An Ontology of Quality for Enterprise Modelling

Henry M. Kim, Mark S. Fox, Michael Gruninger

Department of Industrial Engineering,University of Toronto,

4 Taddle Creek Road, Toronto, Ontario M5S 1A4

tel:1-416-978-6823 fax:1-416-978-3453

internet:{ henryk, msf, gruninger }@ie.utoronto.ca

Abstract

Although there are many quality domains in which ideas and concepts about quality are represented, these representations are often informal. The TOVE Quality Ontology is the formal representation (using first-order logic) of terms, relationships, and axioms about quality which are generic beyond any specific quality domain. The assumption that quality is “conformance to requirements” is used to decompose the domain of quality into sub-domains of measurement, quality analysis, identification, and traceability. An ontological engineering methodology of posing ontological scope, stating competency questions, constructing data models and axioms, and visualization of the answering of competency questions is demonstrated with an example from the engineering of the traceability ontology.

Keywords:

quality, traceability, enterprise modelling

1. Introduction

In this world of global competition, the importance of quality is well-acknowledged, and quality has now become a corporate cliché in North America. But as with most clichés, the term quality is more anecdotally, and less formally defined. Quality gurus like Juran, Deming, and others have espoused the importance of quality from an experiential and philosophical perspective, and helped many a companies remain competitive. Quality tools, such as statistical quality control (SQC) or

quality function deployment (QFD), are defined with more rigour— mathematics or rules for building the “House of Quality” [Hauser 88]— but relate to only a specific domain of quality management. The ISO 9000 and the Baldridge Awards do address the domain of quality from both a formal and generic view, as these documents specify guidelines or requirements for what constitutes a certain quality level for a generic company. The TOVE Quality Ontology endeavors to an even more formal representation of terms, relationships, and axioms about quality. Yet these representations will also be generic beyond any specific quality domain. Some of the goals of the overall TOVE project of the Enterprise Integration Laboratory at the University of Toronto are to: 1) provide a shared terminology, and 2) define precise and unambiguous semantics for the enterprise [Fox et. al. 93]. Thus the TOVE Quality Ontology is generic to satisfy the first goal, and formal to satisfy the second goal.

With effective use of information technology also providing a competitive advantage for organizations [Walton 89], the need for an organization to better manage information about quality will become more emphasized with increasing advances in information technology [Godfrey 93]. Since the TOVE Quality Ontology presents constructs for all strata of knowledge representation [Brachman 79]— i.e. implementation, logical, conceptual, generic, and application— the ontology, in conjunction with the other TOVE ontologies, provides a comprehensive and integrated representation with which sophisticated, deductive decision-making using information technology can be made.

The bedrock for the TOVE Quality Ontology is the TOVE core ontologies: the ontologies for activity, state, causality, time, and resources [Gruninger & Fox 94],[Fadel et. al. 94]. As the next section will demonstrate, constructs are eventually added into the TOVE Quality Ontology if only it can be defined in terms of the data model and axioms of the core ontologies.

In engineering a generic quality ontology, an overall system competency question can be: “What is the quality of a product, process, or system of the enterprise?”. However this overall question is too broad to motivate the requirements of the TOVE Quality Ontology. Thus this must be decomposed into component competency questions that are narrow enough in scope to motivate the development of ontology representations that will answer these questions. This decomposition also constitutes the abstract exploration of the quality domain; that is, organizing the generic domain before formalizing it.

2. Decomposing the Task of Engineering the TOVE Quality Ontology

2.1 What is Quality?

The official definition of quality, stated by the standardization bodies of Europe and North America, 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, pp. 16]

This vague definition can be augmented with a manufacturing-based definition [Garvin 84], as stated by Crosby: *"Quality means conformance to requirements."* [Crosby 79, pp. 15]. Combining the ISO 9000 and Crosby's definitions of quality, the basis of the TOVE Quality Ontology is this:

- 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.

