovetails with the importance that Guida & Mauri [93] place upon the same factors—i.e., completeness, consistency, and competency—for evaluating knowledge bases; since ontologies are used to construct knowledge bases, this dovetailing is expected. Fox [92a] states additional evaluation criteria such as generality, efficiency, perspicuity, transformability, extensibility, granularity, and scalability. Of these, Gruninger and Fox stress the importance of generality, since effective design of general, context-independent ontologies is a key for establishing ontology sharing and re-use. They place importance on constructing ontologies of core enterprise knowledge, for example, because such generic core ontologies support maximum re-useability [Valente & Breuker 97]. Describing any enterprise requires some reference to core enterprise knowledge, so any ontology-based enterprise models can be partially constructed using an ontology of core enterprise knowledge. Gruninger [93] also prescribe a method called reducibility to formally evaluate the generality and re-useability of an ontology.

2.6 Enterprise Models with an ISO 9000 Quality Perspective

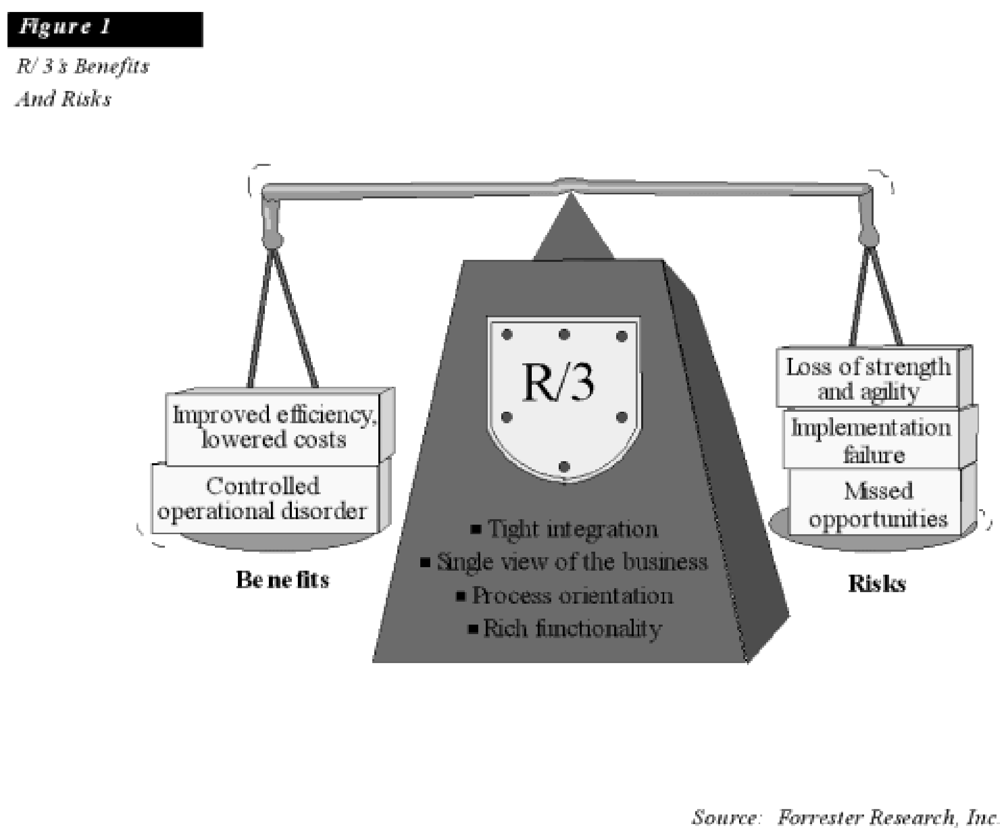
In this section, several specific models of quality are reviewed because they possess the following characteristics:

- they are data models of quality that include a model for ISO 9000 Compliance Analysis
- they are enterprise models that are implemented as an information system

2.6.1 SAP R/3™: An Enterprise-Wide Information System

SAP R/3™ [SAP 98] is the most popular example of an enterprise-wide information system that is used for operating parts of the enterprise. Baan [97] and IFS [97] are other providers of such exhaustive and extensive enterprise-wide enterprise modelling software. The scope of SAP R/3™ is much larger than quality management, and installation and maintenance of an SAP R/3™ system can cost in the millions of dollars. The intent of this review is not to make an assessment of the overall SAP R/3™ system; the intent is to identify only those features related to data modelling of enterprise quality that can be objectively compared with other models in this review. A suitable assessment of SAP R/3™ is offered by [Cameron et. al. 96]:

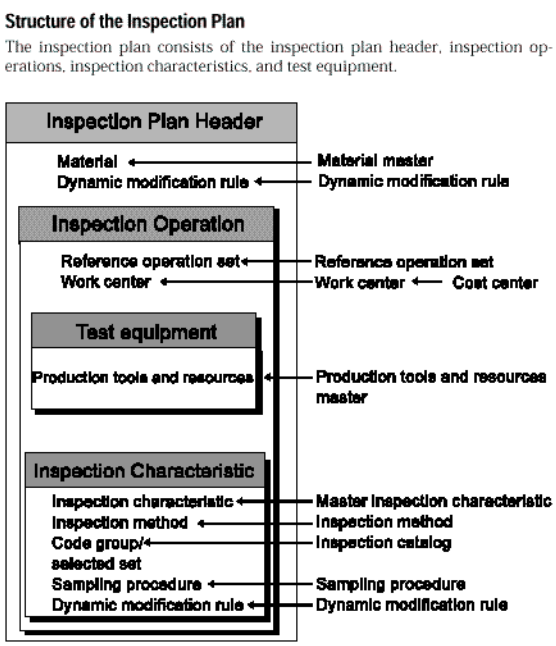
Figure 2.1 SAP R/3™ Benefits and Risks



As part of its offering, SAP R/3™ provides a quality module [SAP 95]. The module is used to centralize quality functions such as document and engineering configuration controls. The four quality sub-modules of SAP R/3™ are quality management in the logistical processes, quality planning, quality inspection, and quality control. This module is linked with the overall enterprise software such that it is possible to, for example, automatically trigger an inspection when goods are received. It is possible to use the module's functions to plan and execute inspection and other quality control activities.

The following figure shows a sample partial reference data model of the quality management module of SAP R/3™ ([SAP 95], pg. 4.6). This data object serves as a reference for modelling objects, attributes, and relations for modelling inspection.

Figure 2.2 **Sample Data Model for SAP R/3™ Quality Management Module**



SAP R/3™ is not based upon a formal data model, since the terms of the data model are identified and classified, but not formally defined in a logical language. As a result, much of the responsibility of interpreting the meaning of, or enforcing constraints upon, the term is left to the user, not the software.

Using the reference models pre-disposes an enterprise to construct their business processes a certain way. In this vein, strict use of the quality module pre-disposes practices that will make ISO 9000 compliance easier. However, there is no SAP R/3™ module that audits the ISO 9000 compliance of an organization. Moreover, SAP R/3™ offers re-useable reference business processes and objects that are specialized for certain industries; for example, there are reference process models for pharmaceutical and automotive industries. These models can be used to construct enterprise models more quickly. However, this can be problematic if the natural mode of business of an enterprise deviates from these models. For example, one Canadian telecom company designed new business processes, not because the old processes were inefficient, but because old processes could not be easily modelled using SAP R/3™. The quality module of SAP R/3™ can be used to address quality problems of the enterprise that other software products do. For example, SAP R/3™ offers some functionality for statistical quality control, but certainly not to the extent as specialized SQC software.

2.6.2 SCOPE: Formal Deductive Model for System Certification

The SCOPE [Hausen 93a] project is an effort to assess the quality of the "products" of software engineering (the programs, specifications, requirements, and documentation); to assess the capability of the processes involved in software engineering; and to evaluate and certify the quality of products and processes of software engineering. So, there are models of software products and processes. There is also a model for measuring software quality characteristics as per the quality model of McCall et. al. [77]. These models consist of an object model and a behaviour model that dictates transition from one state of the world of objects to another state. The behaviour model is described by a set of production rules, called the process representation technique (ProcePT), which are encoded in Prolog [Welzel 93]. Since these rules constrain the proper use of the data model, SCOPE is a formal data model.

The production rules of ProcePT are used to specify and test a software maker's processes. Although it is possible to assess the process model versus some ISO 9001 requirements [Welzel 95], only a few of the requirements are indeed represented, and these requirements can only be applied to software processes. Nevertheless, the SCOPE model supports deductive querying of a formal enterprise model to assess ISO 9000 compliance of the enterprise. The focus of SCOPE is not on formalizing ISO 9000 requirements, but rather on formalizing a "common sense" of software production and testing. Finally, re-useability of SCOPE's model is not emphasized.

2.6.3 WibQuS: A Workflow Quality Information System

The WibQuS (German acronym for knowledge-based systems in quality management) effort seeks to integrate several quality control methods such as SPC, SQC, and QFD by building a federated information system to support various quality control tasks. In WibQuS, information flows between the agents that execute these quality control methods and is analyzed by supporting knowledge-based tools [Peters & Szczurko 94]. The crux of this information system is a deductive object-oriented database management system of generic process models called the ConceptBase repository [Jarke & Peters 94]. Moreover, WibQuS' "methods models" describe specific techniques for solving quality problems, albeit there does not exist an ISO 9000 compliance method in WibQuS.

This model can be used off-line to analyze and design an enterprise, but it is meant to be a workflow model that helps operate the enterprise by facilitating on-line application of quality methods. Another interesting characteristic is that re-useability is emphasized. WibQuS facilitates shareability and re-useability by using existing and emerging standards for modelling repositories (Information Resource Dictionary Standard), process maturity (CMM), knowledge engineering framework (KADS), and representing design descriptions (STEP/Express). This way other researchers who have used some, or all, of these standards can reuse the representations from WibQus.

2.6.4 The Strategic Analyst™: A Software to Help Reach ISO 9000 Compliance

There are many software tools useful for meeting ISO 9000 compliance. Burr [94] offers a survey of such systems. Some tools automate parts of the quality documentation creation, modification, and maintenance processes. Examples are template manuals on paper that can be customized, such as [Carlsen et. al. 92]; forms that can be integrated with a company's word processing package, such as O'Logic™ [95]; and on-line packages that keep track of quality documentation for the whole enterprise, such as Accelerator™ [93]. Other packages extend documentation capability to make ISO 9000 compliance easier to achieve. Both QMS™ [91] and Powerway™ [94] offer enterprise-wide packages that can be used to control documents, control processes, or perform inspection and testing, such that the execution of these functions helps an organization comply with ISO 9000 requirements. Powerway™ also offers an assessment of the degree of compliance to the ISO 9000 requirements. It asks questions and answers to perform internal assessments of the quality system, and it produces reports, including GAP Analysis, that reveal a company's quality system status and its compliance with ISO/QS-9000.

Most software tools that assist directly in ISO 9000 audits only provide sophisticated checklists. Visual Assessor™ [94] merely presents colour graphics displaying conformance to specific requirements, but the degree of conformance is entered by an auditor. Assessor™ [94] extends this further and offers an easy-to-use interface to enter the data on the degree of conformance. However, the questions posed to the users of the software are from almost verbatim sentence-by-sentence dissection of the ISO 9000 requirements.

One that off-loads some of the trivial audit decisions onto a computer is The Strategic Analyst™ [93], an expert system for internal, informal, ISO 9000 audit, used to help prepare for the registrar-led audit. It offers some 500 questions, where the questions asked of the user depend upon the answers given to previous questions. Accompanying these questions are a data dictionary of ISO 8402 terms, as well as special terms used by the software; and an explanation facility to clarify the questions asked. The software also keeps a scorecard to display the degree of conformance. It still uses a checklist approach, because its key input source is the user of the software, not a model of the enterprise to be analyzed. This software is a one-of tool constructed solely for ISO 9000 compliance evaluation and so, as is characteristic of most one-of tools, it is neither formal nor re-useable.

2.7 Observations

Some of the best-known models of quality are models of philosophy of quality gurus such as Deming and Juran. These models are characterized by management philosophies rooted in the guru's definition of enterprise quality. The intended use of these models is to exhort a philosophy to an organization's management. These models are inherently informal, not comprised of restrictive and precise vocabulary, syntax, and semantics. So, they will not be interpreted similarly for all organizations. For example, Deming's "Drive out fear" serves as an important principle, but the interpretation and implementation of this is not uniform for all organizations that follow Deming's teachings. Such informality may lead to ambiguity, vagueness, and interpretations unintended by the model's creator.

Other models of quality are more formal, such as SQC (for ensuring process quality), and QFD (for capturing customer needs into product requirements). Since these formal models can be interpreted unambiguously, information systems are constructed upon these models. One important area for which the benefits of an information systems model can be brought to bear is quality auditing. Specifically, the computer can be used to automate verification methods, especially for dissemination and use of standards for verification. There is a trade-off with respect to existing models of automatic verification to standards. Formal computer models do not generally include standards verification functionality, and if they do, it is a rather trivial application, while models that provide this functionality tend not to be computer-based models rooted in a formalism.

There are formal information systems models of automation of verification to standards, where the standards are applied to the goodness of the software development process. However, the scope of these models is limited to software development. The two best-known, generic, and exhaustive quality standards, ISO 9000 and Baldrige Award, have much wider scope. The focus of ISO 9000 is on conformity of quality practices, whereas the Baldrige seeks to recognize and reward excellent quality. Obviously, the verification of conformance is much more of an appropriate task for an information systems model than the subjective assessment of the excellence of an organization's quality system. Although there are many information systems models that purport to automate the verification—this can also be called evaluation—of ISO 9000 compliance, they are generally bookkeeping aids for ISO 9000 audits. These models do not capture enough information about the enterprise; they rely upon the intuitive understanding of the enterprise of the auditor who uses the model. They are not constructed using a formal, generic enterprise model; this is a source of these models' limitations.

Many formal models have basic representations to model the enterprise. The CIM-OSA enterprise models, for example, offer representations such as business processes and procedural rule sets with which an organization's quality system can be modelled. For modelling quality, some models offer more: Scheer's for example, has a suite of typical quality-related functional departments as reference models. Scheer's model helps expedite the creation of a model of a quality system and offers quality-related reference models that are models of best practices, including models of ISO 9000 compliant quality processes. It offers analysis and guidance for ISO 9000 compliance, but it does not offer ISO 9000 compliance evaluation automation.

An important component of an enterprise model is the ability to explicitly ask questions of the model. ISO 9000 compliance evaluation automation is the capability to ask "Does the enterprise comply to the ISO 9000?" and receive an answer from the enterprise model. There are three types of such query and analysis models: factual, expert, and common-sense. A common-sense model is the most appropriate for the dual tasks of modelling generic quality concepts in an enterprise model of quality and using this model to automatically evaluate ISO 9000 compliance, because the tasks require a more formal model than a factual system, and a more generic model than an expert system. A common-sense model should be built using ontologies. The high degree of re-useability, characteristic of the use of ontologies, facilitates the careful engineering of a model from "building block" ontologies, and parts of the model can be used as building blocks for other ontology-based applications. The TOVE (TOronto Virtual Enterprise) project at the University of Toronto offers enough ontologies to richly describe the core, or the fundamentals, of a business enterprise.

To facilitate re-use, Gruber [93] states that an ontology should contain only minimal ontological commitment to give details or facilitate problem solving about a domain. Commitment beyond this to support specific enterprise tasks requires formalization of problem-solving domains, or micro-theories. So an ontology-based ISO 9000 compliance evaluation automation system would be comprised of ontologies for modelling enterprise quality and micro-theories for modelling ISO 9000 compliance.

A rigourous methodology is required to justify the validity of the representations of the model. One of the chief problems with the ISO 9000 is that registrars who audit organizations interpret the ISO 9000 very differently. Applying ISO 9000 relies too much upon the subjectivity of the auditor; it is too much of an art and not enough of a science. Following a rigourous methodology and using a formal model that restricts possible interpretations of the model ensures the development of a system that objectively automates ISO 9000 compliance evaluation. There exist many methodologies to guide the engineering of ontologies; the evaluation of an ontology should be part of any such methodology. Of the possible evaluation criteria for an ontology-based ISO 9000 compliance evaluation automation system, competency and re-useability are especially important; that is, the system should be competent to evaluate ISO 9000 compliance, and the components of this system should be re-useable.

From these observations, the following are discerned as questions that characterize the design choices for engineering a formal information systems model of quality for automating ISO 9000 compliance evaluation:

- Type of Quality Model: What types of quality processes does the model represent?
- Model's ISO 9000 Evaluation Competency: For what types of tasks related to ISO 9000 compliance can the model be used?
- Model Formality: To what extent are these models formal?
- Model Re-useability: To what extent are these models re-useable?

Select quality models with an ISO 9000 perspective can be reviewed versus these questions.

Table 2.1 Type of Quality Models: What type of quality processes does the model represent?

Models	Quality Planning	Quality Assurance	Quality Control	Quality Improvement
SAP R/3	x	x	x	x
SCOPE		x	x	x
WibQuS		x	x	
Strategic Analyst		x		

Table 2.2 Model's ISO 9000 Evaluation Competency: For what types of tasks related to ISO 9000 compliance can the model be used?

Models	ISO 9000 Evaluation Automation Competency	ISO 9000 Evaluation Support Competency	Ancillary Quality Competency
SAP R/3		x	x
SCOPE		x	x
WibQuS			x
Strategic Analyst	x	x	

Table 2.3 Model Formality: To what extent are these models formal?

Models	Terminology Informally Stated	Terminology Based upon a Data Model	Terminology Defined and Constrained in Axioms
SAP R/3	x	x	
SCOPE	x	x	x
WibQuS	x	x	
Strategic Analyst	x		

Table 2.4 Model Re-useability: To what extent are these models re-useable?

Models	Built on Sharable Representations	Same Competency for Different Enterprises	Different Competency for Same Enterprise
SAP R/3	x	x	x
SCOPE		x	
WibQuS	x	x	x
Strategic Analyst		x	

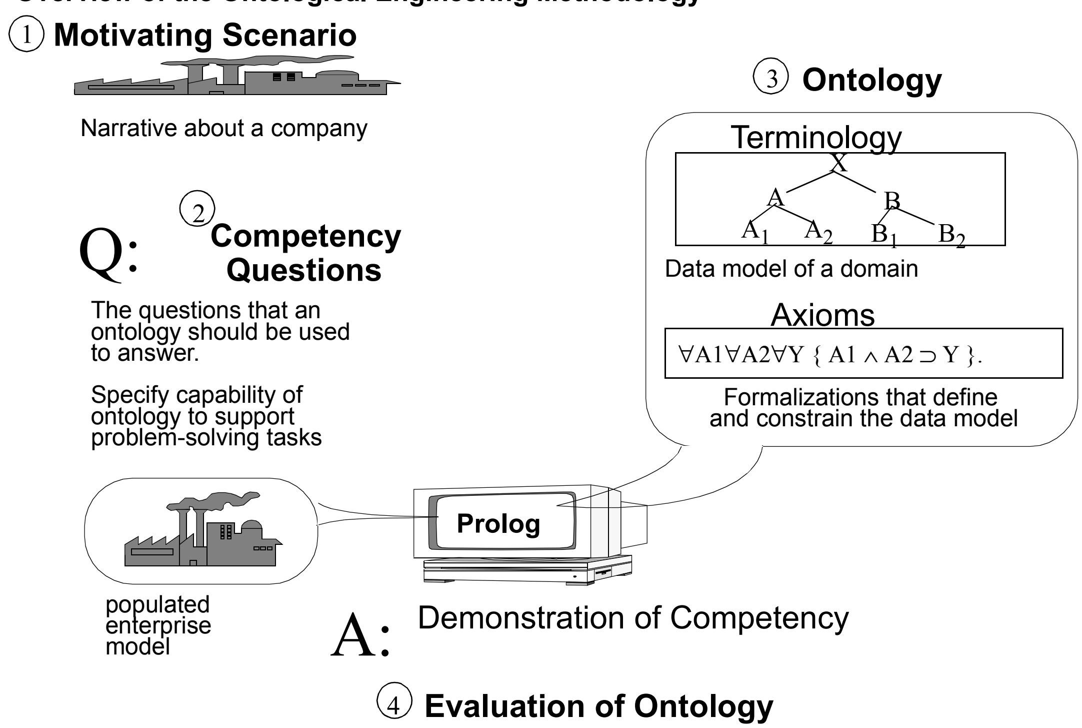
The observations and this review give evidence that a formal, re-useable model of enterprise quality that can be used for automatic ISO 9000 evaluation is needed.

3. Methodology

3.1 Précis

In this chapter, a "road map" for satisfying the objectives of the thesis, developed by Gruninger and Fox [95a], is presented. First, the terms, ontology, micro-theory, and advisor, are defined. Then, the methodology used to develop the Ontologies for Quality Modelling and ISO 9000 Micro-Theory is explained. This methodology, outlined below, is sequenced as 1) presenting motivating scenarios of industrial partners, 2) parsing and analyzing sections of the scenarios and deriving competency questions from this analysis that an ontology should be used to answer, 3) engineering the terminology and axioms that comprise an ontology or micro-theory, and 4) evaluating the ontology or micro-theory by demonstrating that competency questions about a populated enterprise model of the industrial partners can be answered, and by demonstrating reducibility.

Figure 3.1 Overview of the Ontological Engineering Methodology



The full motivating scenarios for this thesis' industrial partners, BHP Steel and deHavilland Manufacturing, are presented, as are the Core Ontologies, the building block representations upon which the ontological engineering methodology is applied to engineer the ontologies and micro-theory of this thesis. A step-by-step outline for the demonstration of competency is given; this demonstration shows that the ISO 9000 Quality Advisor can be a useful software tool, both for a business analyst and an ontology builder. Finally, the methodology for demonstrating reducibility—as a means of demonstrating the re-useability of an ontology or micro-theory's representations—is presented.

3.2 Ontology, Micro-Theory, and Advisor

Since the objectives of this thesis are the design, analysis, and construction of prototypical implementations of the Ontologies of Quality Modelling, ISO 9000 Micro-Theory, the ISO 9000 Quality Advisor, the terms, *ontology*, *micro-theory*, and *advisor*, are defined, and characteristics of these models are discussed.

Ontology

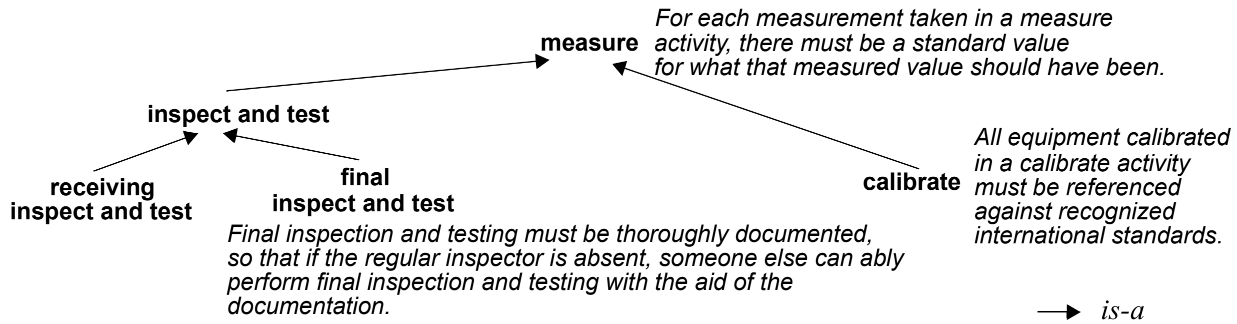
Ontology-based models are useful because:

- "All knowledge-based systems refer to entities in the world, but in order to capture the breadth of human knowledge, we need a well-designed global ontology that specifies at a very high level what kind of things exist and what their general properties are." ([Rich & Knight 91], pg. 292)

Gruber [93] outlines that the ontological engineering process entails the following: Associating human-readable terms in the universe of discourse (the domain) to computer-readable classes or *objects* (a collection of entities organized as one because they share common properties), *relations* (relationships between these entities), and functions; and stating formal axioms (roles within the domain) that constrain the interpretation and proper use of these terms. Thus the ontological engineering process is classified into creating a data dictionary of terms of the domain (terminology) and stating axioms that define and constrain the terminology.

One of the first steps in creating a data dictionary is the development of a *classification hierarchy* of terms, a *taxonomy*. This is a tree structure displaying *is-a* relationships in a hierarchical manner. The figure below is an example of a taxonomy. In it, a *final inspect and test is-a measure activity*, and properties of a *measure activity* are inherited such that these are implied to apply to *final inspect and test*. Because a taxonomy organizes the terms within the domain, and associates implicit information about the terms, it is invaluable for ontological engineering.

Figure 3.2 **Example of an Ontology**



Once a data dictionary is constructed, stating axioms about a domain entails defining and constraining the proper use of the terms in the data dictionary. Figure 3.2 is an example of this. Although some axioms are shown, it is conceivable that many more axioms can also be stated. The concept of *minimal ontological commitment*, though, is a way of bounding the number of axioms [Newell 82][Gruber 93]. This restricts axioms to those required to minimally describe a domain.

Micro-Theory

A micro-theory is a formal model of knowledge required to solve a problem in a domain or describe a subset of the domain in detail Lamprecht, James L., Implementing the ISO 9000 Series, Marcel Dekker Inc., 1993. [Lenat & Guha 90]. It is separate from, but constructed upon, an ontology. In the example, the axiom associated with *measure* is one that meets the minimal commitment criterion and belongs in the ontology of *measurement*, since this axiom defines all valid *measure* activities. The axiom associated with *final inspect and test* is not germane to the description of the *measurement* domain, and hence, does not belong in the *measurement* ontologies: It is inappropriate to categorically state that all the *measure* activities in the world should require thorough documentation. This axiom is an example of formalization that should be included in a micro-theory of, say, quality system auditing. Moreover, it must be ensured that the way the terms are used and the meanings of the terms in these axioms are consistent with the ontology of the domain. Then, the terms and axioms of the Measurement Ontology can be combined with axioms, such as the one associated with *final inspect and test*, to develop a micro-theory of best practices in measurement, such as the ISO 9000 Micro-Theory section on inspection and testing.

Advisor

An *advisor* is a software tool which encapsulates, and enables performing tasks using, ontologies and micro-theories [Fox & Gruninger 94]. The tasks for an advisor can be classified into evaluation, analysis, and guidance. *Evaluation* requires the ability to compare two different enterprise models along a dimension, such as quality or cost, and to evaluate that one model is better as per that dimension. For example, a *to-be* model of the enterprise is ISO 9000 compliant, but an *as-is* model is not. *Analysis* tasks require prediction, monitoring, identification, and explanation. This entails, for example, predicting compliance of a *to-be* model to an ISO 9000 requirement, monitoring noncompliance, identifying possible causes for the noncompliance of the *as-is* model, and explaining actual causes. For *guidance*, the advisor suggests alternatives such as “The ISO 9000 requirement can be complied to if the following best practices are adopted.” Using the advisor, the decision-maker has the capability to repeatedly pose queries to the populated enterprise model and manipulate the enterprise model to test out what-if scenarios. This allows for quick prototyping and analysis of alternate enterprise designs.

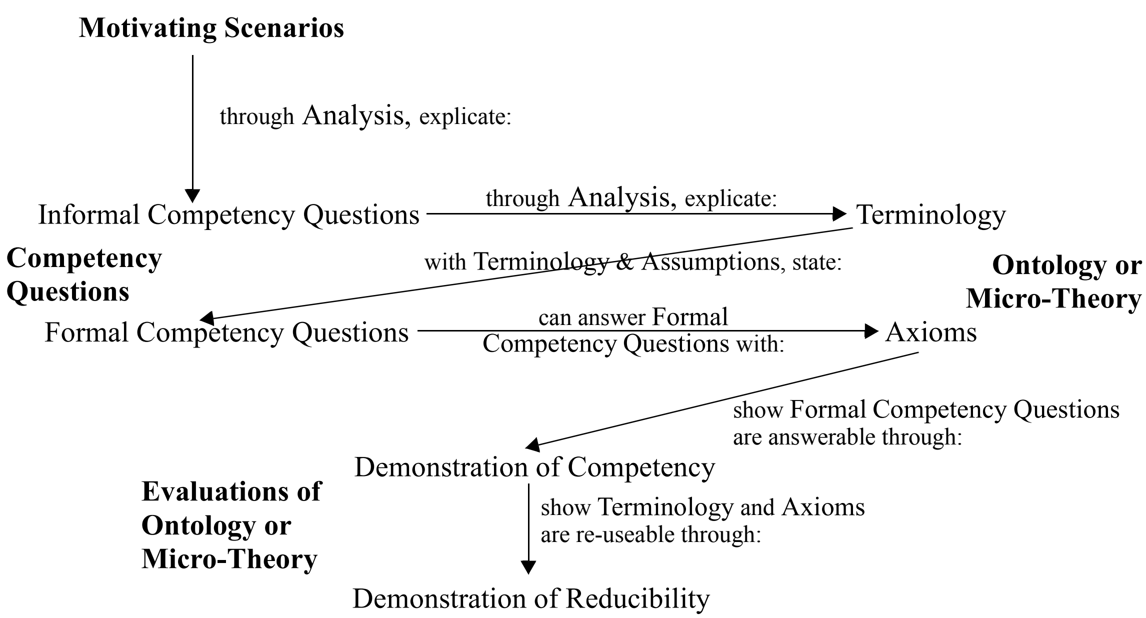
Ontology and Micro-Theory Design

So how should an ontology or micro-theory be designed? Clarity and coherence are two of many design factors offered by Gruber [93]. These factors can be designed into an ontology or micro-theory by following a rigourous, repeatable methodology to engineer, rather than craft, an ontology or micro-theory; the ontological engineering methodology of Gruninger & Fox [95a] provides the template for this. Generality and competence are two of the criteria for evaluating an ontology that Fox et. al. [93a] state. These criteria are used as part of the ontological engineering methodology. Evaluation of generality is operationalized in demonstrations of reducibility, and competence evaluation is operationalized as demonstrations of competency. In the next section, the methodology is further explained.

3.3 Ontological Engineering Methodology

The methodology, developed in the Enterprise Integration Laboratory by Gruninger and Fox [95a] and used for this dissertation, is comprised of the steps shown in the figure below.

Figure 3.3 Methodology for Developing Ontology and Micro-Theory Representations



3.3.1 Motivating Scenario

The motivating scenario for a specific enterprise is the detailed narrative about the enterprise, where special emphasis is placed on the problems that the enterprise is facing, or the tasks that it needs to perform. Ultimately, an application built using an ontology or micro-theory could solve this problem. As such, it depicts an application of the ontology or micro-theory. For example, the motivating scenario for BHP Steel provides background information about the company, statements about its concern about the quality of its products, the terminology that the company uses in talking about quality, explanation of how BHP Steel intends to use the enterprise model, and how it currently handles defects.

3.3.2 Analysis

By analyzing the motivating scenarios, generic concepts, independent of reference to a specific enterprise, are abstracted; such concepts are the basis of an ontology or micro-theory. For example, BHP Steel's motivating scenario highlights the need to model measurement in order to address product quality. This scenario, then, compels explication of design issues such as how to define and represent measurement, and how to represent the attributes of an entity that need to be measured. This analysis leads to representing a system for assessing measurement in the Measurement Ontology.

3.3.3 Informal Competency Questions

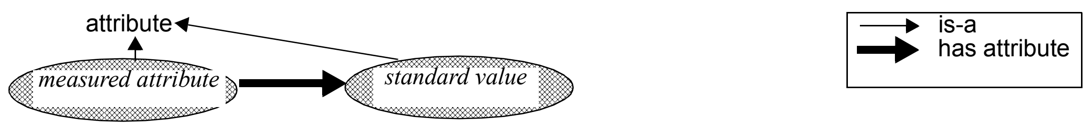
The analysis leads to the asking of competency questions. Competency questions are those that the ontology or micro-theory must be able to answer in support of a software application; they characterize the capability of the ontology or micro-theory to support problem solving tasks. These questions are inherently informal, since the terms required to pose these questions as formal queries in first-order logic have yet to be developed. Hence, the questions are asked in plain English with vocabulary and semantics indicative of the type asked by the users of the ontology or micro-theory. For representing the measurement assessment system, some of these questions are:

- What are the physical characteristics that are measured?
- What ought to be the measured value; that is, what is the expected value for that physical characteristic?

3.3.4 Terminology[1]

In order to more formally ask the competency questions, the terms with which the questions can be posed are identified, and then organized in a data model. A taxonomy of these terms, where the upper nodes are the terms pre-existing in the Core Ontologies, is part of this data model. The terms are expressed as predicates with English word definitions with no accompanying first-order logic definitions. Below is a partial data model, representing a term called *measured attribute*.

Figure 3.4 Partial Data Model of the Measurement Ontology



The following are example predicates of the ontology.

Meas Term: Pred-1. **measured_attribute(At)**

A certain attribute needs to be measured. It is an input into the model (a primitive term) that certain attributes to an object are *measured attributes*.

Meas Term: Pred-2. **has_standard_value(At,Mu)**

A *measured attribute has an attribute called standard value*, where this is what the value of the attribute ought to be.

The annotation Meas Term: Pred-1 means that the term **measured_attribute(At)** is the first predicate of the terminology of the Measurement Ontology.

Predicates that are never formally defined are called *primitive terms*. Enterprise models are populated by instantiating primitive terms with facts and specifying the situations in which the facts hold; an instance of a primitive term is a *ground term*. For example, the predicate **measured_attribute(At)**, is a primitive term. The fact that "average coil thickness" is a *measured attribute* for BHP Steel in a situation called a "verifying actual situation" is represented as a

1. Please note the following conventions which hold throughout all ontology chapters: Words that appear in *italics* denote fundamental concepts, assumptions, terminology, and axioms of the ontology discussed in a chapter. A term that is in *fuzzy italics* denotes a term for which its assumptions, definition and statements of constraint are provided in other ontology chapters of this thesis.

ground term, as *holds*(measured_attribute(average_coil_thickness[2]),sv_actual)[3]; that is, the variable, At, is set to average_coil_thickness and s is set to sv_actual.

3.3.5 Assumptions

Certain assumptions about the domain are made, and in so doing the scope of the ontology and micro-theory are bounded. What must be assumed about the terminology, and what type of potential competency questions can and cannot be answered by the ontology or micro-theory are discerned. For example, some assumptions about *measured attributes* are:

- The quality of a *traceable resource unit*[4] is evaluated by measuring the *measured attributes* of that *traceable resource unit*.
- The quality of an *activity* is entirely evaluated by measuring the *measured attributes* of *traceable resource units* associated with that *activity.*

3.3.6 Formal Competency Questions

Using the terminology, the informal competency questions are re-stated formally in first-order logic. This formally characterizes the competency of the ontology or micro-theory; an ontology or micro-theory satisfies the competency criterion of evaluation only if answers to the questions can be logically deduced by applying the formal ontology or micro-theory axioms to the populated enterprise model. The formal competency questions corresponding to the informal competency questions previously stated are:

- What are the physical characteristics that are measured?
    - Does there exist a *measured attribute* for a *traceable resource unit* (*tru*) κ in a situation σ? ∃At [*holds*(tru(κ),σ) ∧ *holds*(has_attribute(κ,At),σ) ∧ *holds*(measured_attribute(At),σ)].
- What ought to be the measured value; that is, what is the expected value for that physical characteristic?
    - For a *measured attribute* α in a given situation σ, what is its *standard value*? ∃Mu [*holds*(measured_attribute(α),σ) ∧ *holds*(has_standard_value(α,Mu),σ)].

---

2. Another convention is that ground terms as they are actually represented in the computer model are denoted by the font type and size as in average_coil_thickness or sv_actual.
3. *holds*(f,s) is a predicate which states that a fluent predicate f holds or is true in a situation s. A description of the *holds* predicate is provided later in this chapter.
4. A traceable resource unit is a batch or collection of a particular resource. This term is formally defined and constrained in the Traceability Ontology chapter.

The following are guidelines about interpreting formalizations in the thesis, including interpreting formal competency questions.

- Variables are denoted as starting with English letters, and are quantified before they are used. In the first question, At is a variable and ∃ (there exists) is the quantifier. The question is of the form ∃At [...], which can be interpreted as: “There exists a value for the variable, At, such that the conditions within the square brackets hold”.
- Facts are denoted as starting with Greek letters. With the above guideline, the first question can be interpreted as: “There exists a value for the variable, At, such that κ is a traceable resource unit, the value of At is an attribute of κ, and the value of At is a measured attribute, all in a given situation σ”.
- From the BHP Steel enterprise model, here are a set of facts:
  - One particular batch of raw coils is represented in the model; this is expressed as *holds*(tru(tru_wp_raw_coil_1),sv_actual).
  - A noteworthy feature of this batch is its average coil thickness; this fact is represented as *holds*(has_attribute(tru_wp_raw_coil_1,average_coil_thickness),sv_actual).
  - Average coil thickness is a feature that is measured; this fact is represented as *holds*(measured_attribute(average_coil_thickness),sv_actual).
- So, it is possible to pose the competency question as:
  - ∃At [*holds*(tru(tru_wp_raw_coil_1),sv_actual) ∧ *holds*(has_attribute(tru_wp_raw_coil_1,At),sv_actual) ∧ *holds*(measured_attribute(At),sv_actual)].
- It is obvious that this expression is true, if the variable, At, is set to average_coil_thickness. Since the expression is deduced to be true, the competency question has been answered.
- The ontology has sufficient formalizations to answer this competency question. If all of the ontology’s competency questions can be answered, then the ontology is said to be competent with respect to its competency questions.

3.3.7 Axioms

An axiom defines, and constrains the interpretation of, the terms in an ontology or micro-theory. Formalizations in first-order logic that define terminology already introduced are *definitions*; these are defined in terms of existing terms that have been previously defined, or in terms of primitive terms. *Constraining axioms* are first-order logic sentences that constrain the interpretation upon primitive terms and definitions. Below, **measured attribute(At)** is annotated as a primitive term of the axioms of the Measurement Ontology.

Meas Axiom: PT-1. **measured_attribute(At)**

A formal definition for the predicate **has_unit_of_measurement(At,U)** is provided below. It is annotated as the first definition axiom of the Measurement Ontology.

Meas Axiom: Defn-1. **has_unit_of_measurement(At,U)**

If an activity measures a *tru*, then the *unit of measurement* for this *activity*-*tru* pair is the *unit of measurement for the measured attribute*.

- *unit of measurement* and *has attribute* are terms from the Core Ontologies

$$\forall At \forall U \forall s \exists Rt \left[holds(tru(Rt),s) \wedge holds(has\_attribute(Rt,At),s) \wedge holds(measured\_attribute(At),s) \wedge \exists A \exists Un\ holds(unit\_of\_measurement(Rt,Un,U,A),s) \supset holds(\boldsymbol{has\_unit\_of\_measurement(At,U)},s) \right].$$

Meas Axiom: Cons-1. **measured attribute must have a unit of measurement**

All *measured attributes* must *have a unit of measurement*.

$$\forall At \forall s \left[holds(measured\_attribute(At),s) \supset \exists U\ holds(has\_unit\_of\_measurement(At,U),s) \right].$$

A: an activity
Rt: a tru measured by A
At: a measured attribute of both A and Rt
Un: describer for unit of measurement (default is capacity)
U: unit of measurement (default is object, but can be e.g. cm)
s: an extant or hypothetical situation

3.3.8 Demonstration of Competency

The demonstration of competency is explained in detail later in the chapter.

3.3.9 Demonstration of Reducibility

The demonstration of reducibility is explained in detail later in this chapter.

3.4 Motivating Scenarios

With the ontological engineering methodology thus explained, the first step of the methodology can be shown: the motivating scenarios of the industrial partners for this thesis—BHP Steel in Australia and deHavilland Manufacturing in Canada.

3.4.1 BHP Steel

BHP Steel is an international manufacturer of quality steel products. Its Flat Products Division (FPD) produces a wide range of finished and semi-finished flat steel products from two integrated steelworks. Port Kembla Steelworks is Australia's largest integrated steelworks, with a capacity approaching 5 million tonnes. It consists of two blast furnaces, which feed three basic oxygen steelmaking (BOS) vessels. Facilities include a steel ladle injection unit and plant for treatment of special steels under vacuum. Three continuous slab casters supply the division's own rolling mills, and supply other customers. FPD's rolling facilities at Port Kembla include a plate mill, hot strip mill, and tin mill; the facility at Westernport also has a rolling mill that is not so extensive [BHP 96d].

As raw materials are transformed by the different production units of BHP Steel's supply chain, non-prime products may also be produced. These are the products whose physical properties do not satisfy the necessary tolerance specifications; non-prime refers to sub-standard products, not an intended by-product. Non-prime products lead to lost revenue due to regrading and scrapping, increased costs due to additional rework, carrying of excess inventory to meet delivery promises, and increased variability of leadtime performance. Most importantly, non-prime leads to disaffected, even lost, customers. BHP Steel has chosen to construct a prototype model of the flat products division (FPD) to explore how to effectively handle and reduce the occurrence of non-prime products.

The prototype model encompasses all production units from the blast furnace at Port Kembla to the Western Port rolling mills. Especially when the product is shipped to the customer, it is essential that the product be within the tolerance specifications of the customer. So, the product's physical characteristics are measured, compared against their tolerance specifications, and a decision about whether the product is non-prime is made.

If the products are consistently found to be non-prime, this is an indication that there is something faulty in the production unit. A cause for this occurrence is suspected to be an inadequate inspection system. One way to check this is to compare BHP Steel's quality inspection system with established guidelines for conducting inspection, such as the ISO 9001 requirement on inspection and testing.

This check is part of BHP Steel's initiative to achieve ISO 9001 compliance. The general manager of FPD has appointed a chief quality manager, the main authority for ensuring that FPD achieves compliance. The FPD's main customers are export customers and the coating facilities at Port Kembla. The goals of FPD have been translated into a quality policy, and based upon this, the positions of the people at FPD have been carefully examined to make explicit the bearing of these positions on product quality. Another issue for achieving ISO 9000 compliance is the documentation of the revamped quality system, especially since proof of complete documentation is extremely important for compliance.

BHP Steel's flat products division wants to use an information systems model to assess the "goodness" of its quality system, so that it can be improved enough to meet ISO 9000 compliance, thus giving added confidence to its customers about the quality of its products.

3.4.2 deHavilland Manufacturing

deHavilland is one of the aircraft manufacturing divisions of the Bombardier group of companies. The company produces the Dash Series (100, 200, and 300 series) of twin turbo-prop engine aircraft widely used all over the world. The company occupies over one million square feet of manufacturing facilities and administrative offices within the boundaries of Metropolitan Toronto (Downsview) of the Province of Ontario, which is the most industrialized province in Canada. The company has over fifty years of experience in aircraft production. It is a unionized shop and employs approximately 3,200 personnel. With this work-force complement and facilities, it is estimated that maximum plant capacity is about 5 aircraft per month [Tham 98].

It has been identified that quality problems exist for a specific class of products called Leading Edge, used for their aircraft wings. Production responsibility for this product mainly falls within the Methods Fabrication Department of deHavilland. The Production Process Standards (referred to as PPS at deHavilland) for leading edges call for highly specialized raw materials such as reinforced fibre-glass composite sheets that have to be intricately cut and laid up for specialized curing and forming processes that demand highly skilled and expensive labour. After several impregnations of epoxies and resins followed by curing processes that use highly specialized and capital intensive equipment, the initial cloth-like composite fabrics end up as very hard laminated, rounded, and elongated shaped product called the leading edge. This product then goes through a paint line procedure and eventually forms the front portion of an aircraft wing. The leading edge includes the “de-icer boot” system made up of specialized rubber. Through this de-icer boot system, tiny jets of warm air are pumped in order to “de-ice” the wings. The integrity of this system is absolutely essential.

When inspection of a unit of Leading Edge points to a problem, it must be investigated where the defect occurred; the problem could have occurred anywhere between the inspection of the final product all the way back to the raw materials. For this diagnosis, an important piece of information is what quantity of a batch was produced and how much of it was used by a preceding production unit. Knowing the quantity of the batches produced allows for the assessment of the extent to which problems from one process cascaded to subsequent processes. Since tracing material flow is extremely important for product quality, the ISO 9001 requirement upon unique identification and traceability is used to audit the existing traceability system.

3.5 Core Ontologies

The next step is to explain how the facts of the industrial partners are represented in an enterprise model. The schema—as in a database schema—with which the facts are represented also contain the building blocks with which the Ontologies for Quality Modelling and the ISO 9000 Micro-

Theory are constructed: the Core Ontologies. These ontologies represent what is "core" enterprise knowledge, and represent knowledge about activity, states, causality, time, resources, and organizational structure.

3.5.1 Foundations of the Core Ontologies

The Core Ontologies are founded upon the first-order language for representing dynamically changing worlds, called the situation calculus [McCarthy & Hayes 69]. In situation calculus, each perturbation to the modelled world changes the world from one *situation* to another *situation*. In this model are terms that describe an entity or the relationship between entities in this world; if the truth value of such a term varies from situation to situation, then the term is a *fluent*. A *fluent holds* in a given *situation*, if the term is true in that given *situation*. All terms defined in the Ontologies for Quality Modelling are *fluents*. A *fluent occurs in a given time period*, if for all the *situations* which occur during that period, the *fluent holds*.

Sit Calc Term: Pred-1. **s**

s: The world is thought of as being in some situation s; this situation can change only as a consequence of some agent performing an action

Sit Calc Term: Pred-2. ***f***

f: fluents are relations whose truth value may vary from situation to situation

Sit Calc Term: Pred-3. ***holds*(*f*,s)**

f: a fluent: a predicate whose truth value may change
s: situation in which the value of the fluent is true

Sit Calc Term: Pred-4. $\boldsymbol{occurs_T}$**(*f*,T)**

f: a fluent: a predicate whose truth value may change
T: time period for which *f* holds

A special type of a *fluent* is an *agent constraint* [Gruninger 96a]. This is a constraint upon an *organizational agent* (an individual or group of individuals) that must be satisfied in order for that agent to achieve some goal. For instance, ISO 9000 compliance can be represented as goal that is achieved if a set of quality-related agent constraints upon an enterprise is satisfied. An entity satisfies an *agent constraint* in a given situation, if the constraint is true in the modelled world in that situation.

Sit Calc Term: Pred-5. **agent_constraint(A,c($\underline{X}$))**

*holds(**agent_constraint(A,c($\underline{X}$))**,*s) = Φ(A,$\underline{X}$,s)

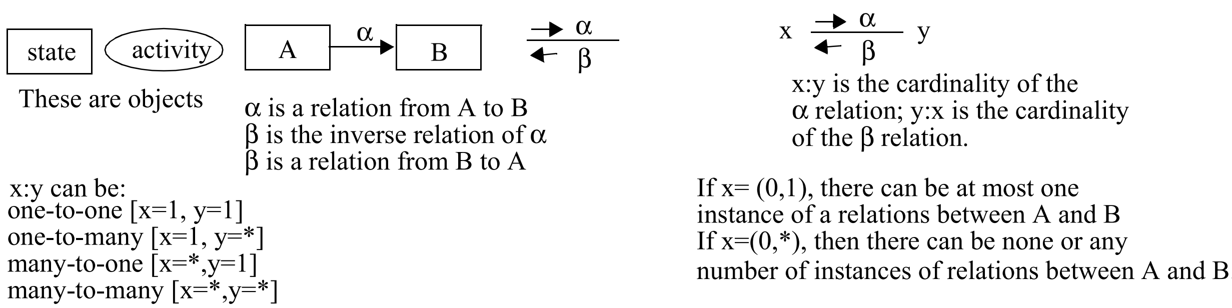
s: a given situation
A: an agent which seeks to achieve a goal in situation s
$\underline{X}$: entities that must be represented in order to represent the constraints on A; $\underline{X}$ is a vector with none, one, or more entities
C($\underline{X}$):predicate name for the agent constraint
Φ(A,$\underline{X}$,s): a first-order logic expression for the constraint described as C($\underline{X}$)

3.5.2 Core Ontologies: Activity, State, Causality, Time, and Resource Ontologies

There exist formal definitions and constraints for the terms of the Core Ontologies. However, only the informal definitions, data models, and statements of the predicates—the terminology of the Core Ontologies—and very few definitions are presented in this thesis. Rather where appropriate, references where formal definitions and explanations can be found are given.

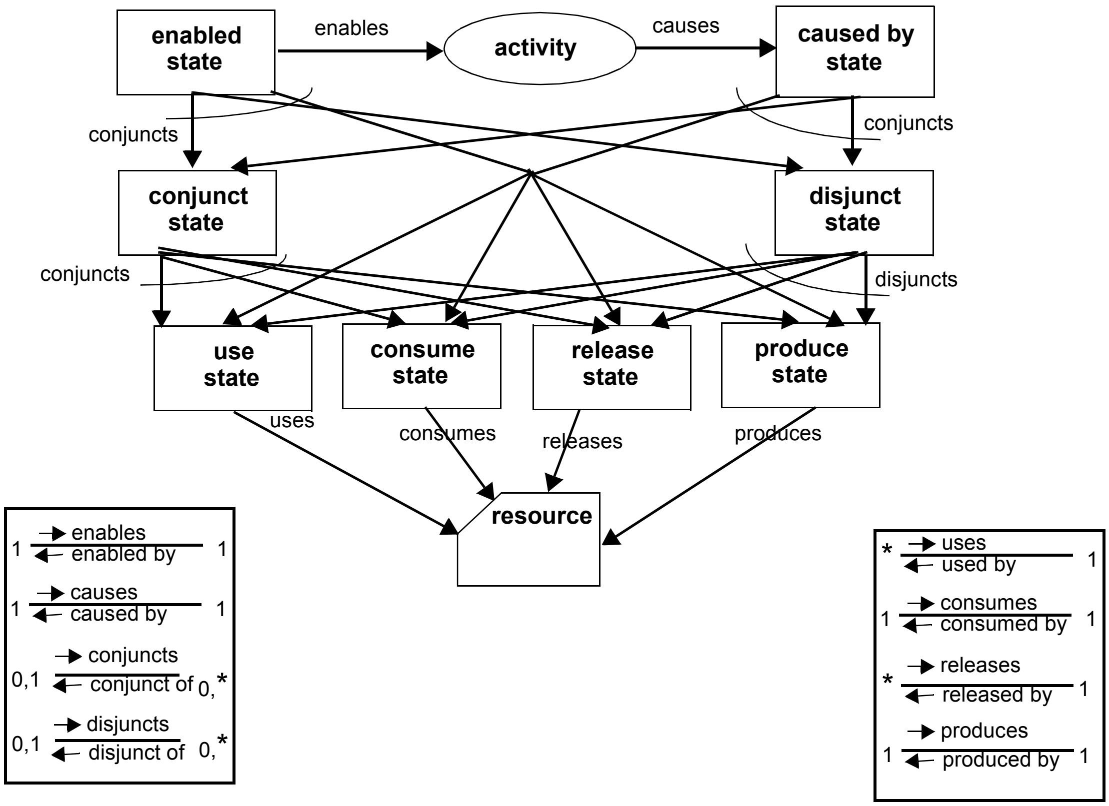
In presenting terminology of the Core Ontologies—or the terminology for the Ontologies for Quality Modelling—graphical presentations of data models are important. The following are the conventions for presenting data models in this thesis.

Figure 3.5 Data Modelling Conventions used throughout this thesis



The fundamental data model for the Core Ontologies is called the activity cluster, and is shown below. At a given point in time, the modelled world is in one situation but activities and resources modelled with activity clusters can be in many states. When there is a state change to a resource or activity of an activity cluster, the situation of the modelled world changes.

Figure 3.6 **Activity Cluster: An Object-Oriented Model of Activities, States, Causality, Time, and Resources**



According to the activity/state, causality, time, and resource ontologies, the usage, consumption, production, or release of a *resource* is represented as a *uses*, *consumes*, *produces*, or *releases* relation, respectively, between a description of the *state* of the world and a *resource*. These state descriptions can be composed using and (*conjuncts*) & or (*disjuncts*) relations to form two composite state descriptions: *enable state* and *caused by state*. An *enable state* is a description of what is true of the world before an *activity* is performed; a *caused by state* is a description of what is true after the *activity* is performed. The predicates below are formally defined by Fox et. al. [93a].

Core Term: Pred-1. **activity(A)**[5]

5. Unless explicitly stated otherwise. All terms defined in all ontology sections are fluents. so for example the form in which this term is used is *holds*(activity(A),s).

- *holds*(activity(autoclave_cure_L_edge),sv_actual).[6]

  A: an activity

Core Term: Pred-2. **state(St)**

- *holds*(state(es_autoclave_cure_L_edge),sv_actual).

  St: a state description of the world: activities and resources can be in different states.

Core Term: Pred-3. **resource(R)**

- *holds*(resource(cured_L_edge),sv_actual).

  R: a resource

Core Term: Pred-4. **produces(St,R)**

- *holds*(produces(produce_cured_L_edge,cured_L_edge),sv_actual).

  A state (St) produces a resource (R) if R is produced or created as a result of an activity.

Core Term: Pred-5. **consumes(St,R)**

- *holds*(consumes(consume_4_L_edge,L_edge_S004),sn-1).

  A state (St) consumes a resource (R) if R's physical characteristics are modelled to change in an activity. R is transformed to produce another resource.

Core Term: Pred-6. **uses(St,R)**

- *holds*(uses(use_686_oper,autocl_686_oper),sn-1).

  A state (St) uses a resource (R) if R participates in the transformation of a consumed resource to produce another resource.

Core Term: Pred-7. **releases(St,R)**

- *holds*(releases(release_686_oper,autocl_686_oper),sn-1).

  St: state which describes the release of a resource by the activity after the activity used that resource

  R: a resource for which its release by an activity is described by St

Core Term: Pred-8. **enables(St,A)**

- *holds*(enables(es_autoclave_cure_L_edge,autoclave_cure_L_edge),sv_actual).

  St: a state description, comprised of all the state descriptions related to the activity, of what is true before the activity executes.

  A: an activity

Core Term: Pred-9. **conjuncts(St_1,St_2)**

- *holds*(conjuncts(es_autoclave_cure_L_edge,cs_autoclave_cure_L_edge),sv_actual).

  St_1: a state description which is composed of other state descriptions

  St_2: a state description which must describe some truth in the world in order for St_1 to be true

Core Term: Pred-10. **disjuncts(St_1,St_2)**

  St_1: a state description which is composed of other state descriptions

  St_2: a state description which may describe some truth in the world in order for St_1 to

6. Where they exist, ground terms from the populated enterprise model for deHavilland Manufacturing are shown to give examples to predicates.

be true

Core Term: Pred-11. **caused_by(St,A)**

- *holds*(caused_by(cs_autoclave_cure_L_edge,autoclave_cure_L_edge),sv_actual).

St: a state description, comprised of all the state descriptions related to the activity, of what is true after the activity executed.
A: an activity

Sub-activities comprise an activity. There can exist a hierarchy of sub-activities, so an *activity may have descendent sub-activities*. Similarly, there can exist a hierarchy of state descriptions, so a *state may have descendent states*.

Core Term: Pred-12. **has_subactivity(A,A_0)**

- *holds*(has_subactivity(plastic_shop_fabrication_1,autoclave_cure_L_edge),sv_actual).

A: activity
A_0: a subactivity of A

Core Term: Defn-1. **has_descendent_subactivity(A,Ao)**

$$\forall A \forall Ao \forall s \Big[\mathit{holds}(\text{has\_subactivity}(A,Ao),s) \vee \exists Ax \,(\mathit{holds}(\text{has\_subactivity}(A,Ax),s) \wedge \mathit{holds}(\text{has\_descendent\_subactivity}(Ax,Ao),s)) \supset \mathit{holds}(\boldsymbol{\mathit{has\_descendent\_subactivity(A,Ao)}},s) \Big].$$

A: activity
A_0: a descendent subactivity of A
s: an extant or hypothetical situation

Core Term: Defn-2. **has_descendent_state(St,Sto)**

$$\forall St \forall Sto \forall s \Big[(\mathit{holds}(\text{conjuncts}(St,Sto),s) \vee \mathit{holds}(\text{disjuncts}(St,Sto),s)) \vee \exists Stx \,(\mathit{holds}(\text{conjuncts}(St,Stx),s) \vee \mathit{holds}(\text{disjuncts}(St,Stx),s)) \wedge \mathit{holds}(\text{has\_descendent\_state}(Stx,Sto),s)) \supset \mathit{holds}(\boldsymbol{\mathit{has\_descendent\_state(St,Sto)}},s) \Big].$$

St: a state
Sto: a descendent state of St
s: an extant or hypothetical situation

There are also relations between *use*, *consume*, *release*, and *produce* states, and the *activity* for which these state descriptions exist.[7]

Core Term: Defn-3. **use(St,A)**

$$\forall St \forall A \forall s \exists Stx \Big[\mathit{holds}(\boldsymbol{\mathit{use(St,A)}},s) \equiv \mathit{holds}(\text{enables}(Stx,A),s) \wedge \mathit{holds}(\text{has\_descendent\_state}(Stx,St),s) \wedge \exists R\, \mathit{holds}(\text{uses}(St,R),s) \Big].$$

St: state which describes the usage of a resource by an activity
A: an activity for which its usage of a resource is described by St
R: a resource for which its use is described by St

7. Often, these predicates are used as convenient terms to write shorter first-order logic sentences.

s: an extant or hypothetical situation

Core Term: Defn-4. **consume(St,A)**

$$\forall St \forall A \forall s \exists Stx \left[holds(\textbf{\textit{consume(St,A)}},s) \equiv holds(\text{enables}(Stx,A),s) \wedge holds(\text{has\_descendent\_state}(Stx,St),s) \wedge \exists R\ holds(\text{consumes}(St,R),s) \right].$$

St: state which describes the consumption of a resource by an activity
A: an activity for which its consumption of a resource is described by St
R: a resource for which its consumption is described by St
s: an extant or hypothetical situation

Core Term: Defn-5. **produce(St,A)**

$$\forall St \forall A \forall s \exists Stx \left[holds(\textbf{\textit{produce(St,A)}},s) \equiv holds(\text{caused\_by}(Stx,A),s) \wedge holds(\text{has\_descendent\_state}(Stx,St),s) \wedge \exists R\ holds(\text{produces}(St,R),s) \right].$$

St: state which describes the production of a resource by an activity
A: an activity for which its production of a resource is described by St
R: a resource for which its production is described by St
s: an extant or hypothetical situation

Core Term: Defn-6. **release(St,A)**

$$\forall St \forall A \forall s \exists Stx \left[holds(\textbf{\textit{release(St,A)}},s) \equiv holds(\text{caused\_by}(Stx,A),s) \wedge holds(\text{has\_descendent\_state}(Stx,St),s) \wedge \exists R\ holds(\text{releases}(St,R),s) \right].$$

St: state which describes the release of a resource by an activity after the activity used that resource
A: an activity for which its release of a resource is described by St
R: a resource for which its release is described by St
s: an extant or hypothetical situation

Since quantities are represented, the *unit of measurement in which these quantities are quantified* is also represented. This term is formally defined by Fadel [94].

Core Term: Pred-13. **unit_of_measurement(R,UName,U,A)**

R: a resource
UName: a description for unit of measurement (default is capacity)
U: unit of measurement (default is object, but can be for example cm)
A: the activity that uses/consumes/releases/produces the resource

There is a *time period for the performance of an activity.* This is composed of the *time periods for the different state descriptions* that describe the activity. These time periods have a *start point* and an *end point*, and it is possible to discern if *a given time point falls within a time period*. The time duration terms are formally defined by Gruninger & Fox [94a], and competency questions for time points are stated in Fox et. al. [93a].

Core Term:Pred-3.14 $occurs_T$**(activity_duration(A,T))**

A: an activity
T: description of period of time between when the activity starts execution, to when the activity completes.

Core Term: Pred-15. *occurs*$_T$**(state_duration(St,T))**

St: a use/consume/release/produce state
T: description of period of time from when St is enabled, until it is completed.

Core Term: Pred-16. **start_point(T,Tp)**

T: a description of a period of time
T_p: time point at which T begins
This is not a fluent

Core Term: Pred-17. **end_point(T,Tp)**

T: description of a time period
T_p: time point at which T ends
This is not a fluent

Core Term: Pred-18. **has_point(T,Tp)**

T: description of a time period
T_p: time point that is in the T time period
This is not a fluent

Underlying the activity cluster are general object-oriented relations such as *has subclass*, *has descendent*, *has attribute*, and *has attribute value* relationships.

Core Term: Pred-19. **has_subclass(X,X_o)**

X: an object
X_o: a subclass of X

Core Term: Defn-7. **has_descendent(X,Xo)**

$$\forall X \forall Xo \forall s \Big[\textit{holds}(\text{has\_subclass}(X,Xo),s) \lor \exists Xx \, (\textit{holds}(\text{has\_subclass}(X,Xx),s) \land \textit{holds}(\text{has\_descendent\_subclass}(Xx,X),s)) \supset \textit{holds}(\boldsymbol{\textit{has\_descendent\_subclass}(X,Xo)},s) \Big].$$

X: an object
X_o: a descendent subclass of X
s: an extant or hypothetical situation

Core Term: Pred-20. **has_attribute(X,Atr)**

X: any object
Atr: an attribute of X

Core Term: Pred-21. **has_attribute_value(X,Atr,V)**

X: any object
Atr: an attribute of X
V: value of At for X

3.5.3 Core Ontologies: Organization Ontology

The representations from the organization ontology are graphically displayed below: These representations are formally defined by Fox et. al. [95].

Figure 3.7 Object-Oriented Model of Organizational Structure

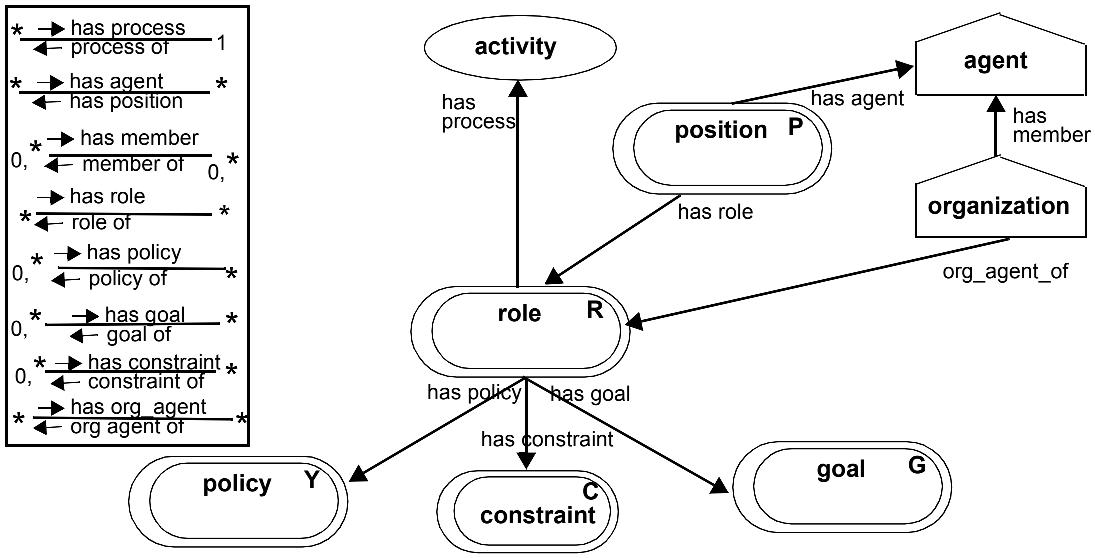
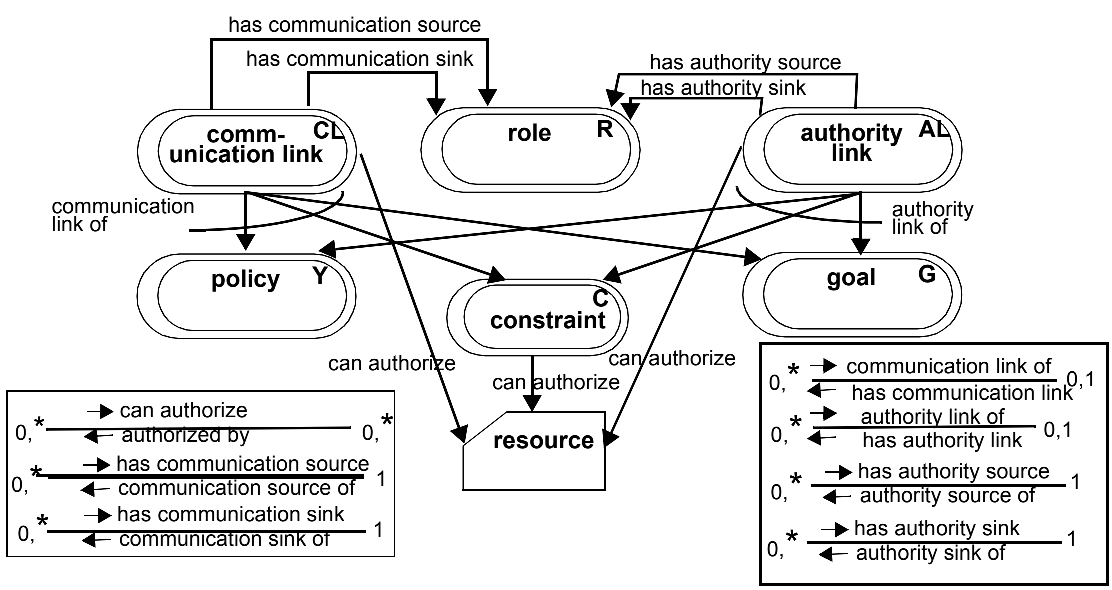


Figure 3.8 Object-Oriented Model of Information Flows



According to the organization ontology, *roles specify* the following: the *agents* fulfilling those roles, a *position* which is associated with one or more *roles*, and the *organization to which the agents fulfilling those roles belong*.

Core Term: Pred-22. **organization_agent(Oa)**

- *holds*(organization_agent(colin_montrose),sv_actual).[8]

Oa: an individual or group of individuals

Core Term: Pred-23. **role(Ro)**

- *holds*(organization_agent(bhp_steel_quality_manager_role_1),sv_actual).

Ro: an organizational role

Core Term: Pred-24. **position(Pt)**

- *holds*(position(bhp_steel_quality_manager).

Pt: name for an organizational position

Core Term: Pred-25. **has_member(Oa_1,Oa_2)**

- *holds*(has_member(bhp_steel_1,wp_qc_1),sv_actual).

Oa_1: an organization agent which is comprised of other organization agents
Oa_2: an organization agent which is a member of Oa_1

If Oa_1 is a grouping of agents, and Oa_2 is one of these agents, then Oa_1 has member Oa_2.

Core Term: Defn-8. **has_descendent_member(Oa_1,Oa_2)**

$$\forall Oa_1 \forall Oa_2 \forall s \Big[\, holds(\text{has\_member}(Oa_1,Oa_2),s) \vee \exists Oa_x \,(\, holds(\text{has\_member}(Oa_1,Oa_x),s) \wedge holds(\text{has\_descendent\_member}(Oa_x,Oa_2),s)\,) \supset holds(\boldsymbol{\text{has\_descendent\_subactivity}(Oa_1,Oa_2)},s) \Big].$$

Oa_1: an organization agent which is comprised of other organization agents
Oa_2: an organization agent which has an ancestor Oa_1
s: an extant or hypothetical situation

If C is a member of B, and B is a member of A, then C and B are descendent members of A

Core Term: Pred-26. **has_agent(Pt,Oa), where Pt is an organization position**

- *holds*(has_agent(bhp_steel_quality_manager,colin_montrose),sv_actual).

Pt: an organization position
Oa: an organization agent

An agent named John Smith may be filling a position of "line quality manager."

Core Term: Pred-27. **has_agent(Ro,Oa), where Ro is an organization role**

- *holds*(has_agent(wp_hcpf_260_management_role_1,wp_hcpf_260_1),sv_actual).

Ro: an organization role
Oa: an organization agent

If an agent is a grouped entity like the QC department, then one of its roles

8. When they exist, ground terms from the populated enterprise model for BHP Steel are shown as examples for predicates.

may be "ensure quality of products."

Core Term: Pred-28. **has_role(Pt,Ro), where Pt is an organization position**

- *holds*(has_agent(bhp_steel_quality_manager,bhp_steel_quality_manager_role_1),sv_actual).

Pt: an organization position

Ro: an organization role which is fulfilled by an agent filling the position Pt.

If an agent is filling the position of line quality manager, then one of the agent's roles in that position may be to "ensure quality of a specific product."

Core Term: Pred-29. **has_role(Oa,Ro), where Oa is an organization agent**

- *holds*(has_role(wp_hcpf_260_1,wp_hcpf_260_management_role_1),sv_actual).

Oa: an organization agent

Ro: an organization role which is fulfilled by the agent Oa

If an agent is a grouped entity like the QC department, then one of its roles may be "ensure quality of products."

As well agents, *fulfilling the roles perform activities*.

Core Term: Pred-30. **has_process(Ro,A)**

- *holds*(has_process(bhp_steel_quality_manager_role_1,process_wp_qc_1),sv_actual).

Ro: an organization role

A: an activity which is performed to fulfill the role R

Roles are related to the *policies* and *constraints that constrain that role* and the *goals that must be satisfied.*

Core Term: Pred-31. **policy(Y)**

- *holds*(policy(bhp_steel_quality_policy_1),sv_actual).

Y: ID for an organizational policy

Core Term: Pred-32. **goal(G)**

- *holds*(goal(bhp_steel_q_objective_1),sv_actual).

G: ID for an organizational goal

Core Term: Pred-33. **organizational_constraint(C)**

C: ID for an organizational constraint

Core Term: Pred-34. **has_goal(Ro,G)**

- *holds*(has_goal(bhp_steel_q_manager_role_1,bhp_steel_q_objective_1),sv_actual).

if *holds*(agent_constraint(A,G),s) $\equiv \Phi(A,G,s)$, then the goal uniquely identified by G is satisfied if $\Phi(A,G,s)$ is true.

The goal (uniquely identified as G) of a role (Ro) is an expression that states the desired outcome for that role; it states the objectives for the role

A goal for the role "inspect line products" may be "the defect rate must be less than 1%".

Core Term: Pred-35. **has_policy(Ro,Y)**

- *holds*(has_policy(bhp_steel_q_manager_role_1,wp_qc_q_procedure_1),sv_actual).

If *holds*(agent_constraint(A,Y,s) ≡ Φ(A,Y,s), then the policy uniquely identified by Y is satisfied if Φ(A,Y,s) is true.

A policy (uniquely identified as Y) of a role (Ro) is an expression that constrains that role; generally, it specifies how that role is performed.

