System Engineering
Analysis Support for Decision-making
Draft #8.1

11/11/05 4:18:10 AM
Harold P. Frisch
NASA/PDES Inc.

Draft #8.1 changes from Draft #8 are (minor). Address AP233 issue 64.

- **Fix misleading symbology used for the property "relate"**
  - **Figs 1,2 provide double lines for "relate" property**
  - **Fix figure 2 typo in one of analysis parameter relationship boxes (one should have been design parameter relationship.**
  - **Note: The definition of relate – this should imply "relating – related" as used in STEP hierarchical relationships**

Draft #8 Incorporates inputs from both PDES Inc. offsite and Hangzhou China meetings. Effort is directed towards bringing Protégé model data content in line with the concept model block diagrams.

Draft #8 supercedes work distributed in Hangzhou; i.e., Draft #6 in the file "AP233 Decision_analysis(1).pdf(.ppt)).

Associated Protégé ontology work has not yet been restarted. The Protégé work is still at the Draft #2 level and it needs significant re-work to bring it up to Draft #8. It's too much work for now with very limited AP233 WD#2 return on investment apparent at this point in time.

Introduction

Decision-making is an integral part of any systems engineering activity. Decisions are assumed to be rational (what, why, how). They are made by people and organizations; somewhere, and at some point in time (who, when, where). The proposed concept model seeks to provide the basic the network of concepts and relationships necessary for the representation and exchange of all computer sensible information associated with the systems engineering decision-making process. It builds upon STEP foundational modules and is intended

for inclusion within AP233, see AP233 working draft #1 at:
https://sourceforge.net/project/showfiles.php?group_id=27487&package_id=154135&release_id=332007

The concept model is presented in block diagram form. The intent is to make this computer sensible via Ontology technology as enable by the open source software capability Protégé, see http://protege.stanford.edu/**.**

Within the associated block diagrams shown herein:

Boxes = Ontology class (noun)
Solid Line = Ontology property (verb)
Dashed line = Ontology property (verb) (detail on another figure)
Colors used have no special meaning. They are simply a visualization aid.

To make this diagram computer sensible triples are provided herein such as:

Systems engineering requires [is required by] Decision-making

In the block diagram figure 1 this triple is presented as the 2 boxes labeled Systems Engineering, and Decision-making and one line labeled require.

This maps to the forward and inverse ontology assertions:

1. **Systems Engineering** requires **Decision making**
2. **Decision making** is required by **Systems Engineering**

Other assertion specification detail enabled by Protégé has not yet been developed.

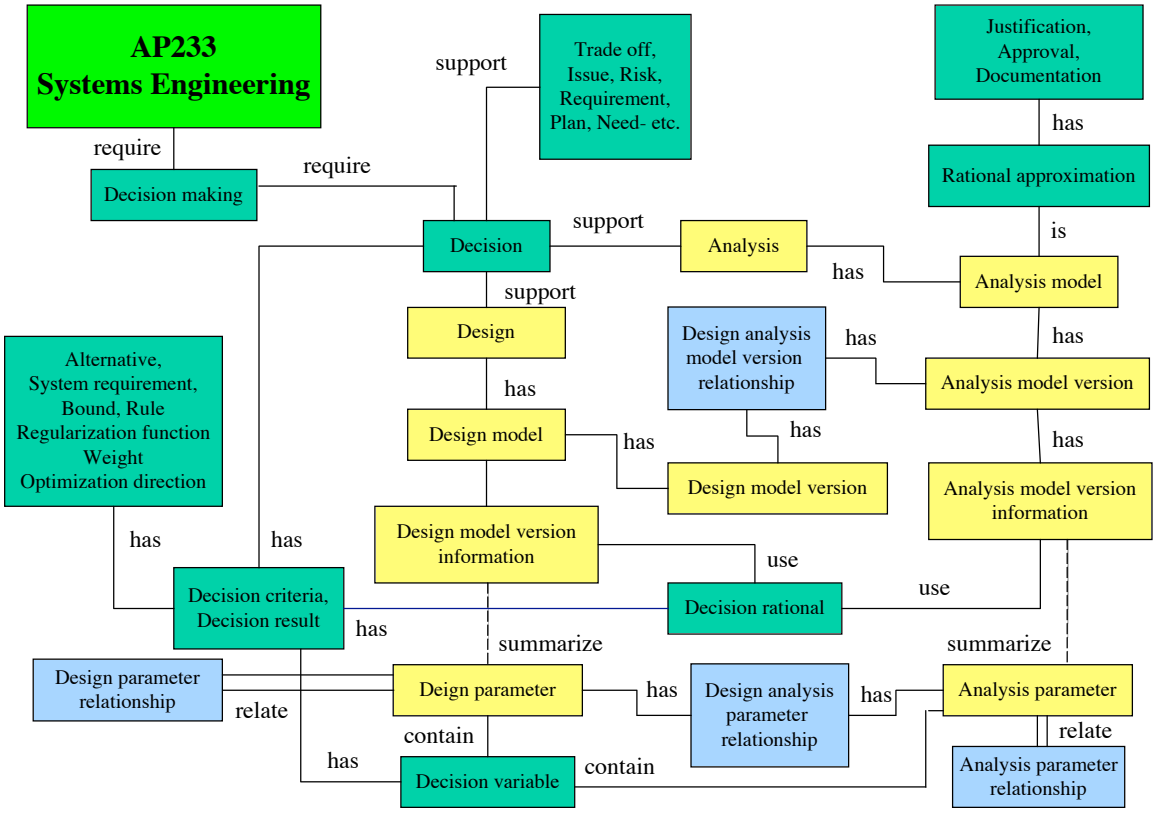
Definitions for all terms used within boxes and associated with lines are provided. Rather than attempt single crisp definitions that are guaranteed to generate endless arguments many definitions consistent with concept model intent are provided. All, but a very few, have been copied directly from sources obtained via http://www.onelook.com/**.**