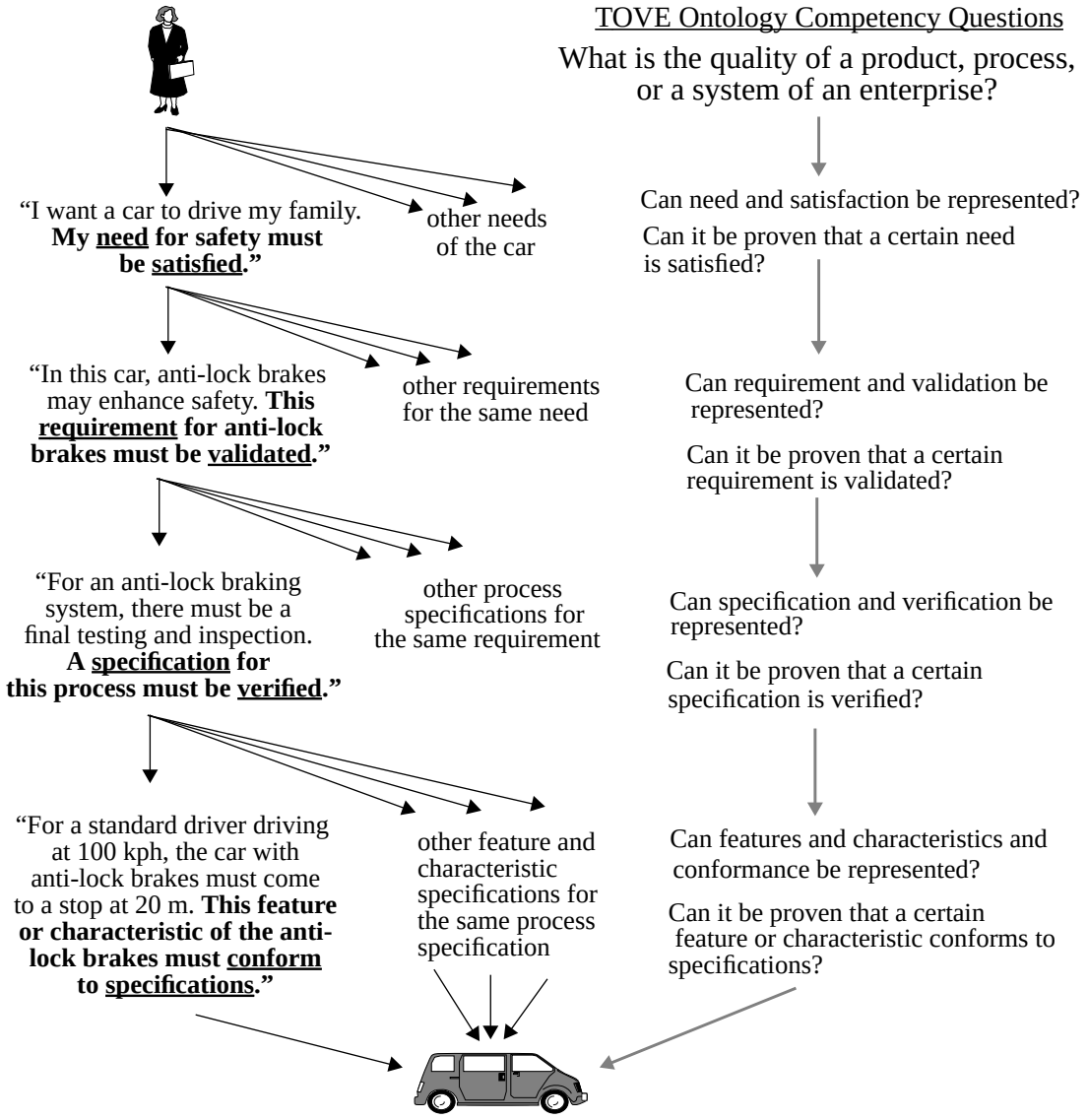
2.2 Decomposing a Quality Need

[Hall 89] stresses the importance of mapping needs into goals, objectives, constraints, input-output requirements, and specifications for the system that will satisfy these needs. These inputs translated into requirements should also be layered into system component requirements and overall system requirements [Blanchard 91]. And [Grady 93] emphasizes a structured methodology for system requirements analysis. The diagram below shows how such a structured methodology is used: it is shown that as a quality need is decomposed, there are concomitant competency questions at each level of decomposition, that drive the engineering of the TOVE Quality Ontology.

The diagram shows that it is possible to decompose a vague need— such as the need for a safe car— into a myriad of very concrete, measurable specifications on features or characteristics— such as the specification for stopping distance. Intermediate nodes in this decomposition are requirements and specifications. That requirements are validated and specifications are verified, are explained in [Boehm 81, pp.37]:

- Verification: "Are we building the product right?"
- Validation: "Are we building the right product?"

FIGURE 1. Engineering the TOVE Quality Ontology: Decomposing a Need



2.3 Towards a Base Representation: Grounding Out to the TOVE Core Ontologies

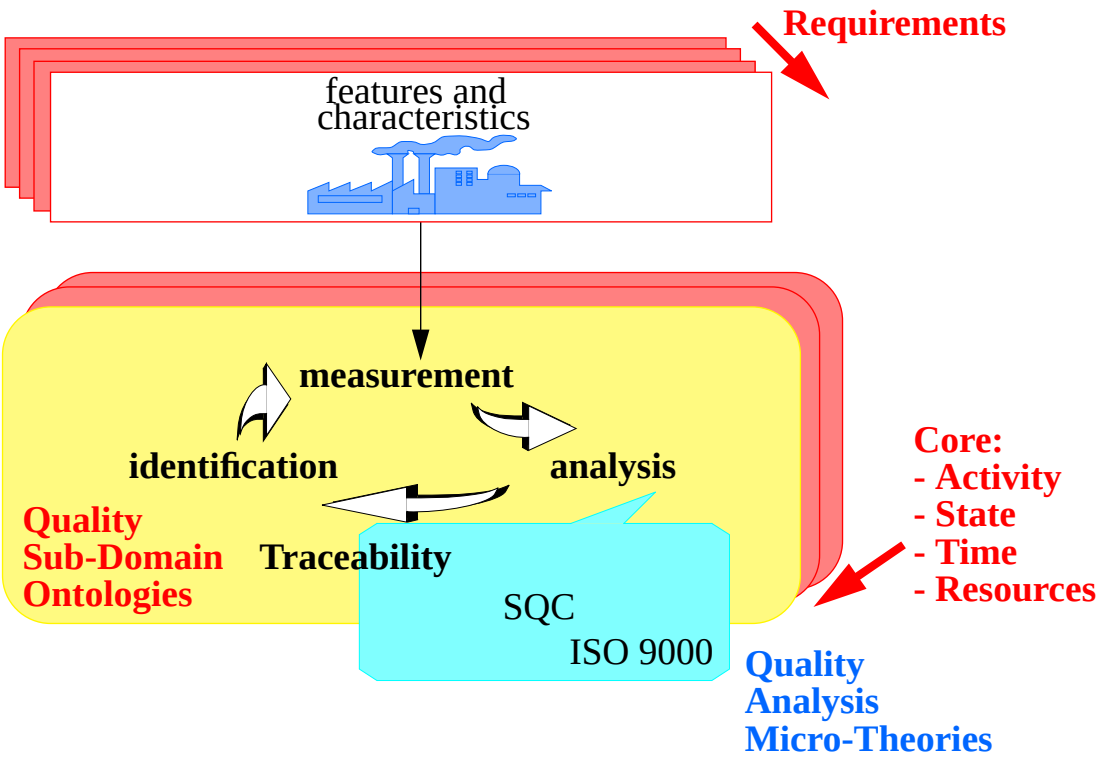
The core ontologies can represent such entities as standard driver, driving, car, anti-lock brakes, and the state of being stopped. However the conformance of the example specification cannot be determined unless the stopping distance is measured. Hence in order to talk about quality, activities, states, times, and resources must be measured, and the measurement values must be represented. It is noted that: *"Things are investigated in physics in so far as it is possible to measure them, and not with the impossible goal of discovering their intimate essence."* [Di Franca

81, pp. 18]. As in physics, measurement is the root of quality management, as evidenced by Federal Express' quality philosophy of "measure, measure, measure".

An ontology of measurement links the features and characteristics specification, decomposed from a quality need, to the representations of activities, states, times, and resources, that exist in the enterprise model. Constructs in the measurement ontology can be defined in terms of constructs in the core ontologies. So the domain of measurement constitutes a sub-domain of quality; that is, an ontology of quality includes an ontology of measurement.

Measurements are taken only because variability exists. Often this variability needs to be analyzed because the variability causes a problem. That analysis proceeds it, is a post-condition to measurement. A pre-condition to measurement is identification; that is, an entity must first be identified as that which is to be measured. The TOVE Quality Ontology, then is comprised of sub-domain ontologies, an ontology of requirements, plus the TOVE Core Ontologies in the following way:

FIGURE 2. TOVE Quality Ontology Decomposition



The TOVE Quality Ontology builds up from the other ontologies being constructed in the Enterprise Integration Laboratory. So the quality sub-domain ontologies of measurement,

identification, and analysis are constructed from existing representations from the core ontologies. Moreover the de-composition of quality needs into measurable specifications on features and characteristics is supported by representations from the TOVE Requirements Ontology.