A policy for the role "inspect line products" may be "inspect a random sample of 20 from each batch".

Core Term: Pred-36. **has_constraint(Ro,C)**

If *holds*(agent_constraint(A,C,s) ≡ Φ(A,C,s), then the constraint uniquely identified by C is satisfied if Φ(A,C,s) is true.

An organizational constraint (uniquely identified as C) is an arbitrary constraint on a role (Ro); it can be a goal or a policy, or neither.

A constraint for the role "inspect line products" may be "the inspection must be performed".

Roles are related to the *communication links* that the role can receive (*communication sink*) and send (*communication source*), and the *authority links* that the role can receive (*authority sink*) and send (*authority source*). The content of these links *can authorize* status changes to entities.

Core Term: Pred-37. **communication_link(L)**

- *holds*(communication_link(wp_hcpf_260_q_evidence_link_1,sv_actual).

A communication link (L) is a modelled communication channel between peer roles

Core Term: Pred-38. **authority_link(L)**

- *holds*(authority_link(bhp_steel_q_objective_link_1),sv_actual).

An authority link (L) models the channel between roles where one role has authority over another

Core Term: Pred-39. **communication_link_of(L,I)**

- *holds*(communication_link_of(wp_hcpf_260_q_evidence_link_1,hcpf_260_q_evidence_1),sv_actual).

If *holds*(agent_constraint(A,I,s) ≡ Φ(A,I,s), then the policy, goal, or constraints uniquely identified by I is the content of the communication link that is uniquely identified by L

L: unique ID for a communication link
I: unique ID for an organization policy, goal, or constraint

A communication link may be the ID for a message sent to the role of the activity which produces the batches that are inspected; the content of the link may be the goal that "the defect rates must be consistently over 1%."

Core Term: Pred-40. **authority_link_of(L,I)**

- *holds*(authority_link_of(bhp_steel_q_objective_link_1,bhp_steel_q_objective_1),sv_actual).

If *holds*(agent_constraint(A,I,s) ≡ Φ(A,I,s), then the policy, goal, or constraints uniquely identified by I is the content of the authority link that is uniquely identified by L

L: unique ID for an authority link
I: unique ID for an organization policy or goal

An authority link may be a message sent to a role for a sub-ordinate; the content of such a link may state the policy "change the sampling method...

the current one may not be random."

Core Term: Pred-41. **has_communication_source(L,Ro)**

- *holds*(has_communication_source(wp_hcpf_260_q_evidence_link_1,wp_hcpf_260_manager_role_1),sv_actual).

L: ID for a communication link
Ro: an organization role which sends the communication link
An agent may, in fulfilling the role "inspect line products," send a message with an unique ID, where the content of this message may be "defective item detected."

Core Term: Pred-42. **has_communication_sink(L,Ro)**

L: ID for a communication link
Ro: an organization role which receives the communication link
An agent may, in fulfilling the role "control defects," receive a message with an unique ID, where the content of this message may be "defective item detected."

Core Term: Pred-43. **has_authority_source(L,Ro)**

- *holds*(has_authority_source(bhp_steel_q_procedure_link_1,bhp_steel_q_manager_role_1),sv_actual).

L: ID for an authority link
Ro: ID for an organization role which sends the authority link
An agent may, in fulfilling the role "manage inspectors," send an authority link to a sub-ordinate which states the policy "change the sampling method... the current one may not be random."

Core Term: Pred-44. **has_authority_sink(L,Ro)**

- *holds*(has_authority_sink(bhp_steel_q_objective_link_1,bhp_steel_q_manager_role_1),sv_actual).

L: ID for an authority link
Ro: ID for an organization role which receives the authority link
An agent may, in fulfilling the role "inspect line products," receive an authority link from a manager which states the policy "change the sampling method... the current one may not be random."

Core Term: Pred-45. **can_authorize(L,X)**

L: ID for a communication or authority link
X: an entity for which the change in its status value is authorized by the receipt of the communication or authority link
An agent may, in fulfilling the role "inspect line products," receive an authority link from a manager which states the policy "change the sampling method... the current one may not be random." The receipt of this message authorizes the agent to change the status of the activity to disenabled while the agent changes the sampling method.

And a goal, policy, or constraint is composed of other goals, policies, or constraints [Baid 94]:

Core Term: Pred-46. **has_requirement(I,Io)**

I: ID for an organizational constraint, goal, or policy
Io: ID an organizational constraint, goal, or policy that must be satisfied in order for I to be satisfied.

3.6 Means of Demonstrating Competency: Using the ISO 9000 Quality Advisor

A demonstration of competency for an ontology or micro-theory entails determining:

- Does it support problem-solving tasks for a specific enterprise?
- Are its competency questions—which, by design, do not refer to any specific enterprise—answered, thus demonstrating that the ontology or micro-theory can be used to support similar problem-solving tasks for other enterprises?

The first question cannot be answered positively unless the second is; that is, ontology-based quality analyses for BHP Steel and deHavilland Manufacturing are possible only because the competency questions for the Ontologies for Quality Modelling and ISO 9000 Micro-Theory can be answered. The first question characterizes the requirements for an enterprise analyst who is unfamiliar with first-order logic and the exact terms and axioms of an ontology or micro-theory; this might be a BHP Steel employee who performs ISO 9000 compliance analysis. The second question characterizes the requirements for the builder of an ontology or micro-theory who may be unfamiliar with the details of a specific organization's processes but has a good understanding of how organizations of the type that is modelled operate.

The ISO 9000 Quality Advisor offers a different view for each of these two types of advisor users for the demonstration of competency: one for the enterprise analyst, and another for the ontology builder. Each step in the demonstration can be delineated into the two views shown below. The ontology or micro-theory representations used by the ontology builder must be translated into queries, terms, and rules understandable by the enterprise analyst. This translation is likely performed by a technical person maintaining the advisor or by the ontology builder.

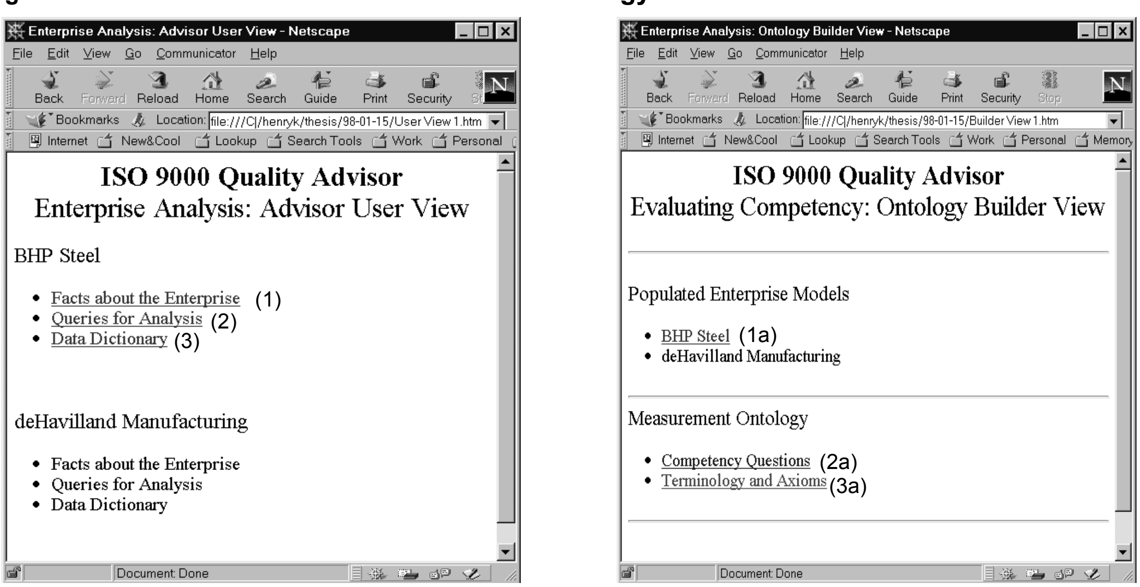
Table 3.1 Steps for Using the ISO 9000 Quality Advisor for Demonstrating Competency

step #	**Enterprise Analyst View:** Using the advisor to analyze a specific enterprise		**Ontology Builder View:** Using the advisor to evaluate competency of ontology or micro-theory
1	Stating facts about an enterprise	⇔	Representing populated enterprise model
2	Stating queries for analyzing enterprise	⇔	Representing formal competency questions
3	Stating data dictionary of enterprise's terms	⇔	Representing ontology or micro-theory terminology and axioms
4	Answering queries	⇔	Deducing answers to formal competency questions
5	Explaining the derivation of answers	⇔	Tracing of deduction and displaying Prolog trace list

⇔: denotes translation between knowledge about a specific enterprise and how that knowledge is represented in an ontology or micro-theory

The initial screens of the advisor, customized for BHP Steel and deHavilland Manufacturing, correspond to these two views.

Figure 3.9 Screens for Advisor User and Ontology Builder Views

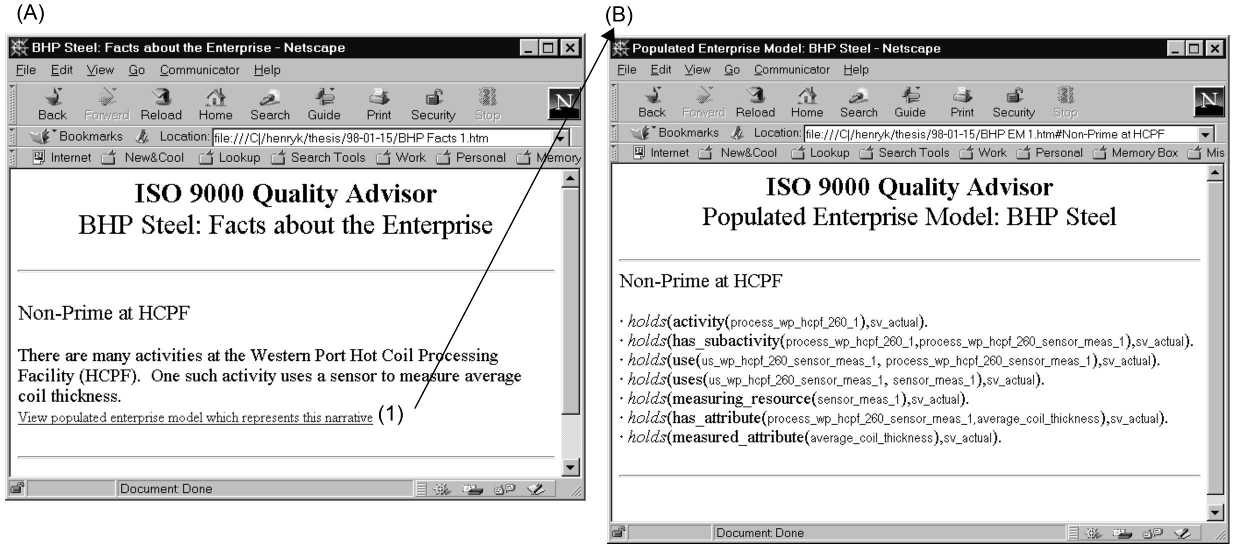


Step 1: Stating Facts about an Enterprise ⇔ Representing Populated Enterprise Models

In stating facts about an enterprise, narratives describing an enterprise, expressed using vocabulary common to the it, are presented. In representing the populated enterprise model, facts about the enterprise are represented using the terminology of the ontologies or micro-theory.

Figure 3.10 **Displaying Facts & Representing them in an Ontology or Micro-Theory**

(A) (B)

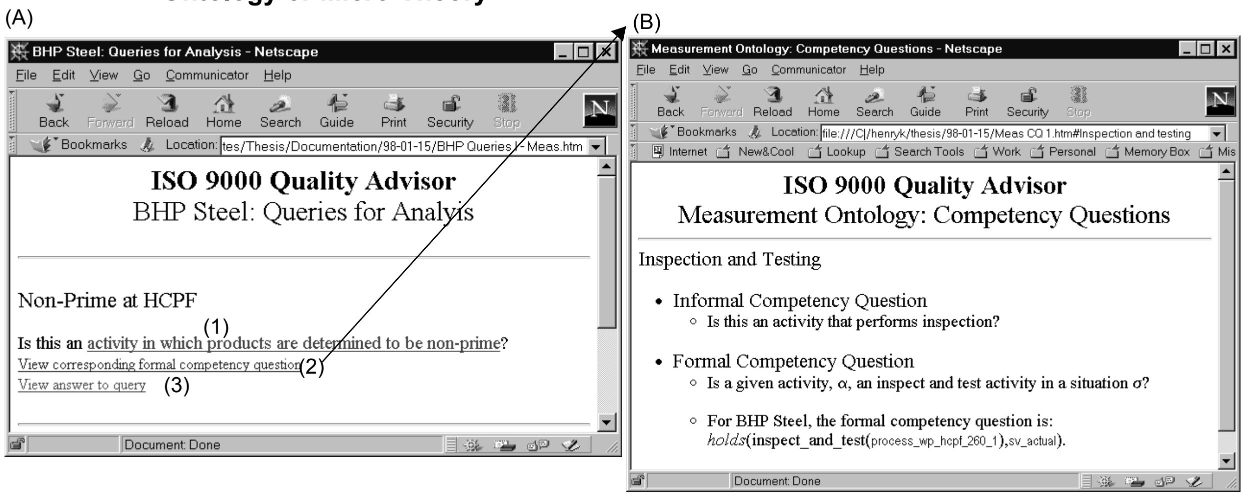


Clicking on the link, labelled (1) in Figure 3.9 Screens for Advisor User and Ontology Builder Views takes the advisor user to the screen above, labelled (A). Clicking on the link, labelled (1) in the diagram above, or the link labelled (1a) in Figure 3.9 Screens for Advisor User and Ontology Builder Views, takes the advisor user to the screen above, labelled (B).

Step 2: Stating Queries for Analyzing Enterprise ⇔ Representing Formal Competency Questions

In stating queries for analyzing an enterprise, questions that support analysis are asked, expressed using the vocabulary common to the enterprise. In stating formal competency questions, the analysis queries are represented using the terms of the ontology or micro-theory.

Figure 3.11 **Displaying Queries and Representing Them as Formal Competency Questions of the Ontology or Micro-Theory**

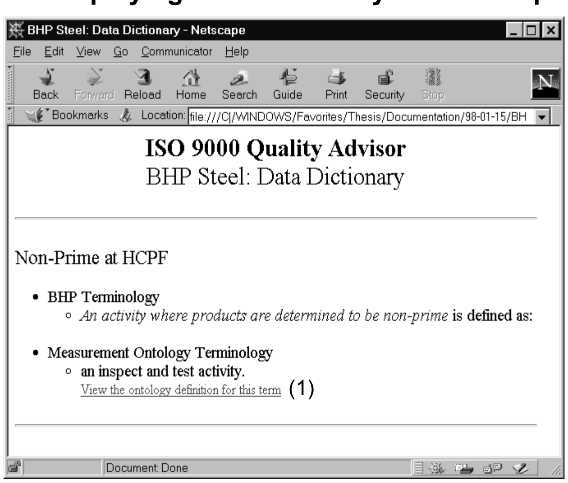


Clicking on the link labelled (2) in Figure 3.9 Screens for Advisor User and Ontology Builder Views takes the advisor user to the screen above, labelled (A). Clicking on the link, labelled (2) in the diagram above, or the link labelled (2a) in Figure 3.9 Screens for Advisor User and Ontology Builder Views takes the advisor user to the screen above, labelled (B).

Step 3: Stating Data Dictionary of Enterprise's Terms ⇔ Representing Ontology or Micro-Theory Terminology and Axioms

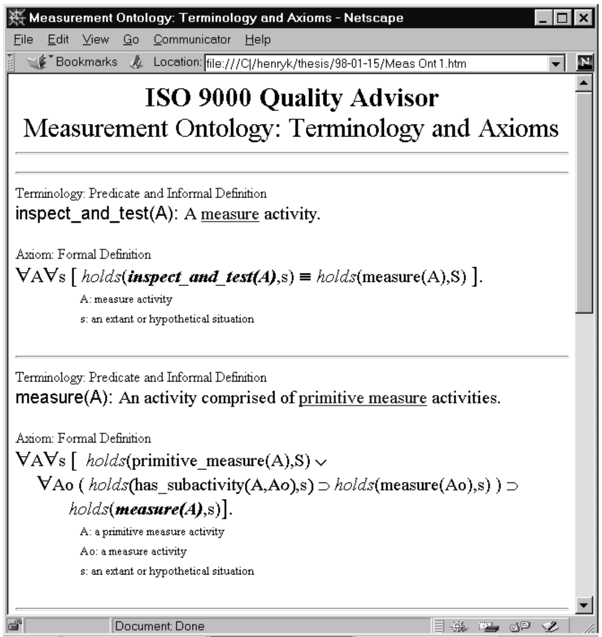
In stating data dictionary of enterprise's terms, the terms of an enterprise's vocabulary are defined using the terminology of an ontology or micro-theory. In representing ontology or micro-theory terminology and axioms, terms are informally stated, then formally defined in first-order logic.

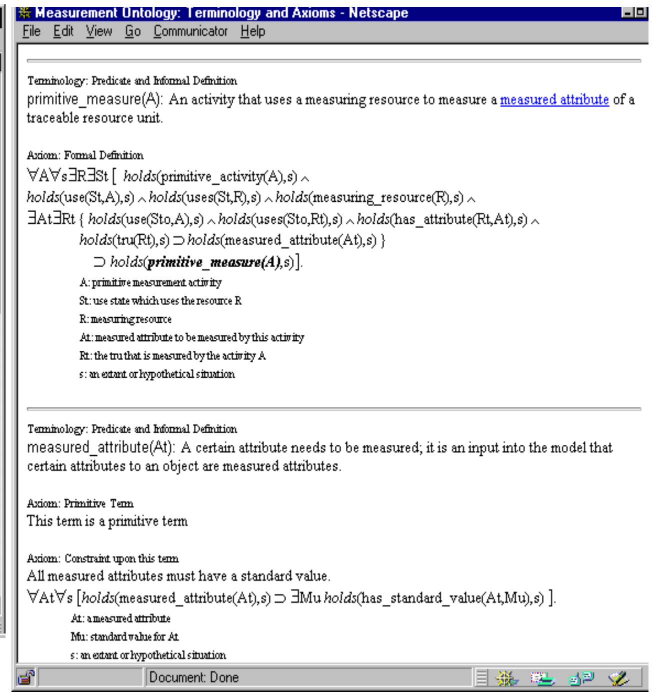
Figure 3.12 **Displaying Data Dictionary of an Enterprise's Terms**



Clicking on the link labelled (3) in Figure 3.9 Screens for Advisor User and Ontology Builder Views takes the advisor user to this screen.

Figure 3.13 Displaying Ontology or Micro-Theory Terminology and Axioms

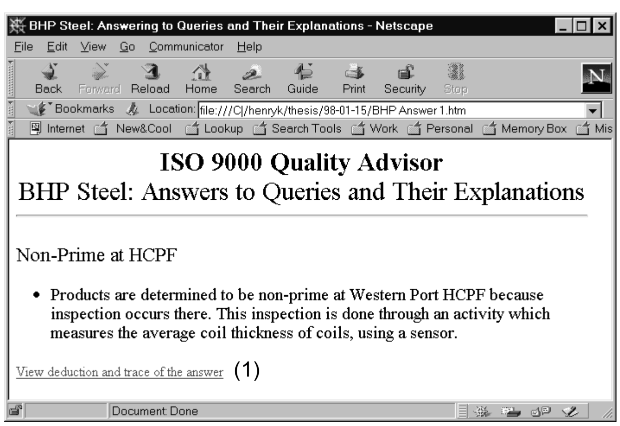




Clicking on the link, labelled (1) in Figure 3.12 Displaying Data Dictionary of an Enterprise's Terms or labelled (3a) in Figure 3.9 Screens for Advisor User and Ontology Builder Views, takes the advisor user to a screen like the one above.

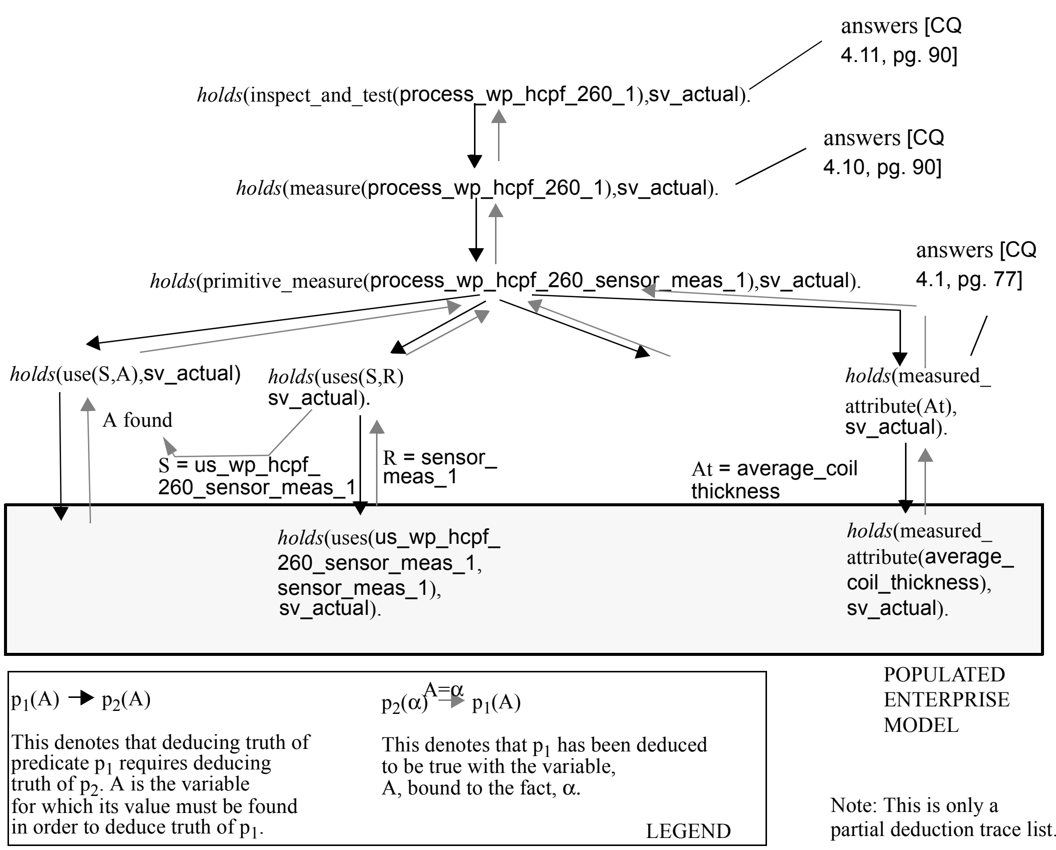
Steps 4 and 5: Answering Queries ⇔ Deducing Answers to Formal Competency Questions; and Explaining the Derivation of Answers ⇔ Tracing Deduction and Displaying Prolog Trace List

Figure 3.14 Displaying Answers to Queries, and Explanations for Answers for BHP Steel



Clicking on the link, labelled (3) in Figure 3.11 Displaying Queries and Representing Them as Formal Competency Questions of the Ontology or Micro-Theory, takes the advisor user to this screen.

Figure 3.15 Displaying Competency Question Deduction for Ontology or Micro-Theory[9]



9. Clicking on (1) in Figure 3.14 Displaying Answers to Queries, and Explanations for Answers for BHP Steel takes the advisor user to this screen.

In deducing answers to formal competency questions, deductions in the Prolog programming environment are presented. In tracing deduction and displaying a Prolog trace list, the steps in the deduction are shown either as a tree-graph of how variables in predicates are bound or as a trace of the deduction execution steps in Prolog.

3.7 Means of Demonstrating Re-useability: Reducibility as a Proof of Generality, and Other Measures

Re-useability is the capability to use portions of a model to solve different problems beyond the problems that initially motivated the development of the model.

3.7.1 Empirical Qualitative and Quantitative Measures

Whereas construction of complex systems such as buildings and cars entails piecing together many off-the-shelf components, this kind of re-use is not done often enough in constructing software systems [Budd 91]. As well, according to Pressman ([92], pg. 769):

- "We have been processing data using computers for over 40 years and extracting information for over two decades. One of the most significant challenges facing the software engineering community is to build systems that take the next step along the spectrum— systems that extract knowledge from data and information in a way that is practical and beneficial."

Putting these two imperatives together, it can be asked: How can knowledge-based systems be constructed by the re-use of components, and how can such re-use be evaluated?

For constructing re-useable knowledge bases, Gennari et. al. [94] propose creating separate domain and method ontologies, as well as domain-method dependent mapping ontologies which allow for the sufficient de-coupling of domain and method knowledge. Their proposal dovetails with the decision to delineate between the Ontologies for Quality Modelling (domain ontologies) and the ISO 9000 Micro-Theory (ontology for solving the ISO 9000 evaluation problem); a more re-useable set of representations can be constructed because of this decision.

Evaluation of effective re-useability is also an issue. For example, Toshiba uses the following ratio: percentage of re-used lines of code to actual lines that are included into a program with little or no modification [Cusamano 91]. An extension of this measure is in terms of re-use of entire software objects rather than re-use of lines of codes [Banker et. al. 93]. This evaluation based upon quantitative empirical evidence is one of several methods for knowledge-based systems evaluation [Guida & Mauri 93]. Other methods are evaluation methods based upon qualitative empirical evidence, ground in knowledge-based systems lifecycle, original evaluation criteria, imported evaluation criteria, and automatic knowledge-base checking methods.

3.7.2 Reducibility

Another model evaluation method is *reducibility* [Gruninger 96]. This evaluation is premised upon the following: If a set of competency questions from a foreign or "target" ontology or micro-theory can be reasonably translated (reduced) to a set of questions answerable using the representations of a "native" ontology or micro-theory, then the competency of the "native" ontology or micro-theory is a superset of the competency of the "target" ontology. The representations of the "native" ontology or micro-theory are then proven to be general enough to be used to solve similar problems for similar enterprises as the "target" ontology. Reducibility is a method to evaluate the re-useability of an ontology's representations, because it is a means to demonstrate that an ontology is *general* enough to answer competency questions for different applications that support solving different problems; and *general* representations can be re-used to construct different applications that solve different problems.

The reduction of competency questions can be expressed as the following meta-theoretic problem [Gruninger 96]:

$$T'_{ontology} \cup T'_{ground} \models Q \Rightarrow T_{ontology} \cup T_{ground} \cup T_{def} \models Q$$

- $T_{ontology}$ and $T'_{ontology}$ denote the TOVE ontology and target ontology representations, respectively.

- T_{ground} and T'_{ground} denote ground terms, represented using the primitive terms of the TOVE ontology and target ontology representations, respectively.
- T_{def} is the set of axioms which translate $T'_{ontology} \cup T'_{ground}$ into the same language as $T_{ontology} \cup T_{ground}$
- Q denotes a first-order sentence which is entailed by $T'_{ontology} \cup T'_{ground}$. Q is also entailed by $T_{ontology} \cup T_{ground} \cup T_{def}$.

Then, the procedure to follow for reduction of competency questions is as follows.

1. Determine the target ontology.
   - $T'_{ontology}$.
2. State the target ontology's competency question in the language of the target ontology.
   - $Q \in L(T'_{ontology})$.
3. Ensure that the competency question is answerable using target ontology representations.
   - $T'_{ontology} \cup T'_{ground} \models Q$.
4. Specify a set of reduction axioms such that the target ontology's competency questions can be posed in the language of the native (TOVE) ontology.
   - $Q \cup T_{def} \in L(T_{ontology})$.
5. Answer the competency question in the language of the native (TOVE) ontology.
   - $T_{ontology} \cup T_{ground} \cup T_{def} \models Q$.

Using this procedure, it is possible to reduce competency questions for an existing software package to competency questions expressible using TOVE Ontologies; that is, TOVE Ontologies *span* some of the competency of the software package. Reducibility demonstrates that existing representations of the TOVE Ontologies can be re-used to construct applications that perform some of the same tasks as the software package.

3.8 Summary and Conclusion

The following summarize the key facets of the methodology used to construct the Ontologies for Quality Modellng and ISO 9000 Micro-Theory:

- The Ontological Engineering Methodology is a structured design methodology for systematically and rigourously engineering ontologies. In the methodology, motivating scenarios of industrial partners are analyzed to explicate informal competency questions for an ontology or micro-theory. These questions are analyzed to explicate terminology, and assumptions that bound the scope, of an ontology or micro-theory. From these, formal competency questions expressed using the terminology are stated. Then, axioms that define and constrain the interpretation of the terminology are engineered. By reasoning using these axioms, the formal competency questions are answered; this is the demonstration of competency of the ontology or micro-theory. Then, the terminology and axioms of the ontology or micro-theory are shown to be re-useable by a demonstration of reducibility. Augmentation or creation of new motivating scenarios then initiates another iteration of the Ontological Engineering Methodology to augment the ontology or micro-theory, or to engineer a new one.

- Motivating scenarios for BHP Steel and deHavilland Manufacturing are stated to develop the Ontologies for Quality Modelling. For one, a motivating scenario depicts a scenario for which an application constructed using an ontology or micro-theory is to be used; it motivates the development of an ontology or micro-theory. Second, it can be used to validate the ontology: Does the application constructed using the ontology or micro-theory address the issues described in the motivating scenario? In this thesis, the ontologies are developed from the motivating scenarios, but not validated against them.

- The TOVE Core Ontologies—Activity, State, Causality, Time, Resource, and Organization Ontologies—are the building blocks for the Ontologies for Quality Modelling; terms and axioms of the Ontologies for Quality Modelling are composed from representations of the Core Ontologies. By using Core Ontologies' representations, additional enterprise ontologies like the Ontologies for Quality Modelling do not have to be built from "first principles," thus reducing engineering time and standardizing ontology development efforts.

- A useful tool for demonstrating the competency of the Ontologies for Quality Modelling and the ISO 9000 Micro-Theory is the ISO 9000 Quality Advisor. The advisor can be used to support both an enterprise analyst using the advisor to perform an ISO 9000 compliance audit, and an ontology builder using the advisor to evaluate the competency of a quality-related ontology or micro-theory.

- A useful means of demonstrating re-useability of ontology or micro-theory representations is reducibility: A procedure to demonstrate that a native ontology is general enough to answer competency questions for a target ontology. This demonstrates then that the natiove ontology representations can be re-used to build similar applications at those for which the target ontology exists.

The Ontological Engineering Methodology is critical for developing models that support the thesis of this dissertation [see pg. 7], because it describes steps and guidelines to follow to systematically and rigourously engineer descriptive and prescriptive models of quality.

4. Measurement Ontology

4.1 Précis

The following is the premise in developing the Measurement Ontology: the customer specifies needs, and the supplier provides a product to satisfy those needs; and the evaluation of quality can be de-composed into evaluating the series of measurements upon features and characteristics of the product. The Measurement Ontology represents a systematic way of describing how a particular feature or characteristic of an entity, called a *measured attribute*, is to be measured. Measured attributes are given a value through *measure activities*, where one value for a *measured attribute* of an entity at a given point in time is called a *measurement point*. Part of the description for a *measured attribute* is the determination of the quality of each *measurement point* as either a *conformance* or *nonconformance point*. By comparing a collection of these points versus *quality requirements* that are translated from the needs of the customers, an entity (a product, process, or the enterprise itself) can be assessed as being *of conforming quality.* The Measurement Ontology, then, is an implementation of a popular view of quality—quality is conformance to requirements—and a fundamental systems engineering view—that satisfaction to requirements can be decomposed to assessments on a series of measurements. In this chapter, one iteration of the ontological engineering methodology applied to develop the Measurement Ontology is presented.

4.2 Introduction

The following are the premises in developing the Measurement Ontology: The customer specifies needs and the supplier provides a product to satisfy those needs; and the evaluation of quality can be de-composed into evaluating the series of measurements upon features and characteristics of the product. Hence, measurement marries the customer perception of quality with supplier capability; before quality can be improved, assured, and controlled, quality must be measured. It is no wonder that those who first studied quality, such as Shewhart and Dodge & Romig, were also statisticians [Banks 89]. Those concerned with measuring, collecting, and analyzing data could see how the data could be used to improve quality of products and processes. Fittingly, then, the first of the Ontologies for Quality Modelling is the Measurement Ontology.

4.3 Motivating Scenario

The first step in the development of the Measurement Ontology is the analysis of the motivating scenario—specifically, to determine the key concepts for modelling measurement from excerpts of the BHP Steel scenario.

The following excerpt describes BHP Steel's losses with respect to cost, time, and revenue, when products of unacceptable quality (called non-prime products by BHP Steel) are produced. The key concept for the development of the Measurement Ontology is the following: *there must be a systematic way of describing how a particular physical characteristic is to be measured and this description must be used to meet the customer expectations of quality.*

> As raw materials are transformed by the different production units of BHP Steel's supply chain, non-prime products may also be produced. These are the products whose physical properties do not satisfy the necessary tolerance specifications; non-prime refers to sub-standard products, not an intended by-product. Non-prime products lead to lost revenue due to regrading and scrapping, increased costs due to additional rework, carrying of excess inventory to meet delivery promises, and increased variability of leadtime performance. Most importantly, non-prime leads to disaffected, even lost, customers.

The next excerpt describes BHP Steel's need to understand and improve its inspection system, the collection of activities that assesses whether a product is non-prime. The key concept here is that *quality assessment is made through a system of activities that perform measurement; this is a view of measurement as an activity.*

> If the products are consistently found to be non-prime, this is an indication that there is something faulty in the production unit. A cause for this occurrence is suspected to be an inadequate inspection system.

The following excerpt specifies what is entailed in determining a product as non-prime. The key concept here is that *every quality assessment is a decision that begins with a value of measurement at a given point in time.*

> Especially when the product is shipped to the customer, it is essential that the product be within the tolerance specifications of the customer. So, the product's physical characteristics are measured, compared against their tolerance specifications, and a decision about whether the product is non-prime is made.

4.4 Measurement Ontology: Measurement Description System

The concept that *there must be a systematic way of describing how a particular physical characteristic is to be measured and that this description must be used to assess quality* is the basis for the following principles about representing measurement:

- In order to measure, there must be a way to systematically describe a measurement. This description system must minimally include the appropriate attributes of an entity to measure, as well as that attribute's mean (μ), distribution (σ), and comparison operator ($\otimes$) for comparing a measured value against μ and σ.
- It must be explicitly represented that measurements are made in order to satisfy some requirements. With respect to quality, measurements are made in order to satisfy quality requirements, where these requirements capture customers' quality expectations.

The competency questions that characterize what needs to be represented so that an ontology is constructed upon these principles are presented next.

4.4.1 Informal Competency Questions

Of course, only some physical characteristics of an entity need be measured, so it should be asked:

CQ 4.1 What are the physical characteristics that are measured?

Why are these characteristics measured? Is there a requirement that needs to be satisfied that necessitates the measurement?

CQ 4.2 Is this a quality requirement?

Also, beyond representing what is measured, what about how to measure it?

CQ 4.3 Is every product that is produced measured?

CQ 4.4 If the product is a batch, is a sample[1] taken from that batch and measured?

CQ 4.5 If a sample is taken and measured, is the value for the measurement some aggregate (e.g., average) of the measurement upon individual units of that sample?

CQ 4.6 Or, is the value of the measurement a measure of whether or not individual units of the sample passed or failed a certain threshold (e.g., % of widgets of the sample which are <10cm)?

Hopefully, the value of the measurement for a physical characteristic falls within certain tolerance specifications. So, it should be asked:

CQ 4.7 What ought to be the measured value; that is, what is the expected value for that physical characteristic?

CQ 4.8 What is the tolerance specification for a physical characteristic that is measured?

Measurements are ambiguous without their relevant units of measurements. So, it should be asked:

CQ 4.9 What is the unit of measurement for a physical characteristic of an entity?

In order to answer these competency questions, the domain of measurement is analyzed, assumptions are stated, and terminology and axioms are developed.

4.4.2 Analysis, Assumptions, Terminology, and Axioms

What is Measurement?

Measurement is defined as the process by which numbers or symbols are assigned to attributes of entities in the real world in such a way as to describe them according to clearly defined rules

1. A sample is defined as “a subset of the population containing those measurements actually obtained by experiment.” ([Scheaffer & McClave 82], pg. 2)

[Finkelstein 84]. In representing measurement, there must be a modelling of the relationship between the entity and its attributes into the more tractable domain of numbers and operators; for this thesis, the relationship is that the attributes bear on the quality of the entity. Assessing entity quality—i.e., "does an entity meet its requirements upon quality?"—necessitates this modelling. According to Grady [93], all requirements upon an entity must be decomposed into a series of equations, A⊗B, where A and B denote qualitative or quantitative measurements upon attributes, and ⊗ denotes a comparison operator. So, before discussing measurement further, requirements must be discussed.

Developing terminology and axioms to formally state and answer:

CQ 4.2 Is this a quality requirement?

How should quality requirements be represented? An expression can be characterized as a quality requirement based upon its content, but how? The content will always refer to domain-dependent knowledge like, for example, "Is an inspector at the activity?" or "Are 95% of the measurements within tolerance specifications?" It is very difficult to state a categorical characterization of all quality requirements; there do not seem to exist axioms about quality requirements that would be true for all enterprises in all industries. Rather than define a quality requirement based upon its content, *it is assumed that a quality requirement is an input to the enterprise model; that is, it must be given as a fact of an enterprise that a given expression is a quality requirement*.

Meas Axiom: PT-1. **quality_requirement(Qr)**

If an *organizational constraint* is a constraint that has a bearing on the quality of an entity, then this constraint is a *quality requirement*; this is a primitive term.

- Qr is just the unique ID for the quality requirement

Moreover, in order to tie a quality requirement to customers' expectations, it is assumed that *customer quality needs are translated into quality requirements before the requirements are input into the enterprise model.* This makes sense since it would be very difficult to represent in an ontology the creative translation of vague needs into concrete requirements in an ontology.

Although specific quality requirements do not belong in an ontology, they can be classified as belonging to a less-generic domain than generic quality. They can be used to examine an enterprise as per a specific quality perspective—for example, a set of requirements that use statistical techniques or prescribe what is acceptable enterprise quality. A micro-theory is comprised of formal expressions of related quality requirements. For example, an SQC micro-theory axiom that expresses a quality requirement may specify that an activity is "in-control" if 95% of measurements of a certain attribute are within $[\mu-3\sigma,\mu+3\sigma]$, with mean μ and variance σ^2. Another example is an axiom that states that written procedures must control all inspection activities; this axiom formalizes one of the ISO 9000 requirements and is part of the ISO 9000 Micro-theory. So, *a quality-related micro-theory provides the axiom that expresses a quality requirement; the Ontologies for Quality Modelling provide the terminology with which that axiom can be composed. By combining the micro-theory and the ontologies, an assessment of an entity's quality can be made.* This assessment can be formalized as:

Meas Axiom: Defn-1. **conforming_quality(X,Qr)**

An entity is *of conforming quality* if an axiom of a micro-theory (called an *agent constraint*) is satisfied and that constraint is a *quality requirement*.

$$\forall X \forall Qr \forall s \left[holds(\textbf{conforming\_quality(X,Qr)},s) \equiv holds(\text{agent\_constraint}(X,Qr),s) \wedge holds(\text{quality\_requirement}(Qr),s) \right].$$

X: an entity
Qr: a quality requirement
s: an extant or hypothetical situation

- Recall that an agent constraint can be expressed in this way: *holds*(agent_constraint(X,Qr),s) ≡ Φ(X,s), where Φ(X,s) is an expression composed of ontology terms. If there are sufficient terms in the ontology to express and deduce the truth of Φ(X,s), then using the definition of the term *conforming quality*, it can be deduced that an entity X is of conforming quality with respect to the quality requirement, identified as Qr.

Now, measurement can be discussed as the means by which an entity is determined to be *of conforming quality.*

Developing terminology and axioms to formally state and answer:

CQ 4.1 What are the physical characteristics that are measured?

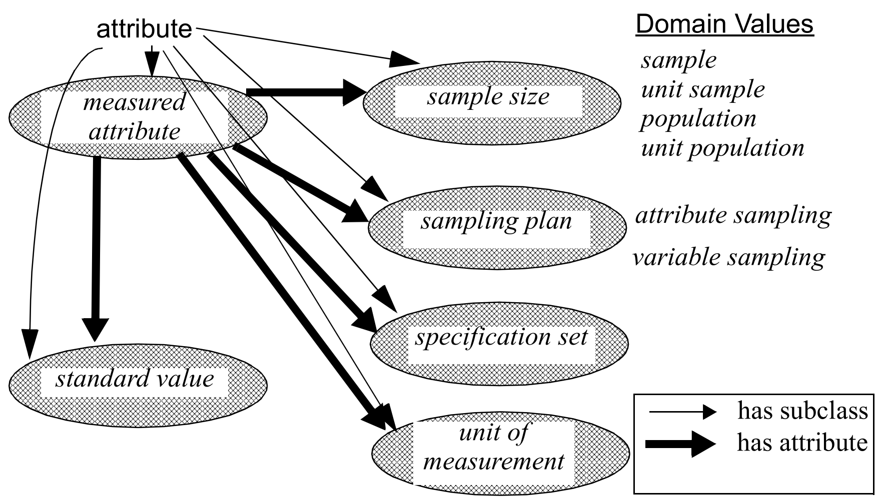
In statistical process control (SPC), the quality of an activity is gauged by measuring its output, where an output is a batch, lot, or an individual unit. According to the Traceability Ontology, all such outputs are homogeneous sets of individual resources, called *traceable resource units* (*trus*).

The Measurement Ontology assumes a similar approach to SPC: *The quality of an activity is evaluated by the quality of trus outputted by that activity; and the quality of a resource is gauged by the quality of trus that it comprises*. *So, attributes of trus are measured*. The expression of what constitutes quality for a *tru*, *activity* or a *resource* is the axiom that corresponds to a *quality requirement*. If *trus* are to be measured, what attributes (features or characteristics) of a given *tru* are measured? These can be modelled as attributes that acquire a value through measurement—e.g., the average coil thickness of a sample of steel coils. Such attributes are called *measured attributes*.

Meas Axiom: PT-2. **measured_attribute(At)**

A certain attribute needs to be measured; it is an input into the model (a primitive term) that certain attributes of a *tru* are *measured attributes*.

Figure 4.1 **Data Model for Measured Attribute**



The objects and relations in this data model are necessary to answer competency questions. Next, the rationale for this data model will be discussed, and the terminology in the data model will be formally defined and constrained.

Developing terminology and axioms to formally state and answer:

CQ 4.3 Is every product that is produced measured?

CQ 4.4 If the product is a batch, is a sample taken from that batch and measured?

When taking measurements, a *tru* is sampled. In sampling, an attribute of an individual unit of a *tru* is measured from a subset (a sample) of the *tru*. There will be as many measurements as there are units in the sample. The result of measuring the sample is an aggregate value. For example, coil thicknesses of different coils in a sample are measured, and the average of these measurements becomes the value for the "average coil thickness" *measured attribute* of the *tru* of coils. A measured attribute of a tru is "sampled" from an attribute of the resource that comprises it.

Meas Axiom: PT-3. **samples_attribute(Atr,At)**

It is an input into the enterprise model that there exists a *samples attribute* relation between the attribute of a *resource* (denoted by the variable Atr, e.g., arm length of an arm assembly) and the *measured attribute* of the *tru* that is comprised of individual units of the *resource* (denoted by the variable At, e.g. average arm length of sample). Hence, *samples attribute* is a primitive term.

There are two additional issues regarding sampling.

- **sample size**: How many individuals in a set are measured in order to model the behaviour of the set?
- **sampling plan**:When determining an aggregate value from the sample, is this aggregate value expressed directly (e.g., average arm length is $\overline{X}$ such that $\overline{X} \in [\alpha,\beta]$, and any individual arm length is X such that $X \in [\alpha,\beta]$) or expressed indirectly (e.g., # nonconforming of a sample, sized N, is Y such that $Y \in [1,2,,N]$, where $Y = \Sigma$ (occurrences of individual arms where arm length is X such that $X \notin [\gamma,\xi]$), and $[\gamma,\xi] \subseteq (\alpha,\beta)$)? In the lexicon of statistics, the former sampling strategy is called variable sampling, and the latter is called attribute sampling.

Any given sample size can be characterized as one of:

- *unit sample*: This is a domain value[2] for sample size, where set size>1 and sample size=1.
- *sample*: This is a domain value for the sample size, where set size >1 and sample size>1.
- *unit population*: This is a domain value for the sample size, where set size=sample size=1
- *population*: This is domain value for the sample size, where set size=sample size>1.

The following primitive term represents the relationship between a *measured attribute* and its *sample sizing plan*.

Meas Axiom: PT-4. **has_sample_sizing(At,Sz)**

At: a measured attribute
Sz: the name for the sample sizing plan

The following axiom constrains the *sample sizing* of a *measured attribute*.

Meas Axiom: Cons-2. All *measured attributes* must have a sample sizing plan, where the only allowable domain values for this attribute are *unit sample*, *sample*, *unit population*, and *population*.

$$\forall s \forall At \Big[\mathit{holds}(\text{measured\_attribute}(At),s) \supset \exists Sz \; \{ \mathit{holds}(\text{has\_sample\_sizing}(At,Sz),s) \wedge (Sz = \text{unit\_sample} \vee Sz = \text{sample} \vee Sz = \text{population} \vee Sz = \text{unit\_population}) \} \Big].$$

At: a measured attribute
Sz: the name for the sample sizing plan
s: an extant or hypothetical situation

Developing terminology and axioms to formally state and answer:

CQ 4.5 If a sample is taken and measured, is the value for the measurement some aggregate (e.g., average) of the measurement upon individual units of that sample?

CQ 4.6 Or, is the value of the measurement a measure of whether or not individual units of the sample passed or failed a certain threshold (e.g., % of widgets of the sample which are <10cm)?

The interpretation of a measurement value also depends on the sampling plan.

- *variable sampling*: This is a domain value for the sampling plan where an attribute of an individual unit is measured and a value aggregating the measurements of all the units of a sample—typically this aggregate is the average—is attained.

2. A domain value is an allowable value for an attribute of an object.

- *attribute_sampling*: This is a domain value for the sampling plan where a measurement upon an attribute of a unit is performed, the acceptability of that unit as per some criteria upon the measurement value is assessed, and the number or percentage of the acceptable units of the sample is counted.

The following primitive term represents the relationship between a *measured attribute* and its *sampling plan*.

Meas Axiom: PT-5. **has_sampling_plan(At,Sp)**

At: a measured attribute
Sp: the name for the sampling plan

The following axiom constrains the *sampling plan* of a *measured attribute*.

Meas Axiom: Cons-3. **All *measured attributes* must have a sampling plan, where the only allowable domain values for this attribute are *variable sampling* and *attribute sampling*.**

$$\forall s \forall At \left[\textit{holds}(\text{measured\_attribute}(At),s) \supset \exists Sp \; \{ \textit{holds}(\text{has\_sampling\_plan}(At,Sp),s) \wedge (Sp = \text{variable\_sampling} \vee Sp = \text{attribute\_sampling}) \} \right].$$

At: a measured attribute
Sp: the name for the sampling plan
s: an extant or hypothetical situation

Developing terminology and axioms to formally state and answer:

CQ 4.7 What ought to be the measured value; that is, what is the expected value for that physical characteristic?

There is inevitable variation in measurement values. *There must be an assessment system for each measured attribute such that its measurement value can be ascertained for its indication of entity quality, using this system.* A measurement value Mp for a *measured attribute* At for a given entity X at a given point in time Tp is called a *measurement point*.

Meas Term: Pred-1. **measurement_pt(X,At,Mp,Tp)**

- This term is formally defined later in the chapter.

For an assessment system, there must be a value for what the measurement point ought to be. The following primitive term represents the *standard value* relationship between a *measured attribute* and its *standard value*, a mean value (μ).

Meas Axiom: PT-6. **has_standard_value(At,Mu)**

A *measured attribute has an attribute called standard value*, where this is what the value of the attribute ought to be.

Meas Axiom: Cons-4. **measured attribute must have a standard value**

All *measured attributes* must *have a standard value*.

$$\forall At \forall s \left[holds(\text{measured\_attribute}(At),s) \supset \exists Mu \; holds(\text{has\_standard\_value}(At,Mu),s) \right].$$

At: a measured attribute
Mu: standard value for At
s: an extant or hypothetical situation

Developing terminology and axioms to formally state and answer:

CQ 4.8 What is the tolerance specification for a physical characteristic that is measured?

There must be an indication of the expected distribution of the measurement points for a given measured attribute; this expected distribution is σ^2. *There must also be a function of* μ, and σ^2 *(*$f(\mu,\sigma^2)$*) and an operator (*$\otimes$*) that compares this function versus the actual measurement point.* Finally, *there must be an assessment value; this is a measure of what can be assessed about the quality of the entity, given the measurement point of the given measured attribute.* Though mathematically tractable, it is difficult to have representations for μ,σ^2, $f(\mu,\sigma^2)$, $\otimes$, and assessment value that are generic for all types of domains and ranges of functions and encompass different operators. For example, what about an assessment system for numerical measurement points distributed normally, with the assessment being assigned a value of "conforming" if the measurement point $\in$ $(\mu-3\sigma,\ \mu+3\sigma)$, versus supporting an assessment system for qualitative measurement points for which assessment value is labelled "acceptable" if the measurement point is an element in a set of acceptable values?

The above examples show that in most cases the assessment value is a binary variable. *So it is assumed that the assessment can only be one of either conforming or nonconforming quality.* The question then is: how can μ, $f(\mu,\sigma^2)$, and $\otimes$ be represented when the measurement points are not numeric, so that it is possible to assess whether a given measurement point is "of conforming quality"? *Although* σ^2 *cannot be expressed qualitatively,* $f(\mu,\sigma^2)$ *can*, *if it is assumed that* $f(\mu,\sigma^2)$ *is a subset of the range of all possible values for measurement points*.

It is also assumed that:

- $f(\mu,\sigma^2)$ *is an ordered set, where* $V_1<V_2<...V_n$ *for the n possible elements of the set.*
- *For measured attributes, with measurement points that are continuous variables,* $f(\mu,\sigma^2)$ *is a set of infinite elements.*
- *For the other types of measured attributes,* $f(\mu,\sigma^2)$ *is a finite set.*
- *If the measurement points are quantitative, then the ordering of the* $f(\mu,\sigma^2)$ *set, regardless of whether it is a finite or infinite set, follows arithmetic rules for the operator <.*
- *If the measurement points are qualitative, then the ordering relationships between the elements of the finite* $f(\mu,\sigma^2)$ *set is determined by the ordering of the elements upon the creation of the* $f(\mu,\sigma^2)$ *set.*

Then, given these assumptions, it is possible to define the "acceptable" limits of the $f(\mu,\sigma^2)$ set. If the measurement values are quantitative, then such a set need only specify the lower limit (V1) and upper limit (V_n). Therefore, any element in this set belongs in the interval $[V_1,V_n]$. If the measurement values are qualitative, then this set needs to specify, in order, V_1, V_2, to V_n. This set is the *specification set*, and it is important for the following reason: If *it is assumed that the standard comparison operator (*$\otimes$*) is the "element of" (*$\in$*) operator, then a conformance point* is assigned if the measurement point is a member of the *specification set*; otherwise a *nonconformance point* is assigned.

The following primitive term represents the relationship between a *measured attribute* and its *specification set*.

Meas Axiom: PT-7. **has_specification_set(At,SL)**

A *measured attribute has an attribute called specification set*, where this is a set of values which denote conforming quality for that *measured attribute*.

- SL is modelled as a multi-valued set of the specification set attribute to the object At.

Meas Axiom: Cons-5. **measured_attributes must have a valid specification set**

All *measured attributes* must *have a specification set*, and the *standard value* for that *measured attribute* must be an element of the *specification set*.

$$\forall At \forall s \left[holds(\text{measured\_attribute}(At),s) \supset \exists SL \exists Mu \left(holds(\text{has\_specification\_set}(At,SL),s) \wedge Mu \in SL \right) \right].$$

At: a measured attribute
SL: the specification set for At
Mu: the standard value for SL
s: an extant or hypothetical situation

The assignment of a *measurement point* as a *conformance* or *nonconformance point* then constitutes the assignment of an assessment value for a *measured attribute* for a point in time.

Meas Term: Pred-2. **conformance_pt(Q,Rt,At,Tp)**

Meas Term: Pred-3. **nonconformance_pt(Q,Rt,At,Tp)**

Q: a unique identifier for the conformance or nonconformance point
Rt: a tru for which there exists a measurement point
At: a measured attribute of Rt
Tp: time point for which Mp is the measurement point for Rt

- These terms are formally defined later in the chapter

Developing terminology and axioms to formally state and answer:

CQ 4.9 What is the unit of measurement for a physical characteristic of an entity?

An assessment system for a *measured attribute* must necessarily include the *unit of measurement*:

Meas Axiom: Defn-2. **has_unit_of_measurement(At,U)**

If an activity measures a *tru*, then the *unit of measurement* for this *activity*-*tru* pair is the *unit of measurement for the measured attribute*.

- *measures tru* is a term defined later; *unit of measurement* and *has attribute* are terms from the Core Ontologies

$$\forall At \forall U \forall s \exists Rt \left[holds(\text{tru}(Rt),s) \wedge holds(\text{has\_attribute}(Rt,At),s) \wedge holds(\text{measured\_attribute}(At),s) \wedge \exists A \exists Un\, holds(\text{unit\_of\_measurement}(Rt,Un,U,A),s) \supset holds(\textbf{has\_unit\_of\_measurement(At,U)},s) \right].$$

Meas Axiom: Cons-6. **measured attribute must have a unit of measurement**

All *measured attributes* must *have a unit of measurement*.

$$\forall At \forall s \left[holds(\text{measured\_attribute}(At),s) \supset \exists U\, holds(\text{has\_unit\_of\_measurement}(At,U),s) \right].$$

A: an activity
Rt: a tru measured by A
At: a measured attribute of both A and Rt
Un: describer for unit of measurement (default is capacity)
U: unit of measurement (default is object, but can be e.g. cm)
s: an extant or hypothetical situation

In the next section, the competency questions are formally posed using the ontology terminology and answered by deduction using the ontology axioms.

4.4.3 Formal Competency Questions

Each competency question is presented in the following manner[3]:

- The informal competency question is stated.
- The informal competency question is re-stated in English with the terminology developed from the ontology; then, the competency question is stated formally in first-order logic. This is a generic question that does not refer to specific facts.
- An example of asking the generic question with facts from an enterprise model is presented. For example, questions about BHP Steel's enterprise model are asked.

CQ 4.1 What are the physical characteristics that are measured?

- Does there exist a *measured attribute* for a *tru* κ in a situation σ? ∃At [*holds*(tru(κ),σ) ∧ *holds*(has_attribute(κ,At),σ) ∧ *holds*(measured_attribute(At),σ)].
- ∃At [*holds*(tru(tru_raw_coil_1),sv_actual) ∧ *holds*(has_attribute(tru_raw_coil_1,At),sv_actual) ∧ *holds*(measured_attribute(At),sv_actual)].

CQ 4.2 Is this a quality requirement?

- Does there exist a *quality requirement* θρ in a situation σ? *holds*(quality_requirement(θρ),σ).
- *holds*(quality_requirement(iso_9000),sv_actual).

CQ 4.3 Is every product that is produced measured?

- For a *measured attribute* α in a given situation σ, does it have a *population sample sizing plan*? *holds*(measured_attribute(α),σ) ∧ *holds*(has_sample_sizing(α,*population*),σ).
- *holds*(measured_attribute(average_coil_thickness),sv_actual) ∧ *holds*(has_sample_sizing(average_coil_thickness,population),sv_actual).

CQ 4.4 If the product is a batch, is a sample taken from that batch and measured?

3. This format holds for all formal competency question sections throughout this thesis.

- For a *measured attribute* α of a *tru* κ in a given situation σ, does it have a *unit sample* or "*sample" sample sizing plan*? *holds*(tru(κ),σ) ∧ *holds*(has_attribute(κ,α),σ) ∧ *holds*(measured_attribute(α),σ) ∧ (*holds*(has_sample_sizing(α,*unit_sample*),σ) ∨ *holds*(has_sample_sizing(α,*sample*),σ)).
- *holds*(tru(tru_raw_coil_1),sv_actual) ∧ *holds*(has_attribute(tru_raw_coil_1,average_coil_thickness),sv_actual) ∧ *holds*(measured_attribute(average_coil_thickness),sv_actual) ∧ (*holds*(has_sample_sizing(average_coil_thickness,unit_sample),sv_actual) ∨ *holds*(has_sample_sizing(average_coil_thickness,sample),sv_actual)).

CQ 4.5 If a sample is taken and measured, is the value for the measurement some aggregate (e.g., average) of the measurement upon individual units of that sample?

- For a *measured attribute* α in a given situation σ, does it have a *variable sampling plan*? *holds*(measured_attribute(α),σ) ∧ *holds*(has_sampling_plan(α,*variable_sampling*),σ).
- *holds*(measured_attribute(average_coil_thickness),sv_actual) ∧ *holds*(has_sampling_plan(average_coil_thickness,variable_sampling),sv_actual).

CQ 4.6 Or, is the value of the measurement a measure of whether or not individual units of the sample passed or failed a certain threshold (e.g., % of widgets of the sample which are <10cm)?

- For a *measured attribute* α in a given situation σ, does it have an *attribute sampling plan*? *holds*(measured_attribute(α),σ) ∧ *holds*(has_sampling_plan(α,*attribute_sampling*),σ).
- *holds*(measured_attribute(average_coil_thickness),sv_actual) ∧ *holds*(has_sampling_plan(average_coil_thickness,attribute_sampling),sv_actual).

CQ 4.7 What ought to be the measured value; that is, what is the expected value for that physical characteristic?

- For a *measured attribute* α in a given situation σ, what is its *standard value*? ∃Mu [*holds*(measured_attribute(α) ,σ) ∧ *holds*(has_standard_value(α,Mu),σ)].
- ∃Mu [*holds*(measured_attribute(average_coil_thickness),sv_actual) ∧ *holds*(has_standard_value(average_coil_thickness,Mu),sv_actual)].

CQ 4.8 What is the tolerance specification for a physical characteristic that is measured?

- For a *measured attribute* α in a given situation σ, what is its *specification set*? $\exists T_1 \exists T_2 \exists \{W_i\}$ [*holds*(measured_attribute(α) ,σ) ∧ (*holds*(has_specification_set(α,$[T_1,T_2]$),σ) ∨ *holds*(has_specification_set(α,$\{W_i\}$),σ))].
- $\exists T_1 \exists T_2 \exists \{W_i\}$ [*holds*(measured_attribute(average_coil_thickness),sv_actual) ∧ (*holds*(has_specification_set(average_coil_thickness,$[T_1,T_2]$),sv_actual) ∨ *holds*(has_specification_set(average_coil_thickness, $\{W_i\}$),sv_actual))].

CQ 4.9 What is the unit of measurement for a physical characteristic of an entity?

- For a *measured attribute* α in a given situation σ, what is its *unit of measurement*? ∃U [*holds*(measured_attribute(α) ,σ) ∧ *holds*(has_unit_of_measurement(α,U),σ)].
- ∃U [*holds*(measured_attribute(average_coil_thickness),sv_actual) ∧ *holds*(has_unit_of_measurement(average_coil_thickness,U),sv_actual)].

4.5 Measurement Ontology: Measurement Activities

The concept that *quality assessment is made through a system of activities that perform measurement* is the basis for the following principle for representing measurement:

- A model that represents measurement should start with a model of an activity that measures one characteristic of one entity, as an elemental measurement activity. With this, it is possible to define any measurement activity, as composed of elemental measurement activities.

The competency questions that characterize what needs to be represented so that an ontology is constructed upon this principle are presented next.

4.5.1 Informal Competency Questions

Using the representations of the Core Ontologies, a system of measurement is represented as comprising of activities that measure. So it should be asked,

CQ 4.10 Is this an activity that performs measurement?

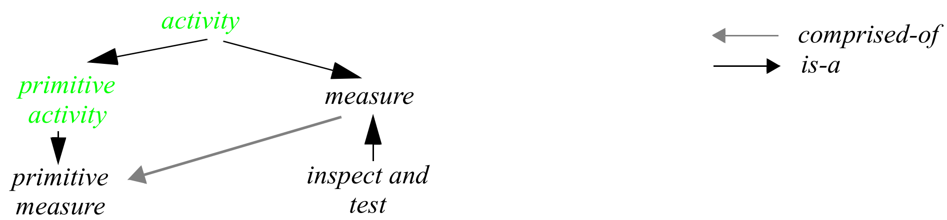
The motivating scenario specifically describes inspection, so it should be asked:

CQ 4.11 Is this an inspection activity?

In order to answer these competency questions, the domain of measurement is analyzed, assumptions are stated, and terminology and axioms are developed.

4.5.2 Analysis, Assumptions, Terminology, and Axioms

Figure 4.2 **Data Model of Measurement Activities**



The objects and relations in this data model are necessary to answer competency questions. Next, the rationale for this data model will be discussed, and the terminology in the data model will be formally defined and constrained.

Developing terminology and axioms to formally state and answer:

CQ 4.10 Is this an activity that performs measurement?

The simplest measurement action is the measurement of one measured attribute of one *tru* at one point in time. Repeated measurements of that same attribute for that same type of *tru* over a period of time constitutes an elemental form of a measurement activity, called a *primitive measure* activity. This activity is a special form of a *primitive activity*, uses a *measuring resource*, and measures the *measured attribute* of a *tru*.

Trace Term-Pred 2. **primitive_activity(A)**

A *primitive activity* is an *activity* without any *subactivities*.

Meas Axiom: PT-8. **measuring_resource(R)**

A given *resource* is *used* for performing measurement; this is a *measuring resource*, a primitive term.

Meas Axiom: Defn-3. **primitive_measure(A)**

$$\forall A \forall s \exists R \exists St \left[\; holds(\text{primitive\_activity}(A),s) \wedge \right.$$
$$holds(\text{use}(St,A),s) \wedge holds(\text{uses}(St,R),s) \wedge holds(\text{measuring\_resource}(R),s) \wedge$$
$$\exists At \exists Sto \exists Rt \; \{ \; holds(\text{use}(Sto,A),s) \wedge holds(\text{uses}(Sto,Rt),s) \wedge$$
$$holds(\text{has\_attribute}(Rt,At),s) \wedge holds(\text{tru}(Rt),s) \supset$$

holds(measured_attribute(At),s) }⊃
holds(***primitive_measure(A)***,s)}].

- *use* and *uses* are term from the Core Ontologies

A: primitive measurement activity
St: use state which uses the resource R
R: measuring resource
At: measured attribute to be measured by this activity
Rt: the tru that is measured by the activity A
s: an extant or hypothetical situation

The following constraints ensure the simplicity of the primitive measurement activity[4].

- A *primitive measure* activity cannot measure two different *measured attributes*.
- Different *measured attributes* of a *tru* must be measured by different *primitive measure* activities.

Then, all measure activities can be composed from *primitive measure* activities.

Meas Term: Pred-4. **measure(A)**

A *measure* activity is a *primitive measure activity* or an *aggregation* of *measure* activities.

Developing terminology and axioms to formally state and answer:

CQ 4.11 Is this an inspection activity?

Since an implicit assumption of this ontology is that measurement and assessment of the measurement (inspection and testing) occur simultaneously as part of a *measure* activity, the following is stated.

Meas Term: Pred-5. **inspect_and_test(A)**

An *inspect and test* activity is a *measure* activity.

In the next section, the competency questions are formally posed using the ontology terminology and answered by deduction using the ontology axioms.

4. Throughout the thesis, for the sake of brevity, some definitions and constraints are stated in English, and not stated formally in first-order logic.

4.5.3 Formal Competency Questions

CQ 4.10 Is this an activity that performs measurement?

- Is α a *measure* activity in a situation σ? *holds*(measure(α),σ) .
- *holds*(measure(process_wp_hcpf_260_1),sv_actual) .

CQ 4.11 Is this an inspection activity?

- Is α an *inspect and test* activity in a situation σ? *holds*(inspect_and_test(α),σ) .
- *holds*(inspect_and_test(process_wp_hcpf_260_1),sv_actual) .

4.6 Measurement Ontology: Measurement Points

The concept that *every quality assessment is a decision made with a value of measurement at a given point in time* is the basis for the following principles about representing measurement:

- The elemental piece of information needed in order to make a quality assessment decision must be represented as the value of a measurement taken at a given point in time.
- The overall quality assessment must be represented as some composition of the quality assessments of each measurement value.

The competency questions that characterize what needs to be represented so that an ontology is constructed upon these principles are presented next.

4.6.1 Informal Competency Questions

CQ 4.12 What is the measured value for a measured attribute at a given point in time?

CQ 4.13 What is the measured value for a measured attribute for a given period of time?

CQ 4.14 Is an entity of conforming quality at a given point in time?

CQ 4.15 Is an entity of conforming quality over a given period of time?

What if the product is not conforming?

CQ 4.16 Is an entity of nonconforming quality at a given point in time?

In order to answer these competency questions, the domain of measurement is analyzed, assumptions are stated, and terminology and axioms are developed.

4.6.2 Analysis, Assumptions, Terminology, and Axioms

Developing terminology and axioms to formally state and answer:

CQ 4.12 What is the measured value for a measured attribute at a given point in time?

CQ 4.13 What is the measured value for a measured attribute for a given period of time?

Measurement points of *trus* can now be formally defined, since the terminology and axioms required for the definition have been explicated.

Meas Axiom: Defn-4. **measurement_pt(Rt,At,Mp,Tp)**

The *measurement point* is the value for the *measured attribute* of a *tru that is measured by a primitive measure activity* at a *time point* included in the *time period* for that *primitive measure activity.*

- *use res tru* is defined in the Traceability Ontology section; *activity duration* and *has point* are terms from the Core Ontologies.

$$\forall Rt \forall At \forall Mp \forall Tp \forall s \exists T [\ \mathit{holds}(\text{measured\_attribute}(At),s) \wedge \mathit{holds}(\text{has\_attribute}(Rt,At),s) \wedge \mathit{holds}(\text{has\_attribute\_value}(Rt,At,Mp),s) \wedge \exists A\ (\ \mathit{holds}(\text{tru}(Rt),s) \wedge \mathit{holds}(\text{use\_res\_tru}(A,Rt),s) \wedge \mathit{holds}(\text{primitive\_measure}(A),s) \wedge \mathit{occurs}_T(\text{activity\_duration}(A,T))\ ) \wedge \text{has\_point}(T,Tp)\ \} \supset \mathit{holds}(\boldsymbol{\mathit{measurement\_pt(Rt,At,Mp,Tp)}},s)\ ].$$

Rt: a tru for which there exists a measurement point
At: a measured attribute of Rt
Tp: time point for which Mp is the measurement point for Rt
Mp: value of that measurement point
T: time period in which the measurement takes place
A: the primitive measurement activity that ascertained the measurement point
s: an extant or hypothetical situation

Developing terminology and axioms to formally state and answer:

CQ 4.14 Is an entity of conforming quality at a given point in time?

CQ 4.15 Is an entity of conforming quality over a given period of time?

CQ 4.16 Is an entity of nonconforming quality at a given point in time?

Now, *conformance* and *nonconformance points* of *trus* can be defined in terms of *measurement points of trus*.

Meas Axiom: Defn-5. **conformance_pt(Q,Rt,At,Tp)**

The *conformance point* is a *measurement point* which is an element of the *specification set*.

$$\forall Rt \forall At \forall Tp \forall s \Big[\exists Mp \exists SL \; (\mathit{holds}(\mathrm{measurement\_pt}(Rt,At,Mp,Tp),s) \wedge \mathit{holds}(\mathrm{tru}(Rt),s) \wedge \mathit{holds}(\mathrm{has\_specification\_set}(At,SL),s) \wedge Mp \in SL) \supset \exists Q \; \mathit{holds}(\boldsymbol{conformance\_pt(Q,Rt,At,Tp)},s) \Big].$$

Meas Axiom: Defn-6. **nonconformance_pt(Q,Rt,At,Tp)**

The *nonconformance point* is a *measurement point* which is not an element of the *specification set*.

$$\forall Rt \forall At \forall Tp \forall s \Big[\exists Mp \exists SL \; (\mathit{holds}(\mathrm{measurement\_pt}(Rt,At,Mp,Tp),s) \wedge \mathit{holds}(\mathrm{tru}(Rt),s) \wedge \mathit{holds}(\mathrm{has\_specification\_set}(At,SL),s) \wedge Mp \notin SL) \supset \exists Q \; \mathit{holds}(\boldsymbol{nonconformance\_pt(Q,Rt,At,Tp)},s) \Big].$$

Q: a unique identifier for the conformance or nonconformance point
Rt: a tru for which there exists a measurement point
At: a measured attribute of Rt
Tp: time point for which Mp is the measurement point for Rt
Mp: value of that measurement point
SL: specification set for At
Tp: time point for which Mp is the measurement point for Rt
s: an extant or hypothetical situation

Now, *conformance points* of other entities like activities or even organizational agents can be defined strictly in terms of *conformance points* of *trus*. For example, *conformance points* of an activity can be defined in terms of *conformance points* of *trus* produced by the activity, and *conformance points* of organization agents can be defined in terms of c*onformance points* of *trus* produced by activities performed to fulfill the agent's *role*.

Recall the discussion on quality requirements. With conformance points formalized, a typical quality requirement can be expressed as a composition of conformance points. So the expression for evaluating the quality requirement, identified as Qr, could look like:

$$\mathit{holds}(\mathrm{conforming\_quality}(X,Qr),s) \equiv \bigcap_i \mathit{holds}(\mathrm{conformance\_pt}(Q_i,X,At_i,Tp_i),s_i)) \wedge \mathit{holds}(\mathrm{quality\_requirement}(Qr),s).$$

So, the Measurement Ontology representations formalize the following major notion of this thesis:

- Quality is conformance to requirements
- Requirements can be decomposed to a finite set of measurements, and quality can be assessed based upon the results of the measurements.

In the next section, the competency questions are formally posed using the ontology terminology and answered by deduction using the ontology axioms.

4.6.3 Formal Competency Questions

CQ 4.12 What is the measured value for a measured attribute at a given point in time?

- What is the *measurement point* for a *measured attribute* κ of an entity ξ at a given point in time τ for a given situation σ?: $\exists$Mp *holds*(measurement_pt(ξ,κ,Mp,τ),σ).
- $\exists$Mp *holds*(measurement_pt(tru_wp_raw_coil_1,average_coil_thickness,Mp,10),sv_actual).