Following this work is some cut and paste 2002 Lambda Calculus work done in support of AP233. This appears to provide the theoretical linkage between the information triplets of Ontology theory and concept model block diagrams to Ontology (assertion, class) doublets. These should align well with the theoretical foundation of STEP. This is a line of investigation to-be-done,

if required, at some future time.

Block Diagrams and Ontology Assertions

Figure 1 presents the concept model overview. This says that systems engineering requires decision-making and this requires rational decisions to be made. The decisions are assumed to support the design of the system and analysis is used to support these decisions. Provision is provided to record all that enters into how the decision was made and how both design and analysis models related (who, when, where, what, why, how). The systems engineer is assumed to be uninterested in deep domain detail. Communication between deep detail domain work and systems engineering is at the abstraction level of decision variables. STEP associated PDM detail is assumed implicit within the concept model and hence not shown.



1 - Overview - Analysis in support of decision making

Foundational concepts (fig 1)

Systems Engineering requires [is required by] **Decision-making**
Decision-making requires [is required by] **Decision**

Decision usage (fig 1)

Decision supports [is supported by] **Trade off**
Decision supports [is supported by] **Stakeholder need**
Decision supports [is supported by] **Issue**
Decision supports [is supported by] **Risk**
Decision supports [is supported by] **Verification plan**
Decision supports [is supported by] **Verification requirement**
Decision supports [is supported by] **Validation plan**
Decision supports [is supported by] **Validation result**
Decision supports [is supported by] **Validation requirement**
Decision supports [is supported by] **Analysis**
Decision supports [is supported by] **Design**

Decision information elements (figs 1,7)

Decision has [is associated with] **Decision criteria**
Decision has [is associated with] **Decision result**
Decision result has [is associated with] **Decision rational**
Decision criteria has [is associated with] **Decision alternative**
Decision criteria has [is associated with] **Decision variable**
Decision criteria has [is associated with] **System requirement**
Decision criteria has [is associated with] **Regularization function**
Decision criteria has [is associated with] **Optimization direction**
Decision criteria has [is associated with] **Weight**
Decision criteria has [is associated with] **Decision rule**
Decision criteria has [is associated with] **Decision bound**

Define elements of a decision rational (fig 1)

Decision rational uses [is used by] **Design model version information**
Decision rational uses [is used by] **Analysis model version information**
Decision rational has [is associated with] **Decision criteria**
Decision rational has [is associated with] **Decision result**

Define analysis model (fig 1)

Analysis has [is associated with] **Analysis model**
Analysis model has [is associated with] **Analysis model version**
Analysis model has [is associated with]
Analysis model version information

Define design model (fig 1)

Design has [is associated with] **Design model**
Design model has [is associated with] **Design model version**
Design model has [is associated with]
Design model version information

Analysis has relationship with Design (fig 1,2)

Design analysis model version relationship has [is associated with]

Analysis model version

Design analysis model version relationship has [is associated with]

Design model version

Define design and analysis parameters and relationship (fig 1,2)

Analysis parameter summarizes [is summarized by]

Analysis model version input

Analysis parameter summarizes [is summarized by]

Analysis model version output

Design parameter summarizes [is summarized by]

Design model version input

Design parameter summarizes [is summarized by]

Design model version output

Design analysis parameter relationship has [is associated with]

Analysis parameter

Design analysis parameter relationship has [is associated with]

Design parameter

Enable iterative design and analysis parameter evaluations (fig 1,2)

Design parameter relationship has relating [is relating to]

Design parameter

Design parameter relationship has related [[is related to]

Design parameter

Analysis parameter relationship has relating [is relating to]

Analysis parameter

Analysis parameter relationship has related [[is related to]

Analysis parameter

Define decision variables (fig 1)