The philosophy of minimal ontological commitment [Gruber 93] underlies the development of the TOVE Quality Ontology. Although it is believed that a rigorous development of the identification and measurement ontologies is warranted, there is a minimal representation for generic analysis in the TOVE Quality Ontology. Why? Because there are a myriad of quality analysis domains, such as SQC, QFD, quality costing, ISO 9000, and Baldridge Awards, which can be used to analyze, control, assure, and improve product, process, or organizational quality. But in committing one or more of these analysis techniques to the TOVE Quality Ontology, it is believed that a bias is introduced, and the desired genericity of the TOVE Quality Ontology is lost. Hence just with this system of integrated ontologies, it is possible to represent only primitive assessment of quality— e.g., what is the quality of the car at this point in time— without the representational capability to even analyze a nonconformity.

Such analysis capability and more is possible through the construction of TOVE Quality Analysis Micro-Theories, as represented in the previous diagram. The two micro-theories that directly use the TOVE Quality Ontology representations are the TOVE SQC (Statistical Quality Control) and ISO 9000 micro-theories. With an implementation of the ISO 9000 Micro-Theory, it is possible to query an enterprise model— constructed upon the TOVE Core and Quality Ontologies— as to whether the enterprise is compliant to one of the ISO 9000 standards.

A primitive analysis capability requires that it be possible to trace back, for example, from a problematic assembly to its sub-assemblies 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. But is traceability then a sub-domain of generic quality to be included in the TOVE Quality Ontology, or is it a basic analysis technique to be represented as one of the TOVE Quality Micro-Theories?

An ontology of traceability is included as part of the TOVE Quality Ontology, because it is a domain of quality that is generic and usable by any quality analysis technique, and hence does not violate the philosophy of minimal ontological commitment. As such traceability is that subset of

analysis that links the measurement domain with the identification domain, and thus it is worthwhile to explore traceability, as an important sub-domain of quality.

2.4 Structure of Competency Questions

For each of the sub-domains which constitute the TOVE Quality Ontology, the ability of the information system— constructed upon the ontology of that sub-domain— to answer the competency questions put forth at the onset of the ontological engineering process thus validates that ontology. The competency of the ontology is motivated in the following manner:

- **Scope**: Certain assumptions about the domain will need to be made, and in so doing the scope of the ontology will become more apparent. As the assumptions about the ontology are made, objects, relationships, and attributes that belong in this ontology are "ferreted out". Of the set of criteria to evaluate a shareable representation of enterprise knowledge as presented by [Fox et. al. 93], this scoping then dictates the extensibility, granularity, and scalability of the ontology.
- **Problem Statement**: This is the one general problem statement which justifies the construction of the ontology. This is a question, motivated from a quality perspective and within the bounds of the scope of the ontology, posed generally to serve as the template for all competency questions. All competency questions are motivated from this question. This problem statement dictates the generality of the ontology.
- **User Level Competency Questions**: Competency questions for the TOVE Quality Ontology must be motivated by questions related to quality scenarios. These questions are of the form of the problem statement, but more specific. These are called user level competency questions, since these are types of questions likely to be asked by the user of the ontology. The competence of the ontology is dictated by these questions.
- **Developer Level Competency Questions**: These competency questions are, in essence, the concrete requirements of the ontology. These questions dictate the engineering of representations that are needed to answered the user level questions, and are likely to be asked by the developer of the ontology. Whereas user level questions require a breadth of representation to answer, these developer level questions are those that directly motivate the depth of representation. What this means is that developer level questions motivate the full development of a small subset of representations, and the answering of these questions are needed to answer the user level questions. As such efficiency, perspicuity, and transformability of the ontology are dictated by these questions.

These stages of discerning scope, stating general problem statement, and posing a hierarchy of competency questions roughly reflect the systems analysis and requirements analysis phases of a classical software engineering project.

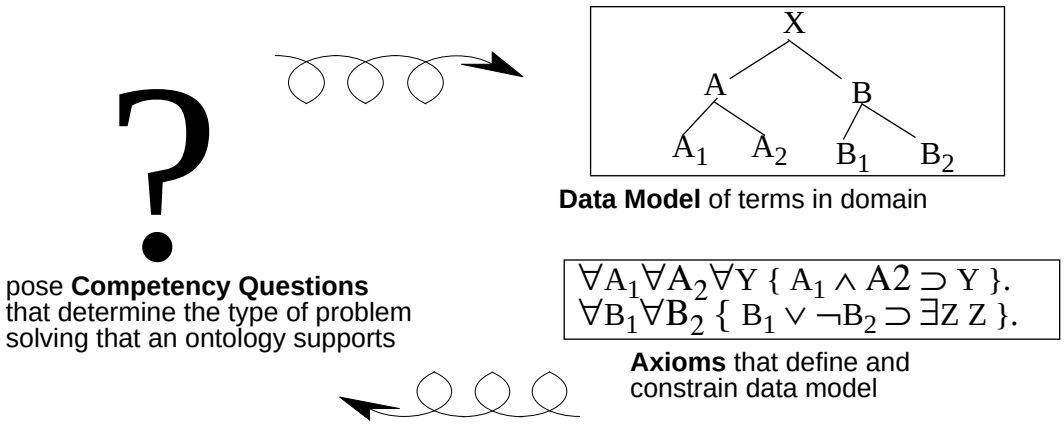
3. A Sub-Domain of the TOVE Quality Ontology: the Traceability Ontology

In this section, a small sample from the TOVE Traceability Ontology is presented to show:

- some of the representations required to explicate the domain of traceability
- concrete examples of application of the ontological engineering methodology
- that it is possible to solve a practical problem, related to a quality scenario, by using this ontology.

The steps in the engineering of the traceability ontology are included in the methodology shown below.

FIGURE 3. Ontological Engineering Methodology



3.1 Competency Questions

Why is traceability important in the context of reasoning about quality? This question must be definitively answered before we start an endeavour to engineer an ontology of traceability, that is to a part of an overall ontology of quality. In order to answer this, a scenario can be put forth to show the importance of traceability:

- Of the resources that were consumed and/or produced along the diagnostic path, the amount of the resources that were needed for specific activities over a period of time is

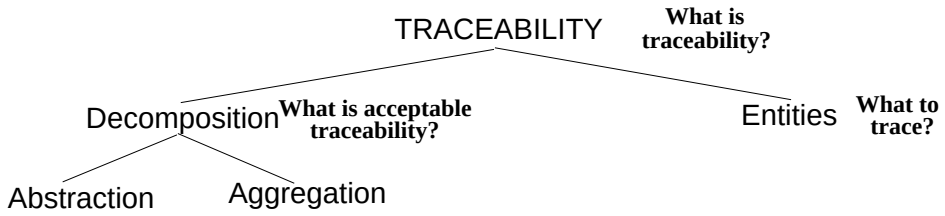
desired. It is known that an electric surge occurred during that period of time, and that this may be the cause of the quality problem. If a given batch of resource existed in that time period (that is, its quantity was greater than zero, and it had not been exhausted), then examination of this batch and activities which consumed or produced it may help diagnose this problem.