CQ 4.13 What is the measured value for a measured attribute for a given period of time?

- What are the *measurement points* for a *measured attribute* κ of an entity ξ for a time duration $[\tau_1,\tau_2]$ for a given situation σ? [$\forall$Tp$\forall$Mp (*holds*(measurement_pt(ξ,κ,Mp,Tp),s) $\wedge$ Tp$\geq\tau_1$ $\wedge$ Tp$\leq\tau_2$ $\supset$ *f*(Tp,Mp))]. Where *f*(Tp,Mp) is just a graphing function that plots Tp vs. Mp.
- [$\forall$Tp$\forall$Mp (*holds*(measurement_pt(tru_wp_raw_coil_1,average_coil_thickness,Mp,Tp),sv_actual) $\wedge$ Tp$\geq$0 $\wedge$ Tp$\leq$30 $\supset$ *f*(Tp,Mp))].

CQ 4.14 Is an entity of conforming quality at a given point in time?

- Is there a *conformance point* for a *measured attribute* κ of an entity ξ at a given point in time τ for a given situation σ?: $\exists$Q *holds*(conformance_pt(Q,ξ,κ,τ),σ).
- $\exists$Q *holds*(conformance_pt(Q,tru_wp_raw_coil_1,average_coil_thickness,10),sv_actual).

CQ 4.15 Is an entity of conforming quality over a given period of time?

- For all time points within a given time duration $[\tau_1,\tau_2]$ for a given situation σ, does there always exist a *conformance point* for different *trus* comprised of *resource* ξ? $\forall$Rt$\forall$Tp [Tp$\geq\tau_1$ $\wedge$ Tp$\leq\tau_2$ $\wedge$ $\tau_1<\tau_2$ $\wedge$ *holds*(has_tru(ξ,Rt),s) $\supset\exists$Q *holds*(conformance_pt(Q,Rt,κ,Tp),σ)].
- $\forall$Rt$\forall$Tp [Tp$\geq$0 $\wedge$ Tp$\leq$30$\wedge$ *holds*(has_tru(raw_coil_1,Rt),s) $\supset$ $\exists$Q *holds*(conformance_pt(Q,Rt,average_coil_thickness,Tp),sv_actual)].

CQ 4.16 Is an entity of nonconforming quality at a given point in time?

- Is there a *nonconformance point* for a *measured attribute* κ of an entity ξ at a given point in time τ for a given situation σ?: $\exists$Q *holds*(nonconformance_pt(Q,ξ,κ,τ),σ).
- $\exists$Q *holds*(nonconformance_pt(Q,tru_wp_raw_coil_1,average_coil_thickness,10),sv_actual).

4.7 Demonstration of Competency: Using the ISO 9000 Quality Advisor

This demonstration shows how the advisor is used for analyzing the inspection system at BHP Steel. In so doing, the advisor is used to answer the following competency question:

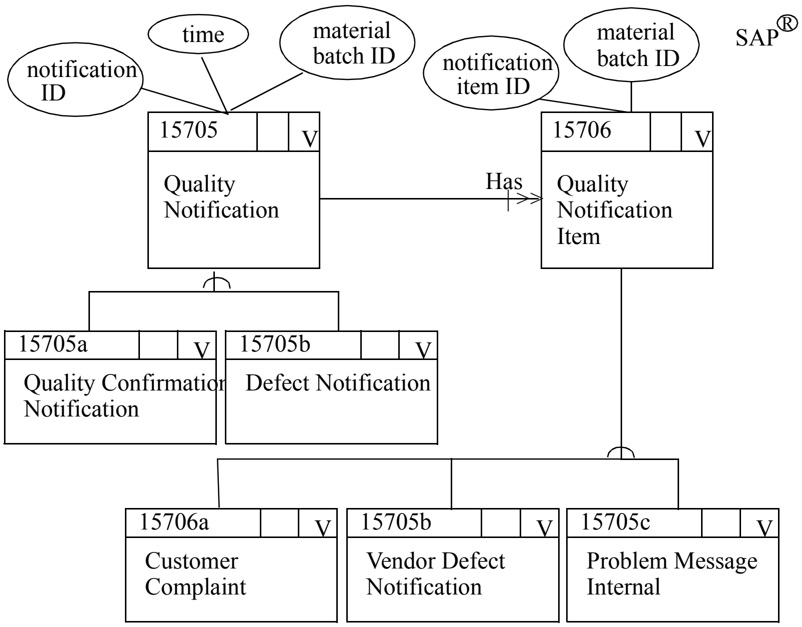
- **CQ 4.11 Is this an inspection activity?** In order to answer this question, the following question is also answered.
  - **CQ 4.10 Is this an activity that performs measurement?** In order to answer this question, the following question is also answered.
    - **CQ 4.1 What are the physical characteristics that are measured?**

This demonstration was already shown in details in Section 3.6 of Chapter 3. Methodology [pg. 64].

4.8 Demonstration of Reducibility: Reducing SAP R/3™ Measurement Competency

As part of its offering, SAP R/3™ offers distributed business objects—a description of complete business processes—and a set of Business Application Programmatic Interfaces (BAPI's)—standardized interfaces that enables external applications to access business objects—for its customers to customize the SAP software [SAP 98a]. One such object that exemplifies the functionality of SAP R/3™'s quality module is described by the following diagram [SAP 97].

Figure 4.3 **SAP R/3™ Quality Notification Business Object**



Documentation:

The business object Quality notification contains the quality notification, that is the documentation of the fulfillment of the quality requirement for a business object in the form of a quality confirmation notification or of the non-fulfillment of the quality requirement for a business object in the form of a quality defect notification. A quality defect notification can take the form of a customer complaint, a vendor defect notification, or a problem message - internal.

The above data model and accompanying description constitute a partial view of SAP R/3™'s quality ontology—call this $T'_{ontology}$—where the partial view is modelled to answer quality notification competency questions. The predicates and axioms of the partial data model are the following:

customer_complaint(Y,Rt)
vendor_defect_notification(Y,Rt)
problem_message_internal(Y,Rt)
quality_notification_item(Y,Rt)
quality_confirmation_notification(X,Tp,Rt)
defect_notification(X,Tp,Rt)
quality_notification(X,Tp,Rt)
has_quality_notification_item(X,Y)

∀Y∀Rt { customer_complaint(Y,Rt) ∨
vendor_defect_notification(Y,Rt) ∨ problem_message_internal(Y,Rt) ⊃
quality_notification_item(Y,Rt) }

∀X∀Rt∀Tp { quality_confirmation_notification(X,Tp,Rt) ∨

$$\text{defect\_notification}(X,Tp,Rt) \supset \text{quality\_notification}(X,Tp,Rt) \}.$$

$$\exists X \exists Y \exists Tp \exists Rt \{ \text{quality\_notification\_item}(Y,Rt) \wedge \text{quality\_notification}(X,Tp,Rt) \supset \text{has\_quality\_notification\_item}(X,Y) \}.$$

X: notification ID
Y: notification item ID
Rt: material batch ID
Tp: time

Informally, a target ontology competency question can be: "What is one internal quality problem at time t_{given}?" Formally, using the terminology of the SAP R/3™ data model, the question can be posed in English as: "What is one defect notification that is notified from an internal problem message at time t_{given}?" More formally, the competency question can be expressed in first-order logic as:

$$\text{(SAP CQ) } \exists X \exists Rt \exists Y \{\text{problem\_message\_internal}(Y,Rt) \wedge \text{defect\_notification}(X,t_{given},Rt)\}.$$

The following are reduction assumptions.

$$\neg \exists Y \neg \exists Rt \text{ customer\_complaint}(Y,Rt).$$
$$\neg \exists Y \neg \exists Rt \text{ vendor\_defect\_notification}(Y,Rt).$$

So, it is assumed that there are no customer complaints and defect notifications from vendors. That is, quality problems are only assumed to be recognized solely as a result of activities performed within the enterprise without intervention from outside enterprises.

The following are reduction axioms.

$$\forall X \forall Rt \forall Tp \exists Y \{\text{defect\_notification}(X,Tp,Rt) \wedge \text{quality\_notification\_item}(Y,Rt) \supset \text{ncp\_tove}(X,Tp,Rt) \}.$$

$$\exists X \exists Rt \exists Tp \{ \text{ncp\_tove}(X,Tp,Rt) \supset \exists At \exists s \; \mathit{holds}(\text{nonconformance\_pt}(X,Rt,At,Tp),s) \}.$$

The axioms state that if a notification of the quality of an item denotes that the item is defective, then that notification is represented as a nonconformance point, where this term is from the Measurement Ontology.

By applying the reduction assumptions and axioms to the predicates and axioms of the SAP R/3™ quality ontology, the competency question (SAP CQ) can be entirely expressed using terms from the Ontologies for Quality Modelling:

$$\forall s \exists X \exists Rt \exists At\ [\ holds(\text{nonconformance\_pt}(X,Rt,At,t_{given}),s) \wedge holds(\text{quality\_evidence}(X),s) \wedge holds(\text{tru}(Rt),s)\ ].$$

where nonconformance_point(X,Rt,At,Tp) denotes the nonconformance point uniquely identified as X of the tru Rt with respect to a measured attribute At at time point Tp.

The previous steps demonstrate how a question answerable using the SAP R/3™ quality module ("SAP R/3™ quality ontology")—*"What is an internal quality problem at time* t_{given}*?"*—is reduced to a question composed of Ontologies for Quality Modelling terminology—*"What is a nonconformance point at time* t_{given}*?"*—and answered using the axioms of the ontologies.

InspLotCharacteristic, InsptLotOperation, InspPoint, and InspectLot are other SAP R/3™ business objects which exist to perform tasks of the quality module [SAP 98b]. It can also be demonstrated that the competency questions for which these objects exist are reducible to questions answerable using Measurement Ontology representations. This demonstrates that the Ontologies for Quality Modelling are re-useable for solving similar problems for similar enterprises as the SAP R/3™ quality module.

4.9 Summary and Conclusion

The following summarize the generic concepts that are formalized in the Measurement Ontology:

- A system for assessing measurements should include the appropriate *attribute of an entity to measure*, as well as the mean (μ), distribution (σ), and comparison operator (⊗) for that attribute. Measurement of attributes should be recorded as *measurement points in time* that are measured as a result of some *measurement activity*. These representations are the basic ones necessary to model any form of measurement.

- Quality can be represented as some composition of *conformance points*, where these are "conforming" *measurement points* with respect to some *quality requirement*. Representing *quality requirements*, *measurement points*, and *conformance points* makes it possible to model and assess the quality of any entity within an enterprise.

These concepts are formalized by posing competency questions, analyzing the domain of measurement, stating assumptions, and developing terminology and axioms. Then, the competency of the ontology and the capability of the ontology to be used to gain insights about an enterprise are demonstrated by automatically deducing answers to competency questions such as:

- *Quality Assessment System Competency Questions*: What is the mean value for a given attribute of an entity? What are its tolerance specifications? How is that attribute *sampled*?
- *Measurement and Conformance Points Competency Questions*: What is the *measured value* for an attribute at a given point in time? Was it within the tolerance specifications? Over a period of time, do the measurements lie within the specs?

Finally, the demonstration of reducibility demonstrates that the Measurement Ontology spans a subset of the competency of SAP R/3™ quality module.

The design, analysis, and prototypical implementation of the Measurement Ontology support the thesis of this dissertation [pg. 7] by:

- Showing that quality description and prescription requires representing measurement.
- Describing a popular view of quality in an enterprise model—"quality is conformance to requirements"—and representing that quality conformance is determined through measurement.
- Representing measurement in the enterprise model, so that ISO 9000 compliance regarding an organization's measurement system can be objectively prescribed.

5. Traceability Ontology

5.1 Précis

When measurements highlight a quality problem, there must be means to diagnose the problem. A basic capability for diagnosis and analysis is traceability: the ability to trace the history, application, or location of an entity by means of recorded identification. In order to represent traceability capability in the Traceability Ontology, conditions for ensuring unique identification and traceability when material are consumed, produced, split and merged are represented. These representations ensure that it is possible to find material and quantity flows between two uniquely identified entities. A uniquely identified and traceable entity that represents material is called a *traceable resource unit* (*tru*). Batches, lots, and all such collections of homogeneous resources are *trus*. The quantity of a *tru* at a given point in time is called the *resource point of a tru*. Constraints upon a *tru* and its *resource point* characterize conditions under which traceability is possible. These are constraints upon creating, splitting and merging trus. In order to perform traceability using the ontology representations, axioms for composing material flows are stated, so that if there exists material flow between two entities, this flow can be found. One set of axioms composes *trace* paths of indirect material flow (e.g., material flow over many activities) from *tru* and *activity trace* paths of direct material flow (e.g., material flow between two activities that are back-to-back in a sequence of activities). Another set of axioms states boundary conditions where tracing beyond certain activities is not possible; activities through which trus enter into, or leave, an enterprise constitute such boundary conditions. So, using the traceability representations, it is possible to, first, ensure conditions for traceability, and then, actually perform traces. In this chapter, one iteration of the ontological engineering methodology applied to develop the Traceability Ontology is presented.

5.2 Introduction

According to ([ISO 94b], pg. 45), traceability is defined as “the ability to trace the history, application, or location of an entity, by means of recorded identification.” Traceability is inexorably linked with unique identification: “Not only are unique identification and traceability necessary to ensure that the series of activities required for a final product operate upon the appropriate resources...” ([Clements 93], pg. 153). For quality management, unique identification and traceability capability is crucial. Without this capability, it would not be possible to trace back to the activities that produced a defective product and identify the cause of the defect. Moreover, since the motivating scenarios require that traceability capability be modelled, and “product identification and traceability” is one of the requirements of the ISO 9001, representations required to demonstrate traceability capability within an enterprise are developed.

5.3 Motivating Scenarios

The first step in the development of the Traceability Ontology is the analysis of the motivating scenario, specifically, to determine the key concepts for modelling traceability from excerpts of the deHavilland Manufacturing scenario.

The following excerpt describes deHavilland Manufacturing’s need to be able to trace quantities of batches produced:

> For this diagnosis, an important piece of information is what quantity of a batch was produced and how much of it was used by a proceeding production unit. Knowing the quantity of the batches produced allows for the assessment of the extent to which problems from one process cascaded to subsequent processes.

If a batch of one type of a resource is consumed and a batch of another resource is produced, how is traceability ensured; that is, how is the ability to trace the history, application, or location of an entity, by means of recorded identification ensured? Traceability is ensured as long as the following key concept is adhered to: *If all or part of a set of a resource is consumed to produce another set of a different resource, then the two sets must be uniquely identified, and the "consume/produce" relationship between the consumed and produced sets must be recorded.*

What about ensuring traceability when a batch is neither consumed nor produced? That is, when it is: 1) idle, 2) used and subsequently released by the same activity, or 3) split or merged. For cases 1) and 2), neither its quantity nor its identity needs to change. When a batch is split and used for different activities, it must be possible to trace back from the split batches to the original batch, in order to ensure traceability. So, the following key concept must be adhered to in designing the Traceability Ontology: *If all or part of a set of a resource is split, it is not necessary to uniquely identify the split sets, just to ensure traceability; if the split sets are uniquely identified, then it must be recorded that these entities are split from an original set*.

Conversely, a composite batch may result from merging different batches. In this situation, a potential traceability problem arises. Take a random unit from the composite batch. From which of the merged batches does this random unit come? The act of merging results in only limited traceability being possible: A unit in a given composite batch must come from one of a finite set of known batches, but it is not possible to know exactly which one. In order to guarantee this limited traceability, the following concept must be adhered to in designing the Traceability Ontology: *When merging all or parts of two or more uniquely identified sets of a resource to a composite set, the composite set must also be uniquely identified, and it must be recorded that the composite set is merged from the merged sets.*

Another key concept motivated from the scenario is the following: *The quantity of a set of a resource that this consumed, produced, split, or merged must be recorded*.

The next scenario describes how traceability is used to enumerate the possible causes of a quality problem.

When inspection of a unit of Leading Edge points to a problem, it must be investigated where the defect occurred; the problem could have occurred anywhere between the inspection of the final product all the way back to the raw materials.

The scenario motivates the following key concept: *Rules for composing material flows must be stated, so that if there exist material flow between two sets of resources, this flow can be found*.

The motivating scenarios refer to an enterprise for which consumption and production occurs discretely. When it is difficult or impossible to determine one set of a resource from another set of the same resource—e.g., it is nearly impossible to model one litre of gas from another litre when gas is continuously consumed—the concept of tracing history of sets of resources is difficult to apply. *As a result, one of the key assumptions of the Traceability Ontology is that its representations cannot be used to model truly continuous consumption and production*.

5.4 Traceability Ontology: Quantity Tracing

This is the first key concept about traceability: *If all or part of a set of a resource is consumed to produce another set of a different resource, then the two sets must be uniquely identified, and the "consume/produce" relationship between the consumed and produced sets must be recorded.* This concept is the basis for the following principles about representing traceability.

- Unique identification should be defined, and conditions for ensuring the unique identification of a set of entities should be represented.
- The relationship between a consumed set of entities and the set of entities that is produced as a result of the consumption should be represented.

This is the second key concept about traceability*: If all or part of a set of a resource is split, it is not necessary to uniquely identify the split sets, just to ensure traceability; if the split sets are uniquely identified, then it must be recorded that these entities are split from an original set.* This concept is the basis for the following principles about representing traceability.

- An activity that splits a set should be represented.
- A relationship between the original set and its split sets should be represented.
- Constraints upon splitting sets should be stated. If these constraints are not satisfied in a given populated model, then unique identification and traceability in the model when entities are split cannot be guaranteed.

This is the third key concept about traceability: *When merging all or parts of two or more uniquely identified sets of a resource to a composite set, the composite set must also be uniquely identified, and it must be recorded that the composite set is merged from the merged sets.* This concept is the basis for the following principles about representing traceability.

- An activity that merges different sets should be represented.
- A relationship between a composite set and the sets that were merged to form the composite entity should be represented.
- Constraints upon merging sets should be stated. If these constraints are not satisfied in a given populated model, then unique identification and limited traceability—where traceability is limited because it is disjunctive (i.e., "A comes from either B, C, or D")—in the model when entities are merged cannot be guaranteed.

This is the fourth key concept about traceability: *The quantity of a set of a resource that this consumed, produced, split, or merged must be recorded*. This concept is the basis for the following principles about representing traceability.

- Tracing the changes in a set's quantity should be represented.
- Conditions for ensuring the accuracy of quantity tracing should be represented.

The competency questions that characterize what needs to be represented so that an ontology is constructed upon these principles are presented next.

5.4.1 Informal Competency Questions

The purpose of traceability is to ensure the means of diagnosing nonconformities in the product. This requires minimally that it be possible to trace from the final product to its raw materials. It must also be possible to trace from the activities that produced the final product to the activities that brought the raw materials into the enterprise. Since raw materials, final products, and activities should be traced, a general question like the following should be asked:

CQ 5.1 Is this a uniquely identified entity that can be traced?

Also, to ensure unique identification and traceability, the following questions should be asked?

CQ 5.2 If a batch is consumed, what is the batch that is produced as a direct result of this consumption? Are these two batches uniquely identified?

CQ 5.3 If an original batch is split, what are the batches that the original batch is split into? Are the original batch and the split batches uniquely identified?

CQ 5.4 Is a given activity a splitting activity?

CQ 5.5 If batches are merged, what is the composite batch that is formed as a result of the merge? Are the merged batches and the composite batch uniquely identified?

CQ 5.6 Is a given activity a merging activity?

Some of the questions to ask to develop the representations for quantity tracing are the following:

CQ 5.7 At which time was a given batch recognized to exist?

CQ 5.8 What is the quantity of a given batch at a given point in time?

CQ 5.9 At what point in time are all the quantities of a given batch exhausted?

CQ 5.10 For a given batch, did the number of units in the batch change during a period of time?

In order to answer these competency questions, the domain of traceability is analyzed, assumptions are stated, and terminology and axioms are developed.

5.4.2 Analysis, Terminology, Assumptions, and Axioms

Developing terminology and axioms to formally state and answer:

CQ 5.1 Is this a uniquely identified entity that can be traced?

CQ 5.2 If a batch is consumed, what is the batch that is produced as a direct result of this consumption? Are these two batches uniquely identified?

It is assumed that unique identification cannot formally be defined. Rather, whether something is uniquely identifiable is a modelling decision. Say there are 3 balls labelled A_1, A_2, and A_3 in a box labelled B in a given scenario. There are two ways to model this, which impact the extent of unique identification in the model. One, A_1, A_2, A_3, and B may all be modelled as objects in the model; all objects then are uniquely identified. Two, B may be modelled as an object with an attribute named # of balls, with a value of 3; only B is uniquely identified, and the fact that the balls are uniquely identified in reality is not reflected in the model. Which of the two models is chosen depends upon its purpose. Similarly, below, the purpose of the Traceability Ontology is analyzed to discern which entities are to be uniquely identified and traced.

Core Ontologies support aggregated views of activities. In order to trace the series of activities performed to transform the raw materials to final products, it must be possible to represent the non-aggregated activities from which aggregated activities are constructed, since the trace must be performed on the most complete model of the activities of an enterprise. Hence, a *primitive activity* is defined as the most elemental activity representation from which the whole series of activities of the enterprise is modelled. A primitive activity is an entity that should be uniquely identified for traceability.

Trace Axiom: Defn-1. **primitive_activity(A)**

A *primitive activity* is an *activity* without any *subactivities*.

$$\forall A \forall s \left[\, holds(\mathbf{primitive\_activity(A)},s) \equiv holds(\text{activity}(A),s) \wedge \neg\exists A_n \, holds(\text{has\_subactivity}(A,A_n),s) \right].$$

A: a primitive activity
s: an extant or hypothetical situation

Next, there must be a term for representing raw materials, the work-in-process that is produced from the raw materials, and the final product that is ultimately produced. But what is exactly meant by these terms? Is a product, a prototypical unit—e.g., “the company makes widgets”—or is it a batch of that unit— e.g., “the company shipped batch #714 of widgets?” According to the

Core Ontologies, the prototypical unit is a *resource*. However, the collection of that *resource* is more relevant for traceability; traceability is possible when it can be stated "there is a problem with batch #714, so let's trace how it was produced," but not when it can only be stated that "there is a problem with widgets." So there needs to be a representation for describing a collection (set) of a prototypical unit. *Traceable resource unit* (*tru*) is a term representing a homogeneous set of a resource type. So, a *tru* is another entity that should be uniquely identified for traceability. Moreover, a *tru* should be an entity for which its quantity variance is traced. In order to formally define a *tru*, the following core ontology predicates are augmented to take a *tru* as an argument:

Core Term: Pred-47. **has_tru(R,Rt)**
A *tru* Rt is *comprised of a resource* R.

Core Term: Pred-48. **amount_produced(Rt,Qu_p)**
The *quantity of a tru Rt produced* is Qu_p.

Core Term: Pred-49. **amount_available(St,Rt,Qu_a)**
There is a *quantity of a tru Rt that is committed* to the *activity* associated with the *state* St. Then, Qu_a is the *quantity of Rt that is available* for other activities.

Core Term: Pred-50. **amount_committed(St,Rt,Qu_c)**
The *quantity Qu_c of the tru Rt is committed* to an *activity*, and the *state* description of this commitment is St.

Core Term: Pred-51. **produces(St,Rt)**
In the Core Ontologies, **produces(St,R)** is presented, where St is a *state* and R could only be a *resource*. The second parameter can now be a *tru*, Rt.

Core Term: Pred-52. **consumes(St,Rt)**
In the Core Ontologies, **consumes(St,R)** is presented, where St is a *state* and R could only be a *resource*. The second parameter can now be a *tru*, Rt.

Core Term: Pred-53. **uses(St,Rt)**
In the Core Ontologies, **uses(St,R)** is presented, where St is a *state* and R could only be a *resource*. The second parameter can now be a tru, Rt.

Core Term: Pred-54. **releases(St,Rt)**
In the Core Ontologies, **releases(St,R)** is presented, where St is a *state* and R could only be a *resource*. The second parameter can now be a tru, Rt.

The following axioms ensure consistency: The same tru cannot be represented to be actually produced more than once, and a tru is always used before it is released.

Trace Axiom: Cons-1. **A tru is produced only once.**

$$\forall A \forall St_1 \forall Rt \forall s \,\textbf{[}\, holds(\text{produce}(St_1,A),s) \wedge holds(\text{produces}(St_1,Rt),s) \supset \neg\exists St_2 \,\{\, holds(\text{produce}(St_2,A),s) \wedge holds(\text{produces}(St_2,Rt),s) \wedge St_1 \neq St_2 \,\}\, \textbf{]}.$$

Rt: a tru
St_1,St_2: the same state describing the production of Rt
A: an activity which produces Rt
s: an extant situation

Trace Axiom: Cons-2. **If there is a release state associated with a primitive activity, then there must also be a use state associated with the same activity that occurs on or before the release state.**

$$\forall St \forall Rt \forall T \forall Tp \forall s \,\textbf{[}\, holds(\text{release}(St,A),s) \wedge holds(\text{primitive\_activity}(A),s) \wedge occurs_T(\text{state\_duration}(St,T)) \wedge \text{start\_point}(T,Tp) \supset \exists Sto \exists To \exists Tpo \exists s_o(\, holds(\text{use}(Sto,A),s_o) \wedge occurs_T(\text{state\_duration}(Sto,To)) \wedge \text{end\_point}(To,Tpo) \wedge Tpo \leq Tp \,)\, \textbf{]}.$$

Rt: a tru
A: an activity that uses and releases Rt
St, Sto:states describing the release and use of Rt respectively
T,To: time durations corresponding to St, and Sto respectively
Tp,Tpo: start point of T and end point of To respectively
s,s_o: extant or hypothetical situations

The following axiom enforces proper use of the term *amount available*.

Trace Axiom: Cons-3. **At any given point in time, the amount available of a tru for any state is the same.**

$$\forall St_1 \forall St_2 \forall Rt \forall Qu_1 \forall Qu_2 \forall s \,\textbf{[}\, holds(\text{amount\_available}(St_1,Rt,Qu_1),s) \wedge holds(\text{amount\_available}(St_2,Rt,Qu_2),s) \wedge occurs_T(\text{state\_duration}(St_1,T_1)) \wedge occurs_T(\text{state\_duration}(St_2,T_2)) \wedge \exists Tp \,(\, \text{has\_point}(T_1,Tp) \wedge \text{has\_point}(T_2,Tp) \,) \supset Qu_1 = Qu_2) \,\textbf{]}.$$

Rt: a tru
St_1,St_2:states which have values for the amount available of Rt
Qu,Qu_2:amount of Rt available for St_1 and St_2 respectively
s: an extant or hypothetical situation

Then, a tru can be simply defined as:

Trace Axiom: Defn-2. **tru(Rt)**

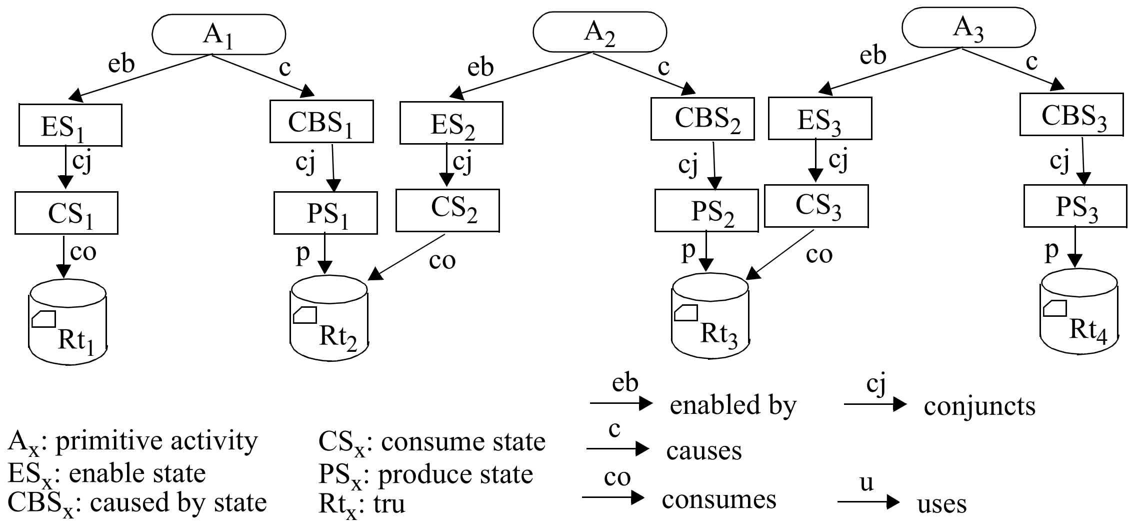
An entity Rt is a *tru* if there exists a *has tru* relation between a *resource* R and Rt.

$$\forall Rt \forall R \forall s \,\textbf{[}\, holds(\text{has\_tru}(R,Rt),s) \supset holds(\textbf{\textit{tru(Rt)}},s) \,\textbf{]}.$$

Rt: a traceable resource unit
R: a resource that comprises Rt
s: an extant or hypothetical situation

A trace path, which includes activity cluster entities like *trus* and *primitive activities*, looks something like this:

Figure 5.1 An Example of Finding a Trace Path in an Activity Cluster



Trace path L from (A_3,Rt_4) to (A_1,Rt_1) = { produce(A_3,Rt_4), consume(A_3,Rt_3), produce(A_2,Rt_3), consume(A_2,Rt_2), produce(A_1,Rt_2), consume(A_1,Rt_1) }

A trace path from one activity to another activity (e.g., from A_3 to A_1) is called an *activity trace*; a path from one tru to another tru (e.g., from R_4 to R_1) is called a *tru trace*. The predicates corresponding to these traces are the following:

Trace Term: Pred-1. **activity_trace(Ay,Ax,L)**

- L is the trace path of the activity trace from Ay to Ax
- this predicate is formally defined later in the chapter

Trace Term: Pred-2. **tru_trace(Rty,Rtx,L)**

- L is the trace path of the tru trace from Ry to Rx
- this predicate is formally defined later in the chapter

In order to represent the quantity of a *tru*, the term, *resource point of a tru*, is used.

Trace Term: Pred-3. **rp_tru(Rt,Qu,Tp,U)**

The *resource point of a tru* Rt is its quantity Qu at a given point in time Tp, expressed in U units of measurement.

- This predicate is formally defined later in the chapter.

Using representations for *trus*, *activity* and *tru trace* paths, and *resource points of trus*, the conditions for ensuring accurate traceability can be formally stated.

Developing terminology and axioms to formally state and answer:

CQ 5.7 At which time was a given batch recognized to exist?

The constraint below sets a boundary for performing a trace.

Trace Axiom: Defn-3. **tru_known(Rt,T_p)**

There must be a point in time when the *tru* is first recognized to exist. It is defined that a *tru is known to exist* the first time that it is *used*, *consumed*, or *produced* by a primitive activity.

$$\forall Rt \forall s \exists T_p \exists A\ (\ \mathit{holds}(\boldsymbol{\mathit{tru\_known}(Rt,T_p)},s) \equiv \Big[\ \mathit{holds}(\mathrm{tru}(Rt),s) \wedge \mathit{holds}(\mathrm{primitive\_activity}(A),s) \wedge \mathit{holds}(\mathrm{curp\_res\_tru}(A,Rt),s) \wedge \exists St \exists T\ [\ (\ (\ \mathit{holds}(\mathrm{uses}(St,Rt),s) \vee \mathit{holds}(\mathrm{consumes}(St,Rt),s)\ ) \wedge \mathit{occurs}_T(\mathrm{state\_duration}(St,T)) \wedge \mathrm{start\_point}(T,T_p)\ ) \vee (\ \mathit{holds}(\mathrm{produces}(St,Rt),s) \wedge \mathit{occurs}_T(\mathrm{state\_duration}(St,T)) \wedge \mathrm{end\_point}(T,T_p)\ )\ ] \wedge \neg\exists St_o \neg\exists T_o \neg\exists s_o\ (\ \mathit{holds}(\mathrm{consumes}(St_o,Rt),s_o) \vee \mathit{holds}(\mathrm{uses}(St_o,Rt),s_o) \vee \mathit{holds}(\mathrm{produces}(St_o,Rt),s_o)\ ) \wedge \mathit{occurs}_T(\mathrm{state\_duration}(St_o,T_o)) \wedge \mathrm{start\_point}(T_o,Tp_o) \wedge St \neq St_o \supset Tp_o < Tp\ \Big].$$

- Rt: ID of the tru
- T_p: time point at which the tru is recognized to exist
- St: the first state that uses/consumes/produces Rt
- T: time duration for state St
- Tp: if St is a use or consume state, this is the start point of T; if St is a produce state, this is the end point of T
- s: an extant or hypothetical situation

- The term *curp res tru* is defined later in the chapter; an activity that *consumes*/*uses*/*releases*/*produces* a tru or resource has a *curp res tru* relation with the resource or tru.

Developing terminology and axioms to formally state and answer:

CQ 5.3 If a batch is consumed, what is the batch that is produced as a direct result of this consumption? Are these two batches uniquely identified?

CQ 5.4 Is a given activity a splitting activity?

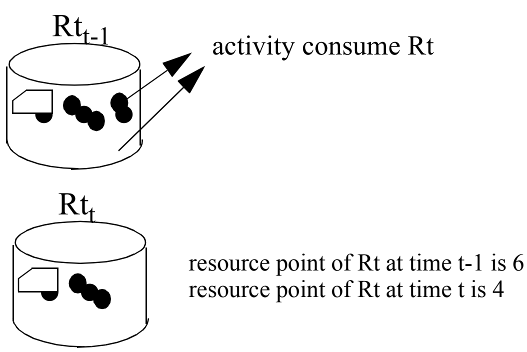
CQ 5.5 If batches are merged, what is the composite batch that is formed as a result of the merge? Are the merged batches and the composite batch uniquely identified?

CQ 5.6 Is a given activity a merging activity?

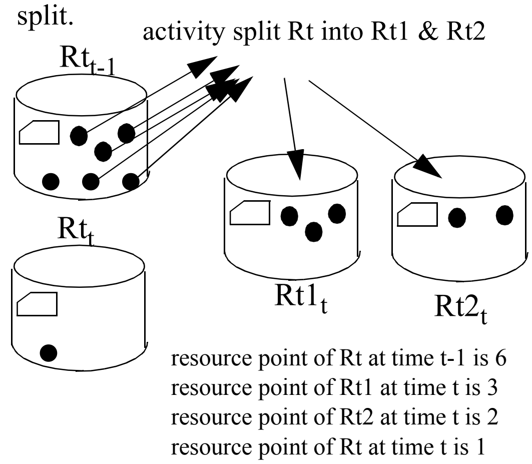
Other conditions constrain the splitting and merging of trus. The following diagram shows the rationale for these constraints.

Figure 5.2 Example of Splitting and Merging Trus

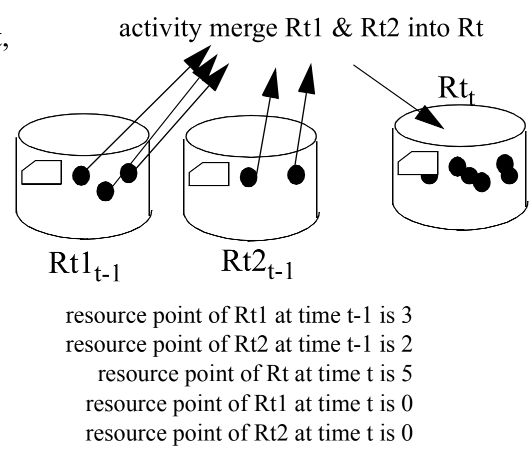
When not to have unique IDs for split trus?
When Rt is split, but the split parts do not need to be described as different from Rt. For instance, if a portion of Rt is consumed, there is no need to uniquely identify the split portion.



When to have unique IDs for split trus?
When Rt is split into different trus that should be described as different from Rt. For instance, if Rt is split into trus that are physically located elsewhere. A split activity represents the activity that performs this split.



When to have unique ID for merged tru?
Always. If the merged tru has a different ID Rt, then it can at least be known that a ball from Rt is from Rt1, Rt2,,, or Rtn. However if the merged tru took the ID of one of the trus that the balls came from, Rt1 for example, then it is incorrect to say that any given ball in that tru came from Rt. A merge activity represents the activity that performs this merge.



First, the terms *split trace* and *split activity*, as well as *merge trace* and *merge activity* are defined:

Trace Axiom: Defn-4. **split_trace(A,Rt,{Rt$_i$})**

If a *primitive activity* *consumes* all or part of one original *tru* and *produces* one or more *trus*, these *trus* are uniquely identified and comprised of the same *resource*, and there exists no quantity loss, then this activity is a *split activity* and there exists a *split trace* relationship.

$$\forall A \forall Rt \forall (Rt_1,,,Rt_n) \forall s \exists R \exists St \exists Quc \exists s_o \Big[\\ holds(\textbf{split\_trace(A,Rt,\{Rt}_1\textbf{,Rt}_2\textbf{,,,Rt}_n\textbf{\})},s) \equiv \\ holds(primitive\_activity(A),s_o) \wedge holds(consume(St,A),s_o) \wedge \\ holds(consumes(St,Rt),s_o) \wedge holds(has\_tru(R,Rt),s_o) \wedge \\ holds(amount\_committed(St,Rt,Quc),s_o) \wedge \\ \neg\exists St_o \neg\exists Rt_o \{ holds(consumes(St_o,Rt_o),s_o) \supset Rt \neq Rt_o \} \wedge \\ \exists (St_1,,,St_n) \exists (Qup_1,,,Qup_n) \Big\{ (\bigcap_{i=1,,n} (holds(produce(St_i,A),s) \wedge \\ holds(produces(St_i,Rt_i),s) \wedge holds(has\_tru(R,Rt_i),s) \wedge \\ holds(amount\_produced(Rt_i,Qup_i),s)) \wedge s_o < s) \wedge \\ Quc = \sum_{i=1,,n} Qup_i \Big\} \Big].$$

Trace Axiom: Defn-5. **split_activity(A)**

$$\forall A \forall s \exists Rt \exists (Rt_1,,,Rt_n) \Big[holds(split\_trace(A,Rt,\{Rt_1,Rt_2,,Rt_n\}),s) \supset \\ holds(\textbf{split\_activity(A)},s) \Big].$$

A: a split activity
Rt: the original tru that is split
R: resource that Rt is comprised of
St: state description for the consumption of Rt by A
Quc: quantity of Rt that in consumed by A
Rt_i: the trus that are formed when Rt is split
St_i: state descriptions for the production of Rt_i by A
Qup_i: quantities of Rt_i that are produced by A
s,s_o: extant or hypothetical situation

Trace Axiom: Defn-6. **merge_trace(A,{Rt$_i$},Rt})**

If a *primitive activity* *consumes* all or part of one or more *trus* and *produces* one *tru*, these *trus* are uniquely identified and comprised of the same *resource*, and there is no quantity loss, then this *activity* is a *merge activity* and there exists a *merge trace* relationship.

$$\forall A \forall s \forall Rt \exists (Rt_1,,,Rt_n) \exists R \exists St \exists Quc \exists s_o \\ \Big[holds(\textbf{merge\_trace(A,\{Rt}_1\textbf{,Rt}_2\textbf{,,,Rt}_n\textbf{\},Rt)},s) \equiv \\ holds(primitive\_activity(A),s) \wedge holds(produce(St,A),s) \wedge \\ holds(produces(St,Rt),s) \wedge holds(has\_tru(R,Rt),s) \wedge \\ holds(amount\_produced(Rt,Qup),s) \wedge \\ \neg\exists St_o \neg\exists Rt_o \{ holds(produces(St_o,Rt_o),s_o) \supset Rt \neq Rt_o \} \wedge \\ \exists (St_1,,,St_n) \exists (Quc_1,,,Quc_n) \Big\{ (\bigcap_{i=1,,n} (holds(consume(St_i,A),s_o) \wedge$$

$$holds(\text{consumes}(St_i,Rt_i),s_o) \wedge holds(\text{has\_tru}(R,Rt_i),s_o) \wedge holds(\text{amount\_committed}(St_i,Rt_i,Quc_i),s_o)\ ) \wedge s_o<s\ ) \wedge Qup = \textstyle\sum_{i=1,,n} Quc_i \ \}\ ].$$

Trace Axiom: Defn-7. **merge_activity(A)**

$$\forall A \forall s \exists Rt \exists (Rt_1,,,Rt_n)\ [\ holds(\text{merge\_trace}(A,\{Rt_1,Rt_2,,Rt_n\},Rt),s) \supset holds(\textbf{\textit{merge\_activity(A)}},s)\ ].$$

A: a merge activity
Rt: the composite tru that results from a merge
R: resource that Rt is comprised of
St: state description for the production of Rt by A
Qup: quantity of Rt that in produced by A
Rt_i: the trus that are merged to form Rt
St_i: state descriptions for the consumption of Rt_i by A
Quc_i: quantities of Rt_i that are consumed by A
s,s_o: extant or hypothetical situation

The following are the unique identification constraints on splitting and merging.

Trace Axiom: Cons-4. If an original tru is split using a split activity, then all the split trus must be identified as different from the original tru. *It is then possible to uniquely trace back from a split tru to the original tru.*

$$\forall A \forall s \forall Rt \forall (Rt_1,,,Rt_n)\ [\ holds(\text{split\_trace}(A,Rt,\{Rt_1,Rt_2,,Rt_n\}),s) \supset \textstyle\bigcap_{i=1,,n} Rt \neq Rt_i\ ].$$

A: a split activity
Rt: the original tru that is split
Rt_i: the trus that are split from Rt
s: extant or hypothetical situation

Trace Axiom: Cons-5. If trus are merged into a composite tru, then all the merged trus must be identified as different from the composite tru. *It is then possible to trace back (but not uniquely) from a composite tru to one of the trus that were merged to form the composite tru.*

$$\forall A \forall s \forall Rt \forall (Rt_1,,,Rt_n)\ [\ holds(\text{merge\_trace}(A,\{Rt_1,Rt_2,,Rt_n\},Rt),s) \supset \textstyle\bigcap_{i=1,,n} Rt \neq Rt_i\ ].$$

A: a merge activity
Rt: the composite tru that is formed
Rt_i: the trus that are merged to form Rt
s: extant or hypothetical situation

By composing representations developed so far—the formal definitions for these compositions are provided in the next section—*it is possible to trace back from a given tru to another tru that* *was used, consumed, merged, or split to produce that given tru*. The adjunct to this can also be stated: *Individual units of a tru are indistinguishable from each other, and hence traceability within a tru is not possible.*

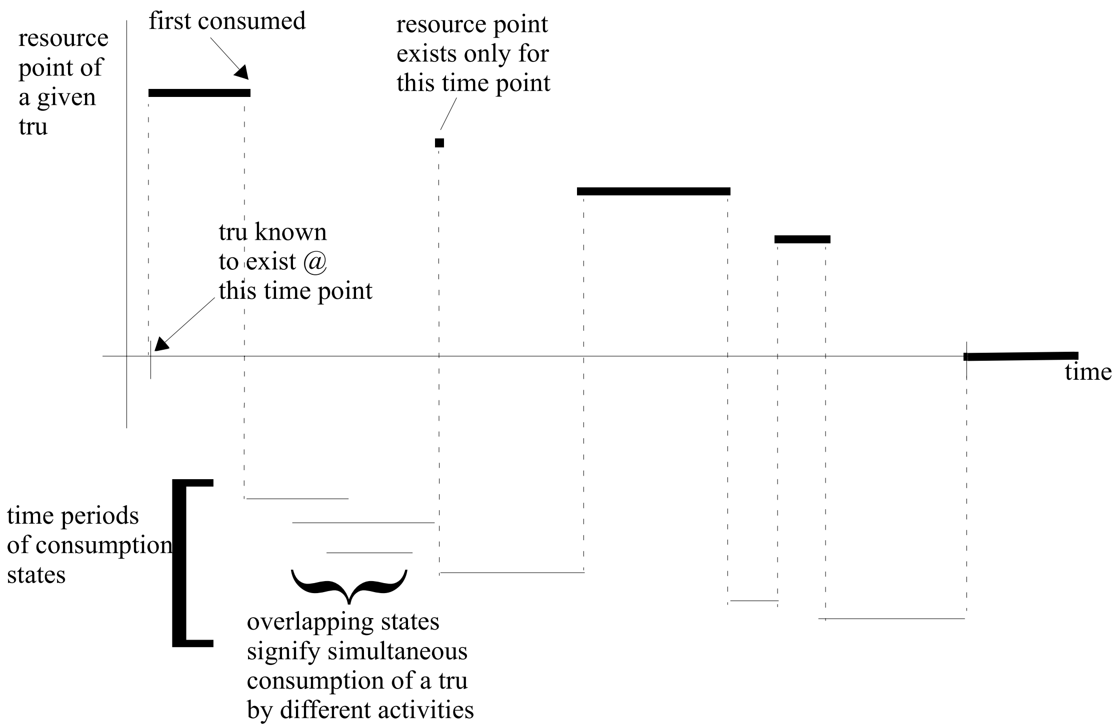
Developing terminology and axioms to formally state and answer:

CQ 5.8 What is the quantity of a given batch at a given point in time?

The rules for calculating the resource point of a given tru at a given point in time are the following:

- Calculate the resource point when the tru is recognized to exist.
- Decrement the resource point of a tru only if it is consumed.
- It is assumed that the resource point of a tru does not exist until it is fully produced. Accordingly, there is no resource point while the tru is in the midst of being produced.
- It is assumed that there is no resource point while a tru is being consumed. Accordingly, at a given point in time during a tru's consumption, it is not possible to determine its quantity without a doubt.
- The resource point of a tru just after its consumption is the prior resource point minus the quantity just consumed.
- A tru may be simultaneously used, consumed, or released by several primitive activities—but only one activity produces a tru. This requires special attention. For instance, if several activities simultaneously consume a tru, then according to the above assumptions, the resource point of the tru cannot be calculated until all consumptions cease. If a tru is always being consumed, then it is possible to know the resource point only when the tru is first consumed and when it is completely exhausted.

Figure 5.3 **Example of Changes in Resource Points of Trus**



The following axioms formalize these rules.

Trace Axiom: Defn-8. If a *tru* is *produced* by a *primitive activity*, then the *resource point of the tru* at the time of activity completion is the *quantity that is produced*.

$$\forall Rt \forall Qu \forall U \forall Tp \forall s \exists A \; \{ \; holds(\text{primitive\_activity}(A),s) \wedge holds(\text{produce\_res\_tru}(A,Rt),s) \wedge occurs_T(\text{activity\_duration}(A,T)) \wedge \text{end\_point}(T,Tp) \wedge holds(\text{amount\_produced}(Rt,Qu),s) \wedge holds(\text{unit\_of\_measurement}(Rt,\text{capacity},U,A),s) \supset holds(\boldsymbol{rp\_tru(Rt,Qu,T_p,U)},s) \; \}.$$

- The term *produce res tru* is defined later in the chapter.
- The term *unit of measurement* is from the Core Ontologies in the methodology chapter.

Rt: a tru
Tp time point for which a resource point of Rt is calculated
Qu quantities of Rt produced
U: unit of measurement for Rt
A: primitive activity that produces Rt
s: an extant or hypothetical situation

Trace Axiom: Defn-9. If a tru exists exactly at the time point when it is used and/or consumed by one or more primitive activities, then the resource point of the tru at that time point is the quantity that is available for other activities plus the quantities that are committed just after that point in time.

$$\forall Rt \forall Qu \forall T_p \forall U \forall s \exists (St_1,,,St_n) \exists (A_1,,,A_n) \exists (T_1,,,T_n) \exists (Quc_1,,,Quc_n) \exists Qua_1$$
$$\exists (s_1,,,s_n) \Big\{ \{ \bigcap_{i=1,,n} (\mathit{holds}(\text{primitive\_activity}(A_i),s_i)) \wedge$$
$$\{ (\mathit{holds}(\text{use}(St_i,A_i),s_i) \wedge \mathit{holds}(\text{uses}(St_i,Rt),s_i) \vee$$
$$(\mathit{holds}(\text{consume}(St_i,A_i),s_i) \wedge \mathit{holds}(\text{consumes}(St_i,Rt),s_i)) \} \wedge$$
$$\mathit{occurs}_T(\text{state\_duration}(St_i,T_i)) \wedge$$
$$\text{start\_point}(T_i,Tp) \wedge \mathit{holds}(\text{tru\_known}(Rt,Tp),s) \wedge$$
$$\mathit{holds}(\text{amount\_committed}(St_i,Rt,Quc_i),s_i) \wedge$$
$$\mathit{holds}(\text{unit\_of\_measurement}(Rt,\text{capacity},U,A_i),s_i)) \supset$$
$$\neg \exists St_J \neg \exists T_J \neg \exists s_J$$
$$((\mathit{holds}(\text{uses}(St_J,Rt),s_J) \vee \mathit{holds}(\text{consumes}(St_J,Rt),s_J)) \wedge$$
$$\mathit{occurs}_T(\text{state\_duration}(St_J,T_J)) \wedge$$
$$\text{start\_point}(T_J,Tp) \supset$$
$$\bigcap_{i=1,,n} St_J \neq St_i) \} \wedge$$
$$\text{amount\_available}(St_1,Rt,Qua_1,s_1) \wedge$$
$$Qu = Qua_1 + \sum_{i=1,,n} Quc_i$$
$$\supset \mathit{holds}(\boldsymbol{rp\_tru(Rt,Qu,T_{p1},U)},s_1) \Big\}.$$

Rt: a tru
Tp time point for which a resource point of Rt is calculated
Qu resource point of Rt at Tp
U: unit of measurement for Rt
$A_1,A_2,A_i,,A_n$:primitive activities that start to use and/or consume Rt at Tp
$St_1,St_2,St_i,,St_n$:states corresponding to the usage and/or consumption of Rt by $A_1,A_2,A_i,,A_n$ respectively
$T_1,T_2,T_i,,T_n$:time periods corresponding to $St_1,St_2,St_i,,St_n$ respectively
$Quc_1,Quc_2,Quc_i,,Quc_n$:amounts committed corresponding to $St_1,St_2,St_i,,St_n$ respectively
Qua_1: amount of Rt not committed and available for other activities at Tp
$s_1,s_2,s,,s_n$:extant or hypothetical situations

Trace Axiom: Defn-10. If there is a period of time in which a tru is not being consumed, then the resource point of that tru for all time points in that period is the resource point just after the last time that the tru was consumed.

$$\forall Rt \forall St_1 \forall St_2 \forall A_1 \forall A_2 \forall T_1 \forall T_2 \forall Tp_1 \forall Tp_2 \forall Qu \forall U \forall s_1 \forall s_2 \Big\{$$
$$\mathit{holds}(\text{consume}(St_1,A_1),s_1) \wedge \mathit{holds}(\text{consumes}(St_1,Rt),s_1) \wedge$$
$$\mathit{holds}(\text{primitive\_activity}(A_1),s_1) \wedge$$
$$\mathit{occurs}_T(\text{state\_duration}(St_1,T_1)) \wedge \text{end\_point}(T_1,Tp_1) \wedge$$
$$\mathit{holds}(\text{rp\_tru}(Rt,Qu,Tp_1,U),s) \wedge$$
$$\mathit{holds}(\text{consume}(St_2,A_2),s_2) \wedge \mathit{holds}(\text{consumes}(St_2,Rt),s_2) \wedge$$

$holds$(primitive_activity(A_2),s_2)
$occurs_T$(state_duration(St_2,T_2)) ∧ start_point(T_2,Tp_2) ∧ Tp_1<Tp_2 ∧
$\neg\exists St_o \neg\exists A_o \neg\exists T_o \neg\exists Tp_o$\{ $holds$(consume(St_o,A_o),s_o) ∧
$holds$(consumes(St_o,Rt),s_o) ∧ $holds$(primitive_activity(A_o),s_1) ∧
$occurs_T$(state_duration(St_o,T_o)) ∧
start_point(T_o,Tp_o) ⊃ Tp_o>Tp_1 ∧ Tp_o<Tp_2 \}⊃
$\forall Tp \forall s$ \{ Tp>Tp_1 ∧ Tp≤Tp_2 ∧ s>s_1 ∧ s≤s_2
⊃ $holds$(***rp_tru(Rt,Qu,T_p,U)***,s) \}\}.

Rt: a tru
A_1,A_2:primitive activities that consume Rt; A_1 is performed before A_2
St_1,St_2:states corresponding to the consumption of Rt by A_1 and A_2 respectively
T_1,T_2:time periods corresponding to St_1,St_2 respectively
Tp_1,Tp_2:time points corresponding to end point of T_1 and start point of T_2 respectively
Qu: resource point of Rt at Tp
U: unit of measurement for Rt
s_1,s_2:extant or hypothetical situations

Trace Axiom: Defn-11. A resource point can only be calculated if the following are true: There is a point in time in which consumption of a tru has ended (at time point Tp_2) and there are no other consumptions of the tru taking place at that point in time, and there was a prior resource point before one or more consumptions of the tru started (at time point Tp_1). Then, the resource point of the tru at Tp_2 is the resource point at Tp_1 minus all quantity of that tru consumed since Tp_1.

$\forall Rt \forall St_1 \forall St_2 \forall A_1 \forall A_2 \forall T_1 \forall T_2 \forall Tp_1 \forall Tp_2 \forall Qu_1 \forall U \forall s_1 \forall s_2$\{
$holds$(consume(St_1,A_1),s_1) ∧ $holds$(consumes(St_1,Rt),s_1) ∧
$holds$(primitive_activity(A_1),s_1) ∧
$occurs_T$(state_duration(St_1,T_1)) ∧ start_point(T_1,Tp_1) ∧
$holds$(rp_tru(Rt,Qu_1,Tp_1,U),s_1) ∧
$holds$(consume(St_2,A_2),s_2) ∧ $holds$(consumes(St_2,Rt),s_2) ∧
$holds$(primitive_activity(A_2),s_2) ∧
$occurs_T$(state_duration(St_2,T_2)) ∧ end_point(T_2,Tp_2) ∧ Tp_1<Tp_2 ∧

$\forall St_3 \forall A_3 \forall T_3 \forall Tp_3 \forall s_3 \forall St_4 \forall A_4 \forall T_4 \forall s_4 \exists Tp_4$
\{ Tp_3<Tp_2 ∧ Tp_3>Tp_1 ∧ Tp_4<Tp_2 ∧ Tp_4>Tp_1 ∧
has_point(T_3,Tp_3) ∧ has_point(T_4,Tp_4) ∧
$occurs_T$(state_duration(St_3,T_3)) ∧
$holds$(consume(St_3,A_3),s_3) ∧ $holds$(consumes(St_3,Rt),s_3) ∧
$holds$(primitive_activity(A_3),s_3) ∧
$occurs_T$(state_duration(St_4,T_4)) ∧
$holds$(consume(St_4,A_4),s_4) ∧ $holds$(consumes(St_4,Rt),s_4) ∧
$holds$(primitive_activity(A_4),s_4) ∧ St_3≠St_4 ⊃
Tp_3=Tp_4 \} ∧

$\exists(St_5,,,St_n)\exists(A_5,,,A_n)\exists(T_5,,,T_n)\exists(T_{s5},,,T_{sn})\exists(T_{e5},,,T_{en})$
$\exists(Quc_5,,,Quc_n)\exists(s_5,,,s_n)$ { $\bigcap_{i=5,,n}$ (*holds*(primitive_activity(A_i),s) $\wedge$
holds(consume(St_i,A_i),s_i) $\wedge$ *holds*(consumes(St_i,Rt),s_i) $\wedge$
$occurs_T$(state_duration(St_i,T_i)) $\wedge$
start_point(T_i,T_{si}) $\wedge$ end_point(T_i,T_{ei}) $\wedge$ $T_{si} \geq Tp_1 \wedge T_{ei} \leq Tp_2 \wedge$
holds(amount_committed(St_i,Rt,Quc_i),s_i) $\supset$
$\neg\exists St_J \neg\exists T_J \neg\exists T_{eJ} \neg\exists T_{sJ} \neg\exists s_J$
(*holds*(consumes(St_J,Rt),s_J) $\wedge$ $occurs_T$(state_duration(St_J,T_J)) $\wedge$
start_point(T_J,T_{sJ}) $\wedge$ end_point(T_J,T_{eJ}) $\wedge$ $T_{sJ} \geq Ts \wedge T_{eJ} \leq Te \supset$
$\bigcap_{i=5,,n} St_J \neq St_i$) }$\wedge$
$Qu = Qu_1 - \sum_{i=5,,n} Quc_i$
$\supset$ *holds*(***rp_tru(Rt,Qu,Tp$_2$,U)***,s_2) }.

Rt: a tru
A_1,A_2:primitive activities that consume Rt; A_1 is performed before A_2
St_1,St_2:states corresponding to the consumption of Rt by A_1 and A_2 respectively
T_1,T_2:time periods corresponding to St_1,St_2 respectively
Tp_1,Tp_2:time points corresponding to start point of T_1 and end point of T_2 respectively
Qu_1 resource point of Rt at Tp_1
U: unit of measurement for Rt
A_3,A_4:primitive activities that consume Rt that start consuming after A_1 and finish consuming before A_2; The consumption periods of these activities overlap
St_3,St_4:states corresponding to the consumption of Rt by A_3 and A_4 respectively
T_3,T_4:time periods corresponding to St_3,St_4 respectively
Tp_3,Tp_4:time points in T_3 and T_4 respectively
A_5,,A_i,,A_n:primitive activities that consume Rt somewhere between Tp_1 and Tp_2
St_5,,St_i,,St_n:states corresponding to the consumption of Rt by A_5,,A_i,,A_n respectively
T_5,T_i,,T_n:time periods corresponding to St_5,,St_i,,St_n respectively
T_{s5},T_{si},,T_{sn}:start points of T_5,T_i,,T_n respectively
T_{e5},T_{ei},,T_{en}:end points of T_5,T_i,,T_n respectively
T_5,T_i,,T_n:time periods corresponding to St_5,,St_i,,St_n respectively
Quc_5,,Quc_i,,Quc_n:amounts committed to St_5,,St_i,,St_n respectively
s_1,s_2,s,,s_n:extant or hypothetical situations

A resource point of a tru is constrained by the following axioms.

Trace Axiom: Cons-6. When a *tru* is being *consumed*, it has no *resource point*.

$\forall Rt \forall St \forall A \forall T \forall Ts \forall Te \forall Tp \forall U \forall s$
[*holds*(consumes(St,Rt),s) $\wedge$ *holds*(consume(St,A),s) $\wedge$*holds*(tru(Rt),s) $\wedge$
holds(primitive_activity(A),s) $\wedge$ $occurs_T$(state_duration(St,T),s) $\wedge$
start_point(T,Ts) $\wedge$ end_point(T,Te) $\wedge$ has_point(T,Tp) $\wedge$ $Tp > Ts \wedge Tp < Te \wedge$
holds(unit_of_measurement(Rt,capacity,U,A),s) $\supset$
$\neg\exists$Qu *holds*(rp_tru(Rt,Qu,Tp,U),s)].

Trace Axiom: Cons-7. Before the time point, Tp, at which point the *tru is known to exist*, there is no *quantity for the tru*.

$\forall Rt\ \forall T_{pa}\ \forall U\ \exists T_{pp} \forall s$ [*holds*(tru_known(Rt,T_{pp}),s) $\wedge$ $T_{pa} < T_{pp}$ $\supset$
$\neg\exists Qu_a \neg\exists s'$ *holds*(rp_tru(Rt,Qu_a,T_{pa},U),s′)].

Rt: ID of the tru
T_{pp}: time point at which the tru is recognized to exist
T_{pa}: any time point before T_{pp}
U: unit of measurement for Rt
Qu_a: the quantity of Rt at T_{pa}. Since Rt does not exist at T_{pa}, there is no value for Qu_a
s: an extant or hypothetical situation
s′: s or a situation that occurs before s

Developing terminology and axioms to formally state and answer:

CQ 5.9 At what point in time are all the quantities of a given batch exhausted?

CQ 5.10 For a given batch, did the number of units in the batch change during a period of time?

How does the quantity of a *tru* change after it is recognized to exist? The quantity decreases if portions or all of the *tru* are consumed. How does this quantity increase? Recall this constraint upon a *tru*: Once *trus are recognized to exist*, aggregating the contents of two or more *trus* does not result in the aggregate quantity maintaining the ID of any of the *trus* that are aggregated. So, any attempt at incrementing the quantity of a *tru* results in a different *tru*. Therefore,

Trace Axiom: Cons-8. The *quantity of a given tru* never increases after it is *recognized to exist*.

$\forall Rt\ \forall Qu_a\ \forall Qu\ \forall U\ \forall T_{pa}\ \forall T_p \forall s\ \forall s'$ [*holds*(tru_known(Rt,T_p),s) $\wedge$ $T_{pa} > T_p$ $\wedge$
holds(rp_tru(Rt,Qu,T_p,U) ,s) $\wedge$ *holds*(rp_tru(Rt,Qu_a,T_{pa},U),s′) $\supset$
$Qu \geq Qu_a$].

Rt: a tru
T_p: time point at which Rt is recognized to exist
T_{pa}: any time point after T_p
Qu: total quantity of Rt at the time that it is recognized to exist
Qu_a: total quantity of Rt at time point T_{pa}
U: unit of measurement for Rt
s: an extant or hypothetical situation
s′: s or a situation that occurs after s

In the next section, the competency questions are formally posed using the ontology terminology and answered by deduction using the ontology axioms.

5.4.3 Formal Competency Questions

CQ 5.1 Is this a uniquely identified entity that can be traced?

- Is α a *primitive activity* in a situation σ? *holds*(primitive_activity(α),σ).
  - e.g. *holds*(primitive_activity(autoclave_cure_L_edge),sv_actual).
- Is κ a *traceable resource unit* in a situation σ? *holds*(tru(κ),σ).
  - e.g. *holds*(tru(tru1_raw_mtl_kit_w_psr),sv_actual).

CQ 5.2 If a batch is consumed, what is the batch that is produced as a direct result of this consumption? Are these two batches uniquely identified?

- Is a uniquely identified κ_1 a *tru that is consumed to produce a uniquely identified tru* κ_2 in a situation σ? $\exists A \exists St_1 \exists St_2 \exists s_1$ { *holds*(consumes(St_1,κ_1),s_1) $\wedge$ *holds*(consume(St_1,A),s_1) $\wedge$ *holds*(produces(St_2,κ_2),σ) $\wedge$ *holds*(produce(St_2,A),σ) $\wedge$ $\kappa_1 \neq \kappa_2$}.
- e.g., *holds*(consumes(St_1,tru1_L_edge_S004,),s_1) $\wedge$ *holds*(consume(St_1,A),s_1) $\wedge$ *holds*(produces(St_2,tru1_cured_L_edge),sv_actual) $\wedge$ *holds*(consume(St_2,A),sv_actual) }.

CQ 5.3 If an original batch is split, what are the batches that the original batch is split into? Are the original batch and the split batches uniquely identified?

- For a uniquely identified tru κ, what are the uniquely identified trus κ_1,κ_2,,,κ_n that κ is split into in a situation σ? $\exists A$ { *holds*(split_trace(A,κ,{κ_1,κ_2,,,κ_n}),σ) $\wedge$ $\kappa \neq \kappa_1 \neq \kappa_2 ... \neq \kappa_n$ }.
- e.g., $\exists A$ { *holds*(split_trace(A,tru1_all_parts,{tru1_part1,tru1_part2,tru1_part3}),sv_actual) }.

CQ 5.4 Is a given activity a splitting activity?

- Is α a split activity in a situation σ? *holds*(split_activity(α),σ).
- e.g., *holds*(split_activity(split_all_parts),sv_actual).

CQ 5.5 If batches are merged, what is the composite batch that is formed as a result of the merge? Are the merged batches and the composite batch uniquely identified?

- For a uniquely identified tru κ, what are the uniquely identified trus κ_1,κ_2,,,κ_n that merge to form κ in a situation σ? $\exists A${*holds*(merge_trace(A,{κ_1,κ_2,,,κ_n},κ),σ) $\wedge$ $\kappa \neq \kappa_1 \neq \kappa_2 ... \neq \kappa_n$}.
- e.g.,
  $\exists A${*holds*(merge_trace(A,{tru1_subass1,tru1_subass2,tru1_subass3},tru1_assembly),sv_actual) }.

CQ 5.6 Is a given activity a merging activity?

- Is α a merge activity in a situation σ? *holds*(merge_activity(α),σ).
- e.g., *holds*(merge_activity(merge_subassemblies),sv_actual) }.

CQ 5.7 At which time was a given batch recognized to exist?

- At what time point was a tru κ *known to exist* in a situation σ? $\exists Tp$ *holds*(tru_known(κ,Tp),σ).
- $\exists Tp$ *holds*(tru_known(tru1_raw_mtl_kit_w_psr,Tp),sv_actual).

CQ 5.8 What is the quantity of a given batch at a given point in time?

- What is the *resource point of a given tru* κ (expressed in *units of measurement* υ) at a point in time τ in a situation σ? $\exists Qu$ *holds*(rp_tru(κ,Qu,τ,υ),σ).

- $\exists$Qu *holds*(rp_tru(tru1_raw_mtl_kit_w_psr,Qu,10,piece),sv_actual).

CQ 5.9 At what point in time are all the quantities of a given batch exhausted?

- At which point in time does the *resource point of a given tru* κ (expressed in *units of measurement* υ) start being 0 in a situation σ? $\exists Tp_o \forall Tp$ (*holds*(rp_tru(κ,0,Tp_o,υ),σ) $\wedge$ *holds*(rp_tru(κ,0,Tp,υ),σ) $\supset Tp_o \leq Tp$).
- $\exists Tp_o \forall Tp$ (*holds*(rp_tru(tru1_raw_mtl_kit_w_psr,0,Tp_o,piece),sv_actual) $\wedge$ *holds*(rp_tru(tru1_raw_mtl_kit_w_psr,0,Tp,piece),sv_actual) $\supset Tp_o \leq Tp$).

CQ 5.10 For a given batch, did the number of units in the batch change during a period of time?

- Plot the *resource point of a tru* κ (expressed in *units of measurement* υ) over time for a duration of time $[\tau_1,\tau_2]$ in a situation σ? $\forall Tp \forall Qu$ (*holds*(rp_tru(κ,Qu,Tp,υ),σ) $\wedge$ $\tau_1 \leq Tp \leq \tau_2 \supset f(Tp,Qu)$). $f(Tp,Qu)$ is just a graphing function that plots Tp vs. Qu.
- $\forall Tp \forall Qu$ (*holds*(rp_tru(tru1_raw_mtl_kit_w_psr,Qu,Tp,piece),sv_actual) $\wedge$ $10 \leq Tp \leq 30 \supset f(Tp,Qu)$). $f(Tp,Qu)$ is just a graphing function that plots Tp vs. Qu

5.5 Traceability Ontology: Entity Classification and Tracing

The concept that *rules for composing material flows must be stated, so that if there exists a material flow between two entities, this flow can be found* is the basis for the following principles about representing traceability capability.

- Classification of traced entities must be represented. Certain types of entities must be treated specially for a trace. For example, it may not be possible to trace from the time after a final product was shipped out and similarly difficult to trace to a time before a raw material was brought into the enterprise, since record-keeping rigour required for traceability cannot be expected beyond the enterprise. It must be possible to classify these special entities.
- The primitive capability required for traceability throughout the enterprise must be represented as the capability to trace between two adjacent entities, where adjacency refers to direct material flow between two entities; i.e., there are no intermediary entities in this material flow.
- All possible trace paths can be represented by composing primitive trace paths.

The competency questions that characterize what needs to be represented so that an ontology is constructed upon these principles are presented next.

5.5.1 Informal Competency Questions

It should not be assumed that there exist enterprise models for suppliers or customers. *So it is assumed that traceability is possible only within the enterprise modelled*. Then the interface between the enterprise and the supplier, and between the enterprise and the customer, constitute boundary conditions upon the trace. What are the traced entities—primitive activities and trus—which are related to these interfaces?

CQ 5.11 Is this a final product?

CQ 5.12 Is this raw material?

CQ 5.13 Does this activity start processing the raw material?

CQ 5.14 Does this activity finish processing the final product?

With respect to direct material flow, it should be asked:

CQ 5.15 What is the direct material flow path through a given activity; that is, what is consumed and what is produced by this activity?

CQ 5.16 What is the direct material flow path that involves a given batch; that is, which activity produces this batch, and which activity subsequently consumes it?

With respect to general material flow, it should be asked:

CQ 5.17 What is the path of the material flow between two batches?

CQ 5.18 What is the path of the material flow between two activities?

Boundary conditions on tracing are required to answer these questions:

CQ 5.19 What is the path of the material flow from a final product to one of its raw materials?

CQ 5.20 What is the path of the material flow from an activity that finishes processing a final product to an activity that starts processing raw materials?

In order to answer these competency questions, the domain of traceability is analyzed, assumptions are stated, and terminology and axioms are developed.

5.5.2 Analysis, Terminology, Assumptions, and Axioms

Developing terminology and axioms to formally state and answer:

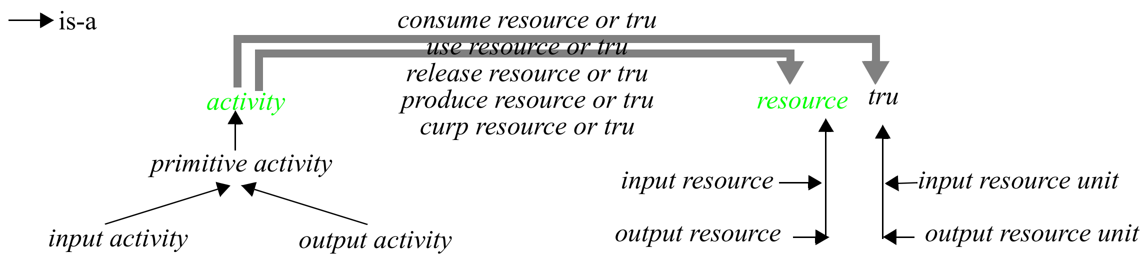
CQ 5.6 Is this a final product?

CQ 5.7 Is this raw material?

CQ 5.8 Does this activity start processing the raw material?

CQ 5.9 Does this activity finish processing the final product?

Figure 5.4 Data Model of Traceability Ontology Terms related to Classification



Activities that are at the interface between a given enterprise and a supplier organization that provide raw materials are *input activities*. Tracing to the activities that are performed to produce the raw materials requires that traceability within the supplier organization be possible. *Since traceability within that organization cannot be automatically assumed, it is assumed that tracing material flow before this activity is not possible*. Similarly, *tracing material flow after activities, called output activities, which are at the interface between a given enterprise and a customer organization that uses or consumes a final product is assumed not to be possible*.