It must then be possible to represent this scenario using some representations from the traceability ontology, and with a combination of different ontologies, it may be possible to answer the question embedded in this quality scenario. In order to start this capability, competency questions about traceability must be posed. But before this, the scope of the traceability ontology must be scoped.

3.1.1 Scope

[Grady 93] states that traceability is a clear knowledge of ancestry, and so a discussion of traceability entails a graphical notation of ancestry: the tree. As such the task of traceability becomes one of recursively decomposing tree nodes, until terminal nodes are encountered. In such a decomposition, nodes are composed of sub-nodes that have a conjunctive or disjunctive relationships to the node. So a problem decomposition task represented by an AND/OR graph [Shinghal 92] can be used to represent the traceability problem. In the TOVE core ontologies, a non-terminal state is abstracted from a conjunction and/or disjunction of terminal states [Fox et. al. 93]. So scope for abstraction is that of questioning acceptable traceability given abstractions of conjunctive and disjunctive states. As well acceptable traceability for aggregations must be addressed. Finally which of the TOVE entities are to be traced must be scoped.

FIGURE 4. Scope Decomposition for Traceability Ontology



The TOVE Core Ontologies support different abstracted views. With such different levels of abstraction, the issue to consider for the TOVE Traceability Ontology is at which level of abstraction a certain level of traceability is possible. Some assumptions to handle this are:

Assumption 1: It must be possible to trace from one entity to another, where neither the entities are abstracted entities.

Assumption 2: If the previous assumption holds, then it will be possible to trace from one entity to another, regardless of the level of abstraction of either the entities.

Next, it must be discerned as to what level of traceability is acceptable. Is it acceptable to state that "A led to B, and then somehow B led C", or to state that "Either A or B led to C". The first statement is incomplete since the path from B to C is uncertain, and the second statement is non-unique, since the exact path is one of two options. Thus it must be scoped as to when complete and unique traceability is necessary, and when incomplete and non-unique traceability is sufficient. Some assumption about the appropriate level of traceability are:

Assumption 3: Given a completely conjunctive traceability tree, it is assumed that complete and unique traceability— e.g. "It is known exactly that A led to B, which led to C"— is possible.

Assumption 4: It is assumed that traceability is required to recall what has already occurred in time; that is, traceability is backwards in time.

Assumption 5: It is assumed that traceability is recalled from the information system implementation records.

Assumption 6: Because of the previous two assumptions, even given a completely disjunctive traceability tree, it is assumed that complete and unique traceability is possible.

The entities to trace depend on which entities are uniquely identified. Of the entities that the ISO 9000 recommends for tracing, only products (one form of traceable resource units) and activities are to be uniquely identified in the TOVE Identification Ontology. Therefore:

Assumption 7: Traceable resource unit (aka a tru—a representation defined in the TOVE Identification Ontology for a batch of a something, e.g. a tru of 100 widgets) is the resource representation that must be traceable, since a tru is neither an abstracted nor aggregated entity.

Assumption 8: Primitive activity (a representation defined in the TOVE Identification Ontology) is the activity representation that must be traceable, since a primitive activity is neither an abstracted nor aggregated entity.

Finally, in tracing between entities, it is desirable to, for example, trace the quantity variance of traceable resource units over time, or trace the processing history of activities. Thus:

Assumption 9: It is possible to trace as per attributes of traceable resource units and primitive activities.

3.1.2 Problem Statement

Now that the scope of what is entailed in traceability and what entities are to be traced have been established, engineering of the representations to enable traceability can ensue. What these representations will be depends upon the quality-related scenario that necessitates the traceability capability. The general problem statement for this quality-related scenario is:

> Q: Given an entity and a state of the enterprise, can all entities and necessary attributes of these entities that had a bearing on the quality of the given entity be *traced* and identified?

The user competency questions posed below are all forms of this general problem statement.

3.1.3 User Competency Questions

This particular question is directly motivated from the quality-related scenario:

> Q 1. How much of a specific batch of resources was used by one or more activities over a given period of time?

This question motivates the asking of more lower-level questions that a developer is likely to ask.

3.1.4 Developer Competency Questions

In order to answer the user competency question, what characterizes the traceable resource unit (a batch) must first be defined. For example, when can a traceable resource unit not be traced?

> Q 2. Under what condition(s) is traceability of a tru not possible?
>
> A: Traceability is not possible before the time point at which the entity is identified to be a tru. But after this time point, traceability is possible, even if at the time of the trace all of the quantity of the tru have been exhausted.
>
> Justification: Axiom T1., pg. 13; Axiom T6., pg. 14; Axiom T7., pg. 14

Note that an important issue to consider is what happens to the traceability of a tru if a portion of it is used or consumed for one activity, while the remaining portion is not. That is:

> Q 3. What happens when a tru is split or disaggregated?
>
> A: If a tru is disaggregated, the disaggregated portions still retain the ID as the original tru.
>
> Justification: Axiom T9., pg. 14

Conversely it must be considered what happens when two or more tru instances of the same tru class are brought together. That is:

> Q 4. What happens when trus are aggregated?

A: If trus are aggregated, the ID of this aggregated tru is different from the IDs of any of the original trus.

Justification: Axiom T8., pg. 14; Axiom T10., pg. 14

Since these previous axioms maintain the traceability of a tru, even if it is aggregated or disaggregated, then it is now possible to axiomatize about the quantity of aggregation or disaggregation of a tru, and quantity changes over time.

Q 5. Quantities of a tru will vary over time. How will this change be represented?

A: Quantity change will be expressed as discrete changes. Representing continuous quantity changes is beyond the scope of this ontology.

Justification: Axiom T2., pg. 14; Axiom T3., pg. 14; Axiom T4., pg. 14; Axiom T5., pg. 14

Q 6. Can the quantity of the tru be represented?

A: *Yes*. The construct for representing quantity of a tru is called *rp_tru* (resource point of a tru).