These modelling assumptions are used to define the following terms.

Trace Term: Pred-4. **input_activity(A)**
An *input activity* is represented as a *primitive activity* that *consumes* an *input resource*.

Trace Axiom: Defn-12. **output_activity(A)**
An *output activity* is represented as a *primitive activity* that *produces* an *output*

resource.

$\forall A \forall s$ [*holds*(***output_activity(A)***,s) $\equiv$ *holds*(primitive_activity(A),s) $\wedge$ *holds*(produce_res_tru(A,R),s) $\wedge$ *holds*(output_resource(R),s)].

A: an output activity
R: an output resource
s: an extant or hypothetical situation

A resource that represents a prototypical raw material is called an *input resource*, and a batch or collection of an *input resource* is an *input resource unit*. Similarly, a prototypical final product and a batch of a final product are called *output resource* and *output resource unit*, respectively. *Tracing material flow to a tru which produces an input resource unit, or tracing from a tru which consumes or uses an output resource unit, is assumed to be not possible.*

Trace Axiom: PT-1. **input_resource(R)**

An *input resource* is a "raw material" for the enterprise; it is a *resource* that is brought in from another enterprise. A *resource* must be asserted as an *input resource*.

Trace Term: Pred-5. **input_ru(Rt)**

An *input resource unit* is a *tru* of an *input resource* that is *consumed*.

Trace Axiom: PT-2. **output_resource(R)**

An *output resource* is a "final product" of an enterprise; it is a *resource* that will be brought to another enterprise. A *resource* must be asserted as an *output resource*.

Trace Axiom: Defn-13. **output_ru(Rt)**

An *output resource unit* is a *tru* of an *output resource* that is produced and neither *consumed* nor *used.*

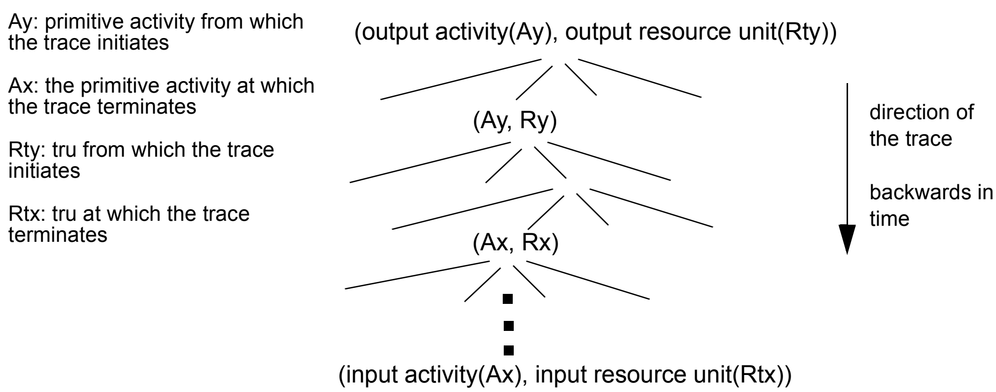
$\forall Rt \forall s \exists A \exists R$ {*holds*(***output_ru(Rt)***,s) $\equiv$ *holds*(has_tru(R,Rt),s) $\wedge$ *holds*(output_resource(R),s) $\wedge$ $\exists St$ *holds*(produces(St,Rt),s) $\wedge$ $\neg\exists St'\neg\exists s'$[*holds*(consumes(St′,Rt),s′) $\vee$ *holds*(uses(St′,Rt),s′)] }.

Rt: an output resource unit
A: output activity which produces the output resource unit
St: produce state which describes the production of the output resource unit by the output activity
s: an extant or hypothetical situation

If traceability is considered as the capability to find a connecting path that links two nodes in a network of nodes, then the diagram below shows how the path can be found. Given a set of possible paths in a network of initial, intermediary, and terminal nodes, the task of performing a trace can be modelled as one of starting from an initiating node for the trace—this may or may not be an initial node for the network—and traversing the network until either a terminating point for the trace or a terminal node for the network is encountered. The path followed in the network from the initiating node of the trace to the terminating node of the trace is the trace path.

In the Traceability Ontology, a node on the traceability network is a tuplet of a *primitive activity*, and a *tru* that it *uses*, *consumes*, *releases*, or *produces*. That is, the traceability network is an abstraction of the activity cluster. Given the special characteristics of input and output activities and input and output resource units, an initial node of the network is the output activity-output resource unit tuplet; a terminal node is the input activity-input resource unit tuplet.

Figure 5.5 **Finding a Trace Path in the Traceability Network**



The following terms describe relations useful for finding a trace path. These terms are used to construct other terms in the Ontologies for Quality Modelling and the ISO 9000 Micro-Theory.

Trace Term: Pred-6. **use_res_tru(A,Rc)**

If a *primitive activity* A *uses* Rc, then there exists a *use resource or tru* relationship between A and Rc; Rc can be either a *tru* or a *resource*

Trace Term: Pred-7. **consume_res_tru(A,Rc)**

If a *primitive activity* A *consumes* Rc, then there exists *a consume resource or tru* relationship between A and Rc; Rc can be either a *tru* or a *resource*.

Trace Term: Pred-8. **produce_res_tru(A,Rc)**

If a *primitive activity* A *produces* Rc, then there exists a *produce resource or tru* relationship between A and Rc; Rc can be either a *tru* or a *resource*.

Trace Term: Pred-9. **release_res_tru(A,Rc)**

If a *primitive activity* A *releases* Rc, then there exists a *release resource or tru* relationship between A and Rc; Rc can be either a *tru* or a *resource*.

Trace Term: Pred-10. **curp_res_tru(A,Rc)**

If a *primitive activity* A *Consumes/Uses/Releases/Produces* Rc ("CURPs Rc"), then there exists a *curp resource or tru* relationship between A and Rc; Rc can be either a *tru* or a *resource*.

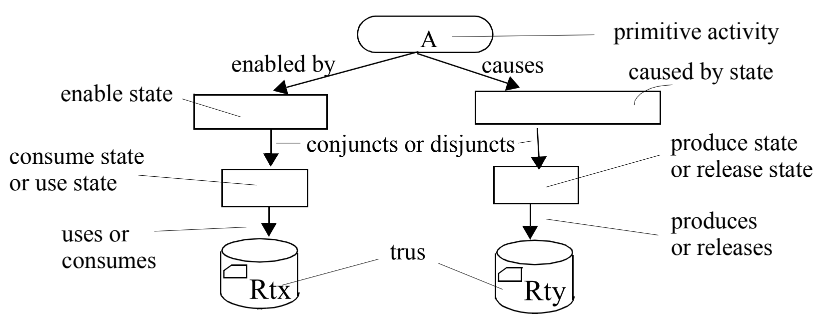
Developing terminology and axioms to formally state and answer:

CQ 5.10 What is the direct material flow path through a given activity; that is, what is consumed and what is produced by this activity?

CQ 5.11 What is the direct material flow path that involves a given batch; that is, which activity produces this batch, and which activity subsequently consumes it?

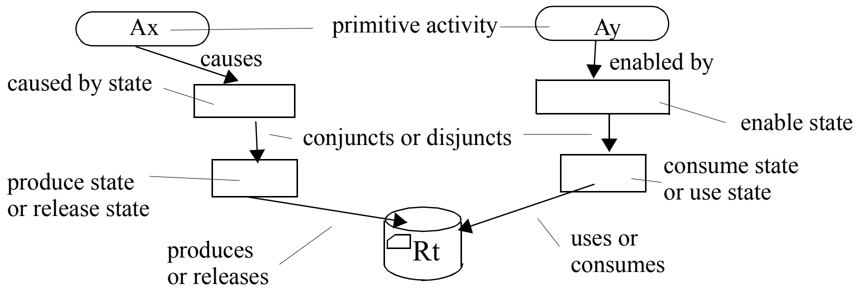
In an activity cluster, a trace is possible from a *tru* to a second *tru*, if the first tru is *produced* or *released* by a given activity and the second *tru* is *used* or *consumed* by that activity. This is the *primitive activity trace* relationship. In a similar manner, the *primitive tru trace* relationship can be defined.

Trace Term: Pred-11. **prim_activity_trace(A,Rty,Rtx,L)**



- e.g. L = { produce(A,Rty), consume(A,Rtx) }

Trace Term: Pred-12. **prim_tru_trace(Rt,Ay,Ax,L)**



- e.g. L = { consume(Ay,Rt), produce(Ax,Rt) }

Developing terminology and axioms to formally state and answer:

CQ 5.12 What is the path of the material flow between two batches?

CQ 5.13 What is the path of the material flow between two activities?

Tru trace paths tracing material flow from two batches, and activity trace paths tracing material flow from two activities can be composed from primitive activity and tru trace paths in the following way:

Trace Axiom: Defn-14. **activity_trace(A_y,A_x,L)**

$$\forall A_x \forall A_y \forall L \forall Rt \forall s \exists L_1 \exists L_2$$
$$\{[\ \mathit{holds}(\text{prim\_tru\_trace}(Rt,A_y,A_x,L),s) \vee$$
$$\exists A_z\ (\ \mathit{holds}(\text{prim\_tru\_trace}(Rt,A_y,A_z,L_1),s) \wedge$$
$$\exists s_1\ \mathit{holds}(\text{activity\_trace}(A_z,A_x,L_2),s_1) \wedge L=L_1+L_2\ )$$
$$\supset \mathit{holds}(\boldsymbol{\mathit{activity\_trace}(A_y,A_x,L)},s)\}.$$

A_y: primitive activity from which the trace initiates
A_x: primitive activity at which the trace terminates
L: list denoting the trace relationships: e.g. L={consume(A_y,Rt), produce(A_x,Rt)}
s,s_1: extant or hypothetical situations

Trace Axiom: Defn-15. **tru_trace(Rt_y,Rt_x,L)**

$$\forall R_x \forall R_y \forall L \forall A \forall s \exists L_1 \exists L_2$$
$$\{[\ \mathit{holds}(\text{prim\_activity\_trace}(A,Rt_y,Rt_x,L),s) \vee$$
$$\exists R_z\ (\ \mathit{holds}(\text{prim\_activity\_trace}(A,Rt_y,Rt_z,L_1),s) \wedge$$
$$\exists s_1\ \mathit{holds}(\text{tru\_trace}(Rt_z,Rt_x,L_2),s_1) \wedge L=L_1+L_2\ )$$
$$\supset \mathit{holds}(\boldsymbol{\mathit{tru\_trace}(Rt_y,Rt_x,L)},s)\}.$$

R_y: tru from which the trace initiates
R_x: tru at which the trace terminates
L: list denoting the trace relationships: e.g. L={produce(A,Rt_y), consume(A,Rt_x)}
s,s_1: extant or hypothetical situations

Developing terminology and axioms to formally state and answer:

CQ 5.14 What is the path of the material flow from a final product to one of its raw materials?

CQ 5.15 What is the path of the material flow from an activity that finishes processing a final product to an activity that starts processing raw materials?

In order to answer these questions, the primitive traces at the boundary conditions must be specially defined. The following are these definitions.

Trace Axiom: Defn-16. **prim_activity_trace(A,Rt,input_ru,L)**

If a *tru* Rt is an *input resource unit* consumed by an *input activity* A, then there is a *primitive activity trace* between A and an *input resource unit* Rt.

∀Rt∀A∀s { *holds*(input_ru(Rt),s) ∧ *holds*(input_activity(A),s) ∧
holds(consume_res_tru(A,Rt),s) ⊃
∃L [L="consume(A,Rt), input_ru(Rt)" ∧
holds(***prim_activity_trace(A,Rt,input_ru,L)***,s)] }.

Trace Axiom: Defn-17. **prim_tru_trace(Rt,Ay,input_activity,L)**

If a *tru* Rt is an *input resource unit* consumed by an *input activity* A, then there is a *primitive tru trace* between Rt and an *input activity* A.

∀Rt∀A∀s { *holds*(input_ru(Rt),s) ∧ *holds*(input_activity(A),s) ∧
holds(consume_res_tru(A,Rt),s) ⊃
∃L [L="consume(A,Rt), input_activity(A)" ∧
holds(***prim_tru_trace(Rt,A,input_activity,L)***,s)] }.

Rt: an input resource unit which is consumed by an input activity A
A: an input activity
L: list denoting the trace relationships; e.g., L="consume(A,Rt), input_ru(Rt)" denotes that A is an input activity which consumes the input resource unit Rt
s: an extant or hypothetical situation

Trace Term: Pred-13. **prim_activity_trace(A,output_ru,Rtx,L)**

If a *tru* Rt is an *output resource unit* produced by an *output activity* A, then there is a *primitive activity trace* between A and an *output resource unit* Rt.

Trace Term: Pred-14. **prim_tru_trace(Rt,output_activity,A,L)**

If a *tru* Rt is an *output resource unit* produced by an *output activity* A, then there is a *primitive tru trace* between Rt and an *output activity* A.

Rt: an output resource unit which is produced by an output activity A
A: an output activity

L: list denoting the trace relationships; e.g., L="output_ru(Rt),produce(A,Rt)" denotes that A is an output activity which produces the output resource unit Rt
s: an extant or hypothetical situation

In the next section, the competency questions are formally posed using the ontology terminology and answered by deduction using the ontology axioms.

5.5.3 Formal Competency Questions

CQ 5.11 Is this a final product?

- Is κ an *output resource* in a situation σ?: *holds*(output_resource(κ),σ).
  - *holds*(output_resource(cured_L_edge),sv_actual).
- Is κ an *output resource unit* in a situation σ?: *holds*(output_ru(κ),σ).
  - e.g. *holds*(output_ru(tru1_cured_L_edge),sv_actual).

CQ 5.12 Is this raw material?

- Is κ an *input resource* in a situation σ?: *holds*(input_resource(κ),σ).
  - e.g. *holds*(input_resource(L_edge_specs),sv_actual).
- Is κ an *input resource unit* in a situation σ?: *holds*(input_ru(κ),σ).
  - e.g. *holds*(input_ru(tru1_L_edge_specs),sv_actual).

CQ 5.13 Does this activity start processing the raw material?

- Is α an *input activity*?: *holds*(input_activity(κ),σ).
- *holds*(input_activity(load_L_edge_p110_002),sv_actual).

CQ 5.14 Does this activity finish processing the final product?

- Is α an *output activity*?: *holds*(output_activity(κ),σ).
- *holds*(output_activity(autoclave_cure_L_edge),sv_actual).

CQ 5.15 What is the direct material flow path through a given activity; that is, what is consumed and what is produced by this activity?

- What is the *primitive activity trace path* between an unknown *tru* Rt_2 to another unknown *tru* Rt_1 via a given *primitive activity* α in a situation σ? $\exists L \exists Rt_1 \exists Rt_2$ *holds*(prim_activity_trace(α,Rt_2,Rt_1,L),σ).
- $\exists L \exists Rt_1 \exists Rt_2$ *holds*(prim_activity_trace(autoclave_cured_L_edge,Rt_2, Rt_1,L),sv_actual).

CQ 5.16 What is the direct material flow path that involves a given batch; that is, which activity produces this batch, and which activity subsequently consumes it?

- What is the *primitive tru trace path* between an unknown *primitive activity* A_2 to another unknown *primitive activity* A_1 via a given *tru* κ in a situation σ? $\exists L \exists A_1 \exists A_2$ *holds*(prim_tru_trace(κ,A_2,A_1,L),σ).

- $\exists L \exists A_1 \exists A_2$ *holds*(prim_tru_trace(tru1_wip_L_edge_S004,A_2,A_1,L),sv_actual).

CQ 5.17 What is the path of the material flow between two batches?

- What is the *tru trace path* from one *tru* κ_2 to another *tru* κ_1 in a situation σ?$\exists L$ *holds*(tru_trace(κ_2,κ_1,L),σ).
- $\exists L$ *holds*(tru_trace(tru1_cured_L_edge,tru1_v_L_kit_w_psr,L),sv_actual).

CQ 5.18 What is the path of the material flow between two activities?

- What is the *activity trace path* from one *primitive activity* α_2 to another *primitive activity* α_1 in a situation σ?$\exists L$ *holds*(activity_trace(α_2,α_1,L),σ).
- $\exists L$ *holds*(activity_trace(autoclave_cure_L_edge , sup_v_pts_L_edge_004,L),sv_actual).

CQ 5.19 What is the path of the material flow from a final product to one of its raw materials?

- What is the tru *trace path* from an *output resource unit* κ_2 to an unknown *input resource unit* Rt_1 in a situation σ? $\exists Rt_1 \exists L$ [*holds*(output_ru(κ_2),σ) $\wedge$ *holds*(input_ru(Rt_1),σ) $\wedge$ *holds*(tru_trace(κ_2,Rt_1,L),σ)].
- $\exists Rt_1 \exists L$ [*holds*(output_ru(tru_cured_L_edge),sv_actual) $\wedge$ *holds*(input_ru(Rt_1),sv_actual) $\wedge$ *holds*(tru_trace(tru_cured_L_edge,Rt_1,L),sv_actual)].

CQ 5.20 What is the path of the material flow from an activity that finishes processing a final product to an activity that starts processing raw materials?

- What is the *activity trace path* from an *output activity* α_2 to an *input activity* α_1 in a situation σ? $\exists L$ [*holds*(output_activity(α_2),σ) $\wedge$ *holds*(input_activity(α_1),σ) $\wedge$ *holds*(activity_trace(α_2,α_1,L),σ)].
- $\exists L$ [*holds*(output_activity(autoclave_cured_L_edge),sv_actual) $\wedge$ *holds*(input_activity(load_L_edge_p110_002),sv_actual) $\wedge$ *holds*(activity_trace(autoclave_cured_L_edge,load_L_edge_p110_002,L),sv_actual)].

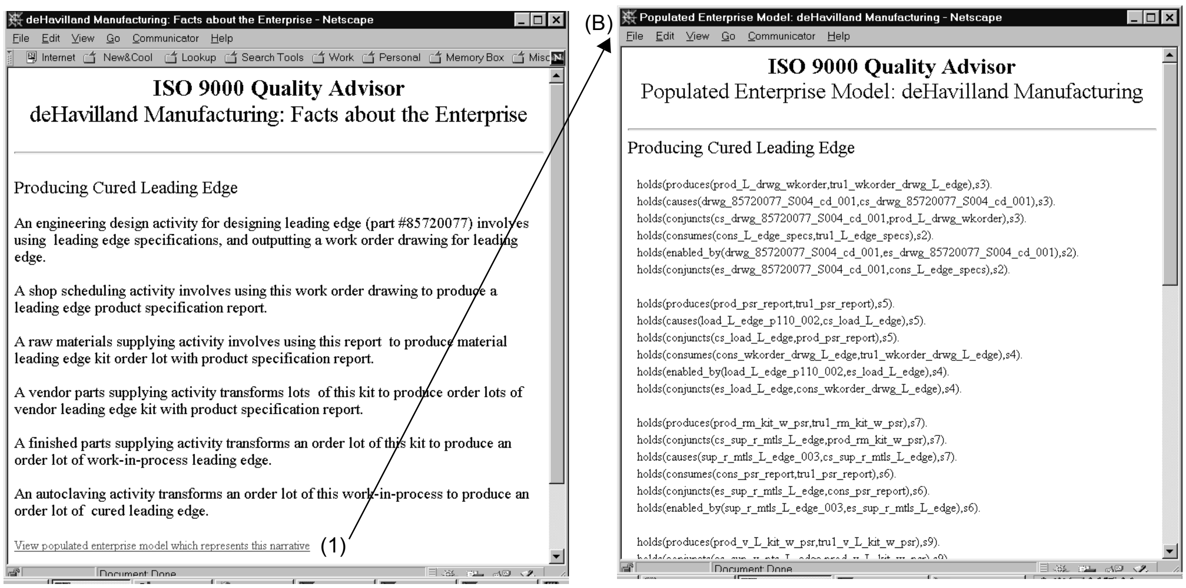
5.6 Demonstration of Competency: Using the ISO 9000 Quality Advisor

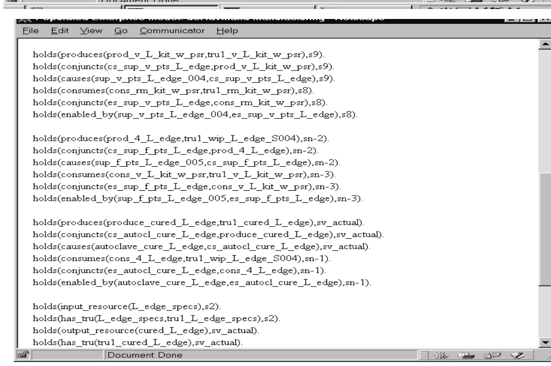
This demonstration shows how the advisor is used for traceability analysis for deHavilland. In so doing, the advisor is used to answer the following competency question.

- **CQ 5.14 What is the path of the material flow from a final product to one of its raw materials?** In order to answer this question, the following question is also answered.
  - **CQ 5.15 What is the direct material flow path through a given activity; that is, what is consumed and what is produced by this activity?**

Step 1: Stating Facts about an Enterprise ⇔ Representing Populated Enterprise Models.

Figure 5.6 Displaying Traceability-Related Facts & Representing them in the Traceability Ontology

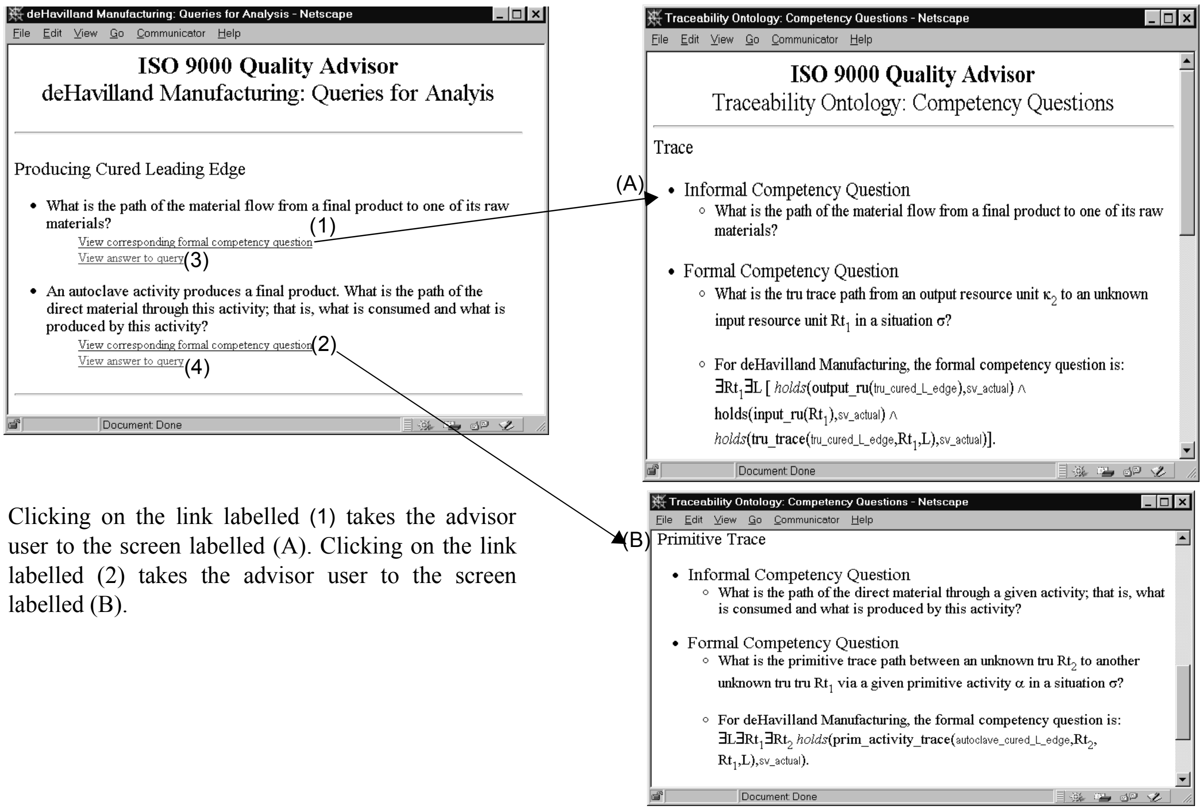




Clicking on the link, labelled (1) in the diagram above takes the advisor user to the screen above, labelled (B).

Step 2: Stating Queries for Analyzing Enterprise ⇔ Representing Formal Competency Questions

Figure 5.7 Displaying Traceability-Related Queries and Representing them as Formal Competency Questions of the Traceability Ontology

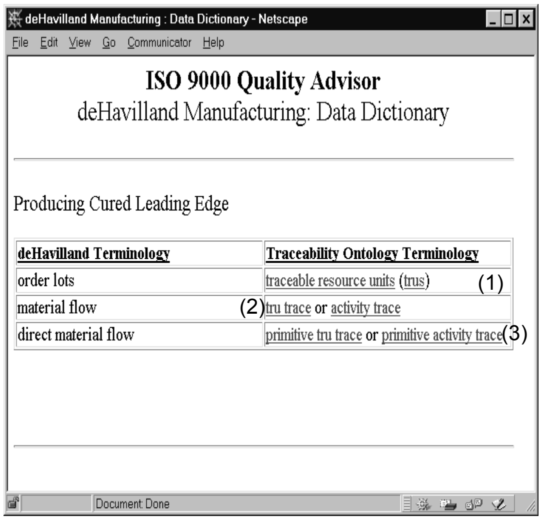


Clicking on the link labelled (1) takes the advisor user to the screen labelled (A). Clicking on the link labelled (2) takes the advisor user to the screen labelled (B).

Step 3: Stating Data Dictionary of Enterprise's Terms ⇔ Representing Ontology or Micro-Theory

Terminology and Axioms

Figure 5.8 **Displaying Data Dictionary of an Enterprise's Traceability-Related Terms**


deHavilland Manufacturing : Data Dictionary - Netscape

File Edit View Go Communicator Help

ISO 9000 Quality Advisor

deHavilland Manufacturing: Data Dictionary

Producing Cured Leading Edge

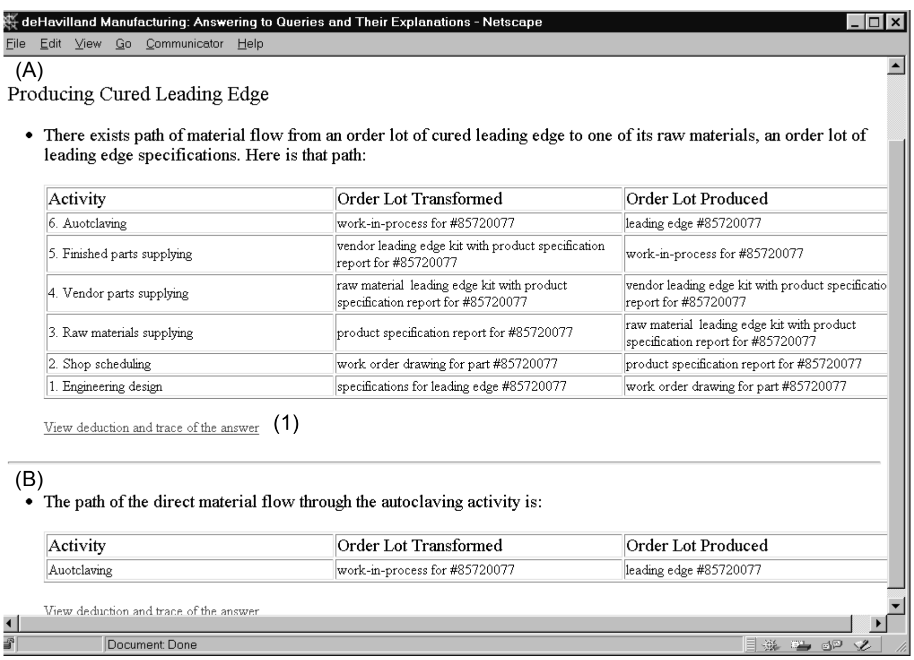
deHavilland Terminology	Traceability Ontology Terminology
order lots	traceable resource units (trus) (1)
material flow (2)	tru trace or activity trace
direct material flow	primitive tru trace or primitive activity trace (3)

Document Done

Clicking on the link, labelled (1), (2), or (3) takes the user to the first-order logic definitions for these terms. The screens corresponding to these definitions are not shown since the definitions have already been presented in this chapter.

Steps 4 and 5: Answering Queries ⇔ Deducing Answers to Formal Competency Questions; and Explaining the Derivation of Answers ⇔ Tracing Deduction and Displaying Prolog Trace List

Figure 5.9 **Displaying Answers to Traceability-Related Queries, and Explanations for Answers**


deHavilland Manufacturing: Answering to Queries and Their Explanations - Netscape

File Edit View Go Communicator Help

(A)

Producing Cured Leading Edge

- There exists path of material flow from an order lot of cured leading edge to one of its raw materials, an order lot of leading edge specifications. Here is that path:

Activity	Order Lot Transformed	Order Lot Produced
6. Auotclaving	work-in-process for #85720077	leading edge #85720077
5. Finished parts supplying	vendor leading edge kit with product specification report for #85720077	work-in-process for #85720077
4. Vendor parts supplying	raw material leading edge kit with product specification report for #85720077	vendor leading edge kit with product specificatio report for #85720077
3. Raw materials supplying	product specification report for #85720077	raw material leading edge kit with product specification report for #85720077
2. Shop scheduling	work order drawing for part #85720077	product specification report for #85720077
1. Engineering design	specifications for leading edge #85720077	work order drawing for part #85720077

View deduction and trace of the answer (1)

(B)

- The path of the direct material flow through the autoclaving activity is:

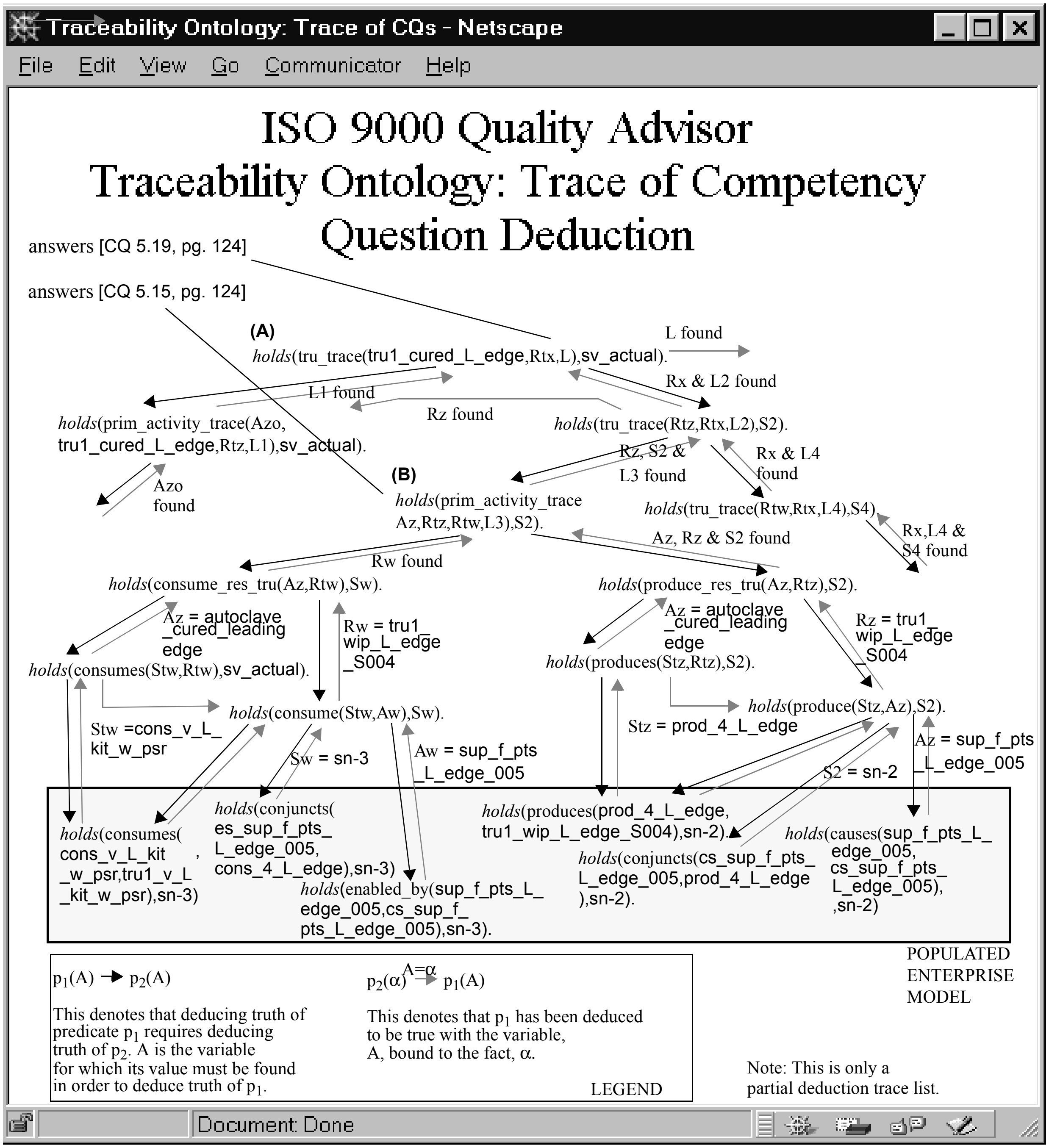
Activity	Order Lot Transformed	Order Lot Produced
Auotclaving	work-in-process for #85720077	leading edge #85720077

View deduction and trace of the answer

Document Done

Clicking on the link, labelled (3) in Figure 5.7 Displaying Traceability-Related Queries and Representing them as Formal Competency Questions of the Traceability Ontology, takes the advisor user to (A) on this screen. Similarly, clicking on (4) in the same figure brings the user to (B).

Figure 5.10 Displaying Competency Question Deduction for Traceability Ontology[1]

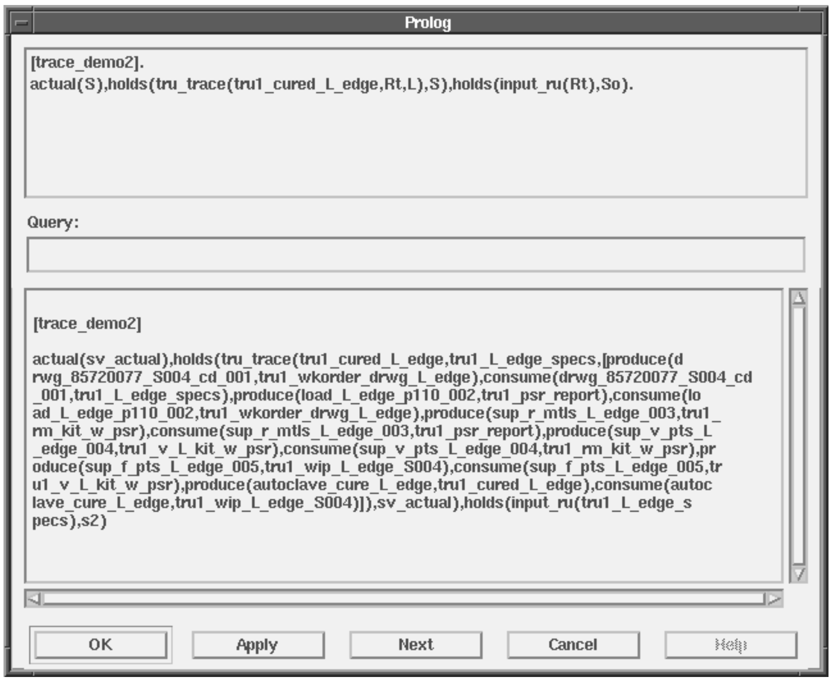


1. Clicking on (1) in Figure 5.9 Displaying Answers to Traceability-Related Queries, and Explanations for Answers takes the advisor user to this screen.

The Prolog query to answer the question denoted by (A) in the diagram took over 800 deductions. Answering question (B) took 30 deductions.

The following is the actual Prolog query screen for the question denoted by (A).

Figure 5.11 Prolog Query to Answer Traceability Ontology Competency Question



5.7 Demonstration of Reducibility

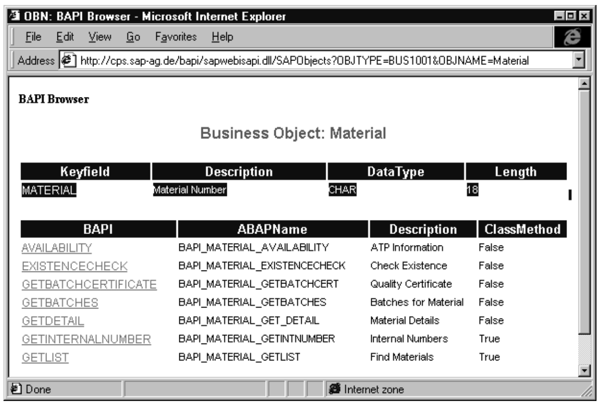
Another of SAP R/3™'s business objects from the Materials Planning module is the Materials object. [Nebraska 98] describes this object; excerpts of this description are given below:

- Different parameters define how activities such as materials planning or quality inspection are carried out for a material.
- Different units of measurement can apply to a material. A conversion factor allowing conversion into the base unit is defined for each such unit.
- An individual piece of a material can be distinguished from others by a serial number.
- Subsets of a material that are manufactured in a particular production run and stocked separately from other subsets of the same material are managed as batches.

- A material is classified by means of the parameters "material type" and "industry sector", which define its possible uses.
- Material groups can be set up, comprising materials with the same attributes.

The following are the Business Application Programmatic Interfaces (BAPI's) related to this object [SAP 98b]:

Figure 5.12 SAP R/3™ Business Object: Material



BAPI Browser

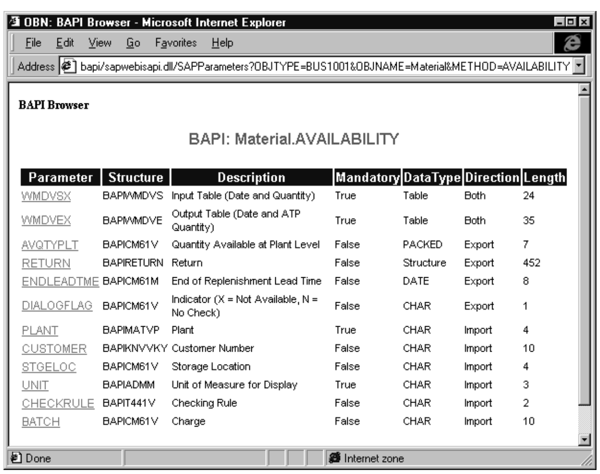
Business Object: Material

Keyfield	Description	DataType	Length
MATERIAL	Material Number	CHAR	18

BAPI	ABAPName	Description	ClassMethod
AVAILABILITY	BAPI_MATERIAL_AVAILABILITY	ATP Information	False
EXISTENCECHECK	BAPI_MATERIAL_EXISTENCECHECK	Check Existence	False
GETBATCHCERTIFICATE	BAPI_MATERIAL_GETBATCHCERT	Quality Certificate	False
GETBATCHES	BAPI_MATERIAL_GETBATCHES	Batches for Material	False
GETDETAIL	BAPI_MATERIAL_GET_DETAIL	Material Details	False
GETINTERNALNUMBER	BAPI_MATERIAL_GETINTNUMBER	Internal Numbers	True
GETLIST	BAPI_MATERIAL_GETLIST	Find Materials	True

Of these BAPI's, AVAILABILITY allows an SAP R/3 user to find out about ATP (Available to Promise) inventory.

Figure 5.13 SAP R/3™ BAPI: Availability



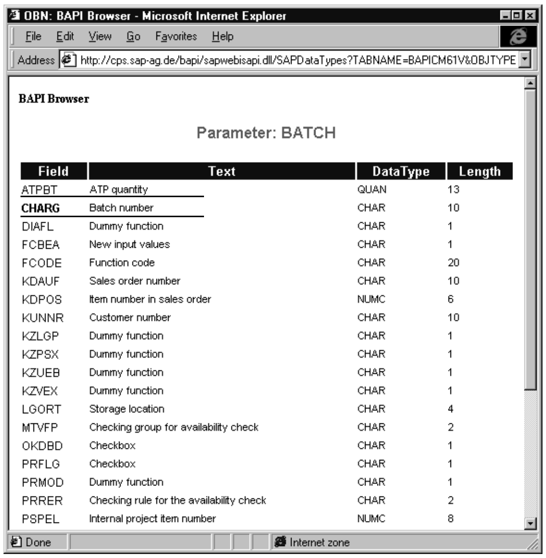
BAPI Browser

BAPI: Material.AVAILABILITY

Parameter	Structure	Description	Mandatory	DataType	Direction	Length
WMDVSX	BAPIWMDVS	Input Table (Date and Quantity)	True	Table	Both	24
WMDVEX	BAPIWMDVE	Output Table (Date and ATP Quantity)	True	Table	Both	35
AVQTYPLT	BAPICM61V	Quantity Available at Plant Level	False	PACKED	Export	7
RETURN	BAPIRETURN	Return	False	Structure	Export	452
ENDLEADTME	BAPICM61M	End of Replenishment Lead Time	False	DATE	Export	8
DIALOGFLAG	BAPICM61V	Indicator (X = Not Available, N = No Check)	False	CHAR	Export	1
PLANT	BAPIMATVP	Plant	True	CHAR	Import	4
CUSTOMER	BAPIKNVVKY	Customer Number	False	CHAR	Import	10
STGELOC	BAPICM61V	Storage Location	False	CHAR	Import	4
UNIT	BAPIADMM	Unit of Measure for Display	True	CHAR	Import	3
CHECKRULE	BAPIT441V	Checking Rule	False	CHAR	Import	2
BATCH	BAPICM61V	Charge	False	CHAR	Import	10

Parameters are the attributes of a business object that are accessible using a BAPI. The following are descriptions of some parameters:

Figure 5.14 Descriptions of Material parameters accessible using Availability BAPI



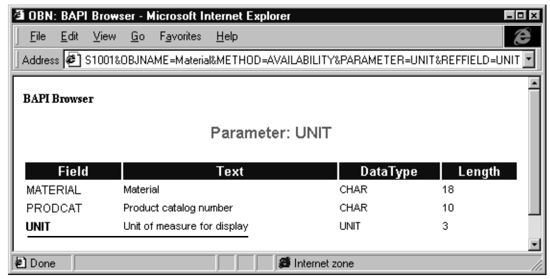
OBN: BAPI Browser - Microsoft Internet Explorer

Address: http://cps.sap-ag.de/bapi/sapwebisapi.dll/SAPDataTypes?TABNAME=BAPICM61V&OBJTYPE

BAPI Browser

Parameter: BATCH

Field	Text	DataType	Length
ATPBT	ATP quantity	QUAN	13
CHARG	Batch number	CHAR	10
DIAFL	Dummy function	CHAR	1
FCBEA	New input values	CHAR	1
FCODE	Function code	CHAR	20
KDAUF	Sales order number	CHAR	10
KDPOS	Item number in sales order	NUMC	6
KUNNR	Customer number	CHAR	10
KZLGP	Dummy function	CHAR	1
KZPSX	Dummy function	CHAR	1
KZUEB	Dummy function	CHAR	1
KZVEX	Dummy function	CHAR	1
LGORT	Storage location	CHAR	4
MTVFP	Checking group for availability check	CHAR	2
OKDBD	Checkbox	CHAR	1
PRFLG	Checkbox	CHAR	1
PRMOD	Dummy function	CHAR	1
PRRER	Checking rule for the availability check	CHAR	2
PSPEL	Internal project item number	NUMC	8



OBN: BAPI Browser - Microsoft Internet Explorer

Address: S1001&OBJNAME=Material&METHOD=AVAILABILITY&PARAMETER=UNIT&REFFIELD=UNIT

BAPI Browser

Parameter: UNIT

Field	Text	DataType	Length
MATERIAL	Material	CHAR	18
PRODCAT	Product catalog number	CHAR	10
UNIT	Unit of measure for display	UNIT	3

* key parameters underlined

A competency question posed by the module user that requires above-mentioned SAPR/3™ representations to answer is: How much of a given material is available right now? What is shown in the screens constitute a partial view of SAP R/3™'s materials management ontology. Some predicates and axioms of the partial data model are stated below so that the competency question can be formally expressed:

material(R): R is a material name at time Tp
batch(Rt): Rt is a batch ID
unit(U): U is a unit of measurement
has_unit(R,U): R is measured in U units of measurement
has_charg(R,Rt,Tp): Rt is a "charg" ID of R at time Tp; i.e., Rt is a batch of R at Tp
has_atp(R,Q_R,Tp): Q_R is the total quantity of R available to promise at time Tp
has_atpbt(Rt,Q_{Rt},Tp): Q_{Rt} is the quantity of the batch Rt available to promise at time Tp

ATP quantity for a material at time Tp is the total ATP quantity for all its batches at that time:

$$\forall R \forall Q_R \forall Tp \exists (Rt_1, Rt_2,,, Rt_n) \exists (Q_{Rt1}, Q_{Rt2},,, Q_{Rtn})$$
$$\left[\cap_i \; \{ \; has\_charg(R, Rt_i, Tp) \wedge has\_atpbt(Rt_i, Q_{Rti}, Tp) \; \} \wedge \right.$$

$$\neg\exists Rt_o \{ \text{has\_charg}(R,Rt_o,Tp) \wedge \cap_i Rt_o \neq Rt_i \} \wedge$$
$$Q_R = \textstyle\sum_{i=1,,n} Q_{Rti} \supset$$
$$\text{has\_atp}(R,Q_R,Tp) \,].$$

Formally, the competency question can be stated as: *What is the ATP quantity for a given material at the time at which the availability request is fulfilled?* Or,

$$\text{(SAP CQ) } \exists Q_R \text{ has\_atp}(\text{material}_{given},Q_R,t_{request}).$$

In order to reduce this question, the following assumption is made: Available to Promise (ATP) refers to quantities that can be committed right now, so it does not refer to quantities that are currently being consumed or in the in the process of being produced.

Next, the following reduction axioms are stated. A key axiom from the situation calculus is used for the reduction: $\forall s\ holds(f,s) \equiv \forall T\ occurs_T(f,T)$; that is, if a fluent holds in a given situation, there is always an accompanying time period in which the fluent occurs.

$$\forall R\ [\ \text{material}(R) \supset \exists T\ occurs_T(\text{resource}(R),T)\ ].$$
$$\forall Rt\ [\ \text{batch}(Rt) \supset \exists T\ occurs_T(\text{tru}(Rt),T)\ ].$$
$$\forall R \forall Rt \forall Tp \exists T\ [\ \text{has\_charg}(R,Rt,Tp) \wedge \text{has\_point}(T,Tp) \supset$$
$$occurs_T(\text{has\_tru}(R,Rt),T)\ ].$$

$$\forall Rt \forall Q_{Rt} \forall Tp \forall U \forall s \exists R \exists T\ [\ \text{has\_unit}(R,U) \wedge \text{has\_atpbt}(Rt,Q_{Rt},Tp) \wedge$$
$$occurs_T(\text{has\_tru}(R,Rt),T) \wedge \text{has\_point}(T,Tp)$$
$$\supset occurs_T(\text{rp\_tru}(Rt,Q_{Rt},Tp,U),T)\ ].$$

The axioms state the following: material is a resource, a batch is a tru, batch/tru is comprised of a material/resource and resource point of a tru/batch is the ATP quantity of that tru/batch at the stamped time at which the availability request is fulfilled, always measured in the same unit of measurement as the material.

Then the competency question can be stated and answered using Traceability Ontology terminology as: *What is the sum of the resource points of all trus that are comprised of a given material at a requested time?* Or, $\exists Q_R \exists T\ occurs_T(\text{rp\_resource}(\text{material}_{given},t_{request},Q_R),T)$.

$$\text{Where, } \forall R \forall Tp \forall Q_{Rt} \forall T \exists U \exists (Rt_1,Rt_2,,,Rt_n) \exists (Q_{Rt1},Q_{Rt2},,,Q_{Rtn})$$
$$\Big[\cap_i \; \{ \; occurs_T(\text{has\_tru}(R,Rt_i),T) \wedge$$
$$occurs_T(\text{rp\_tru}(Rt_i,Q_{Rti},t_{request},U),T) \; \} \wedge$$
$$\neg \exists Rt_o \; \{ \; occurs_T(\text{has\_tru}(\text{material}_{given},Rt_o),T) \wedge$$
$$occurs_T(\text{rp\_tru}(Rt_o,Q_{Rti},t_{request},U),T) \wedge \cap_i \; Rt_o \neq Rt_i \; \} \wedge$$
$$Q_R = \textstyle\sum_{i=1,,n} Q_{Rti} \supset$$
$$occurs_T(\text{rp\_resource}(R,Tp,Q_R),T) \; \Big].$$

Additional exercises to reduce competency questions answerable using the Material business object and its accompanying BAPI's can demonstrate that the Traceability Ontology representations are re-useable for solving similar problems for similar enterprises as the SAP R/3™ materials management module.

5.8 Summary and Conclusion

The following summarize the key concepts that are formalized in the Traceability Ontology:

- Product traceability requires representing homogenous sets of resources, *traceable resource units* (*trus*), where traceability within a *tru* is not possible, and consuming, splitting, or merging different *trus* produces a completely different *tru*. In modelling an enterprise, these representations are necessary in order to guarantee unique identification and traceability of material flow.

- A node in a *trace* path can be represented as the tuplet of a *tru* and the *activity* that *consumes*, *uses*, *releases* or *produces* that *tru*. By stating axioms to link these nodes, subject to conditions on boundary nodes, a *trace path* between two nodes can always be found, if it exists. In modelling an enterprise, these representations are necessary in order to perform a trace of material flow.

These concepts are formalized by posing competency questions, analyzing the domain of traceability, stating assumptions, and developing terminology and axioms. Then, the competency of the ontology and the capability of the ontology to be used to gain insights about an enterprise are demonstrated by automatically deducing answers to competency questions such as:

- *Quantity Tracing*: What is the quantity of a *tru* at a given point in time? When is the *tru* recognized to exist, and what is the quantity of the *tru* at that time? As the quantity of a *tru* decreases over a period of time, what does a plot of this variance look like?
- *Entity Traceability*: Is there a *trace path* from one activity that produces a *tru* to an activity that was performed in the past so that the *tru* could ultimately be produced? What is that path? What is the *trace path* from a given *tru* of a final product to a *tru* of one of its raw materials?

Finally, the demonstration of reducibility demonstrates that the Traceability Ontology spans a subset of the competency of SAP R/3™ materials management module.

The design, analysis, and prototypical implementation of the Traceability Ontology supports the thesis of this dissertation [pg. 7] by:

- Showing that representing traceability capability is important in order to perform quality analysis.
- Representing and enabling unique identification and traceability—the basic capability to diagnose and analyze quality problems—in the enterprise model.
- Representing traceability in the enterprise model, so that ISO 9000 compliance regarding an organization's traceability capability can be objectively prescribed.

6. Quality Management System Ontology

6.1 Précis

An organization's capability to measure and analyze quality (using analysis capabilities such as traceability) is part of its overall quality management system to ensure and improve quality. Hence, for modelling enterprise quality, an ontology of the quality management system is constructed. The focus of this ontology is on the quality-related policies and goals with which one organizational role constrains another. The ontology, however, does not support the assessment of the goodness of policies and goals. It is represented that all roles of an *enterprise*'s quality management system, its *quality system roles*, are planned to satisfy some customers' quality requirements, as stated in a *contract*. The quality management system is modelled as a system for planning and managing *quality objectives*, *quality policy*, and *quality procedures* that constrain the *quality system roles* of the *employees* within an *enterprise*. Important to a quality management system are documents that offer concrete specifications of these policies and goals, and offer evidence of whether policies are followed and goals are satisfied. In the ontology, *quality manuals* document the enterprise's *quality policy*, *quality plans* document the *quality procedures* of activities, and *quality records* document *quality evidence*. In this chapter, one iteration of the ontological engineering methodology applied to develop the Quality Management System Ontology is presented.

6.2 Introduction

To recap, an organization can consistently satisfy its customers if:

- Its system of measurement is carefully designed so that quality problems can be identified and corrected before they get to the customer.
- Proper measurements are taken, appropriate analyses using methods such as traceability capability are performed to address quality problems, and customers are ultimately satisfied with the products they receive.

Thus, measurement systems and traceability capability are important for quality. Just as important are the quality goals and policies of the organization and the people who work to fulfill the goals and follow the policies. In fact, the systems for measurement and traceability only function properly because related goals and policies are appropriately set, and the workers abide by them. In this thesis, the information captured in quality goals and policies, the documentation of this information, and the organizational structure governed by the quality goals and policies are collectively called an organization's quality management system.

Through measurement, quality is assessed; because of measurement and traceability, quality can be analyzed; and because of the quality management system that relates measurement systems and traceability capability to an organization's overall quality goals and policies, quality is consistently delivered to customers. Hence, a Quality Management System Ontology is vital in order to represent and reason about quality and is an integral component of the Ontologies for Quality Modelling.

6.3 Motivating Scenario

The following excerpt describes how BHP Steel currently designs, or wishes to design, its organizational structure for delivering quality to its customers. The key concept that this scenario provides for the development of the Quality Management System Ontology is that *every role of an organization's quality management system exists to satisfy some customers' quality requirements*.

> This check is a part of BHP Steel's initiative to achieve ISO 9001 compliance. The general manager of FPD has appointed a chief quality manager, the main authority for ensuring that FPD achieves compliance. The FPD's main customers are export customers and the coating facilities at Port Kembla. The goals of FPD have been translated into a quality policy, and based upon this, the positions of the people at FPD have been carefully examined to make explicit the bearing of these positions on product quality.

The following excerpt describes the importance that BHP Steel is starting to place upon documenting the various components of the quality management system. The key concept that this scenario provides for the development of the Quality Management System Ontology is that *documentation is the means by which quality roles are concretely defined; it is also the means by which evidence of whether these roles are performed to satisfy customer quality requirements is concretely provided*.

> Another issue for achieving ISO 9000 compliance is the documentation of the revamped quality system, especially since proof of complete documentation is extremely important for compliance.

6.4 Quality Management System Ontology: Quality System Role

The key concept that *every role of an organization's quality management system exists to satisfy customer quality requirements* is the basis for the following principles about representing an organizational quality management system.

- Customer requirements should be represented as the key justification for the design of the quality management system.
- Goals and policies that are developed so that customer requirements can be consistently satisfied should be represented to constrain how people within the quality management system perform their roles.
- It should be represented that a main authority for the design of the quality management system must be identified and empowered. This is the person upon whom the organizational structure of the quality management system is rooted.

The competency questions that characterize what needs to be represented so that an ontology is constructed upon these principles are presented next.

6.4.1 Informal Competency Questions

The questions to ask in order to represent customer quality requirements are:

CQ 6.1 Is this a customer quality requirement?

CQ 6.2 What is the product for which that given requirement exists?

CQ 6.3 For a given enterprise, who is the customer for that given product?

How should the policies and goals that constrain the roles of a quality management system be represented? Also, how do these policies and goals relate to the activities that are performed to fill the roles? For constructing these types of representations, the questions to ask are:

CQ 6.4 Is there a chief policy of the enterprise's quality management system?

CQ 6.5 Is there a goal of the activity that is related to improving quality?

CQ 6.6 Is there a policy of the activity which constrains how the quality goals are to be achieved?

CQ 6.7 Is there evidence from the activity that goals are achieved?

CQ 6.8 Does a given person fill a role within the quality management system?

How should management responsibility, particularly executive management responsibility, within the quality management system be represented? For constructing representations related to management responsibility, the questions to ask are:

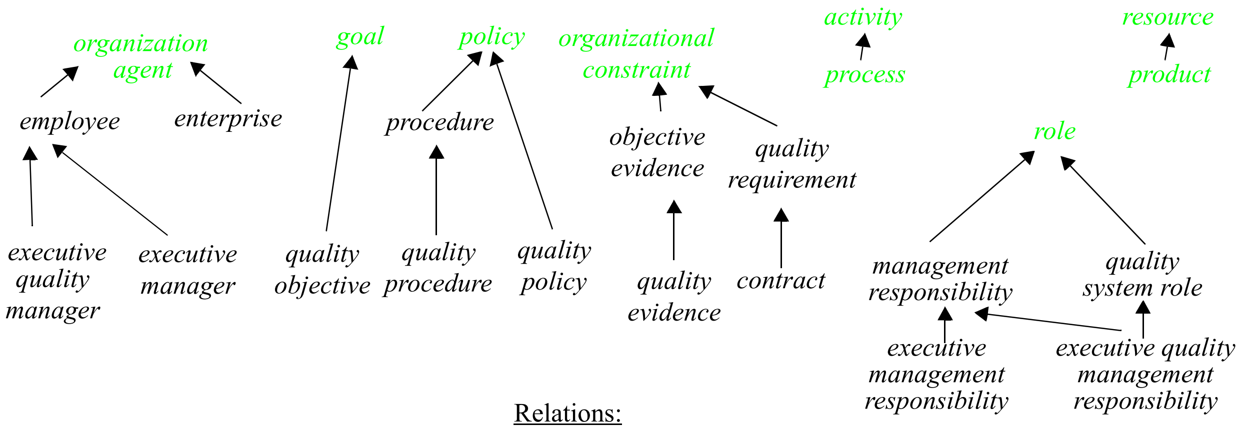
CQ 6.9 Does a given person have quality management responsibility vis-a-vis a given activity?

CQ 6.10 Does a given person have quality management responsibility over the whole quality management system of the enterprise?

In order to answer these competency questions, the domain of quality management is analyzed, assumptions are stated, and terminology and axioms are developed.

6.4.2 Analysis, Terminology, Assumptions, and Axioms

Figure 6.1 Taxonomy Related to Representing Quality System Role



Relations:

product_of(R,O)
contractual_relationship(O,O1,R,Q)
quality_objective_of(G,A)
quality_policy_of(Y,O)
decomposed_from(X1,X)
quality_procedure_of(Y,A)
quality_evidence_of(E,A)

quality_system_role_of(Ro,A)
management_responsibility_of(Ro,A)
quality_management_responsibility_of(Ro,A)
executive_quality_management_responsibility_of(Ro,A)
employee_has_role(M,Ro)
employee_of(M,O)
executive_manager_of(M,O)
executive_quality_manager_of(M,O)

R: resource	Y: organizational policy	
O, O1: enterprise	X,X1: goal, policy, or constraint	
Q: quality requirement	E: objective evidence	
G: organizational goal	Ro: organizational role	
A: activity	M: employee	⟶ is-a

Key Assumption

According to the organization ontology, *policies* and *goals* are *constraints*. A *constraint* has a unique identifier, as well as an expression which evaluates to true or false. *It is assumed that the policies and goals of an enterprise model are represented as unique identifiers, and that no reference is made to the content of the policies and goals. This means that the characterization of the policies and goals is done with respect to the relation of these representations to other ontology representations, rather than through reference to the content of the policies and goals.* So, for example, the ontology defines that a *policy* is an "inspection and testing quality procedure" because it is an *output* from a "define and document" *activity* which *controls* the "document control" *activity*; it does not define an "inspection and testing quality procedure" in terms of what the content of the procedure is. *In general, it is assumed that the ontologies surpass acceptable ontological commitment when its representations introduce bias as to "what is quality?" By representing only that there exist constraints within the enterprise model that state "what is quality?" and by having representations that can evaluate the truth of these constraints, the Ontologies for Quality Modelling can verify that these constraints exist.*

Developing terminology and axioms to formally state and answer:

CQ 6.1 Is this a customer quality requirement?

CQ 6.2 What is the product for which that given requirement exists?

CQ 6.3 For a given enterprise, who is the customer for that given product?

Since ultimately the customer dictates quality, *it is assumed that customer quality requirements for a supplier are decomposed from the needs of the customer*. A *contract* is the explicit representation of the quality requirements between the customer and the supplier. In order to formally represent a *contract*, the term, *enterprise*, is formalized.

QMS Axioms, Defn-1: **enterprise(O)**

An *enterprise* is an *organization agent* which is not a *member of* another *organization agent*.

$$\forall O \forall s \left[\, holds(\boldsymbol{enterprise(O)},s) \equiv holds(\text{organization\_agent}(O),s) \wedge \exists O_1\, holds(\text{has\_member}(O,O_1),s) \wedge \neg\exists O_2\, holds(\text{has\_member}(O_2,O),s) \,\right].$$

O: an enterprise
O_1: an organization agent which is a member of O
s: an extant or hypothetical situation

The products of an enterprise are formalized as:

QMS Axioms, Defn-2: **product_of(R,O)**

An *output resource outputted* from the *enterprise* is a *product of an enterprise*.

$$\forall O \forall R \forall s \left[\, holds(\boldsymbol{product\_of(R,O)},s) \equiv holds(\text{output\_resource}(R),s) \wedge holds(\text{enterprise}(O),s) \wedge \exists A\, (\, holds(\text{process-output}(A,R),s) \wedge holds(\text{descendent-process-organization}(A,O),s)\,) \,\right].$$

R: a product
O: an enterprise which outputs R

A: the activity which is in O that outputs R
s: an extant or hypothetical situation

- *process-output* defines an output to an activity, and *descendent-process-organization* defines an activity that is performed within an organization. These terms are from the activity-process mapping ontology, defined in the appendix.
- *output resource* is a term from the Traceability Ontology.

An *enterprise*'s *products* are provided to the customer. A *contractual relationship* governs this provision.

QMS Term, PT-1: **contractual_relationship(O,O_1,R,Q)**

An *enterprise* O has a contract with another *enterprise* O_1 to provide a *product* R. Q is the unique ID to this *contractual relationship*. This term is a primitive term.

O: an organization which provides a product
O_1: an organization which receives a product from O
R: the product provided from O and received by O_1
Q: unique ID for the contractual relationship between O and O_1 for R
s: an extant or hypothetical situation

QMS Axioms, Defn-3: **contract(Q)**

A *contract* is a *quality requirement* which uniquely identifies a *contractual relationship*, where one *enterprise* provides a *product* to another *enterprise*.

$$\forall Q \forall s \exists O \exists O_1 \exists R \left[\, holds(\mathbf{contract(}\boldsymbol{Q}\mathbf{)},s) \equiv holds(\text{contractual\_relationship}(O,O_1,R,Q),s) \wedge holds(\text{quality\_requirement}(Q),s) \wedge holds(\text{enterprise}(O_1),s) \wedge holds(\text{product\_of}(R,O),s) \right].$$

Q: ID for a contract
O: the supplier in the contractual relationship
O_1: customer in the contractual relationship
R: product provided by O to O_1
s: an extant or hypothetical situation

- *quality requirement* is a term from the Measurement Ontology.

Developing terminology and axioms to formally state and answer:

CQ 6.4 Is there a chief policy of the enterprise's quality management system?

CQ 6.5 Is there a goal of the activity that is related to improving quality?

An *enterprise* should refine its quality management system to meet its *contracts*; that is, the desire to satisfy outstanding *contracts* drives the *enterprise* to define its quality objectives. The term, *quality objective*, is a predicate described as:

QMS Term, PT-2: **quality_objective(G)**

A *quality objective* is an *organizational goal* which is quality-related. This is a primitive term.

G: unique identifier to an organizational goal

In order to relate a *quality objective* to a *contract* for which the objective is defined, the following constraint is stated:

QMS Axioms, Con-1: **A Contract Controls the Definition of any Quality Objective**

For any *quality objective*, the *activity in which it is defined* must be *controlled* by a *contract*.

$$\forall G \forall s \exists A \left[holds(\text{quality\_objective}(G),s) \wedge holds(\text{process-output}(A,G),s) \wedge holds(\text{define\_and\_document}(A),s) \supset \{ holds(\text{process-control}(A,Q),s) \wedge holds(\text{contract}(Q),s) \} \right].$$

G: ID for a quality objective
A: activity that defines and documents G
Q: ID for a contract
s: an extant or hypothetical situation

- the term, *define and document*, is explained and formally defined later in this chapter

By relating an *activity* to its *quality objectives*, the following term can be defined:

QMS Axioms, Defn-4: **quality_objective_of(G,A)**

If one of the *goals (objectives)* of an *activity* is a quality related one, then there is a *quality objective of* relationship between the *goal* (*objective*) and the *activity*

$$\forall G \forall A \forall s \left[holds(\textbf{quality\_objective\_of(G,A)},s) \equiv holds(\text{process-objective}(A,G),s) \wedge holds(\text{quality\_objective}(G),s) \right].$$

G: ID for a quality objective
A: activity for which G is a quality objective
s: an extant or hypothetical situation

- *process-objective* is a term that relates an objective to an activity. This term is from the activity-process mapping ontology.

A *quality objective* for the *activity* "inspect line" may be "the defect rate must be less than 1%."

An enterprise's *quality policy* is not so closely linked with *contracts*. This is because the *quality policy* sets the quality strategy for an *enterprise*, and so, directly attributing it to one or more *contracts* is difficult. The *quality policy* should not change as contracts change, so it should not be closely coupled with *contracts*. From the parlance of quality management literature, the *quality* *policy* is the one policy statement about quality that applies to the whole organization; this is akin to how the mission statement applies to the overall organization.

QMS Term, PT-3: **quality_policy_of(Y,O)**

A special *organizational policy* is called a *quality policy.* This is a primitive term.

Y: unique identifier to a quality policy
O: enterprise for which Y is the quality policy

Just as there is one mission statement for an *enterprise*—and having more than one at a time dilutes its effectiveness—

QMS Axioms, Con-2: **One Quality Policy for an Enterprise**

An *enterprise* must have only one *quality policy.*

$$\forall O \forall s \left[\exists Y \forall Yo\ holds(\text{quality\_policy\_of}(Y,O),s) \wedge holds(\text{enterprise}(O),s)\ holds(\text{quality\_policy\_of}(Yo,O),s) \supset Y=Yo \right].$$

O: an enterprise
Y,Yo: ID of the quality policy of the enterprise O
s: an extant or hypothetical situation

Developing terminology and axioms to formally state and answer:

CQ 6.6 Is there a policy of the activity which constrains how the quality goals are to be achieved?

Policies are decomposable; so, the quality policy decomposes to other policies. Of special note are policies called procedures. This is a predicate, described as:

QMS Term, PT-4: **procedure(Y)**

A *policy* which constrains how an activity is to be performed is called a *procedure*. This is a primitive term. Some, but not all, policies are procedures.

Y: unique identifier to a procedure

In general, goals and organizational constraints (such as contracts) are hierarchically decomposable, as well as policies. This decomposition can be informally defined as:

QMS Term, Pred-1: **decomposed_from(X_1,X)**

A *goal*, *policy* or *organizational constraint* (X_1) is decomposed from another *goal*, *policy*, or *organizational constraint* (X), if X is a *sub-requirement* of a *goal*, *policy*, or *organizational constraint* (Xo), where Xo is decomposed from X or is X itself.

Procedures that constrain how an *activity* is to be performed so as to be consistent with the *quality policy* are called *quality procedures*.

QMS Term, Pred-2: **quality_procedure_of(Y,A)**

A *procedure* (Y) is a *quality procedure* of an *activity* (A) if it is *decomposed from* the *quality policy of the enterprise* in which the *activity* is performed.

A *quality procedure* for the *activity* "inspect line" may be "inspect a random sample of 20 from each batch."

Developing terminology and axioms to formally state and answer:

CQ 6.7 Is there evidence from the activity that goals are achieved?

Quality procedures specify how an *activity* should be done. There also needs to be *evidence* that the *activity* was indeed done right, and that it satisfied its *quality objectives.* The following is the description for the predicate of the term, *objective evidence*.

QMS Term, PT-5: **objective_evidence(E)**

Objective evidence is an *organizational constraint*, where the content of the *objective evidence* is a statement of fulfillment or non-fulfillment of a *goal* or another *constraint*. This term is a primitive term.

E: ID for the objective evidence

Evidence of satisfaction to *quality objectives* is *quality evidence*.

QMS Term, PT-6: **quality_evidence(E)**

This is *objective evidence* that is quality-related. This is a primitive term.

E: ID for the quality evidence

QMS Axioms, Con-3: *Quality evidence* is a subclass of *objective evidence*.

$\forall E \forall s \; holds(\text{quality\_evidence}(E),s) \supset holds(\text{objective\_evidence}(E),s).$

The *quality evidence* of an activity can be informally defined as:

QMS Term, Pred-3: **quality_evidence_of(E,A)**

If *evidence of an activity* (A) is *quality evidence* (E), then it is *quality evidence to that activity.*

If one of the *quality objectives* to an *activity* is that its outputs conform to tolerance specifications, then the *quality evidence* to that *activity* are *conformance points* of *trus produced* from that *activity*.

Developing terminology and axioms to formally state and answer:

CQ 6.8 Does a given person fill a role within the quality management system?

A *quality system role*, then, can be defined as any organizational role which is constrained by *quality objectives* and *quality procedures*:

QMS Axioms, Defn-5: **quality_system_role_of(Ro,A)**

A quality system role for an *activity* has a *quality procedure* as a *policy, quality objective* as an *objective*, and is a *role of the activity*.

$$\forall Ro \forall A \forall s \Big[\; holds(\boldsymbol{quality\_system\_role\_of(Ro,A)},s) \equiv holds(\text{process-responsibility}(A,Ro),s) \wedge \exists Y \; (\; holds(\text{has\_policy}(Ro,Y),s) \wedge holds(\text{quality\_procedure}(Y),s) \;) \wedge \exists G \; (\; holds(\text{has\_goal}(Ro,G),s) \wedge holds(\text{quality\_objective}(G),s) \;) \Big].$$

Ro: a quality system role
Y: ID for a quality policy which constrains Ro
G: ID for a quality objective which is the goal of Ro
A: the activity which is performed to fill Ro
s: an extant or hypothetical situation

- *process-responsibility* is a term from the activity-process mapping ontology and is defined in the appendix; this term relates an organizational role to the activity which is performed to fill that role.

Developing terminology and axioms to formally state and answer:

CQ 6.9 Does a given person have quality management responsibility vis-a-vis a given activity?

CQ 6.10 Does a given person have quality management responsibility over the whole quality management system of the enterprise?