Justification: Axiom T2., pg. 14; Axiom T3., pg. 14; Axiom T4., pg. 14

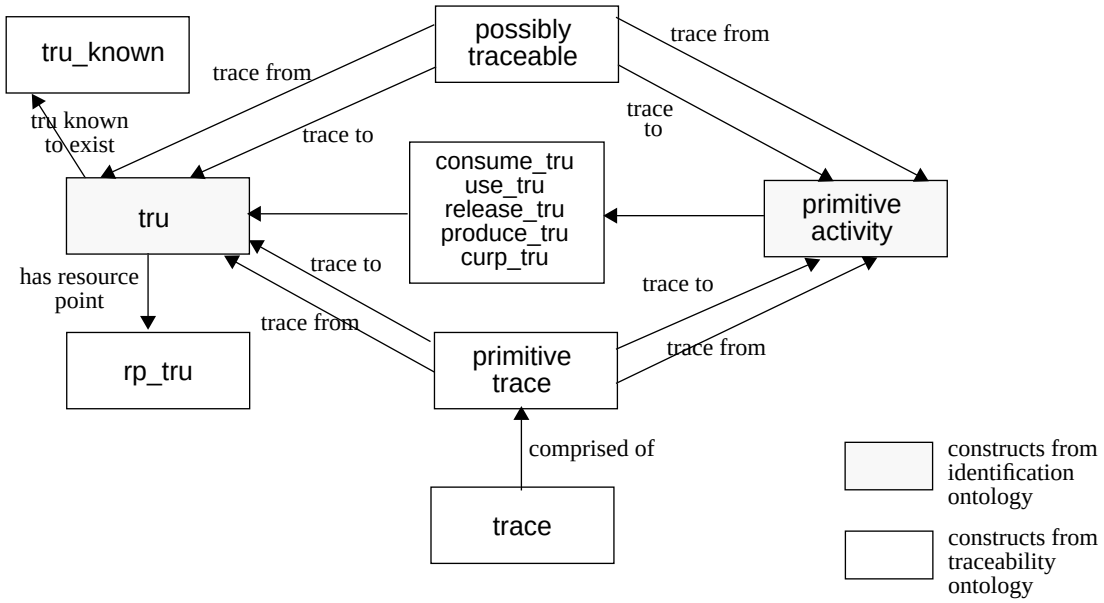
Q 7. Can the quantity of the tru at a certain point in time be represented?

A: *Yes*. The construct for representing quantity of a tru is called *rp_tru* (resource point of a tru).

Justification: Axiom T2., pg. 14; Axiom T3., pg. 14; Axiom T4., pg. 14

3.2 Data Model

FIGURE 5. Traceability Ontology Data Model



The above diagram shows the terms of the traceability ontology. Examples of what can be read from this data model are:

- A primitive activity may consume a tru.
- There may exist a possible trace between a tru and a primitive activity, a tru and another tru, or a primitive activity and another primitive activity.
- There may exist a primitive trace between a tru and a primitive activity, a tru and another tru, or a primitive activity and another primitive activity.
- Trace between one tru-primitive activity pair to another tru-primitive activity pair is comprised of one or more primitive traces.

The details of the relationships between these terms and the constraints upon these terms are stated in first-order logic in the next section.

3.3 Axioms

These are some, but not all, of the axioms of the TOVE Traceability Ontology. These are only the ones necessary to answer the developer level competency question. The first-order logic expressions for these axioms are given section 7 Appendix.

T1. A tru is first known to exist at the time that it is first used, consumed, produced, or

released.

T2. If a tru is produced by a primitive activity, then the resource point of the tru at time T_p (the time at which the produce state completes) is quantity that was produced, Q, measured in U units.

T3. If a tru is released by a primitive activity, then the resource point of the tru at time T_p (the time at which the release state completes) is the quantity released plus the quantity that was already available, measured in U units.

T4. If a tru is used or consumed by a primitive activity, then the resource point of the tru at time T_p (the time at which the use or consume state is enabled) is the amount of the tru that has not been committed, and hence is available for other states.

T5. If there exists a resource point of a tru at time T_{p1}, and there exists a resource point for the same tru at time T_{p2}, and for any point T_p where $T_p \in (T_{p1},T_{p2})$ there does not yet exist a resource point for that tru at T_p, then the resource point of the tru for any point T_p is assigned to be the resource point at time T_{p1}, because the quantity of the tru has not changed since T_{p1}.

T6. Before the time point, T_p, at which the tru is known to exist, there is no quantity for the tru.

T7. However after T_p, there is always a quantity value for the tru (this value = 0, if the tru has been completely consumed).

T8. The quantity of a given tru is never incremented after it is recognized to exist.

T9. Should a tru be produced or released such that the total amount available is Q, and it is used or consumed in quantity Q_0 by a subsequent primitive activity, where $Q_0 \leq Q$, then the units of Q_0 all have the same ID as that previously in Q. The implication is that individual units of a tru are indistinguishable from each other, and hence traceability within a tru is not possible.

T10. If one tru is produced or release such that the total quantity available is Q, and another tru

is used or consumed in quantity Q_0 by a subsequent primitive activity, where Q_0>Q, then the two trus cannot be the same. The implication is that 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.

3.4 Implementation

The enterprise model used in the TOVE project is implemented in C++ using the ROCK^{TM} knowledge representation tool from Carnegie Group. Queries about the enterprise model are made in Prolog. So the representations developed in the traceability ontology are either implemented as objects, relations, and attributes in the enterprise model, or implemented as Prolog predicates that reason about the enterprise model. Finally the enterprise model is visualized using a graphical user interface called oak^{TM}, also from the Carnegie Group.

The user level question is posed again below. A re-statement of this questions with the terminology from the TOVE Ontologies is shown below this. A Prolog implementation of how this question is to be answered is shown. Finally the graphical presentation of the answer to this query is displayed.

Q 1. How much of a specific batch of resources was used by one or more activities over a given period of time?

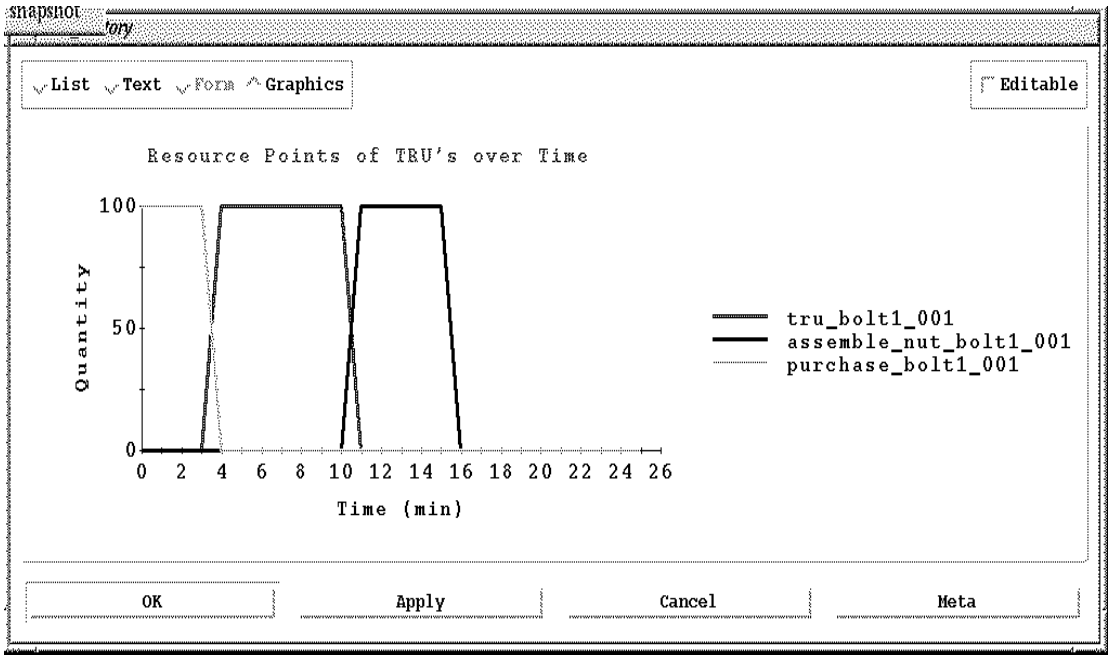
Q 1. *How much of a given tru was used/consumed/produced/released by one or more primitive activities over a given period of time?*

A: It is possible to plot the quantity change of a tru over time. Some of the granularity assumptions in the traceability ontology are that: there are integral units of time, and quantity changes occur discretely, not continuously.