There should be a definition for the roles that manage the quality system: the *quality management responsibility* roles:

QMS Axioms, Defn-6: **management_responsibility_of(Ro,A)**

A management responsibility of an activity is the *role* which *authorizes a goal or policy to another role*, where the activity is performed to fill the role.

$$\forall Ro \forall A \forall s \left[holds(\textbf{management\_responsibility\_of(Ro,A)},s) \equiv \exists I \exists L\ holds(\text{role-authority-source}(Ro,I,L),s) \wedge holds(\text{process-responsibility}(A,Ro),s)\ ) \right].$$

Ro: role as a management responsibility
A: the activity which is performed to fill Ro
I: unique ID for a policy or goal which is sent by Ro
L: unique ID for an authority link for which I is the content
s: an extant or hypothetical situation

- *role-authority-source(Ro,I,L)* is defined as: the role Ro authorizes the organizational goal or policy I, where the ID for this authorization is L. It is composed entirely of terms from the organization ontology. This term is defined in the activity-process mapping ontology.

QMS Axioms, Defn-7: **quality_management_responsibility_of(Ro,A)**

A quality management responsibility of an activity is the *management responsibility* of an *activity* where that *responsibility* is a *quality system role*.

$$\forall Ro \forall A \forall s \left[holds(\textbf{quality\_management\_responsibility\_of(Ro,A)},s) \equiv holds(\text{management\_responsibility\_of}(Ro,A),s) \wedge holds(\text{quality\_system\_role}(Ro),s) \right].$$

Ro: a quality management responsibility role
A: activity which is performed to fill Ro
s: an extant or hypothetical situation

The ultimate management responsibility is for the management of the whole enterprise. Hence, an *executive management responsibility* for the whole *enterprise* is defined as:

QMS Axioms, Defn-8: **executive_management_responsibility_of(Ro,O)**

An executive management responsibility of an enterprise is *management responsibility* that does not *receive any authority links*.

$$\forall Ro \forall O \forall s \exists A \left[holds(\textbf{executive\_management\_responsibility\_of(Ro,O)},s) \equiv holds(\text{management\_responsibility\_of}(Ro,A),s) \wedge \neg\exists I \neg\exists L\ holds(\text{role-authority-info-sink}(Ro,I,L),s) \wedge holds(\text{descendent-process-organization}(A,O),s) \wedge holds(\text{enterprise}(O),s) \right].$$

Ro: an executive management responsibility role

A: activity which is performed to fill Ro
O: enterprise for which Ro is the executive management responsibility
s: an extant or hypothetical situation

- *role-authority-info-sink(Ro,I,L)* is defined as: organizational goal or policy I authorizes an organizational role Ro, where the specific message of this authorization is identified by the information link ID, L. This term is defined in the activity-process mapping ontology, and is defined with terms from the organization ontology.

Similarly, the ultimate responsibility with respect to managing the quality system is the *executive quality management responsibility* for the *enterprise*. This is the *quality management responsibility* which is *authorized* only by the *executive management responsibility*; the *quality procedures* and *quality objectives* that constrain the *executive quality management responsibility* are *authorized* by the *executive management responsibility* and no other role.

QMS Axioms, Defn-9: **executive_quality_management_responsibility_of(Ro,O)**

An executive quality management responsibility is the *management responsibility* that receives its *quality procedures* and *quality objectives* as *authority links* only from the *executive management responsibility*, and no other role.

∀Ro∀O∀s∃A [
holds(***executive_quality_management_responsibility_of(Ro,O)***,s) ≡
holds(management_responsibility_of(Ro,A),s) ∧
holds(quality_system_role_of(Ro,A),s) ∧ *holds*(enterprise(O),s)
¬*holds*(executive_management_responsibility_of(Ro,O),s) ∧
$\exists Ro_1$∀I∀L { *holds*(role-authority-info-sink(Ro,I,L),s) ∧
(*holds*(quality_objective_of(I,A),s) ∨
holds(quality_procedure_of(I,A),s))
⊃ *holds*(role-authority-info-source(Ro_1,I,L),s) ∧
holds(executive_management_responsibility_of(Ro_1,O),s) }].

Ro: executive quality management responsibility role
A: activity which is performed to fulfill Ro
I: ID for a quality objective or quality procedure that Ro receives
L: ID for the authority link which has I as its content
Ro_1: ID for the executive management responsibility that sends I to constrain Ro
s: an extant or hypothetical situation

So, the two people who are the most responsible for managing the quality management system are the *executive manager* and *executive quality manager*. The terms required to define these two terms are:

QMS Term, Pred-4: **employee_has_role(M,Ro)**

An employee (M) fills a role (Ro) if it the employee is an *organization agent* with no *members* which is a member of an *enterprise*, and which fills a *position* which has an associated *role*.

QMS Term, Pred-5: **employee_of(M,O)**

An *organizational agent* (M) is an employee of an enterprise (O) if it is an *employee* which is a *member of* that *enterprise*.

The executive manager and the executive quality manager can then be defined as:

QMS Axioms, Defn-10: **executive_manager_of(M,O)**

An executive manager is an *employee of the enterprise* with *executive management responsibility*.

$$\forall M \forall O \forall s \exists Ro \left[holds(\textbf{executive\_manager\_of(M,O)},s) \equiv holds(\text{employee\_has\_role}(M,Ro),s) \wedge holds(\text{executive\_management\_responsibility\_of}(Ro,O),s) \right].$$

M: an employee
O: an enterprise that M is a member of
Ro: an executive management responsibility role
s: an extant or hypothetical situation

QMS Axioms, Defn-11: **executive_quality_manager_of(M,O)**

An executive quality manager is an *employee of the enterprise* with *executive quality management responsibility.*

$$\forall M \forall O \forall s \exists Ro \left[holds(\textbf{executive\_quality\_manager\_of(M,O)},s) \equiv holds(\text{employee\_has\_role}(M,Ro),s) \wedge holds(\text{executive\_quality\_management\_responsibility\_of}(Ro,O),s) \right].$$

M: an employee
O: an enterprise that M is a member of
Ro: an executive quality management responsibility role
s: an extant or hypothetical situation

In a quality management system, it is important to have accountability. For that, who is in charge must be clearly identified. In order to clearly state that the *executive manager* is solely in charge of the whole *enterprise* and the *executive quality manager* is solely in charge of the quality management system, the following constraints must be stated:

QMS Axioms, Con-4: **Single Executive Manager for an Enterprise**

An *enterprise* only has one *executive manager.*

$$\forall M_1 \forall M_2 \forall s \forall O \left[holds(\text{executive\_manager\_of}(M_1,O),s) \wedge holds(\text{executive\_manager\_of}(M_2,O),s) \supset M_1=M_2 \right].$$

M_1,M_2:executive quality manager

O: the enterprise in which the executive manager is a member
s: an extant or hypothetical situation

QMS Axioms, Defn-12: **Single Executive Quality Manager for an Enterprise**

An *enterprise* only has one *executive quality manager*.

$$\forall M_1 \forall M_2 \forall s \forall O \left[\textit{holds}(\text{executive\_quality\_manager\_of}(M_1,O),s) \wedge \textit{holds}(\text{executive\_quality\_manager\_of}(M_2,O),s) \supset M_1 = M_2 \right].$$

M_1,M_2:executive quality manager
O: the enterprise that M is a member of
s: an extant or hypothetical situation

In the next section, the competency questions are formally posed using the ontology terminology and answered by deduction using the ontology axioms.

6.4.3 Formal Competency Questions

CQ 6.1 Is this a customer quality requirement?

- Is θ the ID for a contract in a situation σ? *holds*(contract(θ),σ).
- *holds*(contract(holden_motors_contract_981),sv_actual).

CQ 6.2 What is the product for which that given requirement exists?

- Is there a product for which θ is the ID for a contract in a situation σ? $\exists R \exists O \exists O_1$ *holds*(contractual_relationship (O,O_1,R,θ),σ).
- $\exists R \exists O \exists O_1$ *holds*(contractual_relationship (O,O_1,R,holden_motors_contract_981),sv_actual).

CQ 6.3 For a given enterprise, who is the customer for that given product?

- What is the enterprise that receives a product ρ from an enterprise Ω as specified in a contract θ in a situation σ? ∃O *holds*(contractual_relationship (Ω,O,ρ,θ),σ).
- ∃O *holds*(contractual_relationship (bhp_steel_1,O,blackform_1,holden_motors_contract_981),sv_actual).

CQ 6.4 Is there a chief policy of the enterprise's quality management system?

- Is there a quality policy for an enterprise Ω in a situation σ? ∃Y *holds*(quality_policy_of (Y,Ω),σ).
- ∃Y *holds*(quality_policy_of (Y,bhp_steel_1),sv_actual).

CQ 6.5 Is there a goal of the activity that is related to improving quality?

- Is there a quality objective for an activity α in a situation σ? ∃G *holds*(quality_objective_of (G,α),σ).
- ∃G *holds*(quality_objective_of (G,process_bhp_steel_1),sv_actual).

CQ 6.6 Is there a policy of the activity which constrains how the quality goals are to be achieved?

- Is there a quality procedure for an activity α in a situation σ? ∃Y *holds*(quality_procedure_of (Y,α),σ).

- ∃Y *holds*(quality_procedure_of (Y,process_wp_hcpf_260_1),sv_actual).

CQ 6.7 Is there evidence from the activity that goals are achieved?

- Is there quality evidence for an activity α in a situation σ? ∃E *holds*(quality_evidence_of(E,α),σ).
- ∃E *holds*(quality_evidence_of(E,process_wp_hcpf_260_1),sv_actual).

CQ 6.8 Does a given person fill a role within the quality management system?

- Is there a quality system role for an employee Γ who performs an activity α in a situation σ? ∃Ro (*holds*(quality_system_role_of (Ro,α),σ) ∧ *holds*(employee_has_role(Γ,Ro),σ)).
- ∃Ro (*holds*(quality_system_role_of (Ro,process_wp_qc_1),sv_actual) ∧ *holds*(employee_has_role(colin_montrose,Ro),sv_actual)).

CQ 6.9 Does a given person have quality management responsibility vis-a-vis a given activity?

- Is there a quality management responsibility for an employee Γ who performs an activity α in a situation σ? ∃Ro (*holds*(quality_management_responsibility_of (Ro,α),σ) ∧ *holds*(employee_has_role(Γ,Ro),σ)).
- ∃Ro (*holds*(quality_management_responsibility_of (Ro,process_wp_qc_1),sv_actual) ∧ *holds*(employee_has_role(colin_montrose,Ro),sv_actual)).

CQ 6.10 Does a given person have quality management responsibility over the whole quality management system of the enterprise?

- Is an employee Γ an executive quality manager for the enterprise Ω in a situation σ? *holds*(executive_quality_manager_of(Γ,Ω),σ).
- *holds*(executive_quality_manager_of(colin_montrose,bhp_steel_1),sv_actual).

6.5 Quality Management System Ontology: Quality System Documentation

The key concept that *documentation is the means by which quality roles are concretely defined and is also the means by which evidence of whether these roles are performed to satisfy customer quality requirements is concretely provided* is the basis for the following principle about modelling the quality management system:

- Quality system documentation should be represented to document the goals, policies, and evidence that relate to quality.

The competency questions that characterize what needs to be represented so that an ontology is constructed upon this principle is presented next.

6.5.1 Informal Competency Questions

The questions to ask to motivate the development of representations regarding quality system documentation are:

CQ 6.11 Is this an activity that defines and documents other activities?

CQ 6.12 Is there a piece of documentation that documents how an activity of the enterprise's quality system is performed?

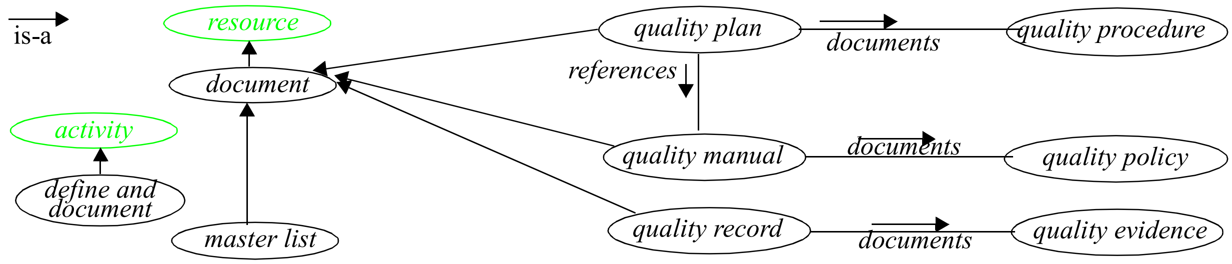
CQ 6.13 Is there a piece of documentation that documents the objective evidence of the fulfillment of the goal of an activity which comprises the enterprise's quality system?

CQ 6.14 Is there a piece of documentation that documents the chief policy of the enterprise's quality system?

In order to answer these competency questions, the domain of quality management is analyzed, assumptions are stated, and terminology and axioms are developed.

6.5.2 Analysis, Assumptions, Terminology, and Axioms

Figure 6.2 Data Model of Ontology Terms related to Quality System Documentation



A *define and document* activity *outputs* a *goal* or a *policy* of a *quality system role of another activity*, and also *outputs documentation* for that *goal* or *policy*:

QMS Axioms, Defn-13: **define_and_document(A)**

$$\forall A \forall s \exists I \left[\, holds(\boldsymbol{define\_and\_document(A)},s) \equiv \right.$$
$$holds(\text{process-output}(A,I),s) \wedge$$
$$\exists D \,(\, holds(\text{documents}(D,I),s) \wedge holds(\text{process-output}(A,D),s)\,) \wedge$$
$$\exists Ro \exists Ao \,(\, holds(\text{quality\_system\_role\_of}(Ro,Ao),s) \wedge$$
$$\left. holds(\text{has\_policy}(Ro,I),s) \vee holds(\text{has\_goal}(Ro,I),s)\,) \,\right].$$

A: a define and document activity
I: ID for a policy or goal defined and documented in A
Ro: a role for which I is a policy or goal defined and documented by A
Ao: an activity that is defined and documented by A
D: document that documents I
s: an extant or hypothetical situation

The outputs of a define and document process are defined this way:

QMS Term, PT-7: **documents(D,I)**

A document *documents goals*, *policies*, and *organizational constraints*. This term is a primitive term.

D: a document resource
I: ID for a goal, policy, or organizational constraint documented by D

QMS Term, PT-8: **references(D,Do)**

A *document references* another *document*.

D: a document that makes a reference to another document within its content
Do: a document referred to by another document

QMS Term, Pred-6: **document*(D)***

A *document* (D) is a *resource* that *documents* other enterprise objects.

QMS Term, PT-9: **master_list(D)**

A *master list* is a *document* that keeps track of the current versions of *documents* of the organization. This is a primitive term.

- *A constraint is that there cannot exist two different master lists at a given point in time for a given enterprise.*

QMS Axioms, Defn-14: **quality_manual_of(D,O)**

A *quality manual* (D) of an *enterprise* (O) *documents* the *quality policy.*

$$\forall D \forall O \forall s \exists Y \left[holds(\textbf{quality\_manual\_of(D,O)},s) \equiv holds(\text{quality\_policy\_of}(Y,O),s) \wedge holds(\text{documents}(D,Y),s) \right].$$

D: a quality manual
O: the enterprise for which D is the quality manual
Y: ID for the quality policy
s: an extant or hypothetical situation

QMS Term, Pred-7: **quality_plan_of(D,A)**

A *quality plan* (D) for an *activity* (A) *documents* the *quality procedure* for that *activity.*

- *All quality plans must be referenced by the quality manual.*

QMS Term, Pred-8: **quality_record(D,A)**

A *quality record* (D) for an *activity* (A) *documents quality evidence* for that *activity.*

In the next section, the competency questions are formally posed using the ontology terminology and answered by deduction using the ontology axioms.

6.5.3 Formal Competency Questions

CQ 6.12 Is this an activity that defines and documents other activities?

- Is α a *define and document* activity in a situation σ? *holds*(define_and_document(α),σ).
- *holds*(define_and_document(def_and_doc_process_wp_hcpf_260_1),sv_actual).

CQ 6.13 Is there a piece of documentation that documents how an activity of the enterprise's quality system is performed?

- Is there a *quality plan of a given activity* α, in a situation σ? ∃D *holds*(quality_plan_of(D,α),σ).
- ∃D *holds*(quality_plan_of(D,wp_hcpf_260_q_plan_1),sv_actual).

CQ 6.14 Is there a piece of documentation that documents the objective evidence of the fulfillment of the goal of an activity which comprises the enterprise's quality system?

- Is there a *quality record of a given activity* α, in a situation σ? ∃D *holds*(quality_record_of(D,α),σ).
- ∃D *holds*(quality_record_of(D,wp_hcpf_260_q_record_1),sv_actual).

CQ 6.15 Is there a piece of documentation that documents the chief policy of the enterprise's quality system?

- Is there a *quality manual for an enterprise* Ω in a situation σ? ∃D *holds*(quality_manual_of(D,Ω),σ).
- ∃D *holds*(quality_manual_of(D,bhp_steel_1),sv_actual).

6.6 Activity-Process Mapping Ontology

There exists a separate ontology comprised of representations that translate core ontology representations into representations which richly characterize a *process*. The basic principles underlying this ontology are:

- *Processes* are *activities*.
- *Inputs*, *outputs*, *controls*, and *mechanisms* of a *process* are: the *resources* and *trus consumed*/*used*/*released*/*produced*; the *organizational goals*, *constraints*, and *policies* that constrain the *organizational roles* for which the *process* is performed; and the *organizational agents* which actually perform the *process*.

Because of this emphasis on the activity-process translation, this ontology is called the activity-process mapping ontology. Although its axioms are used to construct representations of the Quality Management System Ontology and ISO 9000 Micro-Theory, this ontology is not comprised of terms germane to quality. It is a useful ontology for constructing the Ontologies for Quality Modelling, but in and of itself does not warrant full discussion in this chapter. A full discussion of the activity-process mapping ontology is found in Appendix A1.

6.7 Demonstration of Competency: Using the ISO 9000 Quality

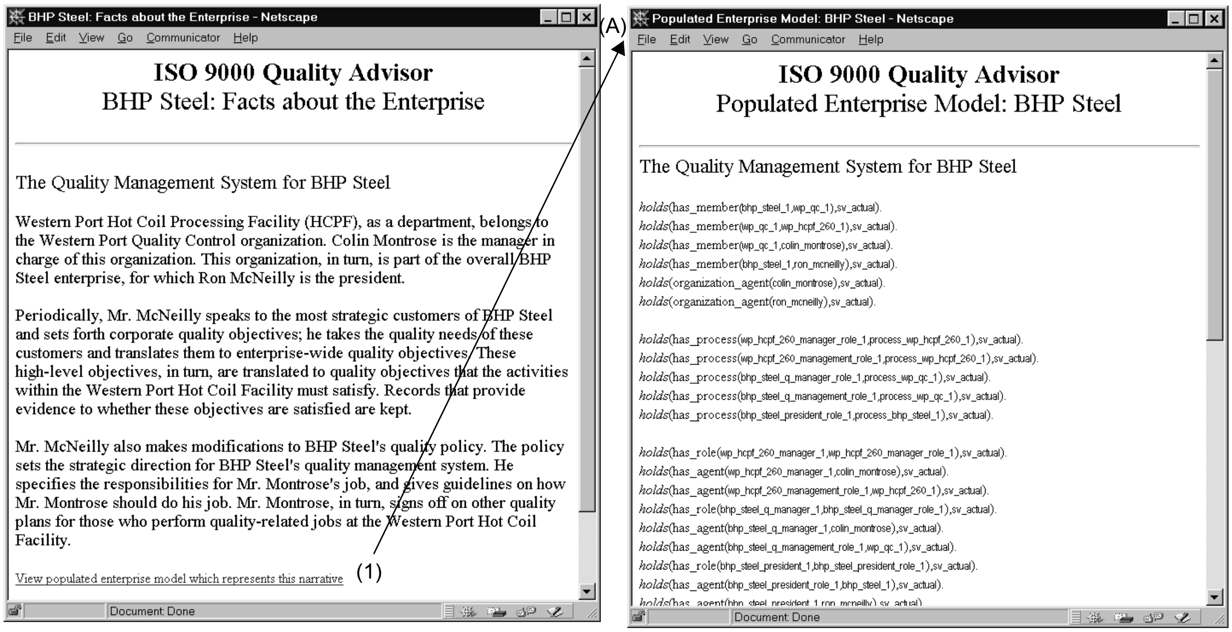
Advisor

This demonstration shows how the advisor is used for analyzing the quality management system at BHP Steel; in so doing, the advisor is used to answer the following competency questions.

- **CQ 6.10 Does a given person have quality management responsibility over the whole quality management system of the enterprise?** In order to answer this question, the following question is also answered.
  - **CQ 6.9 Does a given person have quality management responsibility vis-a-vis a given activity?** In order to answer this question, the following question is also answered:
    - **CQ 6.5 Is there a goal of the activity that is related to improving quality?**

Step 1: Stating Facts about an Enterprise ⇔ Representing Populated Enterprise Models.

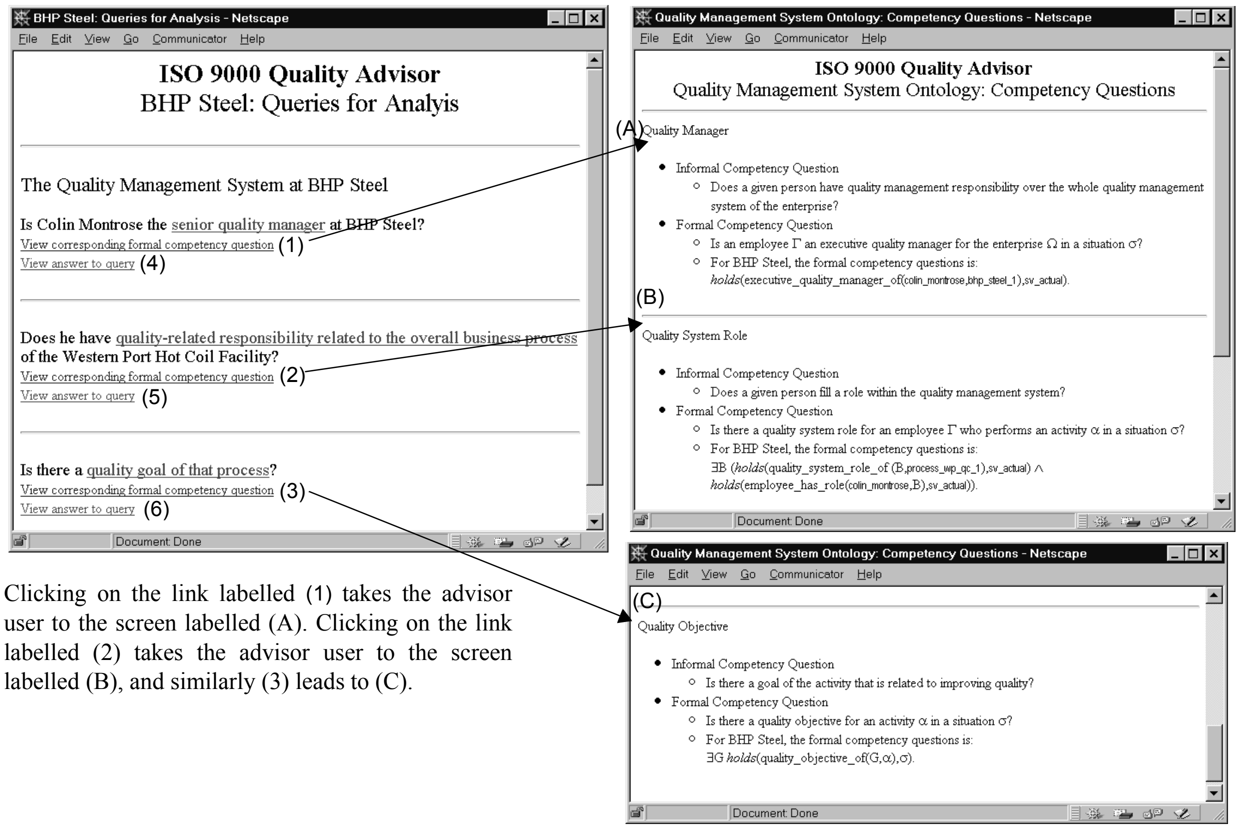
Figure 6.3 **Displaying Quality Management System-Related Facts & Representing them in the Quality Management System Ontology**



Clicking on the link labelled (1) in the diagram above takes the advisor user to the screen above labelled (A).

Step 2: Stating Queries for Analyzing Enterprise ⇔ Representing Formal Competency Questions

Figure 6.4 **Displaying Quality Management System-Related Queries and Representing them as Formal Competency Questions of the Quality Management System Ontology**

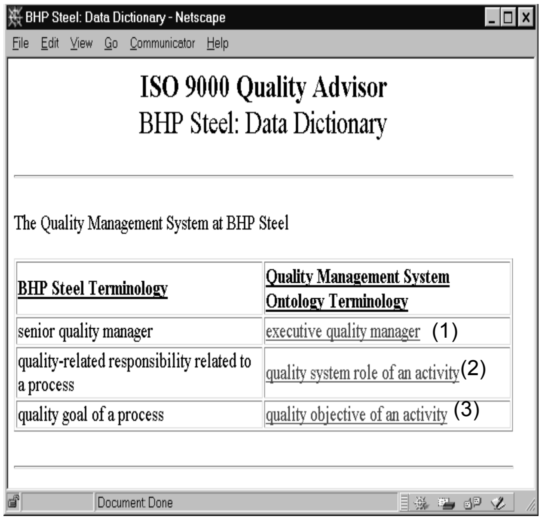


Clicking on the link labelled (1) takes the advisor user to the screen labelled (A). Clicking on the link labelled (2) takes the advisor user to the screen labelled (B), and similarly (3) leads to (C).

Step 3: Stating Data Dictionary of Enterprise's Terms ⇔ Representing Ontology or Micro-Theory

Terminology and Axioms

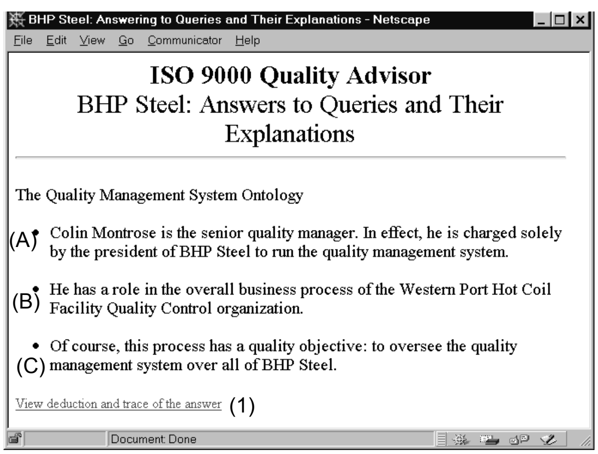
Figure 6.5 Displaying Data Dictionary of an Enterprise's Quality Management System-Related Terms



Clicking on the link labelled (1), (2), or (3) takes the user to the first-order logic definitions for these terms. The screens corresponding to these definitions are not shown since the definitions have already been presented in this chapter.

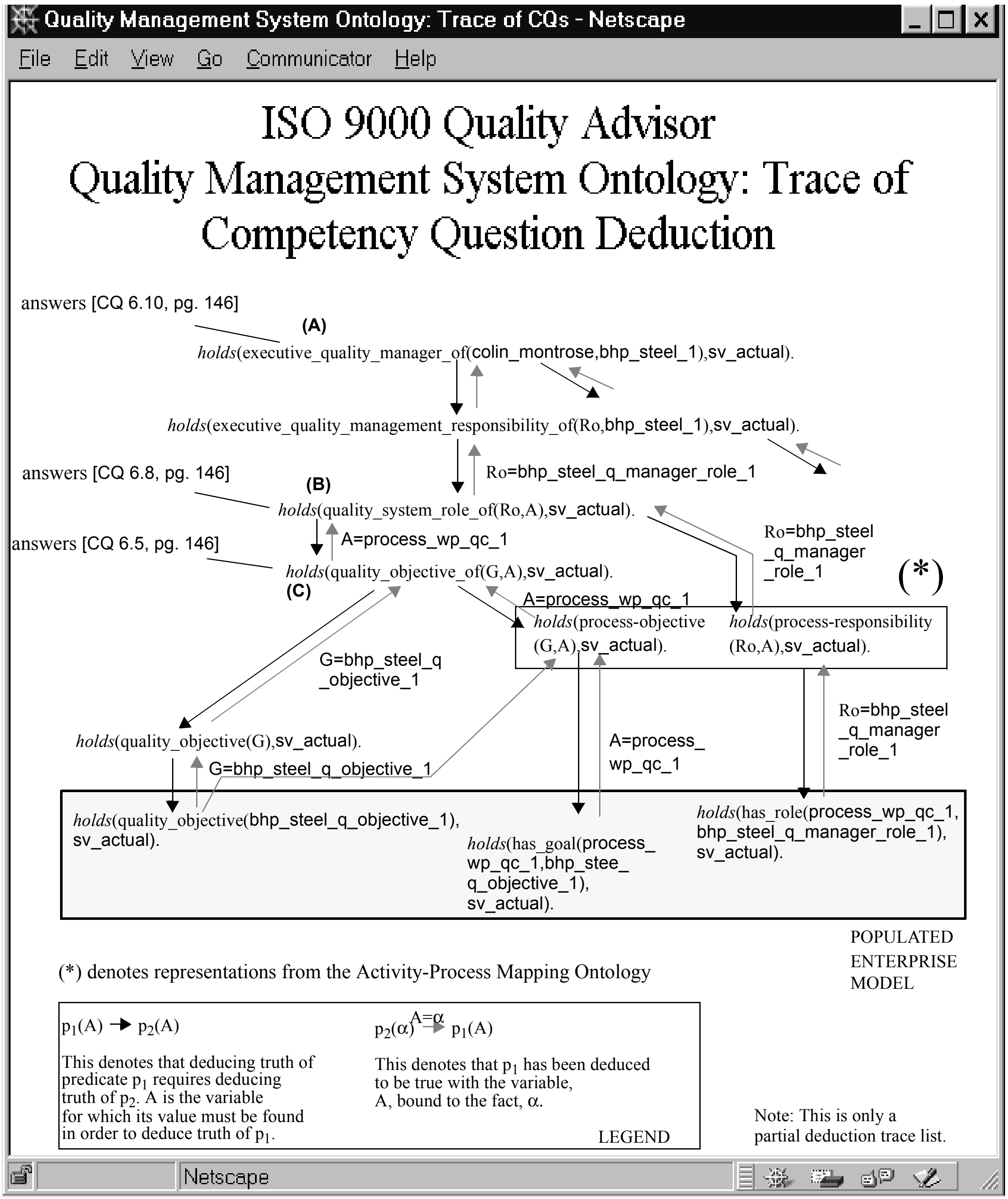
Steps 4 and 5: Answering Queries ⇔ Deducing Answers to Formal Competency Questions; and Explaining the Derivation of Answers ⇔ Tracing Deduction and Displaying Prolog Trace List

Figure 6.6 Displaying Answers to Quality Management System-Related Queries, and Explanations for Answers



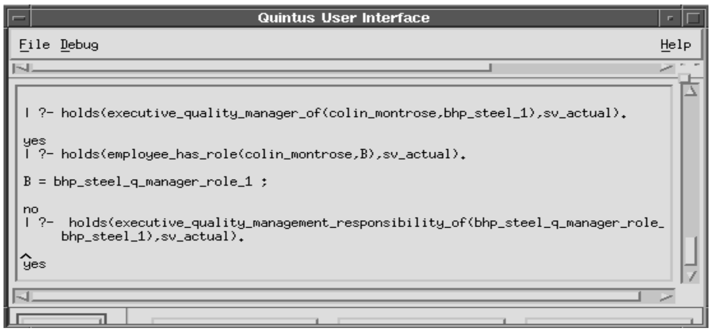
Clicking on the link labelled (4) in Figure 6.4 Displaying Quality Management System-Related Queries and Representing them as Formal Competency Questions of the Quality Management System Ontology, takes the advisor user to (A) on this screen. Similarly, clicking on (5) in the same figure brings the user to (B), and (6) leads to (C).

Figure 6.7 **Displaying Competency Question Deduction for Quality Management System Ontology**[1]



The Prolog query to answer the question denoted by (A) took 178 deductions. Answering (B) and (C) took 14 and 8 deductions, respectively.

Figure 6.8 Prolog Query to Answer Quality Management System Ontology Competency Question



This is an actual Prolog query screen. Below is an excerpt of the deduction trace from the Prolog programming environment.

```
 (2) 0 Call (multifile): holds(executive_quality_manager_of(_10529,bhp_steel_1),_10558)
 (2) 1 Head [10]: holds(executive_quality_manager_of(_10529,bhp_steel_1),_10558)
 (3) 1 Call (multifile): holds(employee_has_role(_10529,_13564),_10558)
 (3) 2 Head [17]: holds(employee_has_role(_10529,_13564),_10558)
 (4) 2 Call (multifile): holds(organization_agent(_10529),_10558)
 (4) 3 Head [66->79]: holds(organization_agent(_10529),_10558)
 (4) 2 Exit (multifile): holds(organization_agent(colin_montrose),sv_actual)
 (5) 2 Call: not(holds(has_member(colin_montrose,_13575),sv_actual))
 (5) 3 Head [1]: not(holds(has_member(colin_montrose,_13575),sv_actual))
 (6) 3 Call (built_in): call(user:holds(has_member(colin_montrose,_13575),sv_actual))
 (7) 4 Call (multifile): holds(has_member(colin_montrose,_13575),sv_actual)
 (7) 5 Head [47->48]: holds(has_member(colin_montrose,_13575),sv_actual)
 (7) 5 Head [48->49]: holds(has_member(colin_montrose,_13575),sv_actual)
 (7) 5 Head [49->50]: holds(has_member(colin_montrose,_13575),sv_actual)
 (7) 5 Head [50]: holds(has_member(colin_montrose,_13575),sv_actual)
 (7) 4 Fail (multifile): holds(has_member(colin_montrose,_13575),sv_actual)
 (6) 3 Fail (built_in): call(user:holds(has_member(colin_montrose,_13575),sv_actual))
 (5) 2 Done: not(holds(has_member(colin_montrose,_13575),sv_actual))

 (177) 7 Fail (multifile): holds(has_member(_14118,bhp_steel_1),sv_actual)
 (176) 6 Fail (built_in): call(user:holds(has_member(_14118,bhp_steel_1),sv_actual))
 (175) 5 Done: not(holds(has_member(_14118,bhp_steel_1),sv_actual))
 (173) 4 Exit (multifile): holds(enterprise(bhp_steel_1),sv_actual)
 (150) 3 Exit (multifile):
holds(executive_management_responsibility_of(bhp_steel_president_role_1,bhp_steel_1),sv_actual
)
 (149) 2 Exit (multifile): holds(executive_management_responsibility(bhp_steel_president_role_1),sv_actual)
 (13) 1 Exit (multifile):
holds(executive_quality_management_responsibility_of(bhp_steel_q_manager_role_1,bhp_steel_1),sv
_actual)
 (2) 0 Exit (multifile): holds(executive_quality_manager_of(colin_montrose,bhp_steel_1),sv_actual)
```

1. Clicking on (1) in Figure 6.6 Displaying Answers to Quality Management System-Related Queries, and Explanations for Answers takes the advisor user to this screen.

6.8 Summary and Conclusion

The following summarizes the key concepts that are formalized in the Quality Management System Ontology:

- A quality management system is modelled to be comprised of an organizational structural system for planning and managing quality-related policies and goals that constrain the roles of people, and a system for disseminating and documenting these policies and goals. In order to assess the capability of the organization to repeatedly satisfy its customers' needs, it is necessary to model an organization using these representations.

These concepts are formalized by posing competency questions, analyzing the quality management system domain, stating assumptions, and developing terminology and axioms. Then, the competency of the ontology and the capability of the ontology to be used to gain insights about an enterprise are demonstrated by automatically deducing answers to competency questions such as:

- *Quality System Role*: Is there a *quality policy* for the company? Does a given person fill an explicit quality-related role in the company? Who is the main person responsible for quality within the company?

- *Quality System Documentation*: Is there a *quality manual* for the company? Are the *quality procedures* for a given activity documented? For a given activity, do records exist that provide documented *evidence* that quality goals are being met?

The design, analysis, and prototypical implementation of the Quality Management System Ontology supports the thesis of this dissertation [pg. 7] by:

- Showing that representing an enterprise's organizational structure and information flows for managing quality, its quality management system, is important for describing and prescribing an enterprise's capability to meet its customers' quality needs.
- Representing an organization's quality goals and policies and their documentation, and the organizational structure governed by these goals and policies.

- Representing quality management system in the enterprise model, so that ISO 9000 compliance regarding an organization's management responsibility, quality system, document control, and control of quality records can be objectively prescribed.

7. ISO 9000 Micro-Theory

7.1 Précis

The ISO 9000 Micro-Theory is a formal model of the ISO 9000 compliance quality perspective of an enterprise, constructed upon generic quality concepts formalized in the Ontologies for Quality Modelling. The micro-theory formalizes those ISO 9000 requirements that are expressible using the Ontologies for Quality Modelling terms. These are requirements related to *inspection and testing*, formalized using Measurement Ontology terms; *product identification and traceability*, formalized using Traceability Ontology terms; and *management of the quality system*, formalized using Quality Management System Ontology terms. The emphasis in formalizing the ISO 9000 *inspection and testing* requirements is upon defining and documenting the proper procedures that dictate how inspection and testing activities should be performed, and on collecting appropriate evidence about the performance of those activities. The micro-theory formalizations that pertain to ISO 9000 requirements upon *product traceability and unique identification* ensure that an organization has the capability to, when need be, identify and trace quality problems. The micro-theory formalizations related to the *management of the quality system* emphasize assessing the organizational commitment to delivering quality products to customers, assessed in terms of allotment of resources and personnel, extensiveness of networks of quality-related activities and information flows, and broad scope of quality-related responsibilities. In this chapter, one iteration of the ontological engineering methodology applied to develop the ISO 9000 Micro-Theory is presented.

7.2 Introduction

Customers seek evidence that their suppliers have systems in place to ensure product quality. Rather than performing an expensive audit on a given supplier, the customer desires ISO 9000 compliance from that supplier. If the supplier demonstrates that it satisfies one of ISO 9001, 9002, or 9003—as audited by a third-party registrar—then the customer is willing to accept this as adequate evidence of the supplier's capability to provide products of good quality. Using the ISO 9000 Micro-Theory, the ISO 9000 requirements can be precisely and unambiguously interpreted throughout an organization. Moreover, this model can be implemented as an information systems model to automatically evaluate the ISO 9000 compliance of an organization.

The ISO 9000 Micro-Theory is an ISO 9000-compliance perspective of quality within the enterprise. It is constructed using the representations of the Ontologies for Quality Modelling. Of these ontologies the Quality Management System Ontology is predominantly used to formalize the following requirements, widely held to be the most important:

- ISO 9001 4.1 Management responsibility
- ISO 9001 4.2 Quality system
- ISO 9001 4.5 Document and data control
- ISO 9001 4.16 Control of quality records

The Traceability Ontology is used to formalize the following requirement:

- ISO 9001 4.8 Product identification and traceability

Finally, the Measurement Ontology is used to formalize the following requirements:

- ISO 9001 4.10 Inspection and testing
- ISO 9001 4.12 Inspection and test status

In the micro-theory, ISO 9001 is formalized, since it is inclusive of the other two standards. Since this micro-theory is a prototype, not all of the ISO 9001's twenty top-level requirements are formalized. Only those that can be formalized using the representations of the Ontologies for Quality Modelling are.

7.3 ISO 9000 Micro-Theory: Inspection and Testing

Requirements

Inspection and testing is at the core of an enterprise's capability to provide quality products to its customers. Inevitably, nonconformities will occur within an enterprise. Inspection and testing is the last means by which these nonconformities are prevented from being delivered to the customers. Cost of dealing with product nonconformities identified by the customers, such as cost of recalls and the intangible cost of losing the customers' future business, is staggering relative to the cost of identifying and fixing problems within the enterprise. Because of this importance and because inspection and testing can be represented using the Measurement Ontology, the ISO 9000 requirements related to inspection and testing are formalized in the ISO 9000 Micro-Theory.

7.3.1 Informal Competency Questions

Complying to ISO 9001 requirement 4.10 Inspection and testing requires complying to its five sub-requirements. So, the informal competency questions are:

- **CQ 7.1** Does the company comply to ISO 9001 requirement 4.10.1 General?
- **CQ 7.2** Does the company comply to ISO 9001 requirement 4.10.2 Receiving inspection and testing?
- **CQ 7.3** Does the company comply to ISO 9001 requirement 4.10.3 In-process inspection and testing?
- **CQ 7.4** Does the company comply to ISO 9001 requirement 4.10.4 Final inspection and testing?
- **CQ 7.5** Does the company comply to ISO 9001 requirement 4.10.5 Inspection and test records?

With the answers to the above questions, the following question can be answered:

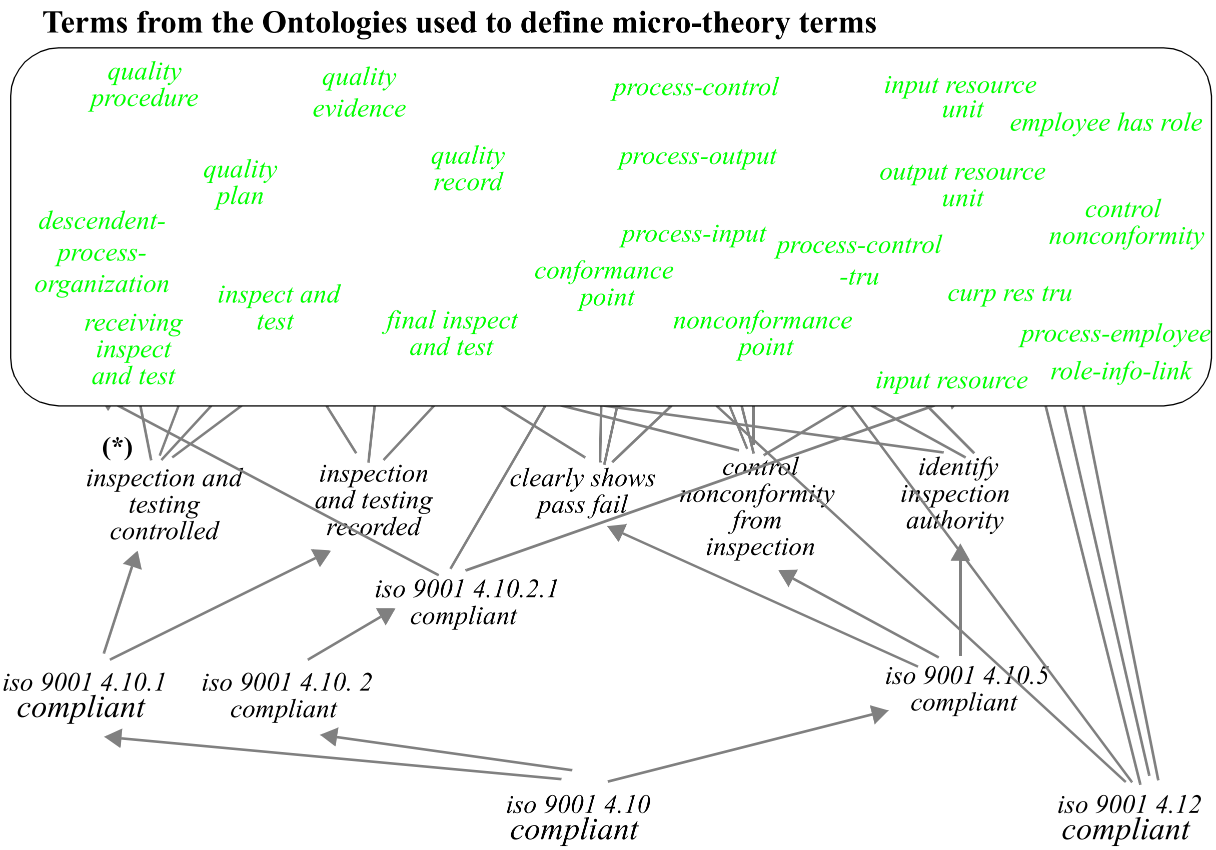
- **CQ 7.6** Does the company satisfy ISO 9001 requirement 4.10 Inspection and testing?

For the requirement upon status of inspection and testing, the question to ask is:

- **CQ 7.7** Does the company satisfy requirement 4.12 Inspection and test status?

7.3.2 Analysis, Assumptions, Terminology, and Axioms

Figure 7.1 Data Model of ISO 9000 Micro-Theory Measurement Terms



The dependency relationship from micro-theory terms to ontology terms are abstracted out for ease-of-reading. So (*) denotes that *inspection and testing controlled* agent constraint is dependent upon 4 terms in the ontologies

all terms except for terms from the ontologies are agent constraints
A → B: A depends upon B; e.g. $B \wedge C \supset A$, where C is another predicate

Compliance to ISO 9001 4.10.1: General

ISO 9001 requirement 4.10.1 states ([ISO 94e], pg. 6):

(1) The supplier shall establish and maintain documented procedures for inspection and testing activities in order to verify that the specified requirements for the product are met.

(2) The required inspection and testing, and the records to be established, shall be detailed in the quality plan or documented procedures.

- The question that characterizes the requirement expressed in (1) is: Does the *enterprise control its inspection and testing*; that is, are all *inspection and testing* activities *controlled* by a *quality procedure* and documentation for the *procedure*, the *quality plan*? This requirement is formalized as:

ISO 9K Axioms, Defn-1: **agent_constraint(O,inspection_and_testing_controlled)**

$$\forall O \forall s \forall A \left[\, holds(\textbf{agent\_constraint(O,inspection\_and\_testing\_controlled)},s) \equiv \right.$$
$$holds(\text{descendent-process-organization}(A,O),s) \wedge$$
$$holds(\text{inspect\_and\_test}(A),s) \supset$$
$$\exists Ra \exists Rb \, \{ \, holds(\text{process-control}(A,Ra),s) \wedge$$
$$holds(\text{process-control}(A,Rb),s) \wedge$$
$$\left. holds(\text{quality\_procedure}(Ra),s) \wedge holds(\text{quality\_plan}(Rb),s) \, \} \right].$$

O: an organization that complies to the ISO 9001 requirement upon control of inspection and testing activity
A: an inspection and testing activity
Ra: an inspection and testing quality procedure
Rb: an inspection and testing quality plan
s: an extant or hypothetical situation

  - *descendent-process-organization* is a term that relates any activity to the organization within which that activity is performed. This term is defined in the activity-process mapping ontology in the Appendix A1.
  - *process-control* is a term that relates an activity to a resource, tru, or information that controls that activity. This term is defined in the activity-process mapping ontology.
  - *inspect and test* is a term from the Measurement Ontology
  - *quality procedure* and *quality plan* are terms from the Quality Management System Ontology.

- The question that characterizes the requirement expressed in (2) is: Does the *enterprise record its inspection and testing*; that is, do all *inspection and testing* activities *output quality evidence*, and a *quality record*, the documentation of this *evidence*? This requirement is formalized as:

ISO 9K Axioms, Defn-2: **agent_constraint(O,inspection_and_testing_recorded)**

$$\forall O \forall s \forall A \left[\, holds(\textbf{agent\_constraint(O,inspection\_and\_testing\_recorded)},s) \equiv \right.$$
$$holds(\text{descendent-process-organization}(A,O),s) \wedge$$
$$holds(\text{inspect\_and\_test}(A),s) \supset$$
$$\exists Ra \exists Rb \{ \, holds(\text{process-output}(A,Ra),s) \wedge holds(\text{process-output}(A,Rb),s) \wedge$$
$$\left. holds(\text{quality\_record}(Ra),s) \wedge holds(\text{quality\_evidence}(Rb),s) \, \} \right].$$

O: an organization that complies to the ISO 9001 requirement upon recording of inspection and testing activity
A: an inspection and testing activity
Ra: an inspection and testing quality record
Rb: an inspection and testing quality evidence
s: an extant or hypothetical situation

  - *quality evidence* and *quality record* are terms from the Quality Management System Ontology.

- *process-output* is a term that relates an activity to a resource, tru, or information that controls that activity. This term is defined in the activity-process mapping ontology.

- If the enterprise *controls its inspection and testing* and *records its inspection and testing*, then the enterprise complies to requirement 4.10.1. This requirement is formalized as:

ISO 9K Axioms, Defn-3: **agent_constraint(O,iso_9001_4.10.1_compliant)**

$$\forall O \forall s \left[holds(\text{agent\_constraint(O,inspection\_and\_testing\_controlled)},s) \wedge holds(\text{agent\_constraint(O,inspection\_and\_testing\_recorded)},s) \supset holds(\textbf{agent\_constraint(O,iso\_9001\_4.10.1\_compliant)},s) \right].$$

O: an organization that complies to ISO 9001 requirement 4.10.1
s: an extant or hypothetical situation

Compliance to ISO 9001 4.10.2: Receiving inspection and testing

ISO 9001 4.10.2 compliance requires compliance to sub-requirements 4.10.2.1 to 4.10.2.3.

ISO 9001 requirement 4.10.2.1 states ([ISO 94e], pg. 6):

(3) The supplier shall ensure that incoming product is not used or processed (except in the circumstances described in 4.10.2.3) until it has been inspected or otherwise verified as conforming to specified requirements.

(4) Verification of conformance to the specified requirements shall be in accordance with the quality plan and/or documented procedures.

- Assumption about the requirement expressed in (3): *Since it is beyond the scope of the ontologies to examine activities that occur at a supplier to the enterprise, it is assumed that receiving inspection and testing is the only means to verify conformance of incoming products.* So, this sub-requirement is complied to if all *input resource units* are *input* into a *receiving inspection and testing* activity. This requirement is formalized as:

ISO 9K Axioms, Defn-4: **agent_constraint(O,iso_9001_4.10.2.1_compliant)**

$$\forall O \forall s \left[\forall A \forall Rt \, [\, holds(\text{descendent-process-organization(A,O)},s) \wedge holds(\text{curp\_res\_tru(A,Rt)},s) \wedge holds(\text{input\_ru(Rt)},s) \supset \exists Ao \{ holds(\text{process-input(Ao,Rt)},s) \wedge holds(\text{receiving\_inspect\_and\_test(Ao)},s) \}] \supset holds(\textbf{agent\_constraint(O,iso\_9001\_4.10.2.1\_compliant)},s) \right].$$

O: an organization that satisfies the ISO 9001 constraint upon the performance of receiving inspection and testing activity
A: any activity of the organization
Ao: a receiving inspection and testing activity
Rt: an input resource unit
s: an extant or hypothetical situation

- *process-input* is a term that relates an activity to a tru or resource that is an input into that activity. This is a term from the activity-process mapping ontology
- *receiving inspect and test* is a term from the Measurement Ontology, but is not discussed in that section for brevity

    - *curp res tru* and *input ru* are terms from the Traceability Ontology.

- Assumption about the requirement expressed in (4): *This requirement is subsumed by the requirement upon verification of all inspection and testing (inspection and testing controlled), so this requirement is not formalized.*

ISO 9001 requirement 4.10.2.2 states ([ISO 94e], pg. 6):

> (5) In determining the amount and nature of receiving inspection, consideration shall be given to the amount of control exercised at the subcontractor's premises and the recorded evidence of conformance provided.

- Assumption about the requirement expressed in (5): *This requirement is subsumed under the assumption, already presented, that activities at the sub-contractors are not examined by the micro-theory. So, the requirement expressed in (5) is not formalized.*

ISO 9001 requirement 4.10.2.3 states ([ISO 94e], pg. 6):

> (6) Where incoming product is released for urgent production purposes prior to verification, it shall be positively identified and recorded in order to permit immediate recall and replacement in the event of nonconformity to specified requirements.

- Assumption about the requirement expressed in (6): *Modelling immediate recall and the release of incoming product for urgent production purposes prior to verification is beyond the scope of this micro-theory.*

- Furthermore, *with the above assumption, this requirement is subsumed by the requirement upon identifying and recording receiving inspection and testing, which in turn, is subsumed by the requirement upon identifying and recording inspection and testing, which has already been formalized. So, the requirement expressed in (6) is not formalized.*

So,

- An *enterprise complies to ISO 9001 requirement 4.10.2*, if it complies to requirement *4.10.2.1*. This requirement can be formalized as:

ISO 9K Axioms, Defn-5: **agent_constraint(O,iso_9001_4.10.2_compliant)**

∀O∀s [*holds*(agent_constraint(O,iso_9001_4.10.2.1_compliant),s) ⊃
holds(***agent_constraint(O,iso_9001_4.10.2_compliant),***s)].

O: an organization that complies to ISO 9001 requirement 4.10.2
s: an extant or hypothetical situation

Compliance to ISO 9001 4.10.3: In-process inspection and testing

The ISO 9001 requirement 4.10.3 states ([ISO 94e], pg. 6):

> (7) The supplier shall inspect and test the product as required by the quality plan and/or documented procedures.
>
> (8) The supplier shall hold product until the required inspection and tests have been completed or necessary reports have been received and verified,
>
> (9) except when product is released under positive-recall procedures. Release under positive-recall procedures shall not preclude the activities outlined in 4.10.3a.

- Assumption about the requirement expressed in (7): *This requirement is subsumed by the requirement on recording all inspection and testing activities. So, the requirement expressed in (7) is not formalized.*
- Assumption about the requirement expressed in (8): *This requirement is subsumed by the requirement upon verification of results as conforming or nonconforming for all inspection and testing activities. The formalization of the latter requirement—called "clearly shows pass or fail"—is a sub-requirement for 4.10.5. So, the requirement expressed in (8) is not formalized.*
- Assumption about the requirement expressed in (9): *Since positive recall is not represented, this requirement is not formalized.*
- So, *Requirement 4.10.3 is not formalized*.

Compliance to ISO 9001 4.10.4: Final inspection and testing

The ISO 9001 requirement 4.10.4 states ([ISO 94e], pg. 7):

> (10)The supplier shall carry out all final inspection and testing in accordance with the quality plan and/or documented procedures to complete the evidence of conformance of the finished product to the specified requirements.
>
> (11)The quality plan and/or documented procedures for final inspection and testing shall require that all specified inspection and tests, including those specified either in receipt of product or in-process, have been carried out and that the results meet specified requirements.
>
> (12)No product shall be dispatched until all the activities specified in the quality plan/or documented procedures have been satisfactorily completed and the associated data and documentation are available and authorized.

- Assumption about the requirement expressed in (10): *This requirement is subsumed by the requirement upon controlling of all inspection and testing. So, the requirement expressed in (10) is not formalized.*

- Assumption about the requirement expressed in (11): *This requirement is subsumed by the requirement upon recording of all inspection and testing. So, the requirement expressed in (11) is not formalized.*

- Assumption about the requirement expressed in (12): *This requirement is subsumed by the requirement upon authorization of all inspection and testing activities. The formalization of this requirement—called "identify inspection authority"—is a sub-requirement for 4.10.5. So, the requirement expressed in (12) is not formalized.*

- So, *Requirement 4.10.4 is not formalized*.

Compliance to ISO 9001 4.10.5: Inspection and test records

The ISO 9001 requirement 4.10.5 states ([ISO 94e], pg. 7):

(13)The supplier shall establish and maintain records which provide evidence that the product has been inspected and/or tested.

(14)These records shall show clearly whether the product has passed or failed the inspections and/or tests according to defined acceptance criteria.

(15)Where the product fails to pass any inspection and/or test, the procedures for control of nonconforming product shall apply.

(16)Records shall identify the inspection authority responsible for the release of product.

- Assumption about the requirement expressed in (13): *This requirement is subsumed by the requirement upon recording of all inspection and testing. So, the requirement expressed in (13) is not formalized.*

- The requirement expressed in (14) can be stated as: An enterprise *clearly shows pass or fail*, if all *trus* that are *output* from an *inspection and testing* activity are marked as either as a *conformance point* or a *nonconformance point*. This requirement is formalized as:

ISO 9K Axioms, Defn-6: **agent_constraint(O,clearly_shows_pass_fail)**

∀O∀s [*holds*(***agent_constraint(O,clearly_shows_pass_fail),***s) ≡
holds(agent_constraint(O,inspection_and_testing_recorded),s) ∧
∀A∀Rt [*holds*(descendent-process-organization(A,O),s) ∧
holds(process-output-tru(A,Rt),s) ∧ *holds*(inspect_and_test(A),s) ⊃
∃X∃At∃Tp (*holds*(conformance_pt(X,Rt,At,Tp),s) ∨
holds(nonconformance_pt(X,Rt,At,Tp),s))]].

O: an enterprise that complies to the requirement upon clearly showing pass or fail in inspection and testing
A: an inspection and testing activity
Rt: an inspected tru
X: the ID for a conformance or nonconformance point
At: a measured attribute of Rt

Tp: the time point of measurement
s: an extant or hypothetical situation

- *process-output-tru* is a term that relates an activity to a tru that is an output from an activity. This is a term from the activity-process mapping ontology
- *conformance* and *nonconformance point* are terms from the Measurement Ontology

- The requirement expressed in (15) can be stated as: An enterprise *controls nonconformities of inspection*, if all *nonconformities* are *input* into a *control nonconformity* activity. This requirement is formalized as:

ISO 9K Axioms, Defn-7: **agent_constraint(O,control_nonconformity_from_inspection)**

∀O∀s [*holds*(***agent_constraint(O,control_nonconformity_from_inspection),***s)
≡
∀A∀Rt [*holds*(descendent-process-organization(A,O),s) ∧
holds(process-output-tru(A,Rt),s) ∧ *holds*(inspect_and_test(A),s) ∧
∃X∃At∃Tp *holds*(nonconformance_pt(X,At,Rt,Tp),s) ⊃
∃Ao (*holds*(process-input(Ao,Rt),s) ∧
holds(control_nonconformity(Ao),s))]].

O: an enterprise that satisfies the constraint upon inspection and testing nonconformity being controlled.
A: an inspection and testing activity
Rt: an inspected and tested tru
Ao: a control nonconformity activity
X: a nonconformance point ID
At: a measured attribute of Rt
Tp: the time point at which Rt is measured
s: an extant or hypothetical situation

- *control nonconformity* is a primitive term that classifies an activity as an activity that controls nonconformities. This is a term from the activity-process mapping ontology

- The requirement expressed in (16) can be stated as: An *enterprise identifies its inspection authority*, if all *trus* that are *output* from an *inspection and testing* activity have *evidence* of its *conformity authorized by* an *employee related to the activity*.

ISO 9K Axioms, Defn-8: **agent_constraint(O,identify_inspection_authority)**

∀O∀s [*holds*(***agent_constraint(O,identify_inspection_authority),***s) ≡
∀A∀Rt [*holds*(descendent-process-organization(A,O),s) ∧
holds(process-output-tru(A,Rt),s) ∧ *holds*(inspect_and_test(A),s) ⊃
∃M∃Ro∃L∃I (*holds*(process-employee(A,M),s) ∧
holds(employee_has_role(M,Ro),s) ∧ *holds*(role-info-link(Ro,L,I),s) ∧
holds(can_authorize(L,Rt),s))]].

O: an enterprise that complies to the requirement upon inspection and testing output being authorized
A: an inspection and testing activity
Rt: an inspected and tested tru
M: the employee who authorizes the inspection and testing of Rt
Ro: the role of M in inspecting and testing Rt

L: the authority link for inspecting and testing Rt
I: the policy, goal, or constraint that authorizes the inspecting and testing of Rt
s: an extant or hypothetical situation

- *process-employee* is a term that relates an employee to the activity where the employee works. This is a term from the activity-process mapping ontology.
- *role-info-link* is a term that relates a communication or authority link to the role that sent or received it. This is a term from the activity-process mapping ontology.
- *employee has role* relates an employee to the role that he/she fills. This is a term from the quality management system ontology.
- *can authorize* is a term from the Core Ontologies explained in the methodology chapter.

- So, an enterprise complies to requirement 4.10.5 if: it satisfies requirements upon *clearly showing inspection pass or fail*, *controlling nonconformities of inspection*, and *identification of all its inspection authorities*. This requirement is formalized as:

ISO 9K Axioms, Defn-9: **agent_constraint(O,iso_9001_4.10.5_compliant)**

∀O∀s [holds(agent_constraint(O,clearly_shows_pass_fail),s) ∧
holds(agent_constraint(O,control_nonconformity_from_inspection),s) ∧
holds(agent_constraint(O,identify_inspection_authority),s)
⊃ *holds*(***agent_constraint(O,iso_9001_4.10.5_compliant),***s)].

O: an organization which complies to ISO 9001 requirement 4.10.5
s: an extant or hypothetical situation

Compliance to ISO 9001 4.10 Inspection and testing

According to the micro-theory, an enterprise complies to ISO 9001 4.10 if it complies to requirements *4.10.1*, *4.10.2*, and *4.10.5*. This requirement is formalized as:

ISO 9K Axioms, Defn-10: **agent_constraint(O,iso_9001_4.10_compliant)**

∀O∀s [*holds*(agent_constraint(O,iso_9001_4.10.1_compliant),s) ∧
holds(agent_constraint(O,iso_9001_4.10.2_compliant),s) ∧
holds(agent_constraint(O,iso_9001_4.10.5_compliant),s)
⊃ *holds*(***agent_constraint(O,iso_9001_4.10_compliant),***s)].

O: an organization which complies to ISO 9001 requirement 4.10
s: an extant or hypothetical situation

Compliance to ISO 9001 4.12 Inspection and test status

The ISO 9001 requirement 4.12 states ([ISO 94e], pg. 8):

(17)The inspection and test status of a product shall be identified by suitable means, which indicate the conformance or nonconformance of product with regard to inspection and tests performed.

(18)The identification of inspection and test status shall be maintained, as defined in the quality plan and/or documented procedures, throughout production, installation, and servicing of the product to ensure that only product that has passed the required inspections and tests [or released under an authorized concession] is dispatched, used, or installed.

- The requirement expressed (17) can be stated as: all *trus* must be identified as either a *conformity* or a *nonconformity* at the end of an *inspection and testing activity*. This requirement is formalized as:

ISO 9K Axioms, Defn-11: **agent_constraint(O,iso_9001_4.12_compliant)**

$\forall O \forall s$ [*holds*(***agent_constraint(O,iso_9001_4.12_compliant),***s) ≡
∀A∀Rt∀T∀Tp [*holds*(descendent-process-organization(A,O),s) ∧
(*holds*(inspect_and_test(A),s) ∧ *holds*(process-output(A,Rt),s) ∧
holds(tru(Rt),s) ∧
$occurs_T$(activity_duration(A,T)) ∧ end_point(T,Tp),s) ⊃
∃X∃At (*holds*(conformance_pt(X,Rt,At,Tp),s) ∨
holds(nonconformance_pt(X,Rt,At,Tp),s))]].

O: an enterprise that satisfies ISO 9001 requirement 4.12
A: an inspect and test activity of O
Rt: a tru
X: the ID for a conformance or nonconformance point
At: a measured attribute of Rt
Tp: the time point of measurement
s: an extant or hypothetical situation

- Assumption about requirement expressed in (18): *It is assumed that the maintenance of the identification of inspection and test status throughout production, installation, and servicing is implied by the previous requirement. So this requirement is not formalized.*

7.3.3 Formal Competency Questions

CQ 7.1 Does the company comply to ISO 9001 requirement 4.10.1 General?

- Is an organization ε compliant to ISO 9001 requirement 4.10.1 in a situation σ?: *holds*(agent_constraint(ε,iso_9001_4.10.1_compliant),σ).
- *holds*(agent_constraint(bhp_steel_1,iso_9001_4.10.1_compliant),sv_actual).

CQ 7.2 Does the company comply to ISO 9001 requirement 4.10.2 Receiving inspection and testing?

- Is an organization ε compliant to ISO 9001 requirement 4.10.2 in a situation σ? *holds*(agent_constraint(ε,iso_9001_4.10.2_compliant),σ).
- *holds*(agent_constraint(bhp_steel_1,iso_9001_4.10.2_compliant),sv_actual).

CQ 7.3 Does the company comply to ISO 9001 requirement 4.10.3 In-process inspection and testing?

- Is not represented in the micro-theory.

CQ 7.4 Does the company comply to ISO 9001 requirement 4.10.4 Final inspection and testing?

- Is not represented in the micro-theory.

CQ 7.5 Does the company comply to ISO 9001 requirement 4.10.5 Inspection and test records?

- Is an organization ε compliant to ISO 9001 requirement 4.10.5 in a situation σ? *holds*(agent_constraint(ε,iso_9001_4.10.5_compliant),σ).
- *holds*(agent_constraint(bhp_steel_1,iso_9001_4.10.5_compliant),sv_actual).

CQ 7.6 Does the company satisfy ISO 9001 requirement 4.10 Inspection and testing?

- Is an organization ε compliant to ISO 9001 requirement 4.10 in a situation σ? *holds*(agent_constraint(ε,iso_9001_4.10_compliant),σ).
- *holds*(agent_constraint(bhp_steel_1,iso_9001_4.10compliant),sv_actual).

CQ 7.7 Does the company satisfy requirement 4.12 Inspection and test status?

- Is an organization ε compliant to ISO 9001 requirement 4.12 in a situation σ? *holds*(agent_constraint(ε,iso_9001_4.12_compliant),σ).
- *holds*(agent_constraint(bhp_steel_1,iso_9001_4.12_compliant),sv_actual).

7.4 ISO 9000 Micro-Theory: Product Identification and Traceability Requirement

Compliance to the ISO 9000 requirement related to product identification and traceability gives confidence to an organization's customers that the organization has an adequate system to:

- identify and locate products in various stages of production throughout the enterprise
- if there is a nonconformity, trace back to the cause of the nonconformity.

Because of this importance, and because product identification and traceability can be modelled by the representations of the Traceability Ontology, this ISO 9000 requirement is modelled in the ISO 9000 Micro-Theory.

7.4.1 Informal Competency Questions

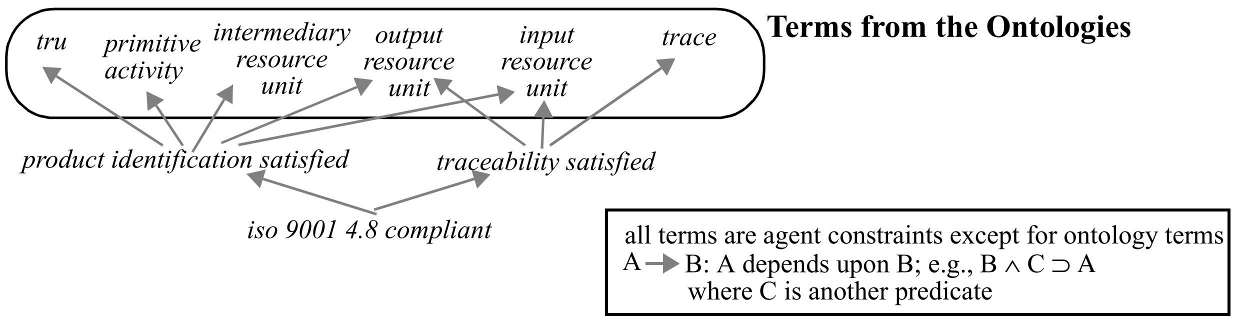
CQ 7.8 Does the enterprise comply to the ISO 9001 requirement upon product identification and classification within the enterprise?

CQ 7.9 Does the enterprise comply to the ISO 9001 requirement upon having product traceability capability within the enterprise?

CQ 7.10 Does the enterprise comply to ISO 9001 requirement 4.8 Product identification and traceability?

7.4.2 Analysis, Terminology, Assumptions, and Axioms

Figure 7.2 Data Model ISO 9000 Micro-Theory Traceability Terms



ISO 9K Terms, Pred-1: **agent_constraint(O,product_identification_satisfied)**

This is a requirement that all the *trus* of a *primitive activity* of an enterprise be identified as an *input*, *output*, or *intermediary resource unit.*

- The term *intermediary resource unit* is defined in the Traceability Ontology, but not shown in the Traceability Ontology chapter.

Another requirement is that any given *tru* must be *traceable* back to at least one *input resource unit*, and it must also be *traceable* forward to at least one *output resource unit*. If this requirement is not complied to, then this demonstrates that there exists a *tru* that should, but cannot, be traced. This is formalized as:

ISO 9K Axioms, Defn-12: **agent_constraint(O,traceability_satisfied)**

$$\forall O \forall A \forall Rt \forall s \exists Aa \exists Ab \exists Rta \exists Rtb \exists s_o \exists s' \,[$$
$$holds(\textbf{agent\_constraint(O,traceability\_satisfied)}, s) \equiv$$
$$holds(\text{descendent-process-organization}(A,O),s) \wedge$$
$$holds(\text{curp\_res\_tru}(A,Rt),s) \wedge holds(\text{tru}(Rt),s) \wedge$$
$$holds(\text{descendent-process-organization}(Aa,O),s') \wedge$$
$$holds(\text{produce\_res\_tru}(Aa,Rta),s') \wedge holds(\text{output\_ru}(Rta),s') \wedge$$
$$holds(\text{descendent-process-organization}(Ab,O),s_o) \wedge$$
$$holds(\text{consume\_res\_tru}(Ab,Rtb),s_o) \wedge holds(\text{input\_ru}(Rtb),s_o) \wedge s_o < s \wedge s < s'$$
$$\supset \{ Rt \neq Rta \supset \exists L_1 \; holds(\text{tru\_trace}(Rta,Rt,L_1),s') \} \wedge$$
$$\{ Rt \neq Rtb \supset \exists L_2 \; holds(\text{tru\_trace}(Rt,Rtb,L_2),s_o) \}].$$

O: an enterprise that complies to the ISO 9001 requirement upon satisfactory traceability
Rta: an output resource unit
Rtb: an input resource unit
Rta_o: an output resource unit that was produced because of the consumption of Rtb
Rtb_o: an input resource unit that was consumed to ultimately produce Rta

Aa: the activity that produces Rta
Ab: the activity that consumes Rtb
L_1: trace list from Rta to Rtb_o
L_2: trace list from Rta_o to Rtb
s,s′: extant or hypothetical situations
s_o: an extant or hypothetical situation that occurs before s

- The terms *tru trace, produce res tru, and consume res tru* are defined in the Traceability Ontology

ISO 9K Terms, Pred-2: **agent_constraint(O,iso_9001_4.8_compliant)**

An enterprise complies to the requirement 4.8 if it complies to the *production identification* and *traceability* requirements.

7.4.3 Formal Competency Questions

CQ 7.8 Does the enterprise comply to the ISO 9001 requirement upon product identification and classification within the enterprise?

- Does an enterprise ε comply to the ISO 9001 requirement upon product identification in a situation σ?: *holds*(agent_constraint(ε,product_identification_satisfied),σ)].
- *holds*(agent_constraint(deh_1,product_identification_satisfied),sv_actual)

CQ 7.9 Does the enterprise comply to the ISO 9001 requirement upon having product traceability capability within the enterprise?

- Does an enterprise ε comply to the ISO 9001 requirement upon traceability in a situation σ?: *holds*(agent_constraint(ε,traceability_satisfied),σ)].
- *holds*(agent_constraint(deh_1,traceability_satisfied),sv_actual)

CQ 7.10 Does the enterprise comply to ISO 9001 requirement 4.8 Product identification and traceability?

- Does an enterprise ε comply to ISO 9001 requirement 4.8 in a situation σ?: *holds*(agent_constraint(ε,iso_9001_4.8_compliant),σ)].
- *holds*(agent_constraint(deh_1,iso_9001_4.8_compliant),sv_actual)

7.5 ISO 9000 Micro-Theory: Management of Quality System Requirements

The central tenets of a quality management system are that:

- roles of workers of the quality management system are planned;
- the expectations of the roles are documented and disseminated; and
- all workers of the quality management system execute their roles.

ISO 9001 requirements 4.1 Management responsibility and 4.2 Quality system address quality system planning. Requirements 4.5 Document and data control and 4.16 Control of quality records address quality system documentation and dissemination. Documentation also provides evidence of the proper execution of roles. Compliance to these requirements gives confidence to the customers that the system of ensuring quality is planned, disseminated, and properly executed. Because of this importance, and because these requirements can be modelled by the representations of the Quality Management System Ontology, these ISO 9000 requirements are represented in the ISO 9000 Micro-Theory.

7.5.1 Informal Competency Questions

The following are some of the examples of the micro-theory competency questions related to Management Responsibility:

CQ 7.11 Is the enterprise compliant to ISO 9001 requirement 4.1?

CQ 7.12 Does the enterprise define and document the quality objectives in accordance with ISO 9001?

CQ 7.13 Does the enterprise implement the quality policy in accordance with ISO 9001?

CQ 7.14 Is the enterprise compliant to ISO 9001 requirement 4.2.1?

CQ 7.15 Does there exist a quality manual in accordance with ISO 9001?

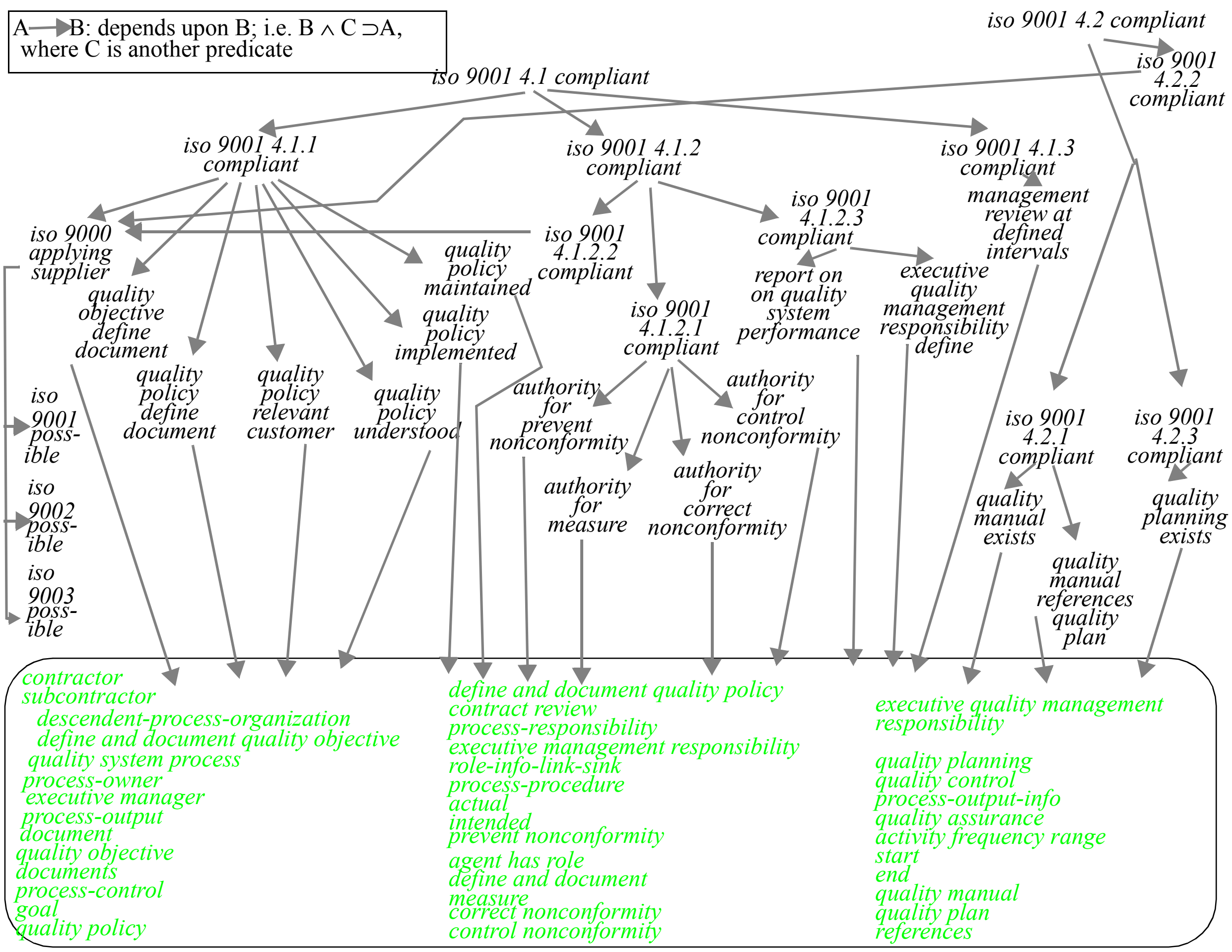
CQ 7.16 Is the enterprise compliant to ISO 9001 requirement 4.5?

CQ 7.17 Does the master list identify current revisions of documents in accordance with ISO 9001?

CQ 7.18 Does the enterprise retain documents for an adequate amount of time in accordance with ISO 9001?

7.5.2 Terminology

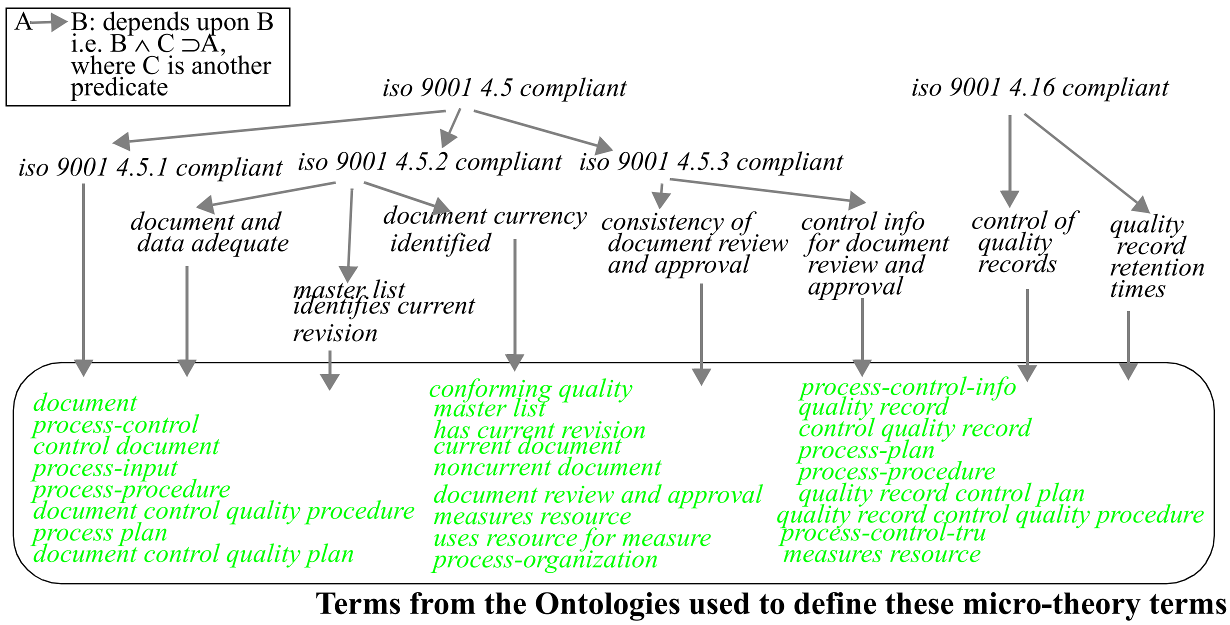
Figure 7.3 Data Model ISO 9000 Micro-Theory Terms as relevant for Management Responsibility for Quality[1]



Terms from the Ontologies used to define these micro-theory terms

1. the dependency relationship from micro-theory terms to ontology terms are abstracted out for ease of reading.

Figure 7.4 Data Model of ISO 9000 Micro-Theory Terms as relevant for Documentation for Quality Management[2]



For brevity, the definition of these terms are not shown.

7.5.3 Formal Competency Questions

These are the formal competency questions corresponding to some of the informal competency questions:

CQ 7.11 Is the enterprise compliant to ISO 9001 requirement 4.1?

- <u>Does an enterprise ε comply to the ISO 9001 requirement 4.1 in a situation σ?</u>: *holds*(agent_constraint(ε,iso_9001_4.1_compliant,σ)].
- *holds*(agent_constraint(bhp_steel_1,iso_9001_4.1_compliant,sv_actual)

CQ 7.12 Does the enterprise define and document the quality objectives in accordance with ISO 9001?

- <u>Does an enterprise ε comply to the ISO 9001 requirement upon defining and documenting quality objectives in a situation σ?</u>: ∃A∃G∃E *holds*(agent_constraint(bhp_steel_1,quality_objective_define_and document(A,G,E),σ)].
- ∃A∃G∃E *holds*(agent_constraint(ε,quality_objective_define_and document(A,G,E),sv_actual)

2. the dependency relationship from micro-theory terms to ontology terms are abstracted out for ease of reading.

CQ 7.13 Does the enterprise implement the quality policy in accordance with ISO 9001?

- Does an enterprise ε comply to the ISO 9001 requirement upon implementing the quality policy in a situation σ?: *holds*(agent_constraint(ε,quality_policy_implemented,σ)].
- *holds*(agent_constraint(bhp_steel_1,quality_policy_implemented,sv_actual)

7.6 Demonstration of Competency: Using the ISO 9000 Quality Advisor

This demonstration shows how the advisor is used for analyzing the quality management system at BHP Steel; in so doing, the advisor is used to answer the following competency questions.

- **CQ 7.6 Does the company satisfy ISO 9001 requirement 4.10 Inspection and testing?** In order to answer this question, the following question is also answered.
  - **CQ 7.1 Does the company comply to ISO 9001 requirement 4.10.1 General?**

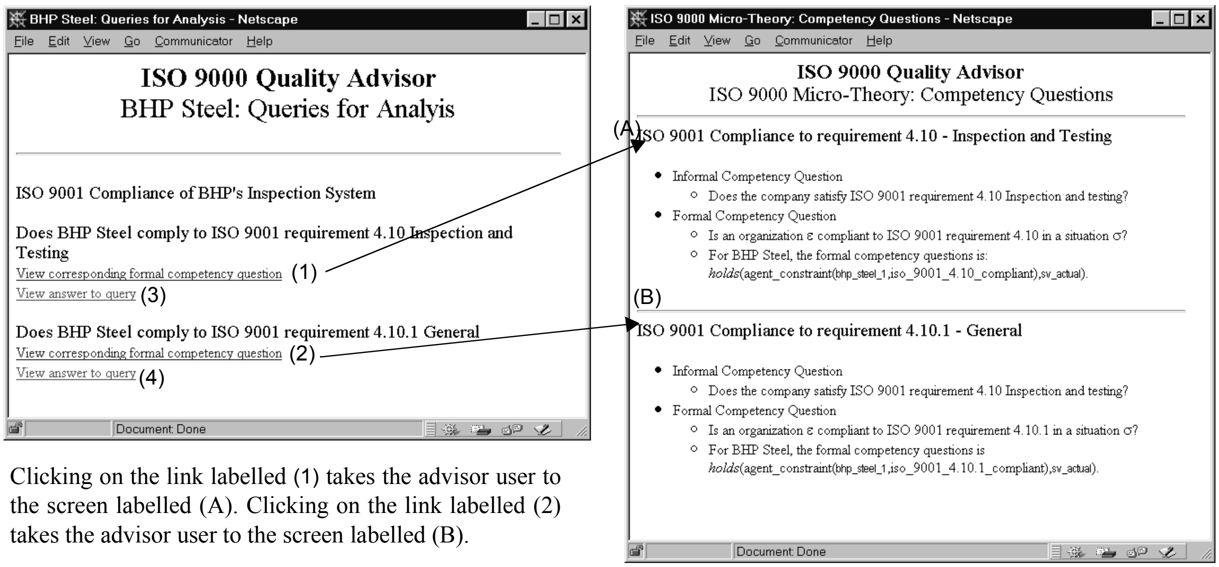
Step 1: Stating Facts about an Enterprise ⇔ Representing Populated Enterprise Models.

There are no additional facts about the enterprise specifically expressed for demonstrating ISO 9000 compliance of an organization; that is, in expressing facts that are represented in the enterprise model, all the facts needed to evaluate ISO 9000 compliance have been represented. This means that the ISO 9000 evaluation can be done on a sufficiently well-populated enterprise model that was designed for some other purpose than for ISO 9000 analysis. For example, the facts about BHP Steel, presented for the demonstrations of competency for the Measurement and Quality Management System Ontologies, are sufficient for the demonstration of micro-theory's competency.

The complete, populated enterprise model for BHP Steel is shown in Appendix A2. For the populated model for deHavilland Manufacturing, refer to [Tham 98].

Step 2: Stating Queries for Analyzing Enterprise ⇔ Representing Formal Competency Questions

Figure 7.5 Displaying ISO 9000-Related Queries and Representing them as Formal Competency Questions of the ISO 9000 Micro-Theory



Clicking on the link labelled (1) takes the advisor user to the screen labelled (A). Clicking on the link labelled (2) takes the advisor user to the screen labelled (B).

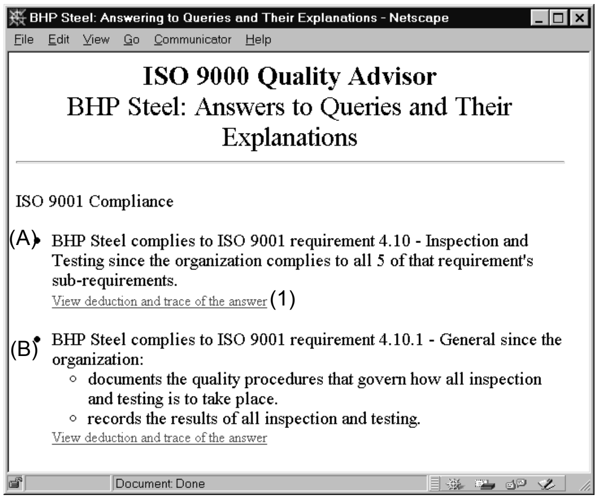
Step 3: Stating Data Dictionary of Enterprise's Terms ⇔ Representing Ontology or Micro-Theory Terminology and Axioms

There is no data dictionary since it is unclear whether many terms used for ISO 9000 compliance within BHP Steel are outside of the ISO 9000 lexicon used by the micro-theory. The screens showing the first-order logic definitions for ISO 9001 compliance axioms are not displayed, since the definitions have already been presented in this chapter.

Steps 4 and 5: Answering Queries ⇔ Deducing Answers to Formal Competency Questions; and

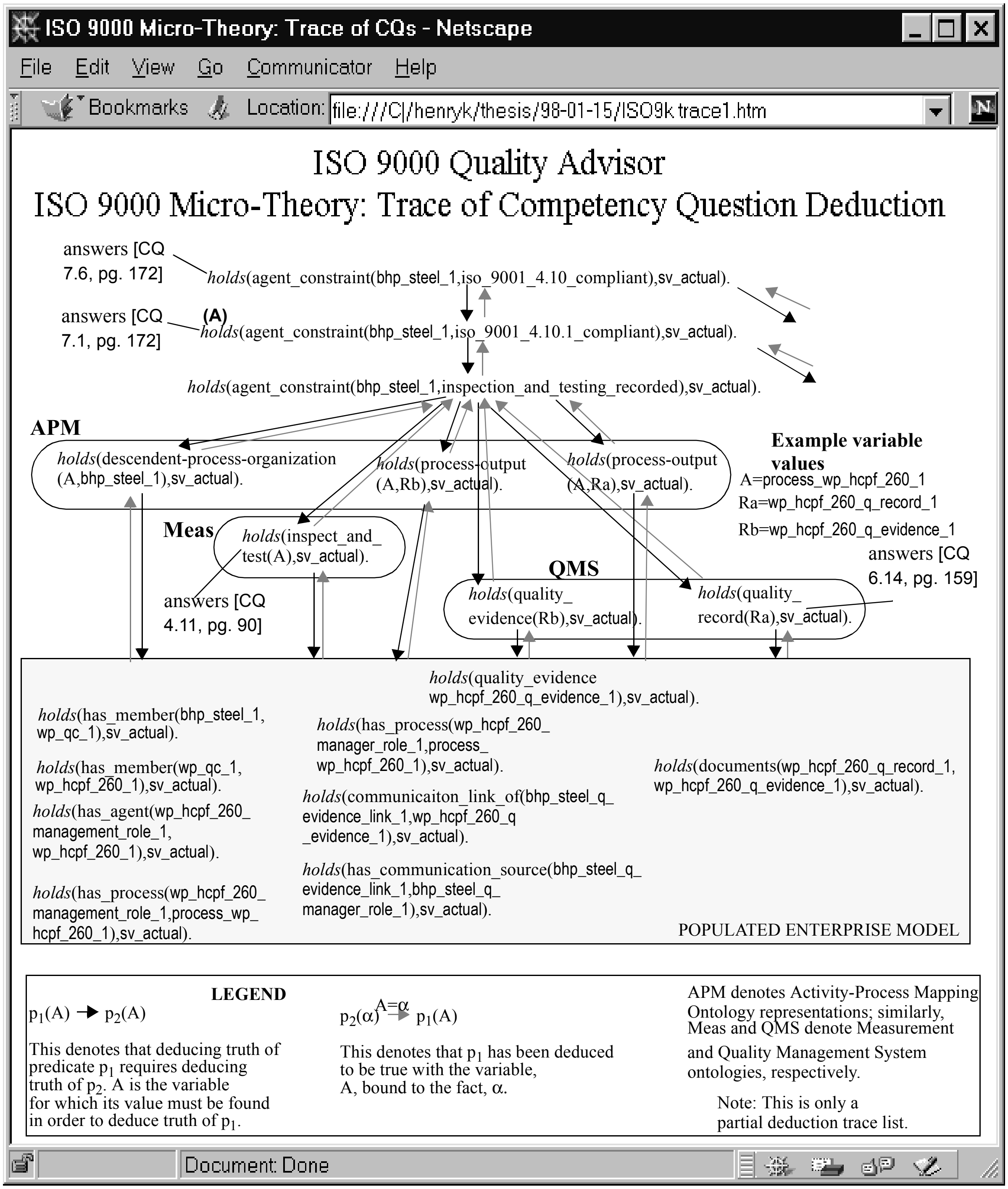
Explaining the Derivation of Answers ⟺ Tracing Deduction and Displaying Prolog Trace List

Figure 7.6 Displaying Answers to ISO 9000-Related Queries, and Explanations for Answers



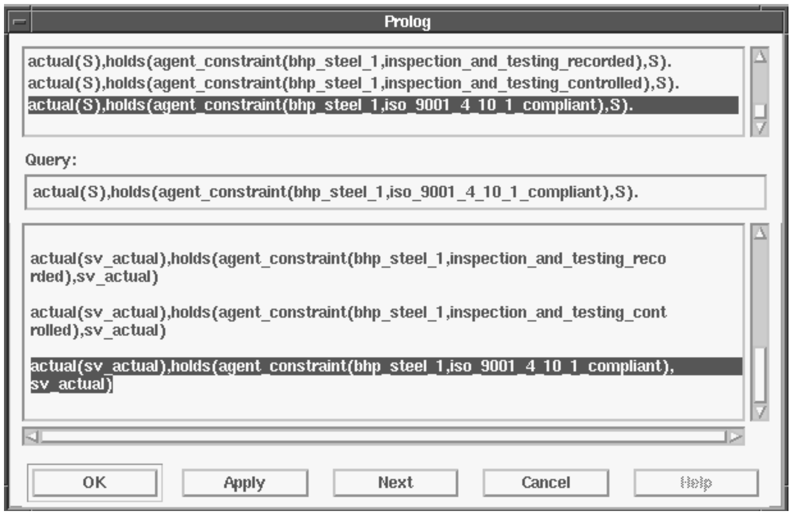
Clicking on the link labelled (1) in Figure 7.5 Displaying ISO 9000-Related Queries and Representing them as Formal Competency Questions of the ISO 9000 Micro-Theory, takes the advisor user to (A) on this screen. Similarly, clicking on (2) in the same figure brings the user to (B).

Figure 7.7 Displaying Competency Question Deduction for ISO 9000 Micro-Theory[3]



3. Clicking on (1) in Figure 7.6 Displaying Answers to ISO 9000-Related Queries, and Explanations for Answers takes the advisor user to this screen.

Figure 7.8 Prolog Query to Answer ISO 9000 Micro-Theory Competency Question



This is an actual Prolog query screen of the query evaluating BHP Steel's compliance to ISO 9001 requirement 4.10.1. Answering this query took 175 deductions. (A) in Figure 7.5 Displaying ISO 9000-Related Queries and Representing them as Formal Competency Questions of the ISO 9000 Micro-Theory denotes the competency question that corresponds to the query.

7.7 Demonstration of Reducibility: Reducing The Strategic Analyst™ ISO 9000 Compliance Competency

The Strategic Analyst™ is a software similar to the ISO 9000 Quality Advisor in that its primary use is for diagnostic internal quality audits, and it provides an easy-to-use interface for the analyst by providing terminology and example help on answering the ISO 9000 requirements. Another similarity to the ISO 9000 Quality Advisor is that, unlike most other ISO 9000 assessment software, it queries much deeper into each of the requirements. For instance, whereas some software will pose a high-level question such as "Does the organization implement the quality policy throughout the organization?" and expect the analyst to answer the question, The Strategic Analyst™ decomposes such high-level questions into lower level questions. This software differs from the ISO 9000 Quality Advisor in that it needs the lowest level queries to be answered by the analyst— who has a cognitive model of the enterprise—whereas the advisor answers the queries by applying the representations of the Ontologies for Quality Modelling and ISO 9000 Micro-Theory to the populated, computer-based models of the specific enterprise. The computer-encodeable ontology for The Strategic Analyst™ is the hierarchical set of questions, the relationships between these questions, and the keywords that are defined in English that comprise these questions.

One of the questions that an analyst must answer in using The Strategic Analyst™ is: “Does the enterprise document its strategic intent especially as it relates to quality?” It is noted that strategic intent is a concept, “sometimes called a mission statement or a corporate vision, it is an organizational framework into whose context short and long term goals comfortably fit” [SA 93]. So a partial list of the predicates, definitions, and constraints of “The Strategic Analyst™ ontology” is the following:

enterprise(O).
document(D).
strategic_intent(S).
mission_statement(S).
corportate_vision(S).
concept(S).
short_term_goal(G).
long_term_goal(G).
quality_related_goal(G).
goal_fits_context_of(G,S).
enterprise_documents(O,D).
concept_is_documented_by(S,D).

If a mission statement or corporate vision has a short or long term goal in its context, then the statement or vision expresses strategic intent.

(1) $\forall S \exists G$ [(mission_statement(S) $\vee$ corporate_vision(S)) $\wedge$
goal_fits_context_of(G,S) $\wedge$ (short_term_goal(G) $\vee$ long_term_goal(G)) $\supset$
strategic_intent(S)].

A quality related goal is a short or long term goal.

(2) $\forall G$ [quality_related_goal(G) $\supset$ short_term_goal(G) $\vee$ long_term_goal(G)].

If a mission statement or corporate vision has a short or long term goal in its context, then the statement or vision expresses strategic intent.

Strategic intent is a concept.

(3) $\forall S$ [strategic_intent(S) $\supset$ concept(S)].

If a document of an enterprise documents a concept than the enterprise documents the concept.

(4) $\forall O \forall S \exists D$ [enterprise(O) $\wedge$ document(D) $\wedge$ concept(S) $\wedge$
enterprise_documents(O,D) $\wedge$ concept_is_documented_by(S,D) $\supset$
enterprise_documents_concept(O,S)].

The competency question can be formally stated in English as "Does the organization document its strategic intent as it relates to a specific quality-related goal?" This can be expressed as the following:

(5) $\exists O \exists S \exists G$ [strategic_intent(S) $\wedge$ goal_fits_context_of(G,S) $\wedge$
quality_related_goal(G) $\wedge$ enterprise_documents_concept(O,S)].

Say the following axiom is asserted as a reduction axiom.

If an enterprise documents its strategic intent which has a goal in its context, then the enterprise documents that goal.

$\exists O \exists G \exists S$ [strategic_intent(S) $\wedge$ goal_fits_context_of(G,S) $\wedge$
enterprise_documents_concept(O,S) $\supset$
enterprise_documents_goal(O,G)].

Then the competency question can be reduced to the following

Does the enterprise document a certain quality-related goal?

(5) $\exists O \exists G$ [quality_related_goal(G) $\wedge$ enterprise_documents_goal(O,G)].

Assuming that a quality-related goal (a The Strategic Analyst™ term) is a quality obective (an ISO 9000 Micro-Theory) term, and that the question is posed in order to achieve ISO 9000 compliance, the competency question can be entirely expressed using the terms of the ISO 9000 Micro-Theory as "Does an organization define and document its quality objective in accordance with the ISO 9001?"

$\exists O \exists G \exists A \exists E \exists s$
holds(agent_constraint(O,quality_objective_define_document(A,G,E)),s).

- The term, *quality objective define and document,* denotes an ISO 9000 Micro-Theory constraint that there be a *define and document activity* A for which the *executive manager* E defines and documents a *quality objective* G.

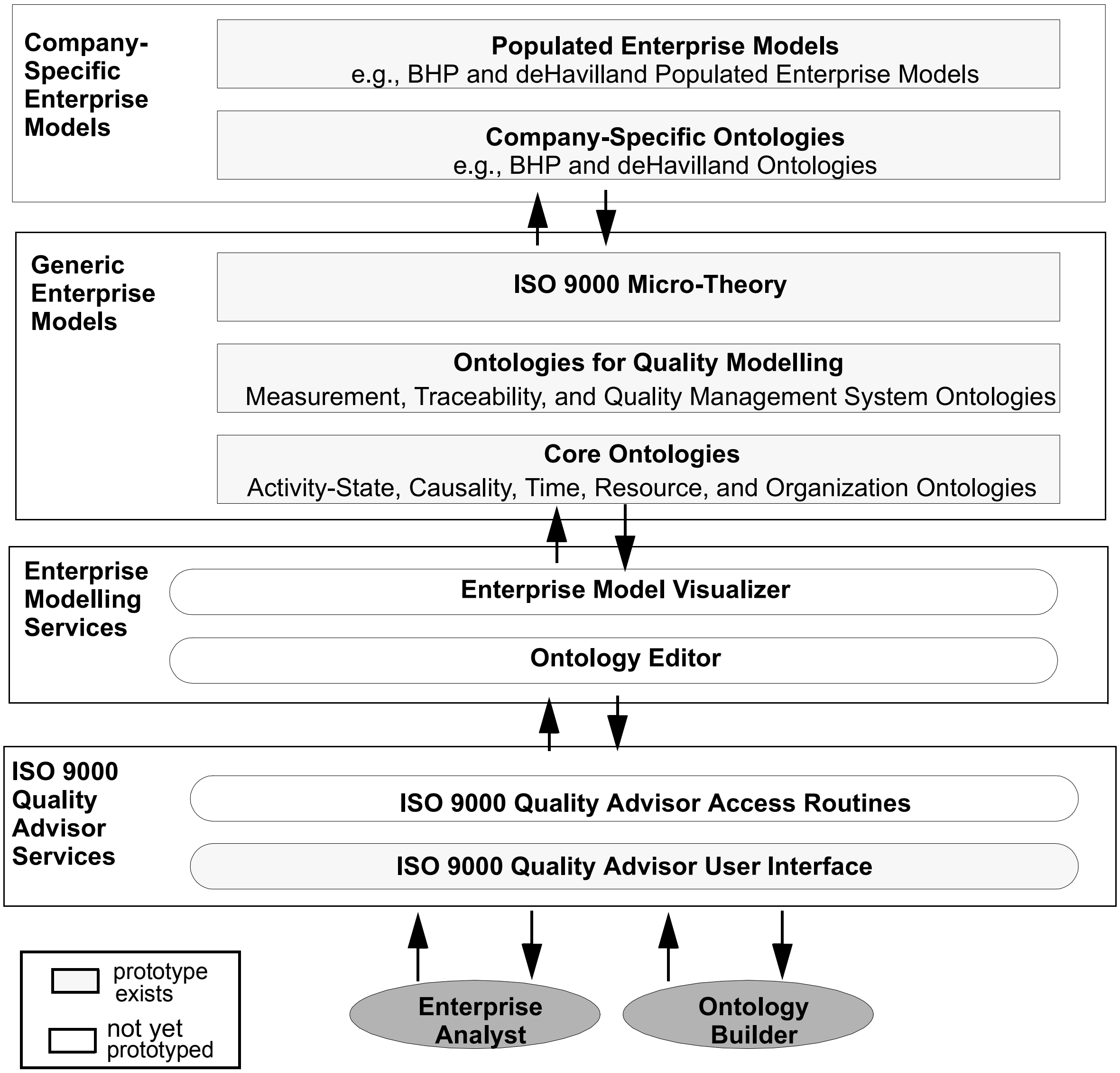
Additional exercises can further demonstrate that The Strategic Analyst™ competency questions about quality management responsibility can be reduced to competency questions regarding management responsibility that can be posed and answered using the ISO 9000 Micro-Theory representations. This is evidence that the representations of the micro-theory can be re-used to evaluate the ISO 9000 compliance of organizations, where the set of such organizations not only includes BHP and deHavilland, but also includes organizations which use The Strategic Analyst™ software.

7.8 Evaluating ISO 9000 Compliance

7.8.1 ISO 9000 Quality Advisor: System Architecture Overview

Throughout this thesis, the use of the advisor has been demonstrated. Properly, the ISO 9000 Quality Advisor is the overall software system that integrates the different enterprise modelling components for ISO 9000 quality analysis. The following is an architectural overview of the advisor.

Figure 7.9 System Architecture Overview for the ISO 9000 Quality Advisor



The front-end of the advisor is of an html user interface which access routines that enable the user to query populated enterprise models that are constructed upon generic models. The access routines also enable the user to access information system services that allow model visualization and editting. Below, the components of the architecture are described.

- **ISO 9000 Quality Advisor User Interface**: HTML documents serve as the user interface for the advisor. Clicking on hyperlinks allows the enterprise analyst or ontology builder to pose and receive answers to queries about ISO 9000 compliance or other quality-related questions, modify ontologies, or visualize a populated enterprise model.

- **ISO 9000 Quality Advisor Access Routines**: This component includes means to translate user input into Prolog queries, process the queries using an inference engine, and then return answers in a format acceptable to the user. A possible feature could be a parser that lets the user query and receive answers in natural language.
- **Core Ontologies, Ontologies for Quality Modelling, and ISO 9000 Micro-Theory**: The representations of these ontologies and micro-theory are encoded in Prolog to support automatic reasoning.
- **Ontology Editor**: This component could work in conjunction with an enterprise model visualizer to make the task of creating an ontology easier and more efficient. An enterprise analyst can use the editor, which may have a natural language or graphical interface, to create a company-specific ontology without having to know details of first-order logic. An ontology builder can use the editor to more efficiently create generic ontologies. The editor may even be used to translate between TOVE Ontologies and ontologies constructed using other languages.
- **Enterprise Model Visualizer**: This component could be used to more efficiently and reliably populate the enterprise model without having to know the details of first-order logic. The component can be used to graphically visualize the relationships between entities in the enterprise model.

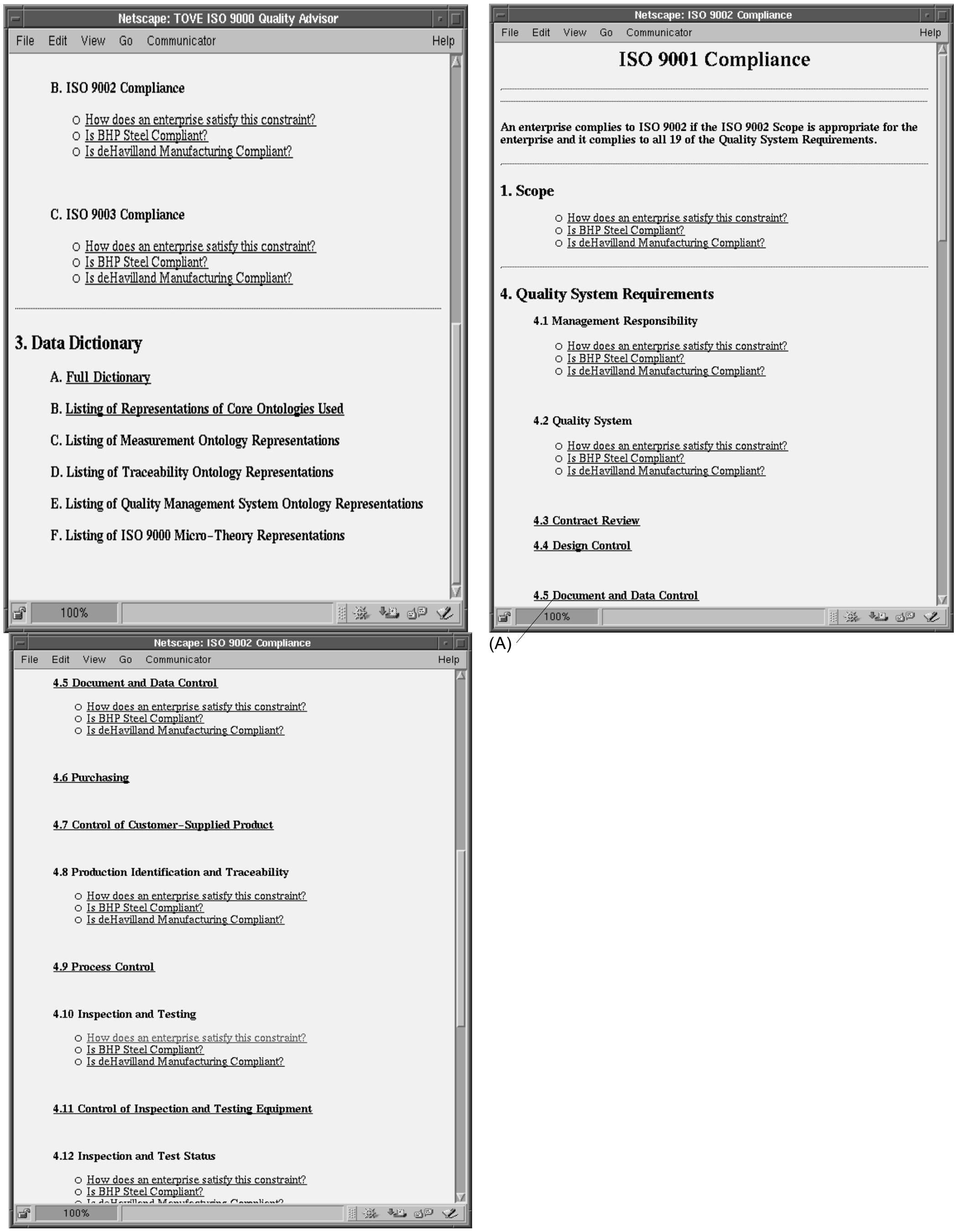
Given robust implementations of the generic components—Generic Enterprise Models and Enterprise Modelling Services—a given enterprise, in order to use the advisor for ISO 9000 quality analysis, must construct company-specific ontologies and then populate the model of it by instantiating objects, relations, and attributes from both company-specific and generic ontologies. An enterprise modeller can do this by using a user-friendly Ontology Editor and Enterprise Model visualizer.

Also, the enterprise modeller must customize the ISO 9000 Quality Advisor user interface. The interface is comprised of generic templates for screens, as well as tools for constructing company-specific screens from these templates. As an example, the top-level screen [Figure 7.10, pg. 199], which lists ISO 9000 requirements and is used to query about compliance to specific requirements, needs to be customized only slightly; more context-specific screens like the screen for explaining non-compliance [Figure 7.12, pg. 200] need much more customization. The access routines must be modified so that they are consistent with the customized user interface. Once again, user-friendly features of these components should make customizing easier.

7.8.2 ISO 9000 Quality Advisor: Step-by-Step Example

The technical design choices made, as shown in the architectural overview, allow the advisor to be used in an interactive manner to analyze different types of information in evaluating ISO 9000 compliance. Here is a step-by-step example of its use. Let’s say that a BHP Steel analyst wants to know about compliance to ISO 9001 requirement 4.12 Inspection and Test Status. The analyst would be presented with the following screens and choose the link marked (A).

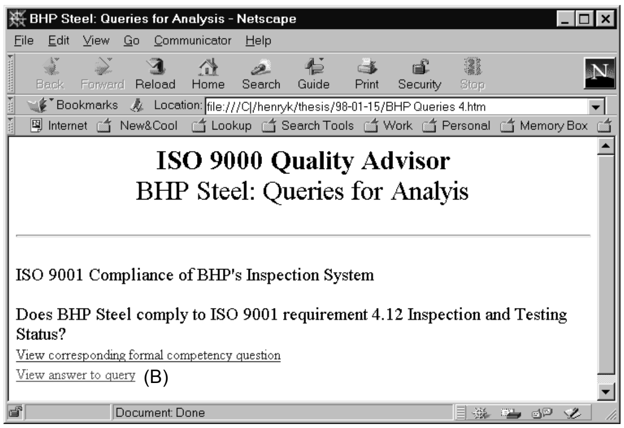
Figure 7.10 ISO 9000 Quality Advisor: Initial Screen for ISO 9001 Compliance Evaluation



Then, the analyst would choose the link labelled (B) in [Figure 7.11, pg. 200]. This choice executes a deduction in the Prolog programming environment to determine whether the ISO 9000 Micro-

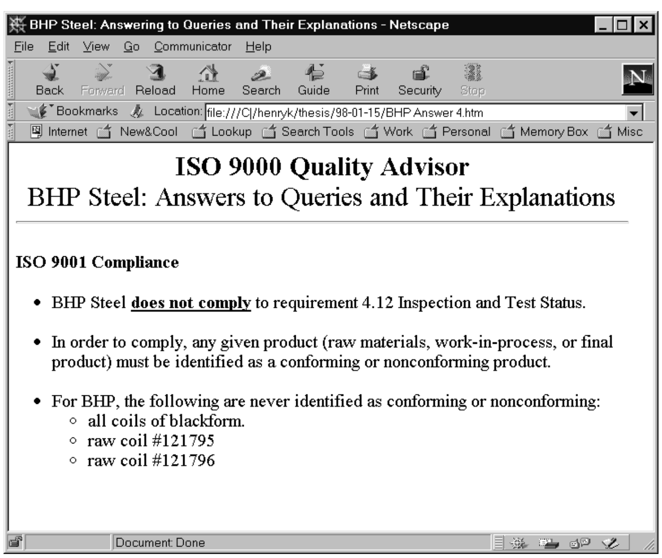
Theory term corresponding to the particular query can be proven true. The analyst is shielded from needing to know ontology or micro-theory terms, or anything about Prolog.

Figure 7.11 Querying about Compliance to ISO 9001 Requirement 4.12



After deduction is performed, the results of the query are shown.

Figure 7.12 Answers to the Query about Compliance to ISO 9001 Requirement 4.12



Assuming that natural language querying and answer capabilities have been augmented to the advisor, it would be possible for the analyst to analyze the causes of noncompliance to requirement 4.12. The analyst can ask questions in natural language to determine the history of a problematic coil. The following are sample questions.

- What was the material flow from raw materials to raw coil #121795, and what were the activities performed for this material flow?
- What was the material flow from raw coil #121795 to a final product, and what were the activities performed for this material flow? What was the time frame for this flow?

These are informal competency questions that can be translated into formal competency questions and answered, using the representations of the Traceability Ontology. The analyst can then compare the coils in question with another coil. A sample question is:

- For a given raw coil other than #121795 or #121796, when and where was it assessed that the coil was prime or non-prime?

This question can be formally posed and answered using the representations of the Measurement Ontology. The answers to these questions may show that for some reason the quality of the coils in question was not assessed at a given process when they should have been. To find out more, the analyst can ask:

- Which person had the quality control role at the process when the quality of coils #121795 and #121796 were not assessed?

Formally posing and answering this question requires the Quality Management System Ontology representations. The analyst then can talk to that person and make recommendations about changes to the overall quality management system so that in the future similar problems will not arise. After addressing the reasons for non-compliance to requirement 4.12, the analyst can then make the changes to the populated enterprise model to see the effect of changes using the enterprise model visualizer. The analyst can follow similar steps to the one just presented to evaluate BHP Steel's compliance to other requirements that have been formalized for the ISO 9000 Micro-Theory.

The capability to use the advisor interactively and analyze disparate information from different sources of the enterprise, and the support of different views of abstraction—e.g., querying about compliance to one of the requirements (a high level of abstraction), as well as querying about details such as who was working when something happened—make the advisor a useful tool not just for ISO 9000 compliance analysis, but for different quality analyses within the enterprise.

7.9 Summary and Conclusion

The following summarizes how the ISO 9000 Micro-Theory is formalized:

- The micro-theory formalizes those ISO 9000 requirements that are expressible using the Ontologies for Quality Modelling representations including: requirements related to inspection and testing formalized using the Measurement Ontology, product identification and traceability formalized using the Traceability Ontology, and management of the quality system formalized using the Quality Management System Ontology.

The micro-theory is formalized by posing competency questions, analyzing the ISO 9000 domain, stating assumptions, and developing terminology and axioms. Then, the competency of the micro-theory is demonstrated by automatically evaluating the ISO 9000 of an enterprise to a subset of requirements. The demonstration of reducibility shows that the ISO 9000 Micro-Theory spans a subset of the competency of The Strategic Analyst™.

The design, analysis, and prototypical implementation of the ISO 9000 Micro-Theory supports the thesis of this dissertation [pg. 7] by:

- Showing that representing and reasoning about ISO 9000 compliance is an appropriate application of a descriptive and prescriptive models of quality.
- Representing ISO 9000 requirements using the Ontologies for Quality Modelling.
- Using these representations to automatically evaluate compliance of organizations to a subset of ISO 9000 requirements and reason about compliance. Used in this way, the micro-theory objectively prescribes ISO 9000 compliance for organizations.

8. Conclusion, Summary, Contributions, and Future Work

8.1 Conclusion

The thesis of this dissertation is that quality within an organization can be described by representing it in an enterprise model, and ISO 9000 compliance of an organization can be objectively prescribed by reasoning about quality using the model. This thesis is supported by the design, analysis, and prototypical implementation of the Ontologies for Quality Modelling, ISO 9000 Micro-Theory, and ISO 9000 Quality Advisor. The design, analysis, and implementation support the thesis in the following way:

- By showing that a descriptive and prescriptive enterprise model of quality can be systematically engineered, rather than haphazardously crafted, using a step-by-step methodology, the Ontological Engineering Methodology.
- By showing that a descriptive and prescriptive enterprise model of quality requires representing measurement.
  - This is done by describing a popular view of quality in an enterprise model—"quality is conformance to requirements"—and representing that quality conformance is determined through measurement. Then by representing measurement in the enterprise model in the *Measurement Ontology*, ISO 9000 compliance regarding an organization's measurement system is objectively prescribed.
- By showing that representing traceability capability is important in order to perform quality analysis.
  - This is done by representing and enabling unique identification and traceability—the basic capability to diagnose and analyze quality problems—in the enterprise model. Then by representing traceability in the enterprise model in the *Traceability Ontology*, ISO 9000 compliance regarding an organization's traceability capability is objectively prescribed.
- By showing that representing an enterprise's organizational structure and information flows for managing quality—its quality management system—is important for describing and prescribing an enterprise's capability to meet its customers' quality needs.
  - This is done by representing an organization's quality goals and policies, the organizational structure governed by these goals and policies, and accompanying documentation. Then by representing quality management system in the enterprise model in the *Quality Management System Ontology*, ISO 9000 compliance regarding an

organization's management responsibility, quality system, document control, and control of quality records is objectively prescribed.

- By showing that representing and reasoning about ISO 9000 compliance is an appropriate application of a descriptive and prescriptive models of quality.
  - This is done by representing ISO 9000 requirements—using representations from Measurement, Traceability, and Quality Management System Ontologies—as the *ISO 9000 Micro-Theory*. Then, this micro-theory is used to evaluate compliance of an organization to some ISO 9000 requirements, and in so doing is used to objectively prescribe improvements to the quality of an organization's products and processes.
- By demonstrating how the Ontologies for Quality Modelling and the ISO 9000 Micro-Theory can be practically used.
  - The representations of the ontologies and micro-theory are implemented in an object-oriented database (ROCK™) and a logic programming language (Prolog), and then are used to populate models of BHP Steel and deHavilland. The ISO 9000 Quality Advisor is a prototypical analysis tool, used to answer quality-related questions about these models; it is an enabling tool for representing and reasoning about quality using enterprise models.

8.2 Summary

8.2.1 Design Rationale for Ontologies for Quality Modelling, ISO 9000 Micro-Theory, and ISO 9000 Quality Advisor

From the outset of the thesis, it was expected that ontology representations would be used to construct the ISO 9000 Micro-Theory. Formalization of ISO 9000 became the primary need for designing the ontologies. ISO 9001 requirement *4.8 Product Identification and Traceability* was formalized first because relative to the other 20 requirements, it is brief and ideal for prototyping; this requirement is written more precisely than others; and there was a need within the TOVE project to extend existing representations of resources to formally represent tracing history of sets of resources. The Traceability Ontology was engineered to formalize traceable resource units and provide representations with which the prototypical ISO 9001 requirement could be formalized.

Traceability is a fundamental capability required to analyze quality problems. This point motivated the need to represent how quality problems could be identified. By assuming that quality is conformance to requirements and requirements can be hierarchically decomposed, and by representing conformance as assessments of measurements upon entities, it was realized that results of these measurement could be used to identify quality problems. This concept became the basis of the Measurement Ontology and linked measurement with traceability. The Measurement Ontology also provided representations with which ISO requirements *4.10 Inspection and Testing* and *4.12 Inspection and Test Status* could be formalized.

It is a widely held belief that the most important factor in achieving ISO 9000 compliance is upper management buy-in. Also, ISO 9000 compliance absolutely requires a detailed and effective system of documentation about quality. So, it was decided to formalize the ISO 9001 requirements relating to these factors: *4.1 Management Responsibility*, *4.2 Quality System*, *4.5 Document Control*, and *4.16 Control of Quality Records*. This, then, became a requirement for constructing the Quality Management System Ontology: to formally represent an enterprise's quality management system as comprised of its management responsibilities and documentation system.

The ISO 9000 Quality Advisor was used as a tool to demonstrate prototypical implementations of the ontologies and the micro-theory and their application to the enterprise models of BHP Steel and deHavilland Manufacturing. The advisor was used to show how the ontologies and micro-theory could support the following tasks: quality-related analyses of enterprises conducted by an analyst unfamiliar with the details of the ontologies or micro-theory; and analyses of ontologies and micro-theory, specifically evaluation of the competency of their representations, conducted by an ontology builder unfamiliar with the details of the modelled enterprises.

8.2.2 Scope

The first challenge of this research was the difficulty in formalizing the concept of quality: "Quality is neither mind nor matter, but a third entity independent of the two... even though it cannot be defined, you know what it is" ([Pirsig 74], pg. 185,213). Adopting the following definitions—"Quality is the totality of features and characteristics of a product or service that bear on its ability to satisfy stated or implied needs." ([ISO 91], pg. 16) and "Quality means conformance to requirements." ([Crosby 79], pg. 15)—was an important design decision. *Therefore, the scope of the Ontologies for Quality Modelling encompasses reasoning about the quality of an enterprise, given that customer needs have been explicitly decomposed to quality requirements. As well, the ontologies' scope does not encompass evaluating appropriateness and validity of these requirements. The ontologies represent the building blocks for defining what is quality for a specific organization; it does not explicitly represent what is quality for a generic organization*.

A second challenge was bounding the sufficiency of the representations of the Ontologies for Quality Modelling. Some could argue that domains other than measurement, traceability, and quality management system should have been formalized as well as, or instead. Also, the specific formalizations and assumptions of a given ontology could be challenged. The ontologies were designed to support the ISO 9000 Micro-Theory and formalize germane concepts as identified from the motivating scenarios. The choice of domains and competencies to represent was consistent with this. *So, the scope of the Ontologies for Quality Modelling encompasses formalizing representations needed to answer measurement, traceability, and quality management system competency questions*. So as to construct ontologies from existing ontologies, *the scope also encompasses formalizing representations by using or extending representations from the Core Ontologies; it does not encompass formalizing representations that cannot be readily constructed from representations of the Core Ontologies*.

Another challenge was the difficulty in finding consistent and appropriate sources of reference for formalizing the ISO 9000. The language of the ISO 9000 is vague—e.g., "The quality policy shall be relevant to the supplier's organizational goals and the expectations and the needs of its customers." ([ISO 94e], pg. 2). The ISO 9000 document does not define "relevance," "expectation," and "needs"; it is left to the ISO 9000 registrars to define these terms and apply them to the companies they audit. The registrars' approaches to ISO 9000 compliance generally lack formality and generality. Many just document their experiences: "This is how we achieved ISO 9000 compliance." Some put forth a repeatable methodology, but these efforts tend to be too detailed and context-dependent; it is difficult to abstract generic concepts to formalize.

The main design decision for the ISO 9000 Micro-Theory was to use the ISO 9000 document as the primary reference, and the writings of the registrars as corroborating reference. This was a practical decision. It was more productive to clearly state the assumptions for defining terms and then make explicit the rationale for formalizing each of the requirements, than to find, abstract, and formalize generic concepts for which there exists widespread consensus amongst the registrars. *So, the scope of the ISO 9000 Micro-Theory encompasses representing a generic, perspective of the ISO 9000; it does not encompass any specific auditor's perspective.*

Many assumptions were stated to simplify the formalization of ISO 9000 requirements. Also, the formalizations themselves could be challenged; that a rigourous, real-life ISO 9000 compliance analysis using the micro-theory was not conducted could support this challenge. *The formalizations and assumptions were consistent with the generic scope of the micro-theory: It may be necessary to build upon micro-theory representations to practically evaluate ISO 9000 compliance of a specific organization, although the risk that the micro-theory representations will be inappropriate for a given organization is minimized*. For the requirements that are formalized, the micro-theory then has a breadth, not a depth, focus.

With respect to which requirements were formalized, Only 7 of ISO 9001's 20 requirements were. Consistent with the scope of the Ontologies for Quality Modelling, *the scope of the micro-theory encompasses formalizing the ISO 9000 requirements that can be represented using measurement, traceability, quality management system, and Core Ontologies representations.*

As for the ISO 9000 Quality Advisor, its construction was always meant to be a secondary objective relative to the other two. So, there were significant limitations to the implementation of the advisor, such as the inability to execute an extemporaneous query from within the advisor. Also, the user must be familiar with ontology and micro-theory terms in order to make changes to the populated enterprise model. A systems architecture overview that addresses these limitations and specifies requirements for a working software was designed. *So, the scope of the advisor encompasses prototyping a tool that enables the practical use of the ontologies and micro-theory; it does not encompass constructing a "production" system*.

In general, satisfying the thesis objectives required prototyping implementations, not constructing "production" systems. The expectations set with the industrial partners reflected this. *The scope of implementing Ontologies for Quality Modelling, ISO 9000 Micro-Theory, and the ISO 9000 Quality Advisor encompasses prototyping how organizations—especially BHP Steel and deHavilland Manufacturing—can use ontology-based enterprise models as analysis tools; it does not encompass explicitly providing results on industrial partners' enterprise quality*. Even though a real-world ISO 9000 compliance evaluation was not undertaken, there was enough evidence through demonstrations of competency that the ontologies, micro-theory, and advisor can feasibly be used as the basis for developing a robust evaluation tool.

Finally, the scope for demonstrations for competency and reducibility were bound. The enterprise models of BHP and deHavilland Manufacturing were populated to provide facts needed to answer some, but not all, competency questions. Terms and axioms required to answer these questions were implemented and tested. *So, the scope of implementation also encompasses answering a prototypical set of competency questions to demonstrate the overall competency of the ontologies and micro-theory; it does not encompass answering all competency questions*. As well, *the scope of demonstrations of reducibility encompasses showing few examples of reductions as a proof of the generality and re-useability of the representations of the ontologies and micro-theory; it encompasses demonstrations of reducibility for the Measurement Ontology and ISO 9000 Micro-Theory, but not for other ontologies*.

8.2.3 Insights from Applying Ontologies, Micro-Theory, and Advisor to Models of BHP Steel and deHavilland Manufacturing

Implementing the prototypes—specifically, having the opportunity to test the ontology and micro-theory representations on real-life data—provided valuable insights about the ontologies, micro-theory, and ontological engineering methodology. Moreover, it provided general insights to the companies about how ontologies could serve as the basis for useful analyses tools. Some of these insights are stated here.

In using the Quality Management System Ontology to augment the BHP model, the difficulty in using the Organization Ontology—one of the Core Ontologies that is necessary to construct the Quality Management System Ontology—became apparent. The organization ontology contains representations that richly describe organizational structure, such as terms like *position*, *role*, and *skill*. However, people often do not delineate between these terms; they implicitly combine *roles* and *skills* with the *position* that a person holds. As a result, the parts of the populated enterprise model constructed using the quality management systems ontology were much more difficult to explain to BHP people than other parts.

In the demonstration of competency sections, company-dependent ontologies that link a company's terminology to the TOVE terminology have been shown. Because the way an organization is structured varies widely across companies, and since organization structure concepts as represented using TOVE ontologies are difficult to explain, the first set of representations of a company-specific ontology should relate a company's organizational structure to TOVE organization and quality management system ontologies. For example, in using an analysis tool that encapsulates the BHP Steel ontology, the tool's users shouldn't necessarily have to delineate between *position*, *skill*, and *role*, but, the BHP Steel ontology should be able to reason, in a given query, whether a user is referring to *position*, *skill*, or *role*.

In using the enterprise model for deHavilland Manufacturing, the surprising ease with which the existing model could be used was noteworthy. It was populated to serve as a testbed to construct a cost ontology, but only little modification was required to test the Traceability Ontology on the model. This experience highlights one of the benefits of using ontologies: that models rigourously populated using ontology representations minimize ambiguity in interpretation. Because the model was populated by properly applying terms, axioms, and conventions of the Core Ontologies, those familiar with the Core Ontologies could easily interpret the model. This bodes great promise. If enterprise models are carefully populated using robust generic and company-dependent ontologies, and the company-dependent ontologies are easy to understand, then it would be possible for someone to understand an enterprise model that they did not create. It may even be possible to represent and maintain the valuable knowledge that is lost when people leave jobs.

These insights constitute some of the contributions of this thesis. Other contributions are detailed in the next section.

8.3 Contributions

The contributions of the Ontologies for Quality Modelling and the ISO 9000 Micro-Theory were the result of formalizing these models and characterizing their re-useability. The main contribution of the ISO 9000 Quality Advisor was that it evaluated and highlighted the potential use of the ontologies and micro-theory, including how an ontology-based system could be used to perform useful ISO 9000 compliance evaluation and analysis.

8.3.1 Measurement Ontology

Contributions as a result of Model Formalization

A measurement ontology is useful because it formally and systematically represents generic concepts about measurement. As a result, implicit axioms are made explicit, ambiguous terms are more precisely and unambiguously defined, terms are organized and structured, and terms and axioms that are generic across different measurement-related domains are identified. The following summarize the generic concepts that were formalized.

- The system for assessing measurements was represented; this included the appropriate attribute of an entity to measure, as well as the mean (μ), distribution (σ), and comparison operator (⊗) for that attribute. Measurement of attributes were recorded as *measurement points in time* that are measured as a result of some *measurement activity*. These representations are the basic ones necessary to model any form of measurement.

- Quality was represented as some composition of *conformance points*, where these were "conforming" *measurement points* with respect to some *quality requirement*. Representing *quality requirements*, *measurement points*, and *conformance points* makes it possible to model and assess the quality of any entity within an enterprise.

Since the semantics of measurement are defined in the ontology as axioms, measurement terms can be interpreted with precision. Additional facts can then be deduced by applying axioms to a known set of facts. By implementing the Measurement Ontology as a set of Prolog axioms, insights about an enterprise were gained and the competency of the ontology was demonstrated by automatically deducing answers to the following competency questions.

- *Quality Assessment System*: What is the mean value for a given attribute of an entity? What are its tolerance specifications? How is that attribute *sampled*?

- *Measurement and Conformance Points*: What is the *measured value* for an attribute at a given point in time? Was it within the tolerance specifications? Over a period of time, do the measurements lie within the specs?

Contributions as a result of Model Re-useability

Because it is generic across different measurement-related domains, representations of the Measurement Ontology can be shared across several domains. If a model of one of these domains is needed, rather than modelling that domain from first principles, ontology representations can be re-used to more rapidly and cheaply develop that model. The following highlight the re-useability of the Measurement Ontology.

- The ontology representations, for example, can be used to construct applications that require models of measurement, such as statistical quality control (SQC) and quality function deployment (QFD) software.
- The ontology representations support the construction of the ISO 9000 Micro-Theory, specifically the formalization of ISO 9001 requirements *4.10 Inspection and Testing* and *4.12 Inspection and Test Status*.

The demonstration of reducibility shows that the Measurement Ontology spans a subset of the competency of SAP R/3^{TM} quality module. This provides evidence that Measurement Ontology representations can even be used to construct commercial software packages.

8.3.2 Traceability Ontology

Contributions as a result of Model Formalization

A traceability ontology is useful because it formally and systematically represents generic concepts about traceability. As a result, implicit axioms are made explicit, ambiguous terms are more precisely and unambiguously defined, terms are organized and structured, and terms and axioms that are generic across different traceability-related domains are identified. The following summarize the generic concepts that were formalized.

- It was identified that product traceability requires representing homogenous sets of resources, *traceable resource units* (*trus*), where traceability within a *tru* is not possible, and merging different *trus* results in a completely different *tru*. In modelling an enterprise, these representations are necessary in order to guarantee unique identification and traceability of material flow.
- A node of a traceability path was represented as the tuplet of a *tru* and the activity that *consumes*, *uses*, *releases* or *produces* that *tru*. By stating axioms to link these nodes, subject to boundary conditions on terminal nodes, a *traceability path* between two nodes could always be found, if it existed. In modelling an enterprise, these representations are necessary in order to perform a trace of material flow.

Since the semantics of traceability are defined in the ontology as axioms, traceability terms can be interpreted with precision. Additional facts can be automatically deduced by applying the axioms to a known set of facts. By implementing the Traceability Ontology as a set of Prolog axioms*,* insights about an enterprise were gained and the competency of the ontology was demonstrated by automatically deducing answers to the following competency questions.

- *Quantity Tracing*: What is the quantity of a *tru* at a given point in time? When is the *tru* recognized to exist, and what is the quantity of the *tru* at that time? As the quantity of a *tru* decreases over a period of time, what does a plot of this variance look like?
- *Entity Traceability*: Is there a *trace path* from one activity that produces a *tru* to an activity that was performed in the past so that the *tru* could ultimately be produced? What is that path? What is the *trace path* from a given *tru* of a final product to a *tru* of one of its raw materials?

Contributions as a result of Model Re-useability

Because it is generic across different traceability-related domains, representations of the Traceability Ontology can be shared across several domains. If a model of one of these domains is needed, rather than modelling that domain from first principles, ontology representations can be re-used to more rapidly and cheaply develop that model. The following highlight the re-useability of the Traceability Ontology.

- The ontology representations, for example, can be used to construct applications that require models of traceability such as workflow and manufacturing requirements planning software.
- The ontology representations support the construction of the ISO 9000 Micro-Theory, specifically the formalization of ISO 9001 requirement *4.8 Product Identification and Traceability.*

The demonstration of reducibility shows that the Traceability Ontology spans a subset of the competency of SAP R/3™ materials management module. This provides evidence that Measurement Ontology representations can even be used to construct commercial software packages

8.3.3 Quality Management System Ontology

Contributions as a result of Model Formalization

A quality management system ontology is useful because it formally and systematically represents generic concepts about the quality management system. As a result, implicit axioms are made explicit, ambiguous terms about quality management system are more precisely and unambiguously defined, terms are organized and structured, and terms and axioms that are generic across different quality management system-related domains are identified. The following summarize the generic concepts that were formalized.

- A quality management system was modelled to be comprised of an organizational structural system for planning and managing quality-related policies and goals that constrain the roles of people, and a system for disseminating and documenting these policies and goals. In order to assess the capability of the organization to repeatedly satisfy its customers' needs, it is necessary to model an organization using these representations.

Since the semantics of the quality management system are defined in the ontology as axioms, quality management system terms can be interpreted with precision. Additional facts can be automatically deduced by applying the axioms to a known set of facts. By implementing the Quality Management System Ontology as a set of Prolog axioms, insights about an enterprise were gained and the competency of the ontology was demonstrated by automatically deducing answers to the following competency questions.

- *Quality System Role*: Is there a *quality policy* for the company? Does a given person fill an explicit quality-related role in the company? Who is the main person responsible for quality within the company?
- *Quality System Documentation*: Is there a *quality manual* for the company? Are the *quality procedures* for a given activity documented? For a given activity, do there exist records that provide documented *evidence* that quality goals are being met?

Contributions as a result of Model Re-useability

Because it is generic across different quality management system-related domains, representations of the Quality Management System Ontology can be shared across several domains. If a model of one of these domains is needed, rather than modelling that domain from first principles, ontology representations can be re-used to more rapidly and cheaply develop that model. The following highlight the re-useability of the Quality Management System Ontology.

- The ontology representations, for example, can be used to construct applications that require models of quality management system such as document control software.
- The ontology representations support the construction of the ISO 9000 Micro-Theory, specifically the formalization of ISO 9001 requirements *4.1 Management Responsibility*, *4.2 Quality System*, *4.5 Document and Data Control*, and *4.16 Control of Quality Records*.

8.3.4 ISO 9000 Micro-Theory

Contributions as a result of Model Formalization

- The micro-theory was used to automatically evaluate an enterprise's compliance to a subset of ISO 9001 requirements, where this subset comprises requirements formalized using the representations from the Ontologies for Quality Modelling.
- The micro-theory is a novel tool for automating ISO 9000 compliance evaluation. Existing compliance packages are either bookkeeping tools, which rely almost entirely on the subjective judgement of the registrar, or detailed one-off tools, which do not adequately separate a model of compliance from the model of a specific enterprise. The micro-theory removes some of the subjectivity by the objective formalization of requirements and is applicable to a generic set of enterprises.
- This formalization of ISO 9000 contributes systematic rigour in representing the standard. ISO 9000 compliance is represented as compliance to hierarchically organized requirements, subject to explicitly stated assumptions. Considering that many ISO 9000 registrars treat compliance checking as an art over which they have mastery, constructing the micro-theory using a transparent and repeatable methodology makes the standard easier to understand.

Contributions as a result of Model Re-useability

- If someone disagrees with the formalization of a requirement or a simplifying assumption, they can modify a portion of the micro-theory without greatly perturbing the rest of the model. One of the benefits of models constructed from re-useable components is the de-coupling between portions of the model, which facilitates easier model modification.

- The demonstration of reducibility shows that the ISO 9000 Micro-Theory spans a subset of the competency of The Strategic Analyst™. This provides evidence that the micro-theory representations can be used to construct commercial software packages.

8.3.5 ISO 9000 Quality Advisor

- The advisor is a prototypical software tool that can be used to perform quality-related analyses of enterprises. An evaluation of the ISO 9000 compliance of an enterprise can be performed. As part of this evaluation, or as part of a separate analysis, this tool can be used to analyze an enterprise's measurement system, traceability capability, and quality management system. Analysis can be performed by analysts familiar with an enterprise. They need not be familiar with the specific terms and axioms of the ontologies and micro-theories encapsulated in the tool.

- The advisor is also a prototypical software tool that can be used to evaluate ontologies and micro-theories. Ontology builders can evaluate the competency of an ontology or micro-theory as long as they are familiar with the terms and axioms of various ontologies; they need not be familiar with the details of any modelled enterprise.

Inasmuch as these contributions highlight what has been achieved with this thesis, there are many ways in which the research can be extended in future work.

8.4 Future Work

These are the different directions in which the Ontologies for Quality Modelling and the ISO 9000 Micro-Theory can be extended:

- **Works Required to Relax Assumptions**: One key assumption of the Traceability is that its representations can only be used to model non-continuous consumption and production. In order to relax this assumption, augmentations to both Activity-State and Traceability Ontologies need to be made, based upon motivating scenarios from, say, a continuous production company. The key is to discern and then formally represent what traceability means in such an enterprise.

- **New Ontologies of Quality**: Motivating scenarios from other industrial partners may necessitate augmenting existing ontologies or engineering new ones. Because of its importance in supporting different types of quality-related problems, augmenting the Measurement Ontology may be warranted. Some possible future Ontologies for Quality Modelling include generic ontologies of improvement and service.

- **New Micro-Theories of Quality**: More requirements can be formalized for the ISO 9000 Micro-Theory. There are many possible micro-theories for quality, as many as there are different quality techniques. By augmenting the current ontologies, it would be possible to formalize micro-theories of statistical quality control and Malcolm Baldrige analysis. Other potential micro-theories are quality costing (using the cost ontology), quality function deployment, and Taguchi Methods (the last two using engineering design ontologies).

- **New Advisors for Quality**: A more immediate extension to the ISO 9000 Quality Advisor would be the interactive query capability to the advisor so that the analyst can engage in real-time dialogue with the advisor. If some of the future work already discussed were in place, a possible endeavour would be a TQM advisor. Such an advisor would add upon the existing features of the ISO 9000 Quality Advisor, but would encapsulate additional Ontologies for Quality Modelling as well as different micro-theories of quality techniques. For instance, the enterprise can be assessed as per different quality perspectives such as an ISO 9000 compliance view as well as a quality costing view.

- **Ontological Engineering Methodology Refinements**: The ontological engineering methodology can be further researched to make constructing an ontology even more systematic. Areas of future work are in building ontologies that are even more re-useable, and formulating better measures of re-useability. Examples include developing more rigourous quantitative measures, as well as developing more formal heuristics for reducing from a target ontology.

References

1. [Accelerator 93]
ISO-Cert, *ISO 9000 Accelerator*, Delta-T, PO Box 108, Somers, CT 06071, 1993.

2. [AIAI 96]
AIAI, *Enterprise Project: Related Work at Other Sites* [Online], Available: http:// www.aiai.ed.ac.uk:80/~entprise/enterprise/other.html, 1996.

3. [Allen 83]
Allen, J. F., "Maintaining Knowledge about Temporal Intervals", *Communications of the ACM*, pp. 832-43, 1983.

4. [Assessor 94]
Motion Knowledge Systems Ltd., *ISO 9000 Assessor*, 59 Washington Street Suite 145, P.O. Box 346, Santa Clara, CA 95052-0346, 1994.

5. [Baan 97]
The Baan Company, *Baan Solutions* [Online], Available: http:// www.baan.com/ solutions/default.htm, January 14, 1997.

6. [Baid 94]
"An Integrated Product and Change Ontology for Collaborative Design", M. Eng. Project Report, Department of Industrial Engineering, University of Toronto, 4 Taddle Creek Rd., Toronto, ONT M5S 1A4, 1994.

7. [Baldrige 94]
National Institute of Standards and Technology, *Malcolm Baldrige National Quality Award— 1994 Award Criteria*, Gaithersburg, MD 20899, 1994.

8. [Banker et. al. 93]
Banker, Rajiv D., Kauffman, Robert J., & Zweig, Dani., "Repository Evaluation of Software Reuse", *IEEE Transactions on Software Engineering*, Vol. 19, No. 4, April 1993, pp. 379-389.

9. [Banks 89]
Banks, Jerry, *Principles of Quality Control*, John Wiley and Sons, 1989.

10. [Bassie et. al. 95]
Bassie, Trienekens, Aerts, Giesberts, *Quality Assurance Software Application Requirements (QUASAR) for Industrial Manufacturing*, IFIP WG5.7 Conference on "Reengineering the Enterprise", Galway, 1995.

11. [Bernus 96]
Bernus, P. & Nemes, L., "A Framework to Define a Generic Enterprise Reference Architecture and Methodology", *Computer Integrated Manufacturing Systems 9*,3, July 1996, pp. 179-191.

12. [Bersbach 92]
Bersbach, Peter, “Quality Information Systems”, in *Quality Engineering Handbook*, ed. Pyzdek,T., Berger, R., Milwaukee, WI: ASQC Quality Press, 1992, pp. 61-83.

13. [BHP 96d]
Broken Hills Proprietary, *About BHP Steel* [Online], Available: http://www.bhp.com.au/steel/about1.htm, November 3, 1997.

14. [Bisgaard 94]
Bisgaard, Soren, “Quality and the Bottom Line”, *Quality Engineering*, Vol. 7, No. 1, 1994-5, pp. 223-235.

15. [Bishara & Wyrick 94]
Bishara, H. & Wyrick, Michael C., “A Systematic Approach to Quality Assurance Auditing”, *Quality Progress*, December 1994, pp. 67-70.

16. [Bonner et. al. 96]
Bonner, A., Shrufi, A., and Rozen, S., “LabFlow-1: a Database Benchmark for High-Throughput Workflow Management”, *International Conference on Extending Database Technology*, Avignon, France, March 1996.

17. [BPR 96]
Enterprise Integration Laboratory, *BPR Advisory Group: Business Process Reengineering Tool Repository*, [Online], Available: http://www.ie.utoronto.ca/EIL/tool/list.html, 1996.

18. [BPR-Bonapart 94]
UBIS, *Bonapart Overheads*, UBIS, Unternegemensbertatung fur Integrierte Systeme GmbH, Alt-Moabit 98, 10559 Berlin, 1994.