Implementation:

```
show_tru_history(R,[Te,Te]) :- rp_tru(R,Q,Te,U).
show_tru_history(R,[Ts,Te]) :-
   rp_tru(R,Q,Ts,U),
   TNew is Ts + 1,
   show_tru_history(R,[TNew,Te]).
```

FIGURE 6. Graphical Answer to Traceability User Level Competency Question



The above diagram shows that the traceable resource unit named tru_bolt1_001 was produced by the primitive activity named purchase_bolt1_001 at time 5 min, where the quantity of the tru was 100. The diagram also shows that all quantity of tru_bolt_001 was consumed for the primitive activity named assemble_nut_bolt1_001 at time 11 min.

4. Conclusion

A logical formalization of quality knowledge is presented in this paper. By formalizing this body of knowledge, these benefits have been accrued:

1. Elucidation of the concept of quality by classification of identification, traceability, and measurement as sub-domains of the TOVE Quality Ontology
2. Clearly identified terminology and axioms: the TOVE Quality Ontology
3. Presentation of a rigorous methodology for ontological engineering
4. Graphical presentation of the ability to use this formalization to make deductions and decisions about quality

What has been presented in this paper is only a portion of the work in formalization of quality knowledge that is being performed in the Enterprise Integration Laboratory. As mentioned, there

are two types of formalizations: engineering of the TOVE Quality Ontology and the TOVE Quality Analysis Micro-Theories. The status of these efforts is as follows:

TOVE Quality Ontology

- **Identification Ontology**. Its problem statements are:

  Q: Given an entity, can that entity be *uniquely identified*, such that the quality of that entity can be discerned from the quality of another entity?

  Q: Given an entity, can that entity be *classified* into an entity type, such that the quality of that entity can be compared to the quality of another entity which is classified in the same entity type, although both entities are *uniquely identified* as being different?

  A full iteration of the ontological engineering methodology has been applied to the development of this ontology. The biggest contribution of this ontology to the overall TOVE Quality Ontology is the explicit scoping of basic assumptions about unique identification.

- **Traceability Ontology**

  Q: Given an entity and a state of the enterprise, can all entities and necessary attributes of these entities that had a bearing on the quality of the given entity be *traced* and identified?

  As display in this paper, a full iteration of the ontological engineering methodology has been applied to this ontology. With this ontology, a basic capability to perform traceability in analyzing quality problems is given.

- **Measurement Ontology**

  Q. Given an entity, can attributes of that entity which have a bearing on the quality of that entity be *measured*, so that such a measurement can be used to assess the quality of that entity?

  This is the crux of the TOVE Quality Ontology. With representations from this ontology— using the assumption that quality is conformance to requirements— it is possible to deduce the quality of an entity.

TOVE Quality Analysis Micro-Theories

- **ISO 9000 Micro-Theory**

  Q. Given an enterprise and a state of the world, can it be determined that the enterprise complies to one of the ISO 9000 standards for quality management?

  This is a formalization of the ISO 9000. Specifically this is a formalization of the ISO 9002 requirements upon an enterprise's quality management system, and the ISO 8402— a glossary of terms. This micro-theory can be used as a software tool

to help a company meet ISO 9000 compliance.

- **SQC Micro-Theory**
  - Q. Given an entity, and a state of the world, can the quality of that entity in past states be stated, and can the quality of that entity in future states be predicted?
  - Q. Can the above capability be used to control the quality of that entity in future states?

  This is a formalization of some of the basic techniques in statistical quality control. This micro-theory can be used as a software tool to control a company's processes.

The TOVE Quality Ontology is the representational basis upon which formalized quality knowledge can be used to integrate the quality-related decision making through out an enterprise.

5. Acknowledgments

This research is supported, in part, by the Natural Science and Engineering Research Council, Digital Equipment Corp., Micro Electronics and Computer Research Corp., and Spar Aerospace.

6. Appendix

T1. *A tru is first known to exist at the time that it is first used, consumed, produced, or released.*

$$\forall Rt\ \exists T_p\ (\ \textbf{\textit{tru\_known(Rt,}}\textbf{\textit{T}}_{\textbf{\textit{p}}}\textbf{\textit{)}} \equiv tru(Rt) \wedge$$
$$\forall S\ \forall T\ \exists S_0\ \exists T_0\ [\ (\ uses(S,Rt) \vee consumes(S,Rt) \vee$$
$$produces(S,Rt) \vee releases(S,Rt)\ ) \wedge$$
$$(uses(S_0,Rt) \vee consumes(S_0,Rt) \vee produces(S_0,Rt) \vee releases(S_0,Rt)\ )$$
$$\wedge state\_duration(S,T) \wedge state\_duration(S_0,T_0) \supset$$
$$\{\ (\ strictly\_before(T_0,T) \vee possibly\_before(T_0,T) \vee T=T_0\ )$$
$$\wedge start\_point(T_0,T_p)\ \}\ ]\ ).$$

Rt: ID of the tru
$\mathbf{T_p}$: time point at which the tru is recognized to exist
S: all states that use/consume/produce/release Rt
S_0: the first state that uses/consumes/produces/releases Rt
T,T_0: time durations for states S, S_0, respectively

T2. *If a tru is produced by a primitive activity, then the resource point of the tru at time T_p (the time at which the produce state completes) is quantity that was produced, Q, measured in U units.*

$\forall R_{tru} \forall T_p \forall A \forall S \forall U \exists Q \{$ primitive_activity(A) $\wedge$
produce(S,A) $\wedge$ produces(S,R_{tru}) $\wedge$
$\exists T_{pp}$ (tru_known(R_{tru},T_{pp}) $\wedge$ $T_p \geq T_{pp}$) $\wedge$
state_duration(S,T) $\wedge$ end_point(T,T_p) $\wedge$
amount_produced(R_{tru},Q) $\wedge$ unit_of_measurement(R,capacity,U,A)
$\supset$ ***rp_tru(R_{tru},Q,T_p,U)*** $\}$.