19. [Budd 91]
Budd, Timothy, *An Introduction to Object-Oriented Programming*, Addison-Wesley, 1991.

20. [Burr 94]
Burr, Adrian, "A Review of Computerised Systems for Implementing ISO 9000", *Managing Service Quality*, Vol. 4, No. 5, 1994, pp. 52-5.

21. [Bush & Dooley 89]
Bush, David & Dooley, Kevin, “The Deming Prize and Balridge Award: How They Compare”, *Quality Progress*, January 1989, pp. 28-30.

22. [Byrne et. al. 93]
Byrne, J.A., Brandt, R., Port, O., “The Virtual Corporation”, *Business Week*, February 8, 1993, pp. 99-103.

23. [Cameron et. al. 96]

Cameron, B., Colony, G., Woodring, S., Rhielander, T., Lieu C., ”The Prudent Approach to R⁄ 3”, *Packaged Application Strategies*, Vol.1, No. 1, April 1996, Forrester Research, Inc., 1033 Mass. Ave., Cambridge, MA 02138.

24. [Campbell & Shapiro 95]
Campbell, A. E. & Shaprio, S. C., “Ontological Mediation: An Overview”, *Proceedings of the IJCAI Workshop on Basic Ontological Issues in Knowledge Sharing*, Menlo Park CA: AAAI Press, 1995.

25. [Carlsen et. al. 92]
Carlsen, Robert D., Gerber, Jo Ann, McHugh, James F., *Manual of Quality Assurance Procedures and Forms*, Englewood Cliffs, NJ: Prentice Hall, 1992.

26. [Cawsey 97]
Cawsey, Alison. *Essence of Artificial Intelligence*, UK: Prentice Hall, 1997.

27. [CIM-OSA 90a]
Jorysz, H.R. and Vernadat, F.B. , “CIM-OSA Part 1: total enterprise modelling and function view”, *International Journal of Computer Integrated Manufacturing*, Vol. 3, Nos. 3 and 4, 1990, pp. 144 - 56

28. [Clements 93]
Clements, Richard B., *Quality Manager’s Complete Guide to ISO 9000*, Prentice Hall, 1993.

29. [CMM 93b]
Paulk, Mark C., Curtis, Bill, Chrissis, Mary Beth, Weber, Charles V., *Capability Maturity Model for Software, Version 1.1*, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA 15213, 1993.

30. [Crosby 79]
Crosby, P.B., *Quality is Free: The Art of Making Quality Certain*, New York: McGraw-Hill, 1988.

31. [Cusamano 91]
Cusamano, M., *Japan’s Software Factories: A Challenge to U.S. Management*, Oxford, England: Oxford University Press, 1991.

32. [Dessousky et. al. 87]
Dessousky, M.I., Kapoor, S.A., Devor, R.E., “A methodology for integrated quality systems”, *Journal of Engineering for Industry*, Vol. 109, 1987, pp. 241-7.

33. [Edwards 68]
Edwards, C.D., “The Meaning of Quality”, *Quality Progress*, October, 1968.

34. [EIL 96]
Fox, Mark S., *Enterprise Integration Laboratory*, [Online] URL— http://

www.ie.utoronto.ca/EIL/eil.html, 1996.

35. [Fadel 94]
Fadel, Fadi G., ”A Resource Ontology for Enterprise Modelling”, M.A.Sc. Thesis, Department of Industrial Engineering, University of Toronto, 4 Taddle Creek Rd., Toronto, ONT, M5S 1A4, 1994.

36. [Fadel et. al. 94]
Fadel,F.G., Fox,M.S., Gruninger,M., “A Generic Enterprise Resource Ontology”, *Proceedings of 3rd Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, Morgantown,WV, April, 1994

37. [Feigenbaum 91]
Feigenbaum, Armand, V., *Total Quality Control*, New York, NY: McGraw- Hill Book Company, 1991.

38. [FEND 96]
Federation for Enterprise Knowledge Development, “Collaborative Projects”, *FEND Newsletter*, Technology Park, Bldg. 206, 48170, Zamudio, Spain, November 1995.

39. [Fernandez et. al. 97]
Fernandez, M., Gomez-Perez, and Juristo. N., “METHONTOLOGY: From Ontological Art towards Ontological Engineering”, *Ontological Engineering Organizing Committee*, AAAI-97, Stanford, CA., March, 1997.

40. [Finger et. al. 92]
Finger, S., Fox, M.S., Prinz, F.B., and Rinderle, J.R., “Concurrent Design”, *Applied Artificial Intelligence*, Vol. 6, New York: Hemisphere Pub, 1992, pp 257-283.

41. [Finin et. al. 94]
Finin T, McKay, D, Fritzson, R, & McEntire R, “KQML: An Information and Knowledge Exchange Protocol”, in Kazuhiro Fuchi & Toshio Yokoi (Ed.), *Knowledge Building & Knowledge Sharing*, Ohmsha&IOS Press, 1994

42. [Finkelstein 84]
Finkelstein, L., “A review of the fundamental concepts of measurement”, *Measurement*, Vol. 2, No. 1, pp. 25-34, 1984.

43. [Fox - Globe 92]
Zeidenberg, Jerry, “Creating Common-Sense Systems”, *The Globe and Mail*, March 3, 1992.

44. [Fox & Gruninger 94]
Fox, M.S., Gruninger, M.,“Enterprise Engineering: An Information Systems Perspective”, *3rd Industrial Engineering Research Conference*, Atlanta, Georgia, 1994.

45. [Fox & Gruninger 96]

Fox, M.S., Gruninger, M., *Enterprise Modelling*, Department of Mechanical and Industrial Engineering, University of Toronto, 4 Taddle Creek Rd., Toronto, Ontario, M5S 3G9, 1996.

46. [Fox & Gruninger 98]
Fox, M.S., Gruninger, M., “Enteprise Modelling”, *AI Magazine*, AIAI Press, To Appear.

47. [Fox 90]
Fox, M. S., “Artificial Intelligence and Expert Systems: Myths, Legends, and Facts”, *IEEE Expert*, 1990.

48. [Fox 92a]
Fox, Mark S., ‘‘The TOVE Project, Towards a Common Sense Model of the Enterprise,’’ in *Enterprise Integration*, Ch. Petrie (Ed), Cambridge, MA, MIT Press, 1992.

49. [Fox et. al. 93a]
Fox, M.S., Chionglo, J.C., Fadel, F.G., “A Common-Sense Model of the Enterprise”, in *2nd IE Research Conference Proceedings*, May 1993, Los Angeles, CA, 1993.

50. [Fox et. al. 95]
Fox, M.S., Barbuceanu, M., and Gruninger, M., “An Organisation Ontology for Enterprise Modelling: Preliminary Concepts for Linking Structure and Behaviour”, *WET-ICE 95*, Morgantown, WV, April 1995, pp. 71-81

51. [Garvin 84]
Garvin, D.A., “What does ’Product Quality’ Really Mean?”, *Sloan Management Review*, Fall, 1984.

52. [Generseth & Fikes 92]
Genesereth, Michael R., Fikes, Richard E., *Knowledge Interchange Format Version 3.0 Reference Manual*

53. [Gennari 94]
Gennari J. H., Tu, S. W., Rothenfluh T. E., Musen, M. A., “Mapping Domains to Methods in Support of Reuse”, *International Journal of Human-Computer Studies*, Vol. 41, 1993, pp. 399-424.

54. [Godfrey 93]
Godfrey, A. Blanton, “Ten Areas for Future Research in Total Quality Management”, *Quality Management Journal*, October, 1993, pp. 47-70.

55. [Gruber 91]
Gruber, Thomas R., “The Role of Common Ontology in Achieving Sharable, Reusable Knowledge Bases, *in Principles of KR & Reasoning: Proceedings of 2nd Intl Conf*,

Allen, Fikes, Sandewall (Eds), San Mateo, CA, 1991.

56. [Gruber 93a]
Gruber, Thomas R., “A Translation Approach to Portable Ontology Specifications”, *Knowledge Acquisition*, 5(2):199-220, 1993.

57. [Gruber 94]
Gruber, T. R., & Olsen, G. R. “An ontology for engineering mathematics”, *4th International Conference on Principles of Knowledge Representation and Reasoning*, Bonn, Germany:Morgan Kaufmann, 1994.

58. [Gruninger & Fox 94a]
Gruninger, M., Fox, M.S., ”The role of competency questions in enterprise engineering”, Proceedings of the IFIP WG5.7 workshop on benchmarking - theory and practice, Trondheim, Norway, Jun, 1994.

59. [Gruninger & Fox 94b]
Gruninger, Michael, Fox, Mark S., “An Activity Ontology for Enterprise Modelling”, *Workshop on Enabling Technologies - Infrastructures for Collaborative Enterprises*, West Virginia University, April,1994

60. [Gruninger & Fox 95a]
Gruninger, M., and Fox, M.S., “Methodology for the Design and Evaluation of Ontologies”, *Workshop on Basic Ontological Issues in Knowledge Sharing*, Montreal, 1995.

61. [Gruninger & Pinto 95]
Gruninger, Michael, and Pinto, Javier A., *A Theory of Complex Action for Enterprise Modelling*,Department of Industrial Engineering, University of Toronto, Toronto, ONT M5S 1A4, 1995.

62. [Gruninger 96]
Gruninger, Michael. “Designing and Evaluating Generic Ontologies”, *Proceedings of the Workshop on Ontological Engineering, European Conference on Artificial Intelligence*, Budapest, 1996, pp. 53-65.

63. [Gruninger 96a]
Gruninger, Michael, “Classes of Theories in the TOVE Organization Ontology”, Department of Industrial Engineering, University of Toronto, 4 Taddle Creek Rd., Toronto, ONT, M5S 1A4, 1996.

64. [Guha et. al. 90]
Guha, R. V., Lenat, D. B., Pittman, K., Pratt, D., , & Shepherd, M., “Cyc: A Midterm Report”, *Communications of the ACM*, Vol. 33, no. 8, August 1990.

65. [Hall 89]

Hall III, Arthur D., *Metasystems Methodology: A New Synthesis and Unification*, Pergamon Press, 1989.

66. [Hammer 90]
Hammer, Michael, "Reengineering Work: Don't Automate, Obliterate", *Harvard Business Review*, July-August 1990, pp. 104-112.

67. [Hancock et. al. 89]
Hancock, W.M., Sathe, P., Edosomwan, J.A., *Quality Assurance*, Technical Report, IOE Department, University of Michigan, 1989.

68. [Harrington 87]
Harrington, H. James, *Poor Quality Cost*, Marcel Dekker Inc., 1987.

69. [Hausen 93a]
Hausen, H. L., Welzel, D., *Specification of Software Evaluation and Certification - Formal Model*, GMD Gesellschaft fur Mathematik und Datenverarbeitung mbH, D-53757 Sankt Augustin, December 1993.

70. [Hayes 85]
Hayes, Patrick J., "Naive Physics I: Ontology for Liquids", in *Theories of the Commonsense World*, eds. J. Hobbs & B. Moore Ablex Publishing Corp., pp. 71-89, 1985.

71. [ICEIMT 92a]
SIG on "Enterprise Modeling: Issues, Problems & Approaches", *Enterprise Modeling: Issues, Problems & Approaches*, Reports of the ICEIMT Special Interest Groups, Hilton Head, SC, June 1992.

72. [IDEF 96]
National Institute of Science and Technology, *IDEF Standards*, Available: http://nemo.ncsl.nist.gov/idef/, 1996.

73. [IFS 97]
Industrial and Financial Systems, Inc., *IFS Applications* [Online], Available: http://www.avalon.com/product.htm, 1997.

74. [Ishikawa 85]
Ishikawa, K., *What is total quality control? The Japanese Way*, Englewood Cliffs, NJ:Prentice-Hall, 1985.

75. [ISO 94a]
International Organization for Standards, *Vision 2000: A Strategy for International Standards' Implementation in the Quality Arena during the 1990s*, Geneva: ISO Central Secretariat, 1994.

76. [ISO 94b]

International Organization for Standards, *ISO 8402: Quality Management and Quality Assurance— Vocabulary*, Geneva: ISO Central Secretariat, 1994.

77. [ISO 94c]
International Organization for Standards, *Quality Management and Quality Assurance Standards— Part 1: Guidelines for Selection and Use*, Geneva: ISO Central Secretariat, 1994.

78. [ISO 94d1]
International Organization for Standards, Quality Management and Quality Assurance Standards— Part 3: Guidelines for the Application of ISO 9001 to the Development, Supply, and Maintenance of Software, Geneva: ISO Central Secretariat, 1994

79. [ISO/IEC 92]
ISO/IEC JTC/SG7WG6: Evaluation and Metrics, *Software Quality Evaluation Guide-Part 1: General Guide (revision 3)*, Ed: Azuma, ISO, September 1992.

80. [Iwasaki et. al. 96]
Iwasaki, Y, Fikes, R., Farquhar, A., and Engelmore, R., "Function-Based Engineering Part Retrieval." *Working Notes of AAAI Workshop on Modeling and Reasoning with Function*, August, 1996.

81. [Jarke & Peters 94]
Jarke, M., Peters, P., "Method Modelling with ConceptBase: Principles and Experiences", *Information and Software Technology, Special Issue on Method Engineering and Meta- Modelling*, 1994.

82. [Jennings et. al. 96]
Jennings, Faratin, Johnson, Norman, O'Brien, Wiegand, "Agent-based business process management", *International Journal of Cooperative InformationSystems*, 5(2&3), 1996, pp. 105-130.

83. [Juran 74]
Juran, J. M., ed., *Quality Control Handbook*, 3rd Ed., New York, NY:McGraw-Hill, 1974.

84. [Juran 88]
Juran, J.M., *Planning for quality*, New York, NY: The Free Press, 1988.

85. [Kautz 85]
Kautz, H., "Formalizing Spatial Concepts and Spatial Language", In *Commonsense Summer: Final Report, CSLI-85-35, Stanford, CA: Center for the Study of Language and Information*, 1985.

86. [Knight 96]
Knight, Kevin, "Learning Word Meanings by Instruction", *Proceedings of the*

American Association of Artificial Intelligence AAAI-96. Portland, OR, 1996.

87. [Kuipers 86]
Kuipers, B. J., "Qualitative Simulation", *Artificial Intelligence*, Vol. 29, No. 3, September 1986, pp. 289-338.

88. [Lenat & Guha 90]
Lenat, D. B., Guha, R. V., *Building Large Knowledge-based Systems*, Addison-Wesley, 1990.

89. [Levesque & Brachman 93]
Levesque, H., and Brachman, R., *Course Notes: Introduction to Knowledge Representation*, Department of Computer Science, University of Toronto, Toronto, Ontario, Canada, 1993.

90. [Liles et. al. 96]
Liles, D.H., Johnson, M.E. & Meade, L. ,"The Enterprise Engineering Discipline" *5th Industrial Engineering Research Conference*, Minneapolis, MN, May 1996.

91. [Lin et. al. 96]
Lin, Jinxin, Fox, M.S., and Bilgic, Taner, "A Requirement Ontology for Enterprise Design", *Concurrent Engineering: Research and Applications*, Vol. 4, No. 3, September 1996, pp. 279-91.

92. [Macintosh 94]
Macintosh, Ann, *Corporate Knowledge Management State-of-the-Art Review*, AIAI, University of Edinburgh, 80 South Bridge, Edinburgh, EH1 1HN, UK, 1994.

93. [Maclean 93]
Maclean, Gary E., *Documenting Quality for ISO 9000 and Other Industry Standards*, Milwaukee, WI: ASQC Press, 1993.

94. [Mahesh & Nirenberg 95]
Mahesh, Kavi & Nirenberg, Sergei., "A Situated Ontology for Practical NLP", in *Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing*, IIJCAI-95, Montreal, Canada, August 1995.

95. [Malone et. al. 93]
Malone, Crowston, Lee, Pentland, "Tools for Inventing Organizations: Towards a Handbook of Orgzanizational Processes", In Proceedings of the 2nd IEEE WET-ICE, Morgantown, WV, April 20-22, 1993.

96. [Marca & McGowan 88]
Marca, D.A., and McGowan, C.L., *SADT: Structured Analysis and Design Technique*, New York: McGraw-Hill Co., 1988.

97. [Mayman 93]

Mayman, Alex, “New System Keeps Users Ahead of Game”, by Adhikari, R., *Info Canada*, Vol. 18, No. 7, pp. 4,38, July 1993.

98. [McCall et. al. 77]
McCall, J.A., Richards, P.K., Walters, G.F., “Concepts and definitions of software quality”, in *Factors in software quality*, Vol. 1, Springfield, VA: NTIS, Nov. 1977.

99. [McCarthy & Hayes 69]
McCarthy, J., & Hayes, P. J., ”Some Philosophical Problems from the Standpoint of AI”, in *Machine Intelligence 4*, Meltzer B., Michie D.(eds.), Edinburgh,UK:Edinburgh University Press,1969, pp. 463-502.

100. [McDermott 82]
McDermott, J., “R1: A Rule-Based Configurer of Computer Systems”, *Artificial Intelligence* 19(1), 1982, pp. 39-88.

101. [McRobb 89]
McRobb, Max, *Specification writing and management*, New York, NY: Marcel Dekker, 1989.

102. [Melan 92]
Melan, Eugene, *Process Management: Methods for Improving Products and Services*, McGraw-Hill, 1992.

103. [Nakhai & Neves 94]
Nakhai, Benham & Neves, Joao S., “The Deming, Baldridge, and European Quality Awards”, *Quality Progress*, December 1994, pp. 33-37.

104. [Nebraska 98]
University of Nebraska Administrative Systems Project, *SAP Words and Terms Glossary,* [Online]: http://asp.uneb.edu/glossary/glossary.htm, September 28, 1998.

105. [Newell 82]
Newell, A., "The Knowledge Level", *Artificial Intelligence*, 18(1), pp. 87-127, 1982.

106. [Noy & Hafner 97]
Noy, N. & Hafner, C., “The State of the Art in Ontology Design: A Comparative Review”, *Ontological Engineering Organizing Committee Working Notes*, AAAI-97 Symposium, Stanford, CA., March, 1997, pp. 84-94.

107. [O’Logic 95]
O’Logic Inc., *ISO Logic Forms*, O’Logic Inc., P.O. Box 4711, Englewood, CO 80155, 1995.

108. [Patil et. al. 92]
Patil, Fikes, Patel-Schneider, McKay, Finin, Gruber & Neches. “The DARPA knowledge sharing effort: Progress report”. In Nebel, Rich, and Swartout (eds.), *KR 92*,

San Mateo CA:Morgan Kaufmann, Nov., 1992.

109. [Peters & Szczurko 94]
Peters, Peter, Szczurko, Peter, “Integrating Models of Quality Management Methods by an Object-Oriented Repository”, *European Joint Conference on Engineering Systems, Design, and Analysis*, London, 1994

110. [Petrie 92]
Petrie, Charles, *Introduction to Enterprise Integration*, MCC Enterprise Integration, 3500 West Balcones center Drive, Austin, TX 78759, 1992.

111. [Pirsig 74]
Pirsig, R. M., *Zen and the Art of Motorcycle Maintenance*, New York, NY: Morrow, 1974.

112. [Pohl & Jarke 92]
Pohl, K., Jarke, M., *Quality Information Systems: Repository Support for Evolving Process Models*, Nr.92-37, RWTH Aachen Fachgruppe Informatik, Aachener Informatik-Berichte, Ahornstr. 55, D-5100 Aachen, 1992.

113. [Powerway 94]
AMS/Powerway, *Powerway 9000*, AMS/Powerway, 43 Pomona St., Suite 300, Buffalo, NY, 14210, 1994.

114. [Pressman 92]
Pressman, Roger S., *Software Engineering: A practitioner's Approach*, McGraw-Hill, 1992.

115. [QIS/TC 92]
ASQC QMD QIS/TC, *Quality Information Systems Draft Document*, IQS Inc., Cleveland, OH 44116-3453, 1992.

116. [QMS 91]
John A. Keane and Associates, *ISO 9000 QMS Programs*, John A. Keane and Associates, 575 Ewing Street, Princeton, NJ 08540., 1991.

117. [Reiger & Grinberg 77]
Reiger, C., Grinberg, M., “The Declarative Representation and Procedural Simulations of Causality in Physical Mechanisms”, *Proceedings of the Joint Conference on Artificial Intelligence*,1977, pp.250-5.

118. [Rich & Knight 91]
Rich, Elaine, and Knight, Kevin, *Artificial Intelligence*, New York: McGraw-Hill, Second Edition 1991.

119. [SA 93]
Omni Data Sciences, *The Strategic Analyst*, Omni Data Sciences, 1345 Monroe NW,

Grand Rapids, MI 49505, 1993.

120. [Saarelainen 96]
Saarelainen, Markku J., “What shall Quality Audits be in 2100?”, *Quality Systems Newsletter*, January 1996.

121. [SAP 95]
SAP AG, *SAP R/3 System: Quality Management*, SAP AG, Neurottstrasse 16, 69190 Walldorf, Germany, 1995.

122. [SAP 98]
SAP AG, *Welcome to the New Dimensions of SAP Solutions* [Online], Available: http://www.sap.com, March 16, 1998.

123. [SAP 96b]
SAP AG, *Business Object: Quality Notification Item* [Online], Available: http://www.sap.com/ interf/ bapis/ preview/ catalog/ qualitynotification.htm, June, 1997.

124. [SAP 98a]
SAP AG, *Business Framework: SAP Business Objects* [Online], Available: http://www.sap.com/bfw/index.htm March 16, 1998.

125. [SAP 98b]
SAP AG,*Welcome to the New Dimensions of SAP Solutions: BAPI Section*, [Online], Available:http://www.sap-ag.de/bfw/interf/bapis/bapi.htm, November 24, 1998.

126. [Sathi et. al. 85]
Sathi, A., Fox, M.S., & Greenberg, M., “Representation of Activity Knowledge for Project Management”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-7, No. 5., 1985.

127. [Scacchi & Mi 93]
Scacchi, Walt, Mi, Peiwei, “Modeling, Inegrating and Enacting Complex Organizational Processes”, *5th International Symposium on Intelligent Systems in Accounting and Management*, Stanford, CA, December 1993.

128. [Scacchi 93]
Scacchi, Walt, Mi, Peiwei, “Modeling, Inegrating and Enacting Complex Organizational Processes”, in Proceedings of the 5th International Symposium on Intelligent Systems in Accounting and Management, Stanford CA, December 1993.

129. [Scheaffer & McClave 82]
Scheaffer, Richard L., McClave, James T., *Statistics for Engineers*, Boston, MA: PWS Publishers, 1982.

130. [Scheer 94a]
Scheer, A-W, *CIM: computer integrated manufacturing: towards the factory of the*

future, Springer-Verlag, 1994.

131. [Schreiber et. al.]
Schreiber, G., Wielinga, B., Jansweijer, W., *The KACTUS View of the 'O' Word*, University of Amsterdam, Social Science Informatics, Roetersstraat 15, NL-1018, Amsterdam, The Netherlands.

132. [Smith & Edge 90]
Smith, D.J., Edge, J., "Essential Quality Procedures", in *Gower Handbook of Quality Management*, Gower Publishing Company Ltd., 1990, pp. 469-494.

133. [Sowa 95]
Sowa, J. F., *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Boston, MA: PWS Publishing Co., 1995.

134. [Spurr et. al. 94]
Spurr, K., Layzell, P., Jennison, L., Richards, N., *Software Assistance for Business Reengineering*, London: Wiley and Sons, 1994.

135. [Stader 96]
Stader, J., "Results of the Enterprise Project", in *Proceedings of Expert Systems '96, the 16th Annual Conference of the BritishComputer Society Specialist Group on Expert Systems*, Cambridge, UK, December 1996

136. [Sullivan 91]
Sullivan, Lawrence, P., *Total Quality Management in the Automotive Industry Worldwide*, Dearborn, MI: Anerican Supplier Institute, 1991.

137. [Suzaki 87]
Suzaki, Kiyoshi, *The new manufacturing challenge: techniques for continous improvement*, New York, NY:Free Press, 1987.

138. [Swenson 93]
Swenson, Keith D., "The Regatta Project", *Proceedings of the First International Conference in Technologies and Theories for human Cooperation, Collaboration, and Coordination*, 22-24 March 1993.

139. [Sylla & Arinze 91]
Sylla, C., & Arinze, B., "A Method for Precoordination in a Quality Assurance Information System", *IEEE Transactions on Engineering Management*, Vol. 38, No. 3, August, 1991, pp. 245-56.

140. [Taguchi et. al. 89]
Taguchi, G., Elsayed, A.E., Hsiang, T., *Quality engineering in production systems*, McGraw-Hill Publishing, 1989.

141. [Tate 95]

Tate, Austin, “Representing Plans as a Set of Constraints - the Model”, in the *Special Issue on “Evaluation of Plans, Planners, and Planning Agents” of ACM SIGART Bulletin*, Vol.6, No.1, January 1995.

142. [Tham 98]
Tham, D., *Enterprise Activity Based Cost Modelling*, PhD Thesis, Department of Mechanical and Industrial Engineering, University of Toronto, 4 Taddle Creek Rd., Toronto, ONT M5S 1A4, 1998.

143. [Tham et. al. 94]
Tham, D.,Fox, M.S.,Gruninger, M., “A cost ontology for enterprise modelling, *Proceedings of third Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, Morgantown,WV, April, 1994.

144. [Turner et. al. 93]
Turner, W., Mize, J., Case, K., Nazemetz, J., *Introduction to Industrial and Systems Engineering*, Englewood Cliffs, NJ: Prentice-Hall, 3rd Ed, 1993.

145. [Uschold & Gruninger 96]
Uschold, M. & Gruninger M., “Ontologies: Principles, Methods, and Applications”. *Knowledge Engineering Review*, Vol. 11, No. 2, 1996.

146. [VA 94]
American Information Systems Inc., *Visual Assessor*, American Information Systems Inc., 1994.

147. [Vallente & Breuker 97]
Valente, A. & Breuker, J., “Ontological Engineering with Principled Core Ontologies”, *Ontological Engineering Organizing Committee Working Notes*, AAAI Spring Symposium, Stanford, CA, March 1997.

148. [Vallespir et. al. 91]
Vallespir,B., Chen,D., Zanettin,M., Doumeingts,G., “Definition of a CIM Architecture within the ESPRIT Project ‘IMPACS’”, in *Computer Applications in Production Engineering*

149. [van der Vet et. al. 94]
van der Vet, P., Speel, P.-H., and Mars, N. ”The Plinius Ontology of Ceramic Materials”. In *Proceedings of ECAI-94 Workshop on Comparison of Implemented Ontologies*, Amsterdam, 1997. pp. 187-205.

150. [Walton 89]
Walton, Richard E., *Up and Running: Integrating IT and the Organization*, Boston, MA: Harvard Business School Press, 1989.

151. [Welzel 93]

Welzel, Dieter, “A Rule-Based Process Representation Technique for Software Process Evaluation”, *Information and Software Technology*, 1993.

152. [Wand & Wang 96]
Wand, Yair and Wang, Richard, “Anchoring Data Quality Dimensions in Ontological Foundations”*,Communications of the ACM,* November 1996.

153. [Welzel 95]
Welzel, Dieter, “Tailoring and Conformance Testing of Software Processes - The ProcePT Approach”, *ERCIM News* No.23 - October 1995.

154. [Whitney 96]
Whitney, D.E., Nevins, J.L., DeFazio, T.L., Gustavson, R.E., *Problems and Issues in Design and Manufacure of Complex Electro-Mechanical Systems*, The Charles Stark Draper Laboratory, Inc., Cambridge MA 02139, 1996.

155. [Williams 92]
Williams, Theodore J., *The Purdue Enterprise Reference Architecture, Instrument Society of America, Research Triangle Park, NC, 1992.*

156. [Yu 93a]
Yu, Eric S. K., “Modelling Organizations for Information Systems Requirements Engineering”, in *1st IEEE International Symposium on Requirements Engineering*, January, 1993, San Diego, USA, 1993.

Glossary

A

advisor
: A software tool which encapsulates, and enables performing tasks using, ontologies and micro-theories. The tasks for an advisor fall into evaluation, analysis, and guidance.

analysis capability of an advisor
: capability to use an advisor to predict, monitor, identify, and explain phenomena in a domain modelled by ontologies and micro-theories encapsulated by the advisor.

axiom
: semantics which define, and constrain the use of, the terminology in a model.

C

classification hierarchy
: a taxonomy; a tree structure displaying is-a relationships in a hierarchical manner.

common-sense model
: A model of a domain in which the core, fundamental concepts of that domain are represented such that the model is able to answer common sense questions by means of deduction.

competency
: The capability of a set of representations to support problem solving within one or more domains.

competency question
: A question which characterizes the problem solving or task support capability of an ontology or micro-theory. It must be possible to answer a competecy question using the representations of the ontology or micro-theory that it motivate

constraining axioms
: First-order logic sentences that constrain the interpretation upon primitive terms and definitions.

D

definitions (in an ontology)
: Formalization in first-order logic of terminology already introduced; these are defined in terms, of existing terms that have been previously defined.

descriptive model
: A model that describes the characteristics of a domain.

domain value
: An allowable value for an attribute of an object.

E

enterprise engineering
: The process of using an enterprise model to analyze and design an enterprise; the process of engineering an enterprise by manipulating a model of that enterprise.

enterprise model
: A computational representation of the structure, activities, processes, information, resources, people, behaviour, goals, and constraints of a business, govenment, or other enterprise.

evaluation capability of an advisor
: the capability of an advisor to compare different models along a dimension, such as quality or cost, and to evaluate that one model is better as per that dimension.

F

first-order logic
: A formal language which supports expressing propositions as well as predicates, where predicates may have quantified variables as arguments. In Higher-order logic, the predicates themselves are quantified.

formal logic
: A language with restrictive syntax and semantics, which supports deduction or inference of propositions given initial propositions and axioms.

formal model
: A data model expressed using formal logic.

formalization
: Expressions, written in the logic-based language, of the semantics of a model's terminology.

G

ground Terms
: Primitive terms instantiated with facts.

guidance capability of an advisor
: the capability of an advisor to suggest alternatives based upon deductions using ontologies and micro-theories it encapsulates.

I

instantiated model
: A formal model populated with data.

ISO 9000 compliance
: Satisfaction to one of the ISO 9000 Standards for Quality Management (ISO 9001, ISO 9002, or ISO 9003).

ISO 9000 Micro-Theory

A formal model of ISO 9000 Compliance

ISO 9000 Quality Advisor
A tool for enterprise engineering which can be used to design and analyze an enterprise towards compliance to the ISO 9000.

M

micro-theory
A formal model required to solve a problem in a domain or to describe in detail a subset of the domain; a contextually-bounded formal model of knowledge that is often task-oriented.

minimal ontological commitment
A design guideline to restrict axioms of an ontology to those required to minimally describe a domain. Thus the ontology offers only minimal commitment to give details or facilitate problem solving about the domain.

O

object
A collection of entities organized as one because they share common properties.

Ontologies for Quality Modelling
Collective name for ontolgies of measurement, traceability, and quality management system.

ontology
A formal description of entities and their properties; it forms a shared terminology for objects of interest in the domain, along with definition for the meaning of each of the terms.

P

predicate
An expression of an object or a relationship between objects in a formal language. In first-order logic, predicates have one or more arguments, where all variable (non-constant) arguments must be quantified.

predicate calculus
See first-order logic

predicate logic
See first-order logic.

prescriptive model
A model that prescribes an alternate state for a domain.

primitive term
Predicates which are never formally defined in first-order logic.

Q

quality
: Totality of the features and characteristics of a product or service that bear on its ability to satisfy stated or implied needs.

queries
: A question, expressed in the formal language, that initiates a deduction about the model.

R

reducibility
: Demonstration that competency questions of another model- spanning a different competency than the evaluated model- can be reasonably translated (reduced) to competency questions answerable using representations of the evaluated model.

relation
: Relationships between objects in a data model.

representations
: Informal statements of terminology and semantics, e.g., in English, as well as formalizations of a model.

re-useability
: Capability to use portions of a model to solve different problems beyond the problems that initially motivated the development of the model.

S

semantics
: Meaning of the terminology of a model.

spans competency
: A native ontology spans the competency of a target ontology if a set of the target ontology's competency questions can be reduced to competency questions expressible and answerable using target ontology representations.

syntax
: The grammar for composing expressions in a model.

T

taxonomy
: a tree structure displaying is-a relationships in a hierarchical manner

terminology
: Terms in the vocabulary of a model.

A1. Appendix I: Activity-Process Mapping Ontology

A1.1 Quality Process Competency

A1.1.1 Motivating Scenario

"FPD wants to look at how the way it controls the processes at the product units, and how it handles the non-prime that are produced, so that quality of the products can be improved." [pg. 50]

In order to control the process, much must be known about the process: in particular as relevant to this section on quality management system ontologies, it is important to engineer representations with which a process can be characterized. This requires translation of existing ontology representations to terminology more prevalent in the quality management literature, for example , is worthwhile. It is stated that since "The organization creates, improves and provides consistent quality in its offerings through the network of processes, a process model is a fundamental conceptual basis for the ISO 9000 family" ([ISO 94c], pg. 5). Hence some questions are:

CQ A1.1 Is a given entity or information transformed by a process?

CQ A1.2 Is a given entity or information the resultant of a transformation by a process?

CQ A1.3 Does a given entity or information control the performance of the process, but not transformed by it?

CQ A1.4 Is a given entity a physical artifact that is required to perform the process, but is not transformed by it?

CQ A1.5 Is this a constraint which defines the scope of a given process?

CQ A1.6 Is this an objective of a given process?

CQ A1.7 Is this a policy which constrains the given process?

CQ A1.8 Is this a responsibility or role to be fulfilled by the performance of a given process?

CQ A1.9 Is a given process performed within a certain enterprise?

CQ A1.10 Is a given employee responsible for performing a certain process?

CQ A1.11 Does a given employee performing a certain process have authority over all other employees performing a certain process?

Once the competency questions that motivate the modelling of the relationship between processes and quality management responsibility are posed, questions that motivate the modelling of specific processes that exist mainly to fulfill quality management responsibility are warranted.

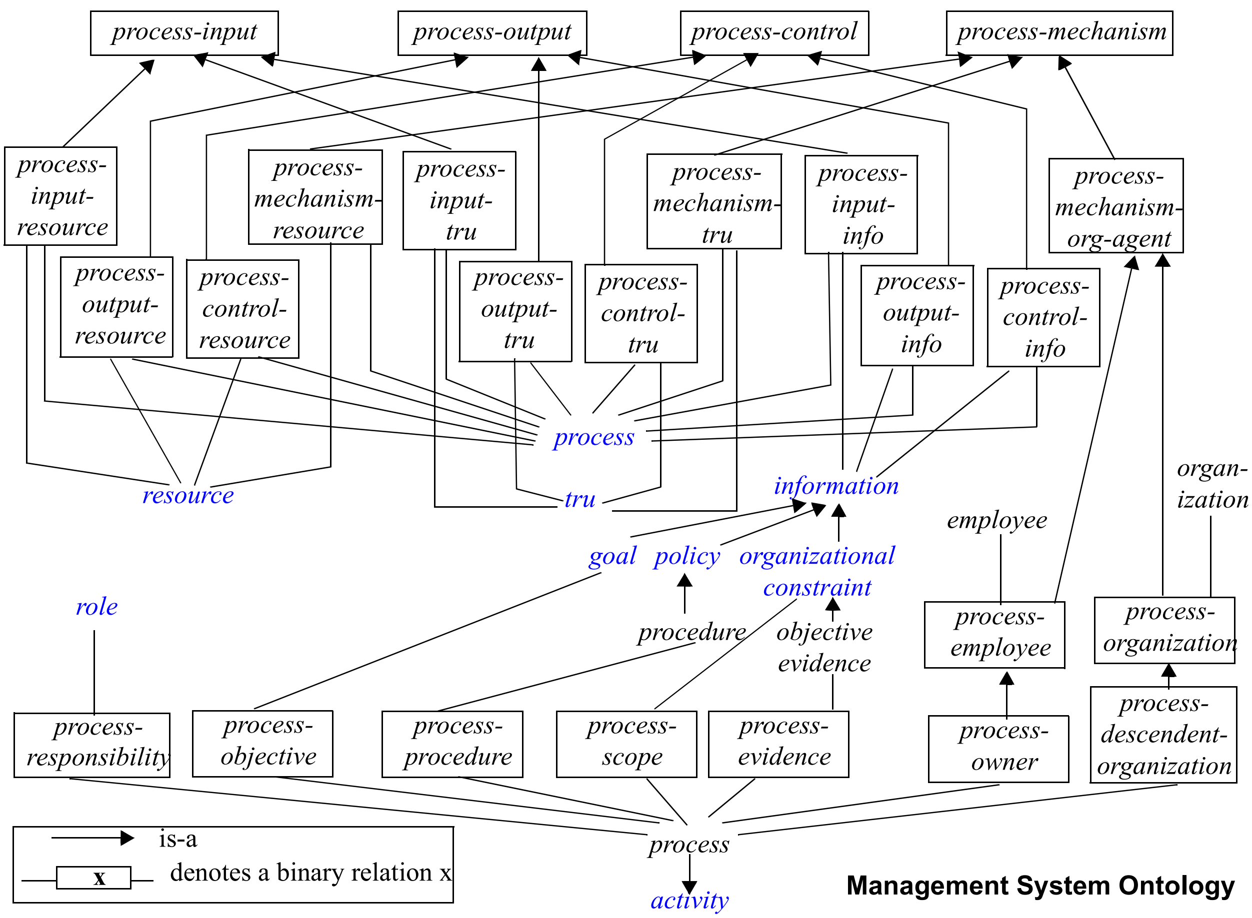
CQ A1.12 Is this process a process which comprises an enterprise's quality system?

CQ A1.13 Which particular function of the enterprise's quality system does this process perform?

A1.1.2 Analysis

Modelling characteristics of a process

Figure A1.1 Data Model of Ontology Terms related to Process Characterization



In order to specify how to perform an activity, it is noted that the documentation of a procedure "usually contains the purposes and scope of an activity; what shall be done and by whom; when, where and how it shall be done; what materials, equipment and documents shall be used; and how it shall be controlled and recorded“ ([ISO 94b], pp. 2). Firstly, a *process* is just an *activity*. Thus same type of information will be needed to characterize a process.

A *procedure* relates the purposes and scope of the *process* for which the *procedure* exists. Building from the activity ontology in which the constraints of an *activity* are related to the *states* of the world before (*enable state*) and *after* (*caused by state*) the activity execution, the scope of a *process* can be considered the pre- and post- conditions upon which the *process* depends [Melan 92]. Hence: ***process-scope(P,I)*** : There exists a process-scope relation between a *process* and an *organizational constraint* of a related *role*

The purpose or the objective for which the process is being performed, as recommended by [Carlsen et. al. 92], should also represented: ***process-objective(P,G)*** : There exists a process-scope relation between a *process* and a *goal* of a related *role*

That roles that are fulfilled by performing the process (*process-responsibility*) must also be explicitly represented:

Expression A1.1 **process-responsibility(P,R)**

$$\forall P \forall R \forall s \left[\mathit{holds}(\textbf{\textit{process-responsibility(P,R)}},s) \equiv \mathit{holds}(\text{has\_role}(P,R),s) \wedge \exists R\ \mathit{holds}(\text{agent\_has\_role}(A,R),s) \right].$$

P: a process
R: role for which P is being performed
A: an agent that possesses the role R
s: an extant or hypothetical situation

According to [McRobb 89], the scope of the process may encompass the different functional departments that are involved in the process; hence the need to define the term *process-organization* and *descendent-process-organization:*

- ***process-organization(P,O)*** : There exists an organization, where its members *perform a process*, if there *exists some role* of the *organization* which requires the performance of the process.
- ***descendent-process-organization(P,O)*** : If there exists an organization where a process is performed within one of its *member organizations*, or the *process is performed within that organization*, then the process is a descendent process to the organization.

- Note one of the terms required to define the term descendent-process-organization— *has descendent member* — is from the core ontologies.

Furthermore, [Maclean 93] states that there should be another section for stating responsibility and ownership for the process. So it is defined that those employees that perform the process is responsible for the process; this motivates the term *process-employee*. It is also defined that an employee who performs a process has ownership of the process if the employee has authority over all the other employees who have responsibility for that process; this is a *process-owner*. So:

- ***process-employee(P,A)*** : There exists an *employee* who is responsible for performing a process, if there *exists some role of the agent* which is fulfilled by performing the process.
- ***process-owner(P,A)*** : There exists an *employee* who has ownership over a *process*, if this *employee* performs the *process* and *has authority* in the agent's role vis-a-vis the *process* over all the other *employees* who perform the *process*.

The de facto standard process model, SADT/IDEF [Marca & McGowan 88][IDEF 96], is used to classify *resources* into inputs, outputs, controls, and mechanisms, where: Inputs are transformed, outputs are the resultant of a transformation, controls represent the things that constrain a process, and usually stands for information that directs what processes do, and mechanisms represent physical aspects of an activity, such as storage places, people, organizations, and equipment . According to [BPR-Bonapart 94], a process model is comprised of information, resource, and functional (organizational) models. Savolainen, T., Beeckmann, D., Groumpos, P. and Jagdev, H., "Positioning of Modelling Approaches, Methods, and Tools", Computers in Industry, Vol 25, No. 3, March 1995, pp. 255-62. [Scacchi & Mi 93] combine aspects of all these models, when they espouse that agents (organizational entities), information technologies (physical resource entities), and tasks (informational entities) be consumed and produced. According to the TOVE resource ontology, physical resource entities that are transformed can be delineated into *resources* or collection of these resources, *traceable resource units*. According to the TOVE organization ontology, organizational entities are *organizational agents*, and information entities are *policies* or *goals* that are *sent* and *received* between these agents.

Below is shown a table of the terms of the Managemeny Systems Ontology, where the terms are delineated by 1) how the "item" is used by the process, and 2) what type of an "item" it is.

Table A1-1 Table of Terms for the Process Model Extensions to the Core Ontologies

item type → item use ↓	resource	tru	organization agent	information
input process-input	process-input-resource	process-input-tru		process-input-info
output process-output	process-output-resource	process-output-tru		process-output-info
control process-control	process-control-resource	process-control-tru		process-control-info
mechanism process-mechanism	process-mechanism-resource	process-mechanism-tru	process-mechanism-org-agent	

- ***process-input-resource(P,Rc)*** : A *resource* is an input resource to a *process* if it is *consumed by that process*, and if it or a *descendent elaboration* of this *resource* is not *produced* by the *process*' *sub-activities*.

- A *tru* is an input tru to a *process* if it is *consumed by that process*, and if it or a *descendent elaboration* of this *tru* is not *produced* by the *process*' *sub-activities*.

Expression A1.2 **process-input-tru(P,Rt)**

$$\forall P \forall Rt \forall s \exists Rc \Big[\mathit{holds}(\textbf{\textit{process-input-tru(P,Rt)}},s) \equiv$$
$$\mathit{holds}(\text{consume\_res\_tru}(P,Rt),s) \wedge \mathit{holds}(\text{has\_tru}(Rc,Rt),s) \wedge$$
$$\forall P_o \forall Rt_o \exists Rc_o \ (\mathit{holds}(\text{has\_subactivity}(P,P_o),s) \wedge$$
$$\mathit{holds}(\text{has\_descendent}(Rc,Rc_o),s) \wedge$$
$$(\mathit{holds}(\text{has\_tru}(Rc_o,Rt_o),s) \vee Rt=Rt_o) \wedge$$
$$\supset \neg \mathit{holds}(\text{produce\_res\_tru}(P_o,Rt_o),s)) \Big].$$

P: a process
Rt: an input tru of P
Rc: an input resource of P
Po: a subactivity to P
Rt_o: a descendent elaboration of Rt
Rc_o: a descendent elaboration of Rc
s: an extant or hypothetical situation

- ***process-input-info(P,I)*** : *Information* is input information if it it is the content of a *link received by an agent fulfilling a role* for the process; the *information* is not a *goal*, *policy*, or *constraint* of the *process*; and if it or a *descendant elaboration* of it is not contained in a *link sent by a role* of the *process*' *subactivities*.

- ***process-input(P,X)*** : Inputs to a process may be its *input resource*, *input tru*, or *input information*.
    - *If an entity or information is an input to a given process and the process has sub-activities, then this or a descendent elaboration of this must be an input to a sub-activity of the process.*

- ***process-output-resource(P,Rc)*** : A *resource* is an output resource to a *process* if it is *produced by that process*, and if it or a *descendent elaboration* of this *resource* is not *consumed by the process*' *sub-activities*.

- ***process-output-tru(P,Rt)*** : A *tru* is an output tru to a *process* if it is *produced by that process*, and if it or a *descendent elaboration* of this *tru* is not *consumed by the process*' *sub-activities*.

- ***process-output-info(P,I)*** : *Information* is output information if it it is a *communication link* sent by an *agent fulfilling a role* for the *process*; if it is not a *goal*, *policy*, or *constraint* of the *process*, and if it or a *descendant elaboration* of this *policy* or *goal* is not *received by a role* of the *process*' *subactivities*.

- ***process-output(P,X)*** : Output to a process may be its *output resource*, *output tru*, or *output information*.
    - *If an entity or information is an output to a given process and the process has sub-activities, then this or a descendent elaboration of this must be an output to a sub-activity of the process.*
    - *Unless all of a process' primitive activities are output activities, a process has output resources and output trus.*

Expression A1.3 **All Processes have Outputs unless Comprised of Output Activities**

$$\forall P \forall s \exists P_o \, [\, holds(\text{has\_descendent\_subactivity}(P,P_o),s) \wedge \text{primitive\_activity}(P_o) \wedge \neg holds(\text{output\_activity}(P_o),s) \supset \exists Rc_o \, holds(\text{process-output}(P,R\ ].$$

P: a process
Rc_o: an output to P
s: an extant or hypothetical situation

- *Information* is control information if it is the content of the *communication link* or *authority link* *received by an agent fulfilling a role* for the *process*, and the *information* is a *goal* or a *policy*.

Expression A1.4 **process-control-info(P,I)**

$$\forall P \forall I \forall s \exists R \exists L \Big[\, holds(\textbf{process-control-info(P,I)},s) \equiv holds(\text{has\_process}(R,P),s) \wedge holds(\text{role-link-info-sink}(R,I,L),s) \wedge [\, holds(\text{has\_goal}(R,I),s) \vee holds(\text{has\_policy}(R,I),s) \vee holds(\text{has\_constraint}(R,I),s)] \Big].$$

P: a process
I: information that is a control to P
R: role of P that receives L

L: a communication or authority link for which I is the content.
T: position filled by O in performing P in order to fulfill R
s: an extant or hypothetical situation

- ***process-control-resource(P,Rc)*** : A *resource* is a control resource if it is a *document* which is *used by the process*, and *documents control information* about the *process*.
    - Note that the term *use resource or tru* is defined in the traceability competency section.

- ***process-control-tru(P,Rt)*** : A *tru* is a control tru if it is a *tru* of a *document* which is *used by the process*, and its *resource* type *documents control information* about the *process*.

- ***process-control(P,X)*** : Control to a process may be its *control resource*, *control tru*, or *control information*.
    - *If an entity or information is an input to a given process and the process has sub-activities, then this or a descendent elaboration of this must be a control to a sub-activity of the process.*

- ***process-mechanism-resource(P,Rc)*** : A *resource* is a mechanism resource if it is a *resource* which is *used by the process*, and is not a *control resource*.

- ***process-mechanism-tru(P,Rt)*** : A *tru* is a mechanism tru if it is a *tru* which is *used by the process*, and is not a *control tru*.

- An *organizational agent* is a mechanism organizational agent to a *process*, if it is an *organization of the process* or an *employee of the process*.

Expression A1.5 **process-mechanism-org-agent(P,A)**

$$\forall P \forall A \forall s \left[\; holds(\mathbf{process\text{-}mechanism\text{-}org\text{-}agent(P,A)},s) \equiv holds(\text{process-organization}(P,A),s) \lor holds(\text{process-employee}(P,A),s) \; \right].$$

P: a process
A: an organization agent required to perform the process
s: an extant or hypothetical situation

- ***process-mechanism(P,X)*** : Mechanisms to a *process* may be its *mechanism resource*, *mechanism tru*, or *mechanism organizational agent*.
    - *If an entity or information is a mechanism to a given process and the process has sub-activities, then this or a descendent elaboration of this must be a mechanism to a sub-activity of the process.*

Expression A1.6 **Abstraction of Process Mechanism**

$$\forall P \forall Rc \forall P_o \forall s \left[\; holds(\text{process-mechanism}(P,Rc),s) \land holds(\text{ has\_subactivity}(P,P_o),s) \supset \exists Rc_o(\{ holds(\text{process-mechanism}(P_o,Rc_o),s\} \land (holds(\text{has\_descendent}(Rc,Rc_o),s) \lor Rc=Rc_o) \} \; \right].$$

P: a process with sub-activities
Rc: a mechanism to P
P_o: a sub-activity to P
Rc_o: a mechanism to P_o, and which is a descendent elaboration of Rc

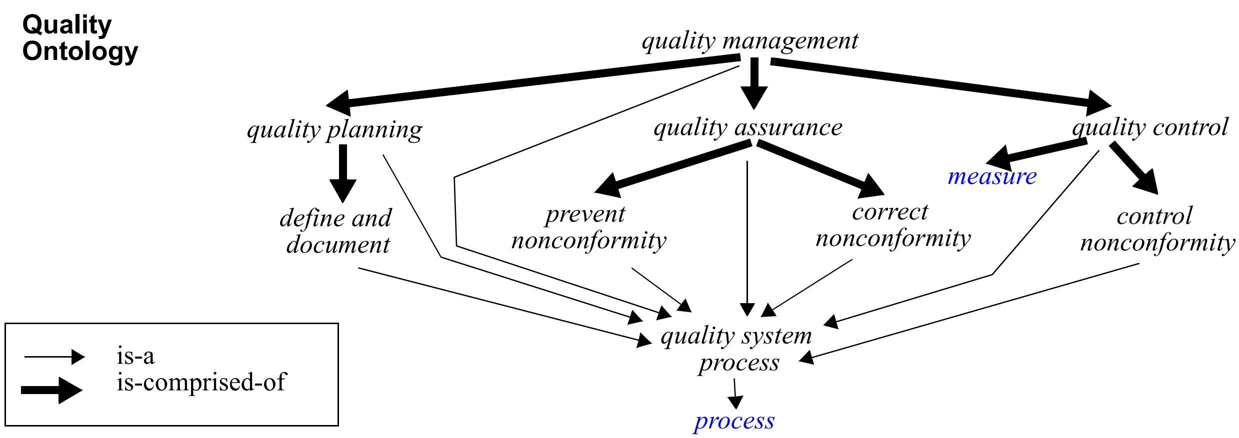
s: an extant or hypothetical situation

With these definitions, it is now possible to define:

- ***process-procedure(P,Q)*** : There exists a process-procedure relation between a *process* and the *procedure* which *controls* it.
- ***process-evidence(P,E)*** : There exists a process-evidence relation between a *process* and the *objective evidence* which is *outputted* from it.

Modelling processes of a quality system

Figure 1.2 Taxonomy of Quality System Processes



According to the activity/state and organization ontologies, activities are performed by agents fulfilling their roles to satisfy goals. Since a process is synonymous with an activity, a *quality system process* then is defined to be a *process* which is performed to fulfill a *quality system role*:

Expression A1.7 **quality_system_process(P)**

$$\forall P \forall s \exists R \left[holds(\mathbf{quality\_system\_process(P)},s) \equiv holds(\text{has\_process}(R,P),s) \wedge holds(\text{quality\_system\_role}(P),s) \right].$$

P: quality system process
R: quality system tole of P
s: an extant or hypothetical situation

A quality system process has the following associated axioms:

- *all quality system processes must have a quality procedure*
- *all quality system processes must have a related organization*

- *all quality system processes must have a related responsibility*
- *all quality system processes must have an employee*
- *all quality system processes must have an objective*
- *all quality system processes must have exactly one owner*

Expression A1.8 **All Quality System Processes must have exactly one Owner**

$$\forall P \forall A \forall Ao \forall s\ [\ holds(\text{process-owner}(P,A),s) \wedge holds(\text{process-owner}(P,Ao),s) \wedge holds(\text{quality\_system\_process}(P),s) \supset A = A_o\ )\ ].$$

P: a process
A: agent which owns P
s: an extant or hypothetical situation

As motivated by [Juran 88] and [ISO 94b], The *quality management* process constitutes the collective *process* of all the *quality system processes* of the *enterprise*, and is comprised of the following *processes*.

- *quality planning*
- *quality control*
- *quality assurance*

And so,

- ***control_nonconformity(P)*** : A *process* must be asserted to be a *process* which controls nonconformities.
- ***correct_nonconformity(P)*** : A *process* must be asserted to be a *process* which corrects nonconformities.
- ***prevent_nonconformity(P)*** : A *process* must be asserted to be a *process* which prevents nonconformities.
- ***quality_planning(P)*** : A quality planning process is a quality system process process with the following sub-activities: only *define and document* processes.
  - *define and document* processes are defined later
- ***quality_control(P)*** : A quality control process is a *quality system process* with the following *sub-activities*, *measure* and *control nonconformity processes*.
- ***quality_assurance(P)*** : A quality assurance process is a *quality system process* which has, as its *sub-activities*, only *prevent nonconformity* and *correct nonconformity processes*.

- ***quality_management(P)*** : A quality management process is a *quality system process* with the the following *sub-activities*: *quality planning*, *quality control*, and *quality assurance*.
  - *Two different quality management processes must never exist for the same enterprise.*

So then a *process* is characterized by the representations related to the *process* defined so far. Since these representations are discerned, in the enterprise model, through relations of various enterprise objects to the enterprise object, *role*, when all the *roles* for a *process* are *defined and documented*, the *process* can be considered to be fully characterized.

A1.1.3 Informal Competency Questions

CQ A1.1 Is a given entity or information transformed by a process?

- Is a given resource κ a process-input-resource to a process α in a situation σ?: *holds*(process-input-resource(α,κ),σ).
- Is a given tru κ a process-input-tru to a process α in a situation σ?: *holds*(process-input-tru(α,κ),σ).
- Is a given information identified by θ a process-input-info to a process α in a situation σ?: *holds*(process-input-info(α,θ),σ).

CQ A1.2 Is a given entity or information the resultant of a transformation by a process?

- Is a given resource κ a process-output-resource to a process α in a situation σ?: *holds*(process-output-resource(α,κ),σ).
- Is a given tru κ a process-output-tru to a process α in a situation σ?: *holds*(process-output-tru(α,κ),σ).
- Is a given information identified by θ a process-output-info to a process α in a situation σ?: *holds*(process-output-info(α,θ),σ).

CQ A1.3 Does a given entity or information control the performance of the process, but not transformed by it?

- Is a given resource κ a process-control-resource to a process α in a situation σ?: *holds*(process-control-resource(α,κ),σ).
- Is a given tru κ a process-control-tru to a process α in a situation σ?: *holds*(process-control-tru(α,κ),σ).
- Is a given information identified by θ a process-control-info to a process α in a situation σ?: *holds*(process-control-info(α,θ),σ).

CQ A1.4 Is a given entity a physical artifact that is required to perform the process, but is not transformed by it?

- Is a given resource κ a process-mechanism-resource to a process α in a situation σ?: *holds*(process-mechanism-resource(α,κ),σ).
- Is a given tru κ a process-mechanism-tru to a process α in a situation σ?: *holds*(process-mechanism-tru(α,κ),σ).
- Is a given agent ω a process-mechanism-agent to a process α in a situation σ?: *holds*(process-mechanism(α,ω),σ).

CQ A1.5 Is this a constraint which defines the scope of a given process?

- Does there exist a process-scope for a process α in a situation σ ?: ∃Q [*holds*(process-scope(α,Q),σ)].

CQ A1.6 Is this an objective of a given process?

- Does there exist a process-objective for a process α in a situation σ ?: ∃G [*holds*(process-goal(α,G),σ)].

CQ A1.7 Is this a policy which constrains the given process?

- Does there exist a process-policy for a process α in a situation σ ?: ∃Y [*holds*(process-policy(α,Y),σ)].

CQ A1.8 Is this a responsibility or role to be fulfilled by the performance of a given process?

- Does there exist a process-responsibility for a process α in a situation σ ?: *holds*(process-responsibility(α,R),σ).

CQ A1.9 Is a given process performed within a certain enterprise?

- Is a given enterprise ε a process-organization to a process α in a situation σ ?: *holds*(process-organization(α,ε),σ).

CQ A1.10 Is a given employee responsible for performing a certain process?

- Is a given employee ω a process-employee to a process α in a situation σ ?: *holds*(process-employee(α,ω),σ).

CQ A1.11 Does a given employee performing a certain process have authority over all other employees performing a certain process?

- Is a given employee ω a process-owner to a process α in a situation σ ?: *holds*(process-owner(α,ω),σ).

CQ A1.12 Is this process a process which comprises an enterprise's quality system?

- Is α a quality system process in a situation σ ?: *holds*(quality_system_process(α),σ)].

CQ A1.13 Which particular function of the enterprise's quality system does this process perform?

- Is α a quality planning process in a situation σ ?: *holds*(quality_planning_process(α),σ).
- Is α a quality control process in a situation σ ?: *holds*(quality_control_process(α),σ).
- Is α a quality assurance process in a situation σ ?: *holds*(quality_asurance_process(α),σ).

A2. Appendix II: Populated Enterprise Model for BHP Steel

```
/*****************************************************************

FILE:           bhp.rock
BHP MODEL VERSION: 3.0
DATE:           January 26, 1996
FILE START DATE:January 26, 1996

*****************************************************************/

class_frameBHP_entity
{     subclassOf:kb_Object;}
```

```
/*****************************************************************

FILE:           reference_attributes.rock
BHP MODEL VERSION: 3.0
DATE:           January 29, 1996
FILE START DATE:January 29, 1996

/*****************************************************************/

base_attribute_classBHP_entity_attribute
{     subclassOf:{ kb_Attribute, BHP_entity };
      value_type:kb_ANY;
      cardinality:multiple;
}


/*****************************************************************/
//    SETS OF ATTRIBUTES ( NOT BHP ATTRIBUTE OBJECTS )


base_attribute_classcost_history
{     subclassOf:BHP_entity_attribute;
      value_type:double;
      cardinality:multiple;
}


base_attribute_classtime_history
{     subclassOf:BHP_entity_attribute;
      value_type:double;
      cardinality:multiple;
}


base_attribute_classstatus_history
{     subclassOf:BHP_entity_attribute;
      value_type:kb_FRAME_ID;
      cardinality:multiple;
}


base_attribute_classactivity_history
{     subclassOf:BHP_entity_attribute;
      value_type:kb_FRAME_ID;
      cardinality:multiple;
}

base_attribute_classquantity_history
{     subclassOf:BHP_entity_attribute;
      value_type:double;
      cardinality:multiple;
}
```

base_attribute_classbatch_history
{ subclassOf:BHP_entity_attribute;
value_type:double;
cardinality:multiple;
}

base_attribute_classbatch_name
{ subclassOf:BHP_entity_attribute;
value_type:kb_FRAME_ID;
cardinality:multiple;
}

base_attribute_classbatch_quantity
{ subclassOf:BHP_entity_attribute;
value_type:double;
cardinality:multiple;
}

base_attribute_classactivity_route
{ subclassOf:BHP_entity_attribute;
value_type:kb_FRAME_ID;
cardinality:multiple;
}

base_attribute_classdowntime
{ subclassOf:BHP_entity_attribute;
value_type:double;
cardinality:single;
}

base_attribute_classplanned_downtime
{ subclassOf:BHP_entity_attribute;
value_type:double;
cardinality:multiple;
}

base_attribute_classrepair_time
{ subclassOf:BHP_entity_attribute;
value_type:double;
cardinality:single;
}

base_attribute_classplanned_repair_time
{ subclassOf:BHP_entity_attribute;
value_type:double;
cardinality:multiple;
}

base_attribute_classquality_classification
{ subclassOf:BHP_entity_attribute;
value_type:kb_FRAME_ID;
cardinality:single;
}

base_attribute_classheld_time_range
{ subclassOf:BHP_entity_attribute;
value_type:double;
cardinality:multiple;
}

base_attribute_class processing_time
{ subclassOf:BHP_entity_attribute;
value_type:double;
cardinality: single;
}

base_attribute_class throughput
{ subclassOf:BHP_entity_attribute;
value_type:double;
cardinality: single;
}

base_attribute_class setup_time
{ subclassOf:BHP_entity_attribute;
value_type:double;
cardinality: single;
}

base_attribute_classprobability
{ subclassOf:BHP_entity_attribute;
value_type:double;
cardinality: multiple;
}

base_attribute_classholding_nonprime_res_cost_unit
{ subclassOf:BHP_entity_attribute;
value_type:kb_FRAME_ID;
cardinality:single;
}

/***

FILE: bhp_reference_relations.rock
BHP MODEL VERSION: 2.0
DATE: September 28, 1995

```
FILE START DATE:September 28, 1995

*****************************************************************/

base_relation_classBHP_relation
{	subclassOf:{ kb_Relation, BHP_entity };
	cardinality:multiple;
	inverse_relation:NULL_VALUE;
}

/*****************************************************************/

base_relation_classhas_schedule
{	subclassOf:BHP_relation;
	cardinality:multiple;
	inverse_relation:single schedule_of;
		/* DOMAIN: bhp_activity
		   RANGE:  bhp_schedule*/
}

base_relation_classschedule_of
{	subclassOf:BHP_relation;
	cardinality:single;
	inverse_relation:multiple has_schedule;
		/* RANGE: bhp_production_unit
		   DOMAIN:  schedule*/
}

base_relation_classhas_feed
{	subclassOf:BHP_relation;
	cardinality:multiple;
	inverse_relation:multiple feed_of;
		/* DOMAIN: bhp_product
		   RANGE:  bhp_feed*/
}

base_relation_classfeed_of
{	subclassOf:BHP_relation;
	cardinality:multiple;
	inverse_relation:multiple has_feed;
		/* RANGE: bhp_produce
		   DOMAIN:  bhp_feed*/
}

base_relation_classhas_facility
{	subclassOf:BHP_relation;
	cardinality:multiple;
	inverse_relation:single facility_of;
		/* DOMAIN: bhp_organization
		   RANGE:  bhp_facility*/
}

base_relation_classfacility_of
{	subclassOf:BHP_relation;
	cardinality:single;
	inverse_relation:multiple has_facility;
		/* RANGE: bhp_organization
		   DOMAIN:  bhp_facility*/
}

base_relation_classhas_product_type
{	subclassOf:BHP_relation;
	cardinality:single;
	inverse_relation:multiple product_type_of;
		/* DOMAIN: bhp_product
		   RANGE:  bhp_product_type*/
}

base_relation_classproduct_type_of
{	subclassOf:BHP_relation;
	cardinality:multiple;
	inverse_relation:single has_product_type;
		/* RANGE: bhp_product
		   DOMAIN:  bhp_product_type*/
}

base_relation_classhas_product_route
{	subclassOf:BHP_relation;
	cardinality:multiple;
	inverse_relation:multiple product_route_of;
		/* DOMAIN: bhp_product
		   RANGE:  bhp_product_route*/
}

base_relation_classproduct_route_of
{	subclassOf:BHP_relation;
	cardinality:multiple;
	inverse_relation:multiple has_product_route;
		/* RANGE: bhp_product
		   DOMAIN:  bhp_product_route*/
```

```
}

named_instance_frame sq_actual
{       instanceOf: bhp_situation;
        attributes:
                situation_type = actual_situation;
}

named_instance_frame sq_intended
{       instanceOf: bhp_situation;
        attributes:
                situation_type = intended_situation;
}

named_instance_frame sv_actual
{       instanceOf: bhp_situation;
        attributes:
                situation_type = actual_situation;
}

named_instance_frame sv_intended
{       instanceOf: bhp_situation;
        attributes:
                situation_type = intended_situation;
}
```

```
/****************************************************************

FILE:           enterprise_bhp_attribute.rock
BHP MODEL VERSION: 2.1
DATE:           January 28, 1996
FILE START DATE:January 28, 1996

/****************************************************************/


/*****************************************************************/
//      BHP QUALITY CLASSIFICATION OBJECTS

/*****************************************************************/

class_frameWP_HSM_240_quality
{
        subclassOf:quality_classification_object;

        relations:
                elaboration_of = quality_classification;
                attribute_of = WP_produce_coil;
                empty has_distribution;

        attributes:
                unit_of_measurement = “days”;
                qualitative_range = { “prime_first_time”,
                                      “prime_held”,
                                      “rework”,
                                      “scrap” };
                probability = { 0.927, 0.03, 0.021, 0.022 };
                held_time_range = { 0, 1, 3, 1 };
}



/*****************************************************************/
//      BHP DOWNTIME & REPAIR TIME OBJECTS

/*****************************************************************/

class_frameWP_HSM_240_MTBF
{       subclassOf:     MTBF_object;

        relations:
                elaboration_of = downtime;
                attribute_of = WP_produce_coil;
                empty has_distribution;

        attributes:
                quantitative_mean = 0.025;
                quantitative_deviation = 0.0087;
                unit_of_measurement = “days”;
}
```

```
class_frame    WP_HSM_240_MTTR
{       subclassOf:     MTTR_object;

        relations:
                elaboration_of = repair_time;
                 attribute_of = WP_produce_coil;
                 empty has_distribution;

        attributes:
                quantitative_mean = 0.01389;
                quantitative_deviation = 0.00234;
                unit_of_measurement = “days”;

}
```

```
/*******************************************************************

FILE:                  enterprise_bhp_cost_object.rock
BHP MODEL VERSION: 3.0
DATE:                  January 28, 1996
FILE START DATE:January 28, 1996

*******************************************************************/

class_framenull_cost_object
{       subclassOf:res_cost_object;
        relations:
                empty cost_object_of;

        attributes:
                one_time_cost = 0;
                cost_time_rate = 0;
                cost_time_quantity_rate = 0;
}


class_frameWP_HSM_240_processing_cost_object
{       subclassOf:bhp_processing_cost_object;
        relations:
                cost_object_of = bhp_processing_start;

        attributes:
                one_time_cost = 0;
                cost_time_rate = 36767;
                cost_time_quantity_rate = 0.44343;
}



class_frameWP_HSM_240_setup_cost_object
{       subclassOf:bhp_setup_cost_object;
        relations:
                cost_object_of = bhp_setup_start;

        attributes:
                one_time_cost = 0;
                cost_time_rate = 36767;
                cost_time_quantity_rate = 0.08018;
}


class_frameWP_HSM_240_in_downtime_cost_object
{       subclassOf:bhp_in_downtime_cost_object;
        relations:
                cost_object_of = bhp_downtime_start;

        attributes:
                one_time_cost = 0;
                cost_time_rate = 99176;
                cost_time_quantity_rate = 0;
```

```
}



class_frameWP_HSM_240_holding_nonprime_cost_object
{       subclassOf:bhp_holding_nonprime_cost_object;
        relations:
                cost_object_of = bhp_hold_nonprime_start;

        attributes:
                one_time_cost = 0;
                cost_time_rate = 36767;
                cost_time_quantity_rate = 0.36325;
}



class_frameWP_HSM_240_idling_cost_object
{       subclassOf:bhp_idling_cost_object;
        relations:
                cost_object_of = bhp_idle_start;

        attributes:
                one_time_cost = 0;
                cost_time_rate = 99176;
                cost_time_quantity_rate = 0;
}
```

```
/****************************************************************

FILE:           enterprise_bhp_activity.rock
DATE:           May 6, 1996
FILE START DATE:May 6, 1996

****************************************************************/

class_frame the_bhp_steel_process
{       subclassOf: bhp_activity;
        relations:
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame the_bhp_steel_process_1
{       instanceOf: the_bhp_steel_process;
        relations:
                has_situations = { sq_actual, sv_actual };
}



class_frame process_wp
{
  subclassOf: bhp_activity;

  relations:
    has_subactivity = { process_wp_hsd, process_wp_prdn,
process_wp_qc };
    subactivity_of = the_bhp_steel_process;
    has_situations = { sq_intended, sv_intended };
}


named_instance_frame process_wp_1
{
  instanceOf: process_wp;

  relations:
        has_subactivity =  {process_wp_hsd_1,
                            process_wp_prdn_1,
                            process_wp_qc_1 };
       subactivity_of = the_bhp_steel_process_1;
        has_situations = { sq_actual, sv_actual };
}



class_frame process_wp_hsd
{       subclassOf: bhp_activity;
        relations:
                has_subactivity = { process_wp_sh_230,
                            process_wp_rcs_250,
                            process_wp_plp_280,
```

```
                process_wp_hcpd_265 };
        subactivity_of = process_wp;
        has_situations = { sq_intended, sv_intended };
}


named_instance_frame process_wp_hsd_1
{   instanceOf: process_wp_hsd;
    relations:
        has_subactivity = { process_wp_sh_230_1,
                process_wp_rcs_250_1,
                process_wp_plp_280_1,
                process_wp_hcpd_265_1 };
        subactivity_of = process_wp_1;
        has_situations = { sq_actual, sv_actual };
}


class_frame process_wp_prdn
{   subclassOf: bhp_activity;
    relations:
        has_subactivity = { process_wp_hsm_240,
                process_wp_pkl_270 };
        subactivity_of = process_wp;
        has_situations = { sq_intended, sv_intended };
}


named_instance_frame process_wp_prdn_1
{   instanceOf: process_wp_prdn;
    relations:
        has_subactivity = { process_wp_hsm_240_1,
                process_wp_pkl_270_1 };
        subactivity_of = process_wp_1;
        has_situations = { sq_actual, sv_actual };
}


class_frame process_wp_qc
{   subclassOf: bhp_activity;
    relations:
        has_subactivity = process_wp_hcpf_260;
        subactivity_of = process_wp;
        has_situations = { sq_intended, sv_intended };
}


named_instance_frame process_wp_qc_1
{   instanceOf: process_wp_qc;
    relations:
        has_subactivity = process_wp_hcpf_260_1;
        subactivity_of = process_wp_1;
        has_situations = { sq_actual, sv_actual };
}

class_frame process_wp_sh_230
{   subclassOf: bhp_activity;
    relations:
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame process_wp_sh_230_1
{   instanceOf: process_wp_sh_230;
    relations:
        has_situations = { sq_actual, sv_actual };
}



class_frame process_wp_hsm_240
{   subclassOf: bhp_activity;
    relations:
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame process_wp_hsm_240_1
{
  instanceOf: process_wp_hsm_240;

  relations:
   ( enabled_by = { es_process_wp_hsm_240_1 };
     slot_control: enabled_by;
   )
      has_situations = { sq_actual, sv_actual };
}



class_frame process_wp_rcs_250
{   subclassOf: bhp_activity;
    relations:
        has_situations = { sq_intended, sv_intended };
}


named_instance_frame process_wp_rcs_250_1
{
  instanceOf: process_wp_rcs_250;

  relations:
   ( enabled_by = { es_wp_rcs_250_1 };
     slot_control: enabled_by;
   )
      has_situations = { sq_actual, sv_actual };
}

named_instance_frame process_wp_rcs_250_2
{
```

```
  instanceOf: process_wp_rcs_250;

  relations:
        has_situations = { sq_actual, sv_actual };
}


class_frame process_wp_hcpf_260
{       subclassOf: bhp_activity;
        relations:
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame process_wp_hcpf_260_1
{       instanceOf: process_wp_hcpf_260;
        relations:
                has_situations = { sq_actual, sv_actual };
}


class_frame process_wp_hcpd_265
{       subclassOf: bhp_activity;
        relations:
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame process_wp_hcpd_265_1
{       instanceOf: process_wp_hcpd_265;
        relations:
                has_situations = { sq_actual, sv_actual };
}



class_frame process_wp_pkl_270
{       subclassOf: bhp_activity;
        relations:
                has_situations = { sq_intended, sv_intended };
}


named_instance_frame process_wp_pkl_270_1
{
  instanceOf: process_wp_pkl_270;

  relations:
    ( enabled_by = { es_wp_pkl_270_1 };
      slot_control: enabled_by;
    )
        has_situations = { sq_actual, sv_actual };
}

class_frame process_wp_plp_280
{       subclassOf: bhp_activity;
        relations:
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame process_wp_plp_280_1
{       instanceOf: process_wp_plp_280;
        relations:
                has_situations = { sq_actual, sv_actual };
}
```

```
/*****************************************************************

FILE:		enterprise_bhp_resource2.rock
BHP MODEL VERSION: 3.0
DATE:		May 6, 1996
FILE START DATE:May 6, 1996

*****************************************************************/



named_instance_frame bhp_facility_1
{	instanceOf: bhp_facility;
	relations:
		has_situations = { sq_actual, sv_actual };
}

named_instance_frame bhp_feed_1
{	instanceOf: bhp_feed;
	relations:
		has_situations = { sq_actual, sv_actual };
}

named_instance_frame bhp_product_type_1
{	instanceOf: bhp_product_type;
	relations:
		has_situations = { sq_actual, sv_actual };
}

class_frame bhp_wp_facility
{	subclassOf: bhp_facility;
	relations:
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame bhp_wp_facility_1
{	instanceOf: bhp_wp_facility;
	relations:
		has_situations = { sq_actual, sv_actual };
}


class_frame bhp_wp_feed
{	subclassOf: bhp_feed;
	relations:
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame bhp_wp_feed_1
{	instanceOf: bhp_wp_feed;
	relations:
		has_situations = { sq_actual, sv_actual };
}

class_frame bhp_wp_product_type
{	subclassOf: bhp_product_type;
	relations:
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame bhp_wp_product_type_1
{	instanceOf: bhp_wp_product_type;
	relations:
		has_situations = { sq_actual, sv_actual };
}


class_frame wp_sh_230_facility
{	subclassOf: bhp_wp_facility;
	relations: has_situations = { sq_intended, sv_intended};
}

named_instance_frame wp_sh_230_facility_1
{	instanceOf: wp_sh_230_facility;
	relations:
		has_situations = { sq_actual, sv_actual };
}


class_frame wp_hsm_240_facility
{	subclassOf: bhp_wp_facility;
	relations:
		has_situations = { sq_intended, sv_intended};
}

named_instance_frame wp_hsm_240_facility_1
{	instanceOf: wp_hsm_240_facility;
	relations:
		has_situations = { sq_actual, sv_actual };
}



class_frame wp_rcs_250_facility
{	subclassOf: bhp_wp_facility;
	relations:
		has_situations = { sq_intended, sv_intended};
}

named_instance_frame wp_rcs_250_facility_1
{	instanceOf: wp_rcs_250_facility;
	relations:
		has_situations = { sq_actual, sv_actual };
}


class_frame wp_hcpf_260_facility
{	subclassOf: bhp_wp_facility;
	relations: has_situations = { sq_intended, sv_intended};
}
```

```
named_instance_frame wp_hcpf_260_facility_1
{	instanceOf: wp_hcpf_260_facility;
	relations:
		has_situations = { sq_actual, sv_actual };
}


class_frame wp_hcpd_265_facility
{	subclassOf: bhp_wp_facility;
	relations: has_situations = { sq_intended, sv_intended};
}

named_instance_frame wp_hcpd_265_facility_1
{	instanceOf: wp_hcpd_265_facility;
	relations:
		has_situations = { sq_actual, sv_actual };
}



class_frame wp_pkl_270_facility
{	subclassOf: bhp_wp_facility;
	relations:
		has_situations = { sq_intended, sv_intended };
}


named_instance_frame wp_pkl_270_facility_1
{	instanceOf: wp_pkl_270_facility;
	relations:
		has_situations = { sq_actual, sv_actual };
}


class_frame wp_plp_280_facility
{	subclassOf: bhp_wp_facility;
	relations:
		has_situations = { sq_intended, sv_intended };
}


named_instance_frame wp_plp_280_facility_1
{	instanceOf: wp_plp_280_facility;
	relations:
		has_situations = { sq_actual, sv_actual };
}



class_frame wp_slab
{	subclassOf: bhp_wp_feed;
	relations:
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_slab_1
{	instanceOf: wp_slab;
	relations:
		has_situations = { sq_actual, sv_actual };
}



class_frame blackform
{	subclassOf: bhp_wp_product_type;
	relations:
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame blackform_1
{	instanceOf: blackform;
	relations:
		has_situations = { sq_actual, sv_actual };
}


class_frame wp_raw_coil
{	subclassOf: bhp_wp_feed;
	relations:
		has_situations = { sq_intended, sv_intended };
}


named_instance_frame wp_raw_coil_1
{	instanceOf: wp_raw_coil;
	relations:
		has_situations = { sq_actual, sv_actual };
}


class_frame wp_pickled_coil
{	subclassOf: bhp_wp_feed;
	relations:
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_pickled_coil_1
{	instanceOf: wp_pickled_coil;
	relations:
		has_situations = { sq_actual, sv_actual };
}



class_frame brightform
{	subclassOf: bhp_wp_product_type;
	relations:
```

```
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame brightform_1
{   instanceOf: brightform;
    relations:
        has_situations = { sq_actual, sv_actual };
}
```

```
/*****************************************************************

FILE:           enterprise_bhp_state.rock
BHP MODEL VERSION: 3.0
DATE:           January 28, 1996
FILE START DATE:January 28, 1996

*****************************************************************/



class_frame es_process_wp_sh_230
{   subclassOf:bhp_state_conjunction;
    relations:
        enables = process_wp_sh_230;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame es_process_wp_sh_230_1
{   instanceOf:es_process_wp_sh_230;
    relations:
        enables = process_wp_sh_230_1;
        has_situations = { sq_actual, sv_actual };
}


class_frame cbs_process_wp_sh_230
{   subclassOf:bhp_state_conjunction;
    relations:
        caused_by = process_wp_sh_230;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame cbs_process_wp_sh_230_1
{   instanceOf:cbs_process_wp_sh_230;
    relations:
        caused_by = process_wp_sh_230_1;
        has_situations = { sq_actual, sv_actual };
}


class_frame us_process_wp_sh_230
{   subclassOf:usage;
    relations:
        conjunct_of = es_process_wp_sh_230;
        uses = wp_sh_230_facility;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame us_process_wp_sh_230_1
{   instanceOf:us_process_wp_sh_230;
    relations:
        uses = wp_sh_230_facility_1;
        conjunct_of = es_process_wp_sh_230_1;
        has_situations = { sq_actual, sv_actual };
```

```
}

class_frame rs_process_wp_sh_230
{       subclassOf:release;
        relations:
                conjunct_of = cbs_process_wp_sh_230;
                released = wp_sh_230_facility;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame rs_process_wp_sh_230_1
{       instanceOf:rs_process_wp_sh_230;
        relations:
                released = wp_sh_230_facility_1;
                conjunct_of = cbs_process_wp_sh_230_1;
                has_situations = { sq_actual, sv_actual };
}

class_frame ps_process_wp_sh_230
{       subclassOf:produce;
        relations:
                conjunct_of = cbs_process_wp_sh_230;
                produced = wp_slab;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame ps_process_wp_sh_230_1
{       instanceOf:ps_process_wp_sh_230;
        relations:
                produced = wp_slab_1;
                conjunct_of = cbs_process_wp_sh_230_1;
                has_situations = { sq_actual, sv_actual };
}

class_frame es_process_wp_hsm_240
{       subclassOf: bhp_state_conjunction;
        relations:
                enables = process_wp_hsm_240;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame es_process_wp_hsm_240_1
{
  instanceOf: es_process_wp_hsm_240;

  relations:
    ( enables = { process_wp_hsm_240_1 };
      slot_control: enables;
    )
        has_situations = { sq_actual, sv_actual };
}

class_frame ps_process_wp_hsm_240
{       subclassOf: produce;
        relations:
                conjunct_of = cbs_process_wp_hsm_240;
                produced = wp_raw_coil;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame ps_process_wp_hsm_240_1
{
  instanceOf: ps_process_wp_hsm_240;

  relations:
    ( produced = wp_raw_coil_1;
      slot_control: produced;
    )
        conjunct_of = cbs_process_wp_hsm_240_1;
        has_situations = { sq_actual, sv_actual };
}

class_frame rs_process_wp_hsm_240
{       subclassOf: release;
        relations:
                conjunct_of = cbs_process_wp_hsm_240;
                released = wp_hsm_240_facility;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame rs_process_wp_hsm_240_1
{
  instanceOf: rs_process_wp_hsm_240;

  relations:
    ( released = wp_hsm_240_facility_1;
      slot_control: released;
    )
        conjunct_of = cbs_process_wp_hsm_240_1;
        has_situations = { sq_actual, sv_actual };
}

class_frame cbs_process_wp_hsm_240
{       subclassOf: bhp_state_conjunction;
        relations:
                caused_by = process_wp_hsm_240;
                has_situations = { sq_intended, sv_intended };
}
```

```
named_instance_frame cbs_process_wp_hsm_240_1
{
 instanceOf: cbs_process_wp_hsm_240;

 relations:
   ( caused_by = { process_wp_hsm_240_1 };
    slot_control: caused_by;
   )
        has_situations = { sq_actual, sv_actual };
}



class_frame cs_process_wp_hsm_240
{    subclassOf: consumption;
     relations:
        consumes = wp_slab;
        conjunct_of = es_process_wp_hsm_240;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame cs_process_wp_hsm_240_1
{
 instanceOf: cs_process_wp_hsm_240;

 relations:
   ( conjunct_of = { es_process_wp_hsm_240_1 };
    slot_control: conjunct_of;
   )
   ( consumes = { wp_slab_1 };
    slot_control: consumes;
   )
        has_situations = { sq_actual, sv_actual };
}



class_frame us_process_wp_hsm_240
{    subclassOf: usage;
     relations:
        uses = wp_hsm_240_facility;
        conjunct_of = es_process_wp_hsm_240;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame us_process_wp_hsm_240_1
{
 instanceOf: us_process_wp_hsm_240;

 relations:
   ( conjunct_of = { es_process_wp_hsm_240_1 };
    slot_control: conjunct_of;
   )
   ( uses = { wp_hsm_240_facility_1 };
    slot_control: uses;
   )
        has_situations = { sq_actual, sv_actual };
}



class_frame es_wp_rcs_250
{    subclassOf: bhp_state_conjunction;
     relations:
        enables = process_wp_rcs_250;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame es_wp_rcs_250_1
{
 instanceOf: es_wp_rcs_250;

 relations:
   ( enables = { process_wp_rcs_250_1 };
    slot_control: enables;
   )
        has_situations = { sq_actual, sv_actual };
}

named_instance_frame es_wp_rcs_250_2
{
 instanceOf: es_wp_rcs_250;

 relations:
   ( enables = { process_wp_rcs_250_2 };
    slot_control: enables;
   )
        has_situations = { sq_actual, sv_actual };
}

class_frame or_wp_rcs_250
{    subclassOf: bhp_state_disjunction;
     relations:
        conjunct_of = es_wp_rcs_250;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame or_wp_rcs_250_1
{    instanceOf: or_wp_rcs_250;
     relations:
        conjunct_of = es_wp_rcs_250;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame or_wp_rcs_250_2
{    instanceOf: or_wp_rcs_250;
     relations:
        conjunct_of = es_wp_rcs_250_2;
        has_situations = { sq_intended, sv_intended };
}
```

```
class_frame use_wp_rcs_250_facility
{       subclassOf: usage;
        relations:
                uses = wp_rcs_250_facility;
                conjunct_of = es_wp_rcs_250;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame use_wp_rcs_250_facility_1
{
  instanceOf: use_wp_rcs_250_facility;

  relations:
    ( uses = wp_rcs_250_facility_1;
      slot_control: uses;
    )
        conjunct_of = es_wp_rcs_250_1;
                has_situations = { sq_actual, sv_actual };
}

named_instance_frame use_wp_rcs_250_facility_2
{
  instanceOf: use_wp_rcs_250_facility;

  relations:
    ( uses = wp_rcs_250_facility_1;
      slot_control: uses;
    )
        conjunct_of = es_wp_rcs_250_2;
                has_situations = { sq_actual, sv_actual };
}



class_frame use_wp_raw_coil
{       subclassOf: usage;
        relations:
                uses = wp_raw_coil;
                disjunct_of = or_wp_rcs_250;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame use_wp_raw_coil_1
{
  instanceOf: use_wp_raw_coil;

  relations:
    ( disjunct_of = {  or_wp_rcs_250_1};
      slot_control: disjunct_of;
    )
    ( uses = { wp_raw_coil_1 };
      slot_control: uses;
    )
                has_situations = { sq_actual, sv_actual };
}

named_instance_frame use_wp_raw_coil_2
{
  instanceOf: use_wp_raw_coil;

  relations:
    ( disjunct_of = {  or_wp_rcs_250_2};
      slot_control: disjunct_of;
    )
    ( uses = { wp_raw_coil_1 };
      slot_control: uses;
    )
                has_situations = { sq_actual, sv_actual };
}


class_frame consume_wp_raw_coil
{       subclassOf: usage;
        relations:
                consumes = wp_raw_coil;
                disjunct_of = or_wp_rcs_250;
                has_situations = { sq_intended, sv_intended };
}



class_frame pro_wp_rcs_250
{       subclassOf: bhp_state_conjunction;
        relations:
                caused_by = process_wp_rcs_250;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame pro_wp_rcs_250_1
{
  instanceOf: pro_wp_rcs_250;

  relations:
    ( caused_by = { process_wp_rcs_250_1 };
      slot_control: caused_by;
    )
    ( conjuncts = { rel_wp_rcs_250_facility_1, rel_wp_raw_coil_1 };
      slot_control: conjuncts;
    )
                has_situations = { sq_actual, sv_actual };
}

named_instance_frame pro_wp_rcs_250_2
{
  instanceOf: pro_wp_rcs_250;

  relations:
    ( caused_by = { process_wp_rcs_250_2 };
```

```
    slot_control: caused_by;
  )
  ( conjuncts = { rel_wp_rcs_250_facility_2, rel_wp_raw_coil_2 };
    slot_control: conjuncts;
  )
        has_situations = { sq_actual, sv_actual };
}


class_frame or2_wp_rcs_250
{    subclassOf: bhp_state_disjunction;
     relations:
        conjuncts = pro_wp_rcs_250;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame or2_wp_rcs_250_1
{    instanceOf: or2_wp_rcs_250;
     relations:
        conjuncts = pro_wp_rcs_250_1;
        has_situations = { sq_actual, sv_actual };
}

named_instance_frame or2_wp_rcs_250_2
{    instanceOf: or2_wp_rcs_250;
     relations:
        conjuncts = pro_wp_rcs_250_2;
        has_situations = { sq_actual, sv_actual };
}


class_frame rel_wp_raw_coil
{    subclassOf: release;
     relations:
        released = wp_raw_coil;
        disjunct_of = or2_wp_rcs_250;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame rel_wp_raw_coil_1
{
 instanceOf: rel_wp_raw_coil;

 relations:
  ( released = wp_raw_coil_1;
    slot_control: released;
  )
  ( disjunct_of = { or2_wp_rcs_250_1 };
    slot_control: disjunct_of;
  )
        has_situations = { sq_actual, sv_actual };
}

named_instance_frame rel_wp_raw_coil_2
{
 instanceOf: rel_wp_raw_coil;

 relations:
  ( released = wp_raw_coil_1;
    slot_control: released;
  )
  ( disjunct_of = { or2_wp_rcs_250_2 };
    slot_control: disjunct_of;
  )
        has_situations = { sq_actual, sv_actual };
}


class_frame ps_wp_rcs_250
{    subclassOf: produce;
     relations:
        produced = blackform;
        disjunct_of = or2_wp_rcs_250;
        has_situations = { sq_intended, sv_intended };
}


class_frame rel_wp_rcs_250_facility
{    subclassOf: release;
     relations:
        released = wp_rcs_250_facility;
        conjunct_of = pro_wp_rcs_250;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame rel_wp_rcs_250_facility_1
{
 instanceOf: rel_wp_rcs_250_facility;

 relations:
  ( released = wp_rcs_250_facility_1;
    slot_control: released;
  )
        has_situations = { sq_actual, sv_actual };
}

named_instance_frame rel_wp_rcs_250_facility_2
{
 instanceOf: rel_wp_rcs_250_facility;

 relations:
  ( released = wp_rcs_250_facility_1;
    slot_control: released;
  )
        has_situations = { sq_actual, sv_actual };
}
```

```
class_frame es_wp_hcpf_260
{	subclassOf: bhp_state_conjunction;
	relations:
		enables = process_wp_hcpf_260;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame es_wp_hcpf_260_1
{	instanceOf: es_wp_hcpf_260;
	relations:
		enables = process_wp_hcpf_260_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame cbs_wp_hcpf_260
{	subclassOf: bhp_state_conjunction;
	relations:
		caused_by = process_wp_hcpf_260;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame cbs_wp_hcpf_260_1
{	instanceOf: cbs_wp_hcpf_260;
	relations:
		caused_by = process_wp_hcpf_260_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame or1_wp_hcpf_260
{	subclassOf: bhp_state_disjunction;
	relations:
		conjunct_of = es_wp_hcpf_260;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame or1_wp_hcpf_260_1
{	instanceOf: or1_wp_hcpf_260;
	relations:
		conjunct_of = es_wp_hcpf_260_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame cs1_wp_hcpf_260
{	subclassOf: consumption;
	relations:
		consumes = wp_raw_coil;
		disjunct_of = or1_wp_hcpf_260;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame cs1_wp_hcpf_260_1
{	instanceOf: cs1_wp_hcpf_260;
	relations:
		consumes = wp_raw_coil_1;
		disjunct_of = or1_wp_hcpf_260_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame cs2_wp_hcpf_260
{	subclassOf: consumption;
	relations:
		consumes = wp_pickled_coil;
		disjunct_of = or1_wp_hcpf_260;
		has_situations = { sq_intended, sv_intended };
}

class_frame us_wp_hcpf_260
{	subclassOf: usage;
	relations:
		uses = wp_hcpf_260_facility;
		conjunct_of = es_wp_hcpf_260;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame us_wp_hcpf_260_1
{	instanceOf: us_wp_hcpf_260;
	relations:
		uses = wp_hcpf_260_facility_1;
		conjunct_of = es_wp_hcpf_260_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame rs_wp_hcpf_260
{	subclassOf: release;
	relations:
		released = wp_hcpf_260_facility;
		conjunct_of = cbs_wp_hcpf_260;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame rs_wp_hcpf_260_1
{	instanceOf: rs_wp_hcpf_260;
	relations:
		released = wp_hcpf_260_facility_1;
		conjunct_of = cbs_wp_hcpf_260_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame or2_wp_hcpf_260
```

```
{	subclassOf: bhp_state_disjunction;
	relations:
		conjunct_of = cbs_wp_hcpf_260;
		has_situations = { sq_intended, sv_intended };
}


named_instance_frame or2_wp_hcpf_260_1
{	instanceOf: or2_wp_hcpf_260;
	relations:
		conjunct_of = cbs_wp_hcpf_260_1;
		has_situations = { sq_intended, sv_intended };
}


class_frame ps1_wp_hcpf_260
{	subclassOf: produce;
	relations:
		produced = blackform;
		disjunct_of = or2_wp_hcpf_260;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame ps1_wp_hcpf_260_1
{	instanceOf: ps1_wp_hcpf_260;
	relations:
		produced = blackform;
		disjunct_of = or2_wp_hcpf_260_1;
		has_situations = { sq_actual, sv_actual };
}



class_frame ps2_wp_hcpf_260
{	subclassOf: produce;
	relations:
		produced = brightform;
		disjunct_of = or2_wp_hcpf_260;
		has_situations = { sq_intended, sv_intended };
}



class_frame es_wp_hcpd_265
{	subclassOf: bhp_state_conjunction;
	relations:
		enables = process_wp_hcpd_265;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame es_wp_hcpd_265_1
{	instanceOf: es_wp_hcpd_265;
	relations:
		enables = process_wp_hcpd_265_1;
		has_situations = { sq_actual, sv_actual };
}


class_frame cbs_wp_hcpd_265
{	subclassOf: bhp_state_conjunction;
	relations:
		caused_by = process_wp_hcpd_265;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame cbs_wp_hcpd_265_1
{	instanceOf: cbs_wp_hcpd_265;
	relations:
		caused_by = process_wp_hcpd_265_1;
		has_situations = { sq_actual, sv_actual };
}


class_frame us_wp_hcpd_265
{	subclassOf: usage;
	relations:
		uses = wp_hcpd_265_facility;
		conjunct_of = es_wp_hcpd_265;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame us_wp_hcpd_265_1
{	instanceOf: us_wp_hcpd_265;
	relations:
		uses = wp_hcpd_265_facility_1;
		conjunct_of = es_wp_hcpd_265_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame or_wp_hcpd_265
{	subclassOf: bhp_state_disjunction;
	relations:
		conjunct_of = es_wp_hcpd_265;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame or_wp_hcpd_265_1
{	instanceOf: or_wp_hcpd_265;
	relations:
		conjunct_of = es_wp_hcpd_265_1;
		has_situations = { sq_actual, sv_actual };
}


class_frame cs1_wp_hcpd_265
{	subclassOf: consumption;
	relations:
		consumes = blackform;
		disjunct_of = or_wp_hcpd_265;
		has_situations = { sq_intended, sv_intended };
```

```
}

named_instance_frame cs1_wp_hcpd_265_1
{       instanceOf: cs1_wp_hcpd_265;
        relations:
                consumes = blackform_1;
                disjunct_of = or_wp_hcpd_265_1;
                has_situations = { sq_actual, sv_actual };
}

class_frame cs2_wp_hcpd_265
{       subclassOf: consumption;
        relations:
                consumes = brightform;
                disjunct_of = or_wp_hcpd_265;
                has_situations = { sq_intended, sv_intended };
}

class_frame rs_wp_hcpd_265
{       subclassOf: release;
        relations:
                released = wp_hcpd_265_facility;
                conjunct_of = cbs_wp_hcpd_265;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame rs_wp_hcpd_265_1
{       instanceOf: rs_wp_hcpd_265;
        relations:
                released = wp_hcpd_265_facility_1;
                conjunct_of = cbs_wp_hcpd_265_1;
                has_situations = { sq_actual, sv_actual };
}

class_frame es_wp_pkl_270
{       subclassOf: bhp_state_conjunction;
        relations:
                enables = process_wp_pkl_270;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame es_wp_pkl_270_1
{
  instanceOf:es_wp_pkl_270;

  relations:
    ( enables = { process_wp_pkl_270_1 };
      slot_control: enables;
    )

    ( conjuncts = { use_wp_pkl_270_facility_1, cons_wp_raw_coil_1 };
      slot_control: conjuncts;
    )
                has_situations = { sq_actual, sv_actual };
}

class_frame cons_wp_raw_coil
{       subclassOf: consumption;
        relations:
                consumes = wp_raw_coil;
                conjunct_of = es_wp_pkl_270;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame cons_wp_raw_coil_1
{
  instanceOf: consumption;

  relations:
    ( conjunct_of = { es_wp_pkl_270_1  };
      slot_control: conjunct_of;
    )
    ( consumes = { wp_raw_coil_1 };
      slot_control: consumes;
    )
                has_situations = { sq_actual, sv_actual };
}

class_frame use_wp_pkl_270_facility
{       subclassOf: usage;
        relations:
                uses = wp_pkl_270_facility;
                conjunct_of = es_wp_pkl_270;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame use_wp_pkl_270_facility_1
{
  instanceOf: use_wp_pkl_270_facility;

  relations:
    ( conjunct_of = { es_wp_pkl_270_1 };
      slot_control: conjunct_of;
    )
    ( uses = { wp_pkl_270_facility_1 };
      slot_control: uses;
    )
```

```
            has_situations = { sq_actual, sv_actual };
}


class_frame pro_wp_pkl_270
{       subclassOf: bhp_state_conjunction;
        relations:
                caused_by = process_wp_pkl_270;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame pro_wp_pkl_270_1
{
  instanceOf: pro_wp_pkl_270;

  relations:
    ( caused_by = { process_wp_pkl_270_1 };
      slot_control: caused_by;
    )
                has_situations = { sq_actual, sv_actual };
}



class_frame pro_wp_pickled_coil
{       subclassOf: produce;
        relations:
                produced = wp_pickled_coil;
                conjunct_of = pro_wp_pkl_270;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame pro_wp_pickled_coil_1
{
  instanceOf: pro_wp_pickled_coil;

  relations:
    ( produced = wp_pickled_coil_1;
      slot_control: produced;
    )
    ( conjunct_of = { pro_wp_pkl_270_1 };
      slot_control: conjunct_of;
    )
                has_situations = { sq_actual, sv_actual };
}



class_frame rel_wp_pkl_270_facility
{       subclassOf: release;
        relations:
                released = wp_pkl_270_facility;
                conjunct_of = pro_wp_pkl_270;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame rel_wp_pkl_270_facility_1
{
  instanceOf: rel_wp_pkl_270_facility;

  relations:
    ( released = wp_pkl_270_facility_1;
      slot_control: released;
    )
                has_situations = { sq_actual, sv_actual };
}



class_frame es_wp_plp_280
{       subclassOf: bhp_state_conjunction;
        relations:
                enables = process_wp_plp_280;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame es_wp_plp_280_1
{
  instanceOf: es_wp_plp_280;

  relations:
    ( enables = { process_wp_plp_280_1 };
      slot_control: enables;
    )
                has_situations = { sq_actual, sv_actual };
}



class_frame or_wp_plp_280
{       subclassOf: bhp_state_disjunction;
        relations:
                conjunct_of = es_wp_plp_280;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame or_wp_plp_280_1
{       instanceOf: or_wp_plp_280;
        relations:
                conjunct_of = es_wp_plp_280_1;
                has_situations = { sq_intended, sv_intended };
}



class_frame use_wp_plp_280_facility
{       subclassOf: usage;
        relations:
                uses = wp_plp_280_facility;
```

```
        conjunct_of = es_wp_plp_280;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame use_wp_plp_280_facility_1
{
  instanceOf: use_wp_plp_280_facility;

  relations:
    ( uses = wp_plp_280_facility_1;
      slot_control: uses;
    )
      conjunct_of = es_wp_plp_280_1;
        has_situations = { sq_actual, sv_actual };
}



class_frame us_wp_pickled_coil
{     subclassOf: usage;
      relations:
        uses = wp_pickled_coil;
        disjunct_of = or_wp_plp_280;
        has_situations = { sq_intended, sv_intended };
}


class_frame cs_wp_pickled_coil
{     subclassOf: consumption;
      relations:
        consumes = wp_pickled_coil;
        disjunct_of = or_wp_plp_280;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame cs_wp_pickled_coil_1
{     instanceOf: cs_wp_pickled_coil;
      relations:
        consumes = wp_pickled_coil_1;
        disjunct_of = or_wp_plp_280_1;
        has_situations = { sq_actual, sv_actual };
}



class_frame pro_wp_plp_280
{     subclassOf: bhp_state_conjunction;
      relations:
        caused_by = process_wp_plp_280;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame pro_wp_plp_280_1
{
  instanceOf: pro_wp_plp_280;

  relations:
    ( caused_by = { process_wp_plp_280_1 };
      slot_control: caused_by;
    )
        has_situations = { sq_actual, sv_actual };
}


class_frame or2_wp_plp_280
{     subclassOf: bhp_state_disjunction;
      relations:
        conjunct_of = pro_wp_plp_280;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame or2_wp_plp_280_1
{     instanceOf: or2_wp_plp_280;
      relations:
        conjunct_of = pro_wp_plp_280_1;
        has_situations = { sq_actual, sv_actual };
}



class_frame rel_wp_pickled_coil
{     subclassOf: release;
      relations:
        released = wp_pickled_coil;
        conjunct_of = or2_wp_plp_280;
        has_situations = { sq_intended, sv_intended };
}


class_frame ps_wp_pickled_coil
{     subclassOf: produce;
      relations:
        produced = brightform;
        conjunct_of = or2_wp_plp_280;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame ps_wp_pickled_coil_1
{     instanceOf: ps_wp_pickled_coil;
      relations:
        produced = brightform_1;
        conjunct_of = or2_wp_plp_280_1;
        has_situations = { sq_actual, sv_actual };
}


class_frame rel_wp_plp_280_facility
```

```
{       subclassOf: release;
        relations:
                released = wp_plp_280_facility;
                conjunct_of = pro_wp_plp_280;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame rel_wp_plp_280_facility_1
{
  instanceOf: rel_wp_plp_280_facility;

  relations:
    ( released = wp_plp_280_facility_1;
      slot_control: released;
    )
                has_situations = { sq_actual, sv_actual };
}



class_frame es_the_bhp_steel_process
{       subclassOf:bhp_state_conjunction;
        relations:
                enables = the_bhp_steel_process;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame es_the_bhp_steel_process_1
{       instanceOf: es_the_bhp_steel_process;
        relations:
                enables = the_bhp_steel_process_1;
                has_situations = { sq_actual, sv_intended };
}


class_frame cbs_the_bhp_steel_process
{       subclassOf:bhp_state_conjunction;
        relations:
                caused_by = the_bhp_steel_process;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame cbs_the_bhp_steel_process_1
{       instanceOf: cbs_the_bhp_steel_process;
        relations:
                caused_by = the_bhp_steel_process_1;
                has_situations = { sq_actual, sv_intended };
}

class_frame us_the_bhp_steel_process
{       subclassOf: usage;
        relations:
                uses = bhp_facility;
                conjunct_of = es_the_bhp_steel_process;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame us_the_bhp_steel_process_1
{       instanceOf: us_the_bhp_steel_process;
        relations:
                uses = bhp_facility_1;
                conjunct_of = es_the_bhp_steel_process_1;
                has_situations = { sq_actual, sv_actual };
}

class_frame rs_the_bhp_steel_process
{       subclassOf: release;
        relations:
                released = bhp_facility;
                conjunct_of = cbs_the_bhp_steel_process;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame rs_the_bhp_steel_process_1
{       instanceOf: rs_the_bhp_steel_process;
        relations:
                released = bhp_facility_1;
                conjunct_of = rs_the_bhp_steel_process_1;
                has_situations = { sq_actual, sv_actual };
}



class_frame cs_the_bhp_steel_process
{       subclassOf: consumption;
        relations:
                consumes = bhp_feed;
                conjunct_of = es_the_bhp_steel_process;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame cs_the_bhp_steel_process_1
{       instanceOf: cs_the_bhp_steel_process;
        relations:
                consumes = bhp_feed_1;
                conjunct_of = es_the_bhp_steel_process_1;
                has_situations = { sq_actual, sv_actual };
}

class_frame ps_the_bhp_steel_process
{       subclassOf: produce;
        relations:
                produced = bhp_product_type;
                conjunct_of = cbs_the_bhp_steel_process;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame ps_the_bhp_steel_process_1
{       instanceOf: ps_the_bhp_steel_process;
        relations:
```

```
		produced = bhp_product_type_1;
		conjunct_of = rs_the_bhp_steel_process_1;
		has_situations = { sq_actual, sv_actual };
}


class_frame es_process_wp
{	subclassOf: bhp_state_conjunction;
	relations:
		enables = process_wp;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame es_process_wp_1
{	instanceOf: es_process_wp;
	relations:
		enables = process_wp_1;
		has_situations = { sq_actual, sv_actual };
}


class_frame cbs_process_wp
{	subclassOf: bhp_state_conjunction;
	relations:
		caused_by = process_wp;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame cbs_process_wp_1
{	instanceOf: cbs_process_wp;
	relations:
		caused_by = process_wp_1;
		has_situations = { sq_actual, sv_actual };
}


class_frame us_process_wp
{	subclassOf: usage;
	relations:
		conjuncts = es_process_wp;
		uses = bhp_wp_facility;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame us_process_wp_1
{	instanceOf: us_process_wp;
	relations:
		conjuncts = es_process_wp_1;
		uses = bhp_wp_facility_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame rs_process_wp
{	subclassOf: release;
	relations:
		conjuncts = cbs_process_wp;
		released = bhp_wp_facility;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame rs_process_wp_1
{	instanceOf: rs_process_wp;
	relations:
		conjuncts = cbs_process_wp_1;
		released = bhp_wp_facility_1;
		has_situations = { sq_actual, sv_actual };
}


class_frame cs_process_wp
{	subclassOf: consumption;
	relations:
		conjuncts = es_process_wp;
		consumes = bhp_wp_feed;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame cs_process_wp_1
{	instanceOf: cs_process_wp;
	relations:
		conjuncts = es_process_wp_1;
		consumes = bhp_wp_feed_1;
		has_situations = { sq_actual, sv_actual };
}


class_frame ps_process_wp
{	subclassOf: produce;
	relations:
		conjuncts = cbs_process_wp;
		consumes = bhp_wp_product_type;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame ps_process_wp_1
{	instanceOf: ps_process_wp;
	relations:
		conjuncts = cbs_process_wp_1;
		consumes = bhp_wp_product_type_1;
		has_situations = { sq_actual, sv_actual };
}
```

```
class_frame es_process_wp_hsd
{       subclassOf: bhp_state_conjunction;
        relations:
                enables = process_wp_hsd;
                has_situations = { sq_intended, sv_intended };
                conjuncts = { us_process_wp,
                                use_wp_raw_coil,
                                us_wp_pickled_coil,
                                cs1_wp_hcpd_265,
                                cs2_wp_hcpd_265 };
}

named_instance_frame es_process_wp_hsd_1
{       instanceOf: es_process_wp_hsd;
        relations:
                enables = process_wp_hsd_1;
                has_situations = { sq_actual, sv_actual };
                conjuncts = { us_process_wp_1,
                                use_wp_raw_coil_1,
                                cs1_wp_hcpd_265_1 };
}

class_frame cbs_process_wp_hsd
{       subclassOf: bhp_state_conjunction;
        relations:
                caused_by = process_wp_hsd;
                has_situations = { sq_intended, sv_intended };
                conjuncts = { rs_process_wp,
                                ps_process_wp_sh_230,
                                rel_wp_raw_coil,
                                rel_wp_pickled_coil };
}

named_instance_frame cbs_process_wp_hsd_1
{       instanceOf: cbs_process_wp_hsd;
        relations:
                enables = process_wp_hsd_1;
                has_situations = { sq_actual, sv_actual };
                conjuncts = { rs_process_wp_1,
                                ps_process_wp_sh_230_1,
                                rel_wp_raw_coil_1 };
}

class_frame es_process_wp_prdn
{       subclassOf: bhp_state_conjunction;
        relations:
                conjunct_of = es_process_wp;
                enables = process_wp_prdn;
                has_situations = { sq_intended, sv_intended };
                conjuncts = { cs_process_wp_hsm_240 };
}

named_instance_frame es_process_wp_prdn_1
{       instanceOf: es_process_wp_prdn;
        relations:
                conjunct_of = es_the_bhp_steel_process_1;
                enables = process_wp_prdn_1;
                has_situations = { sq_actual, sv_actual };
                conjuncts = { cs_process_wp_hsm_240_1 };
}

class_frame cbs_process_wp_prdn
{       subclassOf: bhp_state_conjunction;
        relations:
                conjunct_of = cbs_process_wp;
                caused_by = process_wp_prdn;
                has_situations = { sq_intended, sv_intended };
                conjuncts = { ps_process_wp_hsm_240,
                                pro_wp_pickled_coil };
}

named_instance_frame cbs_process_wp_prdn_1
{       instanceOf: cbs_process_wp_prdn;
        relations:
                conjunct_of = cbs_process_wp;
                enables = process_wp_prdn;
                has_situations = { sq_actual, sv_actual };
                conjuncts = { ps_process_wp_hsm_240_1,
                        pro_wp_pickled_coil_1 };
}

class_frame es_process_wp_qc
{       subclassOf: bhp_state_conjunction;
        relations:
                conjunct_of = es_process_wp;
                enables = process_wp_qc;
                has_situations = { sq_intended, sv_intended };
                conjuncts = es_wp_hcpf_260;
}

named_instance_frame es_process_wp_qc_1
{       instanceOf: es_process_wp_qc;
        relations:
                conjunct_of = es_the_bhp_steel_process_1;
                enables = process_wp_qc_1;
                has_situations = { sq_actual, sv_actual };
                conjuncts = es_wp_hcpf_260_1;
}
```

```
class_frame cbs_process_wp_qc
{	subclassOf: bhp_state_conjunction;
	relations:
		conjunct_of = cbs_process_wp;
		caused_by = process_wp_qc;
		has_situations = { sq_intended, sv_intended };
		conjuncts = cbs_wp_hcpf_260;
}

named_instance_frame cbs_process_wp_qc_1
{	instanceOf: cbs_process_wp_qc;
	relations:
		conjunct_of = cbs_process_wp;
		enables = process_wp_qc;
		has_situations = { sq_actual, sv_actual };
		conjuncts = cbs_wp_hcpf_260_1;
}

/*****************************************************************

FILE:		enterprise_bhp_organization2.rock
bhp MODEL VERSION: 3.0
DATE:		May 6, 1996
FILE START DATE:May 6, 1996

*****************************************************************/

class_framebhp
{	subclassOf: bhp_company;
	relations:
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame bhp_1
{	instanceOf:bhp;
	relations:
		has_situations = { sq_actual, sv_actual };
}



/******************************************************************/


class_framebhp_steel
{	subclassOf:bhp_group;
	relations:
		has_member = { sppd, scpd, lpd };
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame bhp_steel_1
{	instanceOf: bhp_steel;
	relations:
		has_situations = { sq_actual, sv_actual };
}


class_framebhp_steel_executive_management
{	subclassOf: bhp_management;
	relations:
		has_situations = { sq_intended, sv_intended };
}


named_instance_frame bhp_steel_executive_management_1
{	instanceOf: bhp_steel_executive_management;
	relations:
		has_situations = { sq_actual, sv_actual };
}



class_frame bhp_steel_president_agent
```

```
{	subclassOf: bhp_worker;
	relations:
		member_of = bhp_steel_executive_management;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame ron_mcneilly
{	instanceOf: bhp_steel_president_agent;
	relations:
		member_of = bhp_steel_executive_management_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame bhp_steel_quality_manager_agent
{	subclassOf: bhp_worker;
	relations:
		member_of = bhp_steel_executive_management;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame colin_montrose
{	instanceOf: bhp_steel_quality_manager_agent;
	relations:
		member_of = bhp_steel_executive_management_1;
		has_situations = { sq_actual, sv_actual };
}


class_frame bhp_steel_company_role
{	subclassOf: bhp_role;
	relations:
		has_org_agent = bhp_steel;
		has_situations = { sq_intended, sv_intended };
		has_process = the_bhp_steel_process;
}

named_instance_frame bhp_steel_company_role_1
{	instanceOf: bhp_steel_company_role;
	relations:
		has_org_agent = bhp_steel_1;
		has_situations = { sq_actual, sv_actual };
		has_process = the_bhp_steel_process_1;
}


class_frame bhp_steel_executive_management_role
{	subclassOf: bhp_role;
	relations:
		has_org_agent = bhp_steel_executive_management;
		has_situations = { sq_intended, sv_intended };
		has_process = the_bhp_steel_process;
}

named_instance_frame bhp_steel_executive_management_role_1
{	instanceOf: bhp_steel_executive_management_role;
	relations:
		has_org_agent = bhp_steel_executive_management_1;
		has_situations = { sq_actual, sv_actual };
		has_process = the_bhp_steel_process_1;
}


class_frame bhp_steel_president_position
{	subclassOf: bhp_position;
	relations:
		has_situations = { sq_intended, sv_intended };
		has_agent = bhp_steel_president_agent;
}

named_instance_frame bhp_steel_president
{	instanceOf: bhp_steel_president_position;
	relations:
		has_situations = { sq_actual, sv_actual };
		has_agent = ron_mcneilly;
}

class_frame bhp_steel_quality_manager_position
{	subclassOf: bhp_position;
	relations:
		has_situations = { sq_intended, sv_intended };
		has_agent = bhp_steel_quality_manager_agent;
}

named_instance_frame bhp_steel_quality_manager
{	instanceOf: bhp_steel_quality_manager_position;
	relations:
		has_situations = { sq_actual, sv_actual };
		has_agent = colin_montrose;
}


class_frame bhp_steel_president_role
{	subclassOf: bhp_role;
	relations:
		has_process = the_bhp_steel_process;
		role_of = bhp_steel_president_position;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame bhp_steel_president_role_1
{	instanceOf: bhp_steel_president_role;
	relations:
		has_process = the_bhp_steel_process_1;
		role_of = ron_mcneilly;
		has_situations = { sq_actual, sv_actual };
}

class_frame bhp_steel_quality_manager_role
{	subclassOf: bhp_role;
```

```
	relations:
		has_process = process_wp_qc;
		role_of = bhp_steel_quality_manager_position;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame bhp_steel_quality_manager_role_1
{	instanceOf: bhp_steel_quality_manager_role;
	relations:
		has_process = process_wp_qc_1;
		role_of = bhp_steel_quality_manager;
		has_situations = { sq_actual, sv_actual };
}


class_frame bhp_steel_hsd_role
{	subclassOf: bhp_role;
	relations:
		has_process = process_wp_hsd;
		role_of = bhp_steel_president_position;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame bhp_steel_hsd_role_1
{	instanceOf: bhp_steel_hsd_role;
	relations:
		has_process = process_wp_hsd_1;
		role_of = ron_mcneilly;
		has_situations = { sq_actual, sv_actual };
}


class_frame bhp_steel_prdn_role
{	subclassOf: bhp_role;
	relations:
		has_process = process_wp_prdn;
		role_of = bhp_steel_president_position;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame bhp_steel_prdn_role_1
{	instanceOf: bhp_steel_prdn_role;
	relations:
		has_process = process_wp_prdn_1;
		role_of = ron_mcneilly;
		has_situations = { sq_actual, sv_actual };
}


class_framesppd
{	subclassOf:bhp_division;
	relations:
		member_of = bhp_steel;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame sppd_1
{	instanceOf: sppd;
	relations:
		member_of = bhp_steel_1;
		has_situations = { sq_actual, sv_actual };
}


class_framescpd
{	subclassOf:bhp_division;
	relations:
		member_of = bhp_steel;
		has_situations = { sq_intended, sv_intended };
}


named_instance_frame scpd_1
{	instanceOf: scpd;
	relations:
		member_of = bhp_steel_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame scpd_management
{	subclassOf: bhp_management;
	relations:
		member_of = bhp_steel_executive_management;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame scpd_management_1
{	instanceOf: scpd_management;
	relations:
		member_of = bhp_steel_executive_management_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame scpd_gm_agent
{	subclassOf: bhp_worker;
	relations:
		member_of = scpd_management;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame john_gross
{	instanceOf: scpd_gm_agent;
	relations:
		member_of = scpd_management_1;
		has_situations = { sq_intended, sv_intended };
}
```

```
class_frame scpd_division_role
{	subclassOf: bhp_role;
	relations:
		has_org_agent = scpd;
		has_situations = { sq_intended, sv_intended };
		has_process = process_wp;
}

named_instance_frame scpd_division_role_1
{	instanceOf: scpd_division_role;
	relations:
		has_org_agent = scpd_1;
		has_situations = { sq_actual, sv_actual };
		has_process = process_wp_1;
}

class_frame scpd_management_role
{	subclassOf: bhp_role;
	relations:
		has_org_agent = scpd_management;
		has_situations = { sq_intended, sv_intended };
		has_process = process_wp;
}

named_instance_frame scpd_management_role_1
{	instanceOf: scpd_management_role;
	relations:
		has_org_agent = scpd_management_1;
		has_situations = { sq_actual, sv_actual };
		has_process = process_wp_1;
}

class_frame scpd_gm_position
{	subclassOf: bhp_position;
	relations:
		has_agent = scpd_gm_agent;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame scpd_gm
{	instanceOf: scpd_gm_position;
	relations:
		has_agent = john_gross;
		has_situations = { sq_actual, sv_actual };
}

class_frame scpd_gm_role
{	subclassOf: bhp_role;
	relations:
		role_of = scpd_gm_position;
		has_process = process_wp;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame scpd_gm_role_1
{	instanceOf: scpd_gm_role;
	relations:
		role_of = scpd_gm;
		has_process = process_wp_1;
		has_situations = { sq_actual, sv_actual };
}

class_framelpd
{	subclassOf:bhp_division;
	relations:
		member_of = bhp_steel;
		empty has_member;
		empty has_position;
		empty has_process;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame lpd_1
{	instanceOf: lpd;
	relations:
		member_of = bhp_steel_1;
		has_situations = { sq_actual, sv_actual };
}

class_framewp_hsm_240
{	subclassOf:bhp_production_unit;
	relations:
		member_of = scpd;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_hsm_240_1
{	instanceOf: wp_hsm_240;
	relations:
		member_of = scpd_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame wp_hsm_240_role
{	subclassOf: bhp_role;
	relations:
		has_org_agent = wp_hsm_240;
		has_process = process_wp_hsm_240;
```

```
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_hsm_240_role_1
{	instanceOf: wp_hsm_240_role;
	relations:
		has_situations = { sq_actual, sv_actual };
		has_org_agent = wp_hsm_240_1;
		has_process = process_wp_hsm_240_1;
}

class_frame wp_hsm_240_management
{	subclassOf: bhp_management;
	relations:
		member_of = scpd_management;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_hsm_240_management_1
{	instanceOf: wp_hsm_240_management;
	relations:
		member_of = scpd_management_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame wp_hsm_240_management_role
{	subclassOf: bhp_role;
	relations:
		has_org_agent = wp_hsm_240_management;
		has_situations = { sq_intended, sv_intended };
		has_process = process_wp_hsm_240;
}

named_instance_frame wp_hsm_240_management_role_1
{	instanceOf: wp_hsm_240_management_role;
	relations:
		has_org_agent = wp_hsm_240_management_1;
		has_situations = { sq_actual, sv_actual };
		has_process = process_wp_hsm_240_1;
}

class_frame wp_hsm_240_manager_agent
{	subclassOf: bhp_worker;
	relations:
		member_of = wp_hsm_240_management;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame john_gray
{	instanceOf: wp_hsm_240_manager_agent;
	relations:
		member_of = wp_hsm_240_management_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame wp_hsm_240_manager_role
{	subclassOf: bhp_role;
	relations:
		has_situations = { sq_intended, sv_intended };
		has_process = process_wp_hsm_240;
}

named_instance_frame wp_hsm_240_manager_role_1
{	instanceOf: wp_hsm_240_manager_role;
	relations:
		has_situations = { sq_actual, sv_actual };
		has_process = process_wp_hsm_240_1;
}

class_frame wp_hsm_240_manager_position
{	subclassOf: bhp_position;
	relations:
		has_role = wp_hsm_240_manager_role;
		has_agent = wp_hsm_240_manager_agent;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_hsm_240_manager
{	instanceOf: wp_hsm_240_manager_position;
	relations:
		has_role = wp_hsm_240_manager_role_1;
		has_agent = john_gray;
		has_situations = { sq_actual, sv_actual };
}

class_framewp_sh_230
{	subclassOf:bhp_production_unit;
	relations:
		member_of = scpd;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_sh_230_1
{	instanceOf: wp_sh_230;
	relations:
		member_of = scpd_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame wp_sh_230_role
{	subclassOf: bhp_role;
	relations:
		has_org_agent = wp_sh_230;
		has_process = process_wp_sh_230;
		has_situations = { sq_intended, sv_intended };
}
```

```
named_instance_frame wp_sh_230_role_1
{	instanceOf: wp_sh_230_role;
	relations:
		has_situations = { sq_actual, sv_actual };
		has_org_agent = wp_sh_230_1;
		has_process = process_wp_sh_230_1;
}

class_frame wp_sh_230_management
{	subclassOf: bhp_management;
	relations:
		member_of = scpd_management;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_sh_230_management_1
{	instanceOf: wp_sh_230_management;
	relations:
		member_of = scpd_management_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame wp_sh_230_management_role
{	subclassOf: bhp_role;
	relations:
		has_org_agent = wp_sh_230_management;
		has_situations = { sq_intended, sv_intended };
		has_process = process_wp_sh_230;
}

named_instance_frame wp_sh_230_management_role_1
{	instanceOf: wp_sh_230_management_role;
	relations:
		has_org_agent = wp_sh_230_management_1;
		has_situations = { sq_actual, sv_actual };
		has_process = process_wp_sh_230_1;
}

class_frame wp_sh_230_manager_agent
{	subclassOf: bhp_worker;
	relations:
		member_of = wp_sh_230_management;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame peter_mccarthy
{	instanceOf: wp_sh_230_manager_agent;
	relations:
		member_of = wp_sh_230_management_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame wp_sh_230_manager_role
{	subclassOf: bhp_role;
	relations:
		has_situations = { sq_intended, sv_intended };
		has_process = process_wp_sh_230;
}

named_instance_frame wp_sh_230_manager_role_1
{	instanceOf: wp_sh_230_manager_role;
	relations:
		has_situations = { sq_actual, sv_actual };
		has_process = process_wp_sh_230_1;
}

class_frame wp_sh_230_manager_position
{	subclassOf: bhp_position;
	relations:
		has_role = wp_sh_230_manager_role;
		has_agent = wp_sh_230_manager_agent;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_sh_230_manager_position_1
{	instanceOf: wp_sh_230_manager_position;
	relations:
		has_role = wp_sh_230_manager_role_1;
		has_agent = peter_mccarthy;
		has_situations = { sq_actual, sv_actual };
}

class_framewp_rcs_250
{	subclassOf:bhp_production_unit;
	relations:
		member_of = scpd;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_rcs_250_1
{	instanceOf: wp_rcs_250;
	relations:
		member_of = scpd_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame wp_rcs_250_role
{	subclassOf: bhp_role;
	relations:
		has_org_agent = wp_rcs_250;
		has_process = process_wp_rcs_250;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_rcs_250_role_1
{	instanceOf: wp_rcs_250_role;
	relations:
```

```
		has_situations = { sq_actual, sv_actual };
		has_org_agent = wp_rcs_250_1;
		has_process = process_wp_rcs_250_1;
}

class_frame wp_rcs_250_management
{	subclassOf: bhp_management;
	relations:
		member_of = scpd_management;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_rcs_250_management_1
{	instanceOf: wp_rcs_250_management;
	relations:
		member_of = scpd_management_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame wp_rcs_250_management_role
{	subclassOf: bhp_role;
	relations:
		has_org_agent = wp_rcs_250_management;
		has_situations = { sq_intended, sv_intended };
		has_process = process_wp_rcs_250;
}

named_instance_frame wp_rcs_250_management_role_1
{	instanceOf: wp_rcs_250_management_role;
	relations:
		has_org_agent = wp_rcs_250_management_1;
		has_situations = { sq_actual, sv_actual };
		has_process = process_wp_rcs_250_1;
}

class_frame wp_rcs_250_manager_agent
{	subclassOf: bhp_worker;
	relations:
		member_of = wp_rcs_250_management;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame kevin_george
{	instanceOf: wp_rcs_250_manager_agent;
	relations:
		member_of = wp_rcs_250_management_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame wp_rcs_250_manager_role
{	subclassOf: bhp_role;
	relations:
		has_situations = { sq_intended, sv_intended };
		has_process = process_wp_rcs_250;
}

named_instance_frame wp_rcs_250_manager_role_1
{	instanceOf: wp_rcs_250_manager_role;
	relations:
		has_situations = { sq_actual, sv_actual };
		has_process = process_wp_rcs_250_1;
}

class_frame wp_rcs_250_manager_position
{	subclassOf: bhp_position;
	relations:
		has_role = wp_rcs_250_manager_role;
		has_agent = wp_rcs_250_manager_agent;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_rcs_250_manager_position_1
{	instanceOf: wp_rcs_250_manager_position;
	relations:
		has_role = wp_rcs_250_manager_role_1;
		has_agent = kevin_george;
		has_situations = { sq_actual, sv_actual };
}

class_framewp_hcpf_260
{	subclassOf:bhp_production_unit;
	relations:
		member_of = scpd;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_hcpf_260_1
{	instanceOf: wp_hcpf_260;
	relations:
		member_of = scpd_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame wp_hcpf_260_role
{	subclassOf: bhp_role;
	relations:
		has_org_agent = wp_hcpf_260;
		has_process = process_wp_hcpf_260;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_hcpf_260_role_1
{	instanceOf: wp_hcpf_260_role;
	relations:
		has_situations = { sq_actual, sv_actual };
		has_org_agent = wp_hcpf_260_1;
		has_process = process_wp_hcpf_260_1;
```

```
}

class_frame wp_hcpf_260_management
{	subclassOf: bhp_management;
	relations:
		member_of = scpd_management;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_hcpf_260_management_1
{	instanceOf: wp_hcpf_260_management;
	relations:
		member_of = scpd_management_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame wp_hcpf_260_management_role
{	subclassOf: bhp_role;
	relations:
		has_org_agent = wp_hcpf_260_management;
		has_situations = { sq_intended, sv_intended };
		has_process = process_wp_hcpf_260;
}

named_instance_frame wp_hcpf_260_management_role_1
{	instanceOf: wp_hcpf_260_management_role;
	relations:
		has_org_agent = wp_hcpf_260_management_1;
		has_situations = { sq_actual, sv_actual };
		has_process = process_wp_hcpf_260_1;
}

class_frame wp_hcpf_260_manager_agent
{	subclassOf: bhp_worker;
	relations:
		member_of = wp_hcpf_260_management;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame adrian_smythe
{	instanceOf: wp_hcpf_260_manager_agent;
	relations:
		member_of = wp_hcpf_260_management_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame wp_hcpf_260_manager_role
{	subclassOf: bhp_role;
	relations:
		has_situations = { sq_intended, sv_intended };
		has_process = process_wp_hcpf_260;
}

named_instance_frame wp_hcpf_260_manager_role_1
{	instanceOf: wp_hcpf_260_manager_role;
	relations:
		has_situations = { sq_actual, sv_actual };
		has_process = process_wp_hcpf_260_1;
}

class_frame wp_hcpf_260_manager_position
{	subclassOf: bhp_position;
	relations:
		has_role = wp_hcpf_260_manager_role;
		has_agent = wp_hcpf_260_manager_agent;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_hcpf_260_manager_position_1
{	instanceOf: wp_hcpf_260_manager_position;
	relations:
		has_role = wp_hcpf_260_manager_role_1;
		has_agent = adrian_smythe;
		has_situations = { sq_actual, sv_actual };
}

class_framewp_hcpd_265
{	subclassOf:bhp_production_unit;
	relations:
		member_of = scpd;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_hcpd_265_1
{	instanceOf: wp_hcpd_265;
	relations:
		member_of = scpd_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame wp_hcpd_265_role
{	subclassOf: bhp_role;
	relations:
		has_org_agent = wp_hcpd_265;
		has_process = process_wp_hcpd_265;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_hcpd_265_role_1
{	instanceOf: wp_hcpd_265_role;
	relations:
		has_situations = { sq_actual, sv_actual };
		has_org_agent = wp_hcpd_265_1;
		has_process = process_wp_hcpd_265_1;
}

class_frame wp_hcpd_265_management
```

```
{       subclassOf: bhp_management;
        relations:
                member_of = scpd_management;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_hcpd_265_management_1
{       instanceOf: wp_hcpd_265_management;
        relations:
                member_of = scpd_management_1;
                has_situations = { sq_actual, sv_actual };
}

class_frame wp_hcpd_265_management_role
{       subclassOf: bhp_role;
        relations:
                has_org_agent = wp_hcpd_265_management;
                has_situations = { sq_intended, sv_intended };
                has_process = process_wp_hcpd_265;
}

named_instance_frame wp_hcpd_265_management_role_1
{       instanceOf: wp_hcpd_265_management_role;
        relations:
                has_org_agent = wp_hcpd_265_management_1;
                has_situations = { sq_actual, sv_actual };
                has_process = process_wp_hcpd_265_1;
}

class_frame wp_hcpd_265_manager_agent
{       subclassOf: bhp_worker;
        relations:
                member_of = wp_hcpd_265_management;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame michael_lohaus
{       instanceOf: wp_hcpd_265_manager_agent;
        relations:
                member_of = wp_hcpd_265_management_1;
                has_situations = { sq_actual, sv_actual };
}

class_frame wp_hcpd_265_manager_role
{       subclassOf: bhp_role;
        relations:
                has_situations = { sq_intended, sv_intended };
                has_process = process_wp_hcpd_265;
}

named_instance_frame wp_hcpd_265_manager_role_1
{       instanceOf: wp_hcpd_265_manager_role;
        relations:
                has_situations = { sq_actual, sv_actual };
                has_process = process_wp_hcpd_265_1;
}

class_frame wp_hcpd_265_manager_position
{       subclassOf: bhp_position;
        relations:
                has_role = wp_hcpd_265_manager_role;
                has_agent = wp_hcpd_265_manager_agent;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_hcpd_265_manager_position_1
{       instanceOf: wp_hcpd_265_manager_position;
        relations:
                has_role = wp_hcpd_265_manager_role_1;
                has_agent = michael_lohaus;
                has_situations = { sq_actual, sv_actual };
}

class_framewp_pkl_270
{       subclassOf:bhp_production_unit;
        relations:
                member_of = scpd;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_pkl_270_1
{       instanceOf: wp_pkl_270;
        relations:
                member_of = scpd_1;
                has_situations = { sq_actual, sv_actual };
}

class_frame wp_pkl_270_role
{       subclassOf: bhp_role;
        relations:
                has_org_agent = wp_pkl_270;
                has_process = process_wp_pkl_270;
                has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_pkl_270_role_1
{       instanceOf: wp_pkl_270_role;
        relations:
                has_situations = { sq_actual, sv_actual };
                has_org_agent = wp_pkl_270_1;
                has_process = process_wp_pkl_270_1;
}

class_frame wp_pkl_270_management
{       subclassOf: bhp_management;
        relations:
                member_of = scpd_management;
```

```
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_pkl_270_management_1
{   instanceOf: wp_pkl_270_management;
    relations:
        member_of = scpd_management_1;
        has_situations = { sq_actual, sv_actual };
}

class_frame wp_pkl_270_management_role
{   subclassOf: bhp_role;
    relations:
        has_org_agent = wp_pkl_270_management;
        has_situations = { sq_intended, sv_intended };
        has_process = process_wp_pkl_270;
}

named_instance_frame wp_pkl_270_management_role_1
{   instanceOf: wp_pkl_270_management_role;
    relations:
        has_org_agent = wp_pkl_270_management_1;
        has_situations = { sq_actual, sv_actual };
        has_process = process_wp_pkl_270_1;
}

class_frame wp_pkl_270_manager_agent
{   subclassOf: bhp_worker;
    relations:
        member_of = wp_pkl_270_management;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame richard_mulgrave
{   instanceOf: wp_pkl_270_manager_agent;
    relations:
        member_of = wp_pkl_270_management_1;
        has_situations = { sq_actual, sv_actual };
}

class_frame wp_pkl_270_manager_role
{   subclassOf: bhp_role;
    relations:
        has_situations = { sq_intended, sv_intended };
        has_process = process_wp_pkl_270;
}

named_instance_frame wp_pkl_270_manager_role_1
{   instanceOf: wp_pkl_270_manager_role;
    relations:
        has_situations = { sq_actual, sv_actual };
        has_process = process_wp_pkl_270_1;
}

class_frame wp_pkl_270_manager_position
{   subclassOf: bhp_position;
    relations:
        has_role = wp_pkl_270_manager_role;
        has_agent = wp_pkl_270_manager_agent;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_pkl_270_manager_position_1
{   instanceOf: wp_pkl_270_manager_position;
    relations:
        has_role = wp_pkl_270_manager_role_1;
        has_agent = richard_mulgrave;
        has_situations = { sq_actual, sv_actual };
}

class_framewp_plp_280
{   subclassOf:bhp_production_unit;
    relations:
        member_of = scpd;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_plp_280_1
{   instanceOf: wp_plp_280;
    relations:
        member_of = scpd_1;
        has_situations = { sq_actual, sv_actual };
}

class_frame wp_plp_280_role
{   subclassOf: bhp_role;
    relations:
        has_org_agent = wp_plp_280;
        has_process = process_wp_plp_280;
        has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_plp_280_role_1
{   instanceOf: wp_plp_280_role;
    relations:
        has_situations = { sq_actual, sv_actual };
        has_org_agent = wp_plp_280_1;
        has_process = process_wp_plp_280_1;
}

class_frame wp_plp_280_management
{   subclassOf: bhp_management;
    relations:
        member_of = scpd_management;
        has_situations = { sq_intended, sv_intended };
}
```

```
named_instance_frame wp_plp_280_management_1
{	instanceOf: wp_plp_280_management;
	relations:
		member_of = scpd_management_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame wp_plp_280_management_role
{	subclassOf: bhp_role;
	relations:
		has_org_agent = wp_plp_280_management;
		has_situations = { sq_intended, sv_intended };
		has_process = process_wp_plp_280;
}

named_instance_frame wp_plp_280_management_role_1
{	instanceOf: wp_plp_280_management_role;
	relations:
		has_org_agent = wp_plp_280_management_1;
		has_situations = { sq_actual, sv_actual };
		has_process = process_wp_plp_280_1;
}

class_frame wp_plp_280_manager_agent
{	subclassOf: bhp_worker;
	relations:
		member_of = wp_plp_280_management;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame vincent_kotsopoulous
{	instanceOf: wp_plp_280_manager_agent;
	relations:
		member_of = wp_plp_280_management_1;
		has_situations = { sq_actual, sv_actual };
}

class_frame wp_plp_280_manager_role
{	subclassOf: bhp_role;
	relations:
		has_situations = { sq_intended, sv_intended };
		has_process = process_wp_plp_280;
}

named_instance_frame wp_plp_280_manager_role_1
{	instanceOf: wp_plp_280_manager_role;
	relations:
		has_situations = { sq_actual, sv_actual };
		has_process = process_wp_plp_280_1;
}

class_frame wp_plp_280_manager_position
{	subclassOf: bhp_position;
	relations:
		has_role = wp_plp_280_manager_role;
		has_agent = wp_plp_280_manager_agent;
		has_situations = { sq_intended, sv_intended };
}

named_instance_frame wp_plp_280_manager_position_1
{	instanceOf: wp_plp_280_manager_position;
	relations:
		has_role = wp_plp_280_manager_role_1;
		has_agent = vincent_kotsopoulous;
		has_situations = { sq_actual, sv_actual };
}
```

```
/*** BHP Steel Measurement Processes ***/

/*** measured attribute: average_coil_thickness ***/

holds(measured_attribute(average_coil_thickness),sv_actual).
holds(has_sample_sizing(average_coil_thickness,unit_population),s
v_actual).
holds(has_sampling_plan(average_coil_thickness,variable_sampling
),sv_actual).
holds(has_standard_value(average_coil_thickness,10),sv_actual).
holds(has_specification_set(average_coil_thickness,[9.85,10.15]),sv
_actual).
holds(has_unit_of_measurement(average_coil_thickness,mm),sv_ac
tual).


/*** activity: process_wp_hcpf_260_sensor_meas_1 ***/

holds(activity(process_wp_hcpf_260_sensor_meas_1),sv_actual).
holds(use(us_wp_hcpf_260_sensor_meas_1,process_wp_hcpf_260
_sensor_meas_1),sv_actual).
holds(uses(us_wp_hcpf_260_sensor_meas_1,sensor_meas_1),sv_a
ctual).
holds(measuring_resource(sensor_meas_1),sv_actual).
holds(has_attribute(process_wp_hcpf_260_sensor_meas_1,average
_coil_thickness),sv_actual).
holds(has_attribute(tru_wp_raw_coil_1,average_coil_thickness),sv_
actual).
holds(has_tru(wp_raw_coil_1,tru_wp_raw_coil_1),sv_actual).



/*** activity: process_wp_hcpf_260_1 ***/

holds(activity(process_wp_hcpf_260_1),sv_actual).
holds(has_subactivity(process_wp_hcpf_260_1,process_wp_hcpf_2
60_sensor_meas_1),sv_actual).
holds(has_process(wp_hcpf_260_manager_role_1,process_wp_hcp
f_260_1),sv_actual).



/*** organization: wp_hcpf_260_1 ***/

holds(has_agent(wp_hcpf_260_manager_role_1,wp_hcpf_260_1),sv
_actual).
holds(has_policy(wp_hcpf_260_manager_role_1,wp_hcpf_260_q_pr
ocedure_1),sv_actual).
holds(procedure(wp_hcpf_260_q_procedure_1),sv_actual).
holds(evidence(wp_hcpf_260_q_evidence_1),sv_actual).
holds(quality_evidence(wp_hcpf_260_q_evidence_1),sv_actual).hol
ds(has_communication_source(wp_hcpf_260_q_evidence_link_1,wp
_hcpf_260_manager_role_1),sv_actual).
holds(communication_link_of(wp_hcpf_260_q_evidence_link_1,wp_
hcpf_260_q_evidence_1),sv_actual).
holds(evidence(wp_hcpf_260_q_evidence_1),sv_actual).
```

holds(has_member(bhp_steel_1,wp_qc_1),sv_actual).
holds(has_member(wp_qc_1,wp_hcpf_260_1),sv_actual).
holds(has_member(sppd_1,colin_montrose),sv_actual).

holds(organization_agent(colin_montrose),sv_actual).
holds(organization_agent(ron_mcneilly),sv_actual).

holds(has_process(wp_hcpf_260_manager_role_1,process_wp_hcpf_260_1),sv_actual).
holds(has_process(wp_hcpf_260_management_role_1,process_wp_hcpf_260_1),sv_actual).
holds(has_process(bhp_steel_q_manager_role_1,process_wp_qc_1),sv_actual).
holds(has_process(bhp_steel_q_management_role_1,process_wp_qc_1),sv_actual).
holds(has_process(bhp_steel_president_role_1,process_bhp_steel_1),sv_actual).

holds(has_agent(wp_hcpf_260_manager_role_1,colin_montrose),sv_actual).
holds(has_agent(wp_hcpf_260_manager_role_1,wp_hcpf_260_1),sv_actual).
holds(has_agent(bhp_steel_q_manager_1,colin_montrose),sv_actual).
holds(has_agent(bhp_steel_q_management_role_1,wp_qc_1),sv_actual).
holds(has_agent(bhp_steel_president_role_1,bhp_steel_1),sv_actual).
holds(has_agent(bhp_steel_president_1,ron_mcneilly),sv_actual).

holds(has_role(bhp_steel_q_manager_1,bhp_steel_q_manager_role_1),sv_actual).
holds(has_role(bhp_steel_president_1,bhp_steel_president_role_1),sv_actual).

holds(authority_link_of(bhp_steel_q_procedure_link_1,wp_hcpf_260_q_procedure_1),sv_actual).
holds(has_authority_source(bhp_steel_q_procedure_link_1,bhp_steel_q_manager_role_1),sv_actual).
holds(has_authority_sink(bhp_steel_q_objective_link_1,bhp_steel_q_manager_role_1),sv_actual).
holds(has_authority_source(bhp_steel_q_objective_link_1,bhp_steel_president_role_1),sv_actual).
holds(authority_link_of(bhp_steel_q_objective_link_1,bhp_steel_q_objective_1),sv_actual).
holds(has_communication_source(wp_hcpf_260_q_evidence_link_1,wp_hcpf_260_manager_role_1),sv_actual).
holds(communication_link_of(wp_hcpf_260_q_evidence_link_1,wp_hcpf_260_q_evidence_1),sv_actual).

holds(has_policy(wp_hcpf_260_manager_role_1,wp_hcpf_260_q_procedure_1),sv_actual).
holds(has_policy(bhp_steel_q_manager_role_1,wp_qc_q_procedure_1),sv_actual).

holds(has_goal(bhp_steel_q_manager_role_1,bhp_steel_q_objective_1),sv_actual).

holds(evidence(wp_hcpf_260_q_evidence_1),sv_actual).
holds(quality_evidence(wp_hcpf_260_q_evidence_1),sv_actual).
holds(procedure(wp_hcpf_260_q_procedure_1),sv_actual).
holds(procedure(wp_qc_q_procedure_1),sv_actual).
holds(quality_objective(bhp_steel_q_objective_1),sv_actual).
holds(has_requirement(wp_qc_q_procedure_1,wp_hcpf_260_q_procedure_1),sv_actual).
holds(has_requirement(bhp_steel_q_policy_1,wp_qc_q_procedure_1),sv_actual).

holds(documents(wp_hcpf_260_q_record_1,wp_hcpf_260_q_evidence_1),sv_actual).
holds(documents(wp_hcpf_260_q_plan_1,wp_hcpf_260_q_procedure_1),sv_actual).
holds(quality_policy_of(bhp_steel_q_policy_1,bhp_steel_1),sv_actual).