R_{tru}: unique ID of the traceable resource unit
Q: quantity of R_{tru} that was produced
T_p: time point at which R_{tru} is produced
U: unit of measurement in capacity measurement units; if the traceable resource unit is produced discretely, then this can be “objects”, and if produced continuously, this can be “litres” or “tons”
A: primitive activity that produces R_{tru}
S: state associated with the produced tru R_{tru}

T3. *If a tru is released by a primitive activity, then the resource point of the tru at time T_p (the time at which the release state completes) is the quantity released plus the quantity that was already available, measured in U units.*

$\forall R_{tru} \forall T_p \forall A \forall S \forall U \exists Q \{$ primitive_activity(A) $\wedge$
release(S,A) $\wedge$ releases(S,R) $\wedge$
$\exists T_{pp}$ (tru_known(R_{tru},T_{pp}) $\wedge$ $T_p \geq T_{pp}$) $\wedge$
state_duration(S,T) $\wedge$ end_point(T,T_p) $\wedge$
$\exists Q_1$ $\exists Q_2$ (amount_available(S,R_{tru},Q_1) $\wedge$ amount_committed(S,R_{tru},Q_2) $\wedge$
$Q=Q_1+Q_2$) $\wedge$
unit_of_measurement(R,capacity,U,A)
$\supset$ ***rp_tru(R_{tru},Q,T_p,U)*** $\}$.

R_{tru}: unique ID of the traceable resource unit
Q: quantity of R_{tru} available for other states just after R_{tru} was released
T_p: time point at which R_{tru} is released
U: unit of measurement in capacity measurement units; if the traceable resource unit is produced discretely, then this can be “objects”, and if produced continuously, this can be “litres” or “tons”
A: primitive activity that releases R_{tru}
S: state associated with the released tru R_{tru}

T4. *If a tru is used or consumed by a primitive activity, then the resource point of the tru at time T_p (the time at which the use or consume state is enabled) is the amount of the tru that has not been committed, and hence is available for other states.*

$\forall R_{tru} \forall T_p \forall A \forall S \forall U \exists Q \{$ primitive_activity(A) $\wedge$
[(use(S,A) $\wedge$ uses(S,R)) $\vee$ (consume(S,A) $\wedge$ consumes(S,R))] $\wedge$
$\exists T_{pp}$ (tru_known(R_{tru},T_{pp}) $\wedge$ $T_p \geq T_{pp}$) $\wedge$
state_duration(S,T) $\wedge$ start_point(T,T_p) $\wedge$
amount_available(S,R_{tru},Q) $\wedge$ unit_of_measurement(R,capacity,U,A) }
$\supset$ ***rp_tru(R_{tru},Q,T_p,U)*** $\}$.

R_{tru}: unique ID of the traceable resource unit
Q: quantity of R_{tru} that is available for other states once R_{tru} is committed to be used or consumed
T_p: time point at which R_{tru} is committed to be used or consumed
U: unit of measurement in capacity measurement units; if the traceable resource unit is produced discretely, then this can be "objects", and if produced continuously, this can be "litres" or "tons"
A: primitive activity that uses or consumes R_{tru}
S: state associated with the used or consumed tru R_{tru}

T5. *If there exists a resource point of a tru at time T_{p1}, and there exists a resource point for the same tru at time T_{p2}, and for any point T_p where $T_p \in (T_{p1},T_{p2})$ there does not yet exist a resource point for that tru at T_p, then the resource point of the tru for any point T_p is assigned to be the resource point at time T_{p1}, because the quantity of the tru has not changed since T_{p1}.*

$$\forall R_{tru}\forall U\forall T_p\exists Q\ \exists Q_1\exists T_{p1}\exists Q_2\exists T_{p2}$$
$$\{\{\ rp\_tru(R_{tru},Q_1,T_{p1},U) \wedge rp\_tru(R_{tru},Q_2,T_{p2},U) \wedge$$
$$T_p>T_{p1} \wedge T_p<T_{p2} \wedge$$
$$\exists T_{pp}\ (\ tru\_known(R_{tru},T_{pp}) \wedge T_p \geq T_{pp}\ ) \wedge$$
$$\forall T_{p0}\forall Q_0\ [\ T_{p0}>T_{p1} \wedge T_{p0}<T_{p2} \wedge rp\_tru(R_{tru},Q_0,T_{p0},U) \supset Q_0=Q_1]$$
$$\supset Q=Q_1\ \}$$
$$\supset \boldsymbol{rp\_tru(R_{tru},Q,T_p,U)}\ \}.$$

R_{tru}: unique ID of the traceable resource unit
Q: quantity of R_{tru} that is available for other states at time T_p
T_p: time at which the resource point of the tru is desired
U: unit of measurement in capacity measurement units; if the traceable resource unit is produced discretely, then this can be "objects", and if produced continuously, this can be "litres" or "tons"
Q_1: this is the resource point associated with the last state (before time point T_p) in which R_{tru} was used/consumed/produced/released.
Q_2: this is the resource point associated with exactly the next state(after time point T_p) in which R_{tru} was used/consumed/produced/released.
T_{p1}: time at which the resource point of the tru was assessed to be Q_1
T_{p2}: time at which the resource point of the tru was assessed to be Q_2

T6. *Before the time point, T_p, at which the tru is known to exist, there is no quantity for the tru.*

$$\forall Rt\ \forall T_{pa}\ \forall U\ \exists T_p\ [\ tru\_known(Rt,T_p) \wedge T_{pa}<T_p \supset$$
$$\neg\exists Q_a\ rp\_tru(Rt,Q_a,T_{pa},U)\ ].$$

Rt: ID of the tru
T_p: time point at which the tru is recognized to exist
T_{pa}: any time point before T_p
U: unit of measurement for Rt
Q_a: the quantity of Rt at T_{pa}. Since Rt does not exist at T_{pa}, there is no value for Q_a

T7. *However after T_p, there is always a quantity value for the tru (this value = 0, if the*

tru has been completely consumed).

$$\forall Rt\ \forall T_{pz}\ \forall Q_z\ \forall U\ \exists T_p\ [\ tru\_known(Rt,T_p) \wedge T_{pz} \geq T_p \supset rp\_tru(Rt,Q_z,T_{pz},U) \wedge Q_z \geq 0\ ].$$

Rt: ID of the tru
T_p: time point at which the tru is recognized to exist
T_{pz}: any time point equal to or after after T_p
U: unit of measurement for Rt
Q_z: the quantity of Rt at T_{pz}

T8. *The quantity of a given tru is never incremented after it is recognized to exist.*

$$\forall Rt\ \forall Q_a\ \forall Q\ \forall U\ \forall T_{pa}\ \forall T_p\ [\ tru\_known(Rt,T_p) \wedge T_{pa} \geq T_p \wedge rp\_tru(Rt,Q,T_p,U) \wedge rp\_tru(Rt,Q_a,T_{pa},U) \supset Q \geq Q_a\ ].$$

Rt: ID of the tru
T_p: time point at which Rt is recognized to exist
T_{pa}: any time point after T_p
Q: total quantity of Rt at the time that it is recognized to exist
Q_a: total quantity of Rt at time point T_{pa}
U: unit of measurement for Rt

T9. *Should a tru be produced or released such that the total amount available is Q, and it is used or consumed in quantity Q_0 by a subsequent primitive activity, where $Q_0 \leq Q$, then the units of Q_0 all have the same ID as that previously in Q. The implication is that individual units of a tru are indistinguishable from each other, and hence traceability within a tru is not possible.*

$$\forall Rt \exists Rt_0 \forall S \forall \mathbf{Rt_{class}}\ \exists S_0\ \{\ tru(Rt) \wedge tru(\ Rt_0) \wedge has\_tru(\mathbf{Rt_{class}},Rt) \wedge has\_tru(\mathbf{Rt_{class,}}\ Rt_0) \wedge \{\ (\ produces(S,Rt) \vee releases(S,Rt)\ ) \wedge (\ uses(S_0,Rt_0) \vee consumes(S_0,Rt_0)\ )\ \} \wedge \forall Q \forall T \forall T_p \forall U \exists Q_0 \exists T_0 \exists T_{p0}\ [\ \{\ state\_duration(S,T) \wedge state\_duration(S_0,T_0) \wedge (\ strictly\_before(T,T_0) \vee possibly\_before(T,T_0)\ )\ \} \wedge \{\ has\_point(T,T_p) \wedge has\_point(T_0,T_{p0}) \wedge rp\_tru(Rt,Q,T_p,U) \wedge rp\_tru(Rt_0,Q_0,T_{p0},U) \supset Q_0 \leq Q\ \} \supset Rt = Rt_0\ ]\ \}.$$

Rt, Rt_0 :ID of a tru that is produced or released, and then subsequently used or consumed
Rt_{class}: tru type, for which Rt is an instance
S: a produce or release state for Rt
S_0: a use or consume state for Rt
T,T_0: state durations for A and A_0, respectively
T_p,T_{p0}: any time point in T and T_0, respectively
Q,Q_0: total quantity of Rt at states S and S_0, respectively
U: unit of measurement for Rt

T10. *If one tru is produced or release such that the total quantity available is Q, and another tru is used or consumed in quantity Q_0 by a subsequent primitive activity, where Q_0>Q, then the two trus cannot be the same. The implication is that 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.*

$$\forall Rt \exists Rt_0 \forall S \forall Rt_{class} \exists S_0 \{ tru(Rt) \wedge tru(Rt_0) \wedge$$
$$has\_tru(\mathbf{Rt_{class}}, Rt) \wedge has\_tru(\mathbf{Rt_{class}}, Rt_0) \wedge$$
$$\{ (produces(S,Rt) \vee releases(S,Rt)) \wedge$$
$$(uses(S_0,Rt_0) \vee consumes(S_0,Rt_0)) \} \wedge$$
$$\forall Q \forall T \forall T_p \forall U \exists Q_0 \exists T_0 \exists T_{p0} [\{ state\_duration(S,T) \wedge$$
$$state\_duration(S_0,T_0) \wedge$$
$$(strictly\_before(T,T_0) \vee possibly\_before(T,T_0)) \} \wedge$$
$$\{ has\_point(T,T_p) \wedge has\_point(T_0,T_{p0}) \wedge$$
$$rp\_tru(Rt,Q,T_p,U) \wedge rp\_tru(Rt_0,Q_0,T_{p0},U) \supset Q_0 > Q \} \supset$$
$$Rt \neq Rt_0] \}.$$

Rt : ID of a tru that is produced or released
Rt_0 : ID of a tru that is used or consumed after Rt is produced or released
Rt_{class}: tru type, for which Rt and Rt_0 are instances
S: a produce or release state for Rt
S_0: a use or consume state for Rt
T,T_0: state durations for S and S_0, respectively
T_p,T_{p0}: any time point in T and T_0, respectively
Q,Q_0: total quantities for Rt and Rt_0 at states S and S_0, respectively
U: unit of measurement for Rt and Rt_0

7. References

1. [Blanchard 91] Blanchard, Benjamin S., *Systems Engineering Management*, John Wiley & Sons, Inc., 1991.

2. [Boehm 81] Boehm, Barry, *Software Engineering Economics*, Prentice Hall, 1981.

3. [Brachman 79] Brachman R. J., *On the Epistemological Status of Semantic Networks*, in Associative Networks: Representations and Use of Knowledge by Computers, in Findler, N.V: Academic Press, 1979, pp. 3-50.

4. [Crosby 79] Crosby, P.B., *Quality is Free: The Art of Making Quality Certain*, New York: McGraw-Hill, 1988.

5. [Di Franca 81] Di Franca, G.T., The Investigation of the Physical World, Cambridge, England: Cambridge University Press, 1981.

6. [Fadel et. al. 94] Fadel, F.G., Fox, M.S., Gruninger, M., *A Generic Enterprise Resource Ontology*, Proceedings of Third Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, Morgantown, WV, April 1994, pp. 117-128.

7. [Fox et. al. 93] Fox, M.S., Chionglo, J.C., Fadel, F.G., *A Common-Sense Model of the Enterprise*, in 2nd IE Research Conference Proceedings, Los Angeles, CA, May 1993.

8. [Garvin 84] Garvin, D.A., *What does 'Product Quality' Really Mean?*, Sloan Management Review, Fall, 1984.

9. [Godfrey 93] Godfrey, A. Blanton, *Ten Clear Trends for the Next Ten Years*, Quality Quotes, Vol. 19, Spring 1993, No. 2.

10. [Grady 93] Grady, Jeffrey O., *System Requirements Analysis*, McGraw-Hill Inc, 1993.

11. [Gruber 93] Gruber, Thomas, R., *Towards Principles for the Design of Ontologies Used for Knowledge Sharing*, Technical Report KSL 93-4, Knowledge Systems Laboratory, Computer Science Department, Stanford University, Stanford, CA 94305.

12. [Hall 89] Hall III, Arthur D., *Metasystems Methodology: A New Synthesis and Unification*, Pergamon Press, 1989.

13. [Hauser & Clausing 88] Hauser, J.R. & Clausing, D., *The House of Quality*, Harvard Business Review, May-June, 1988, pp. 63-73.

14. [ISO 91] ISO, *ISO 9000 International Standards for Quality Management*, Geneva,Switzerland: ISO General Secretariat, 1991.

15. [Shinghal 92] Shinghal, Rajjan, *Formal Concepts in Artifical Intelligence Fundamentals*, London: Chapman & Hall Computing, 1992

16. [Walton 89] Walton, Richard E., *Up and Running: Integrating IT and the Organization*, Boston, MA: Harvard Business School Press, 1989.