Decision variable is generalization of [is specialization of]

Design parameter

Decision variable is generalization of [is specialization of]

Analysis parameter

Basis for Analysis model version (fig 1)

Analysis model version is described by [describes]

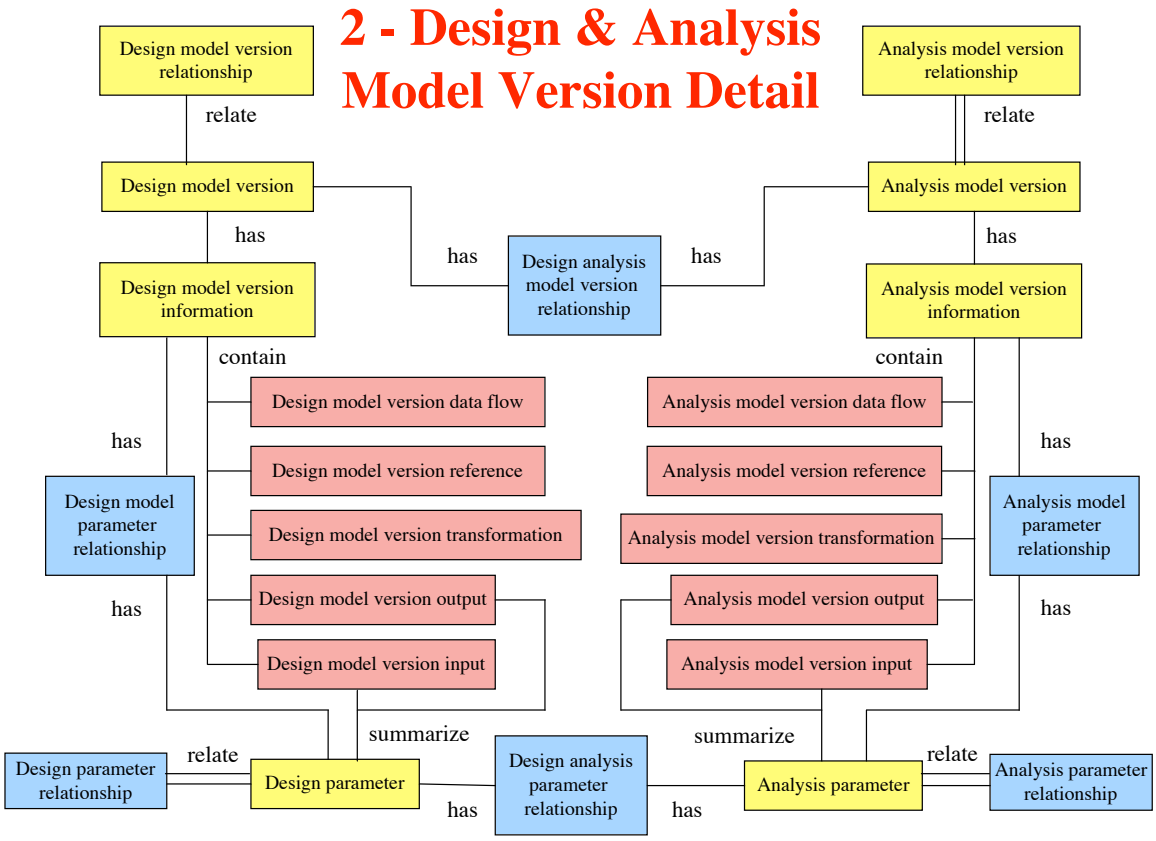
Rational approximation

Rational approximation has [is associated with] **Justification**

Rational approximation has [is associated with] **Approval**

Rational approximation has [is associated with] **Documentation**

Figure 2 presents the detail associated with both design and analysis model versions. This is intended to be fully consistent with STEP foundational modules and with analysis design relationship connectivity used within STEP-AP209 (Engineering analysis). In this concept model relationships between design and analysis parameters are identified. These parameters are those that the systems engineer uses to communicate with the respective support domains. Both design and analysis models have input, transformation and output; additionally they are assumed to have some reference base and may be embedded within some complex analysis model represented by some data flow diagram, see figure 3.



Define design and analysis parameter relationships (fig 2)

Analysis model parameter relationship has [is associated with]
Analysis model version information

Analysis model parameter relationship has [is associated with]
Analysis parameter

Design model parameter relationship has [is associated with]
Design model version information
Design model parameter relationship has [is associated with]
Design parameter

Form simple hierarchy of analysis and design model versions (fig 2)

Analysis model version relationship has relating [is relating to]
Analysis model version
Analysis model version relationship has related [is related to]
Analysis model version
Design model version relationship has relating [is relating to]
Design model version
Design model version relationship has related [is related to]
Design model version

Analysis model version information subtypes (fig 2,3)

Analysis model version information
is generalization of [is specialization of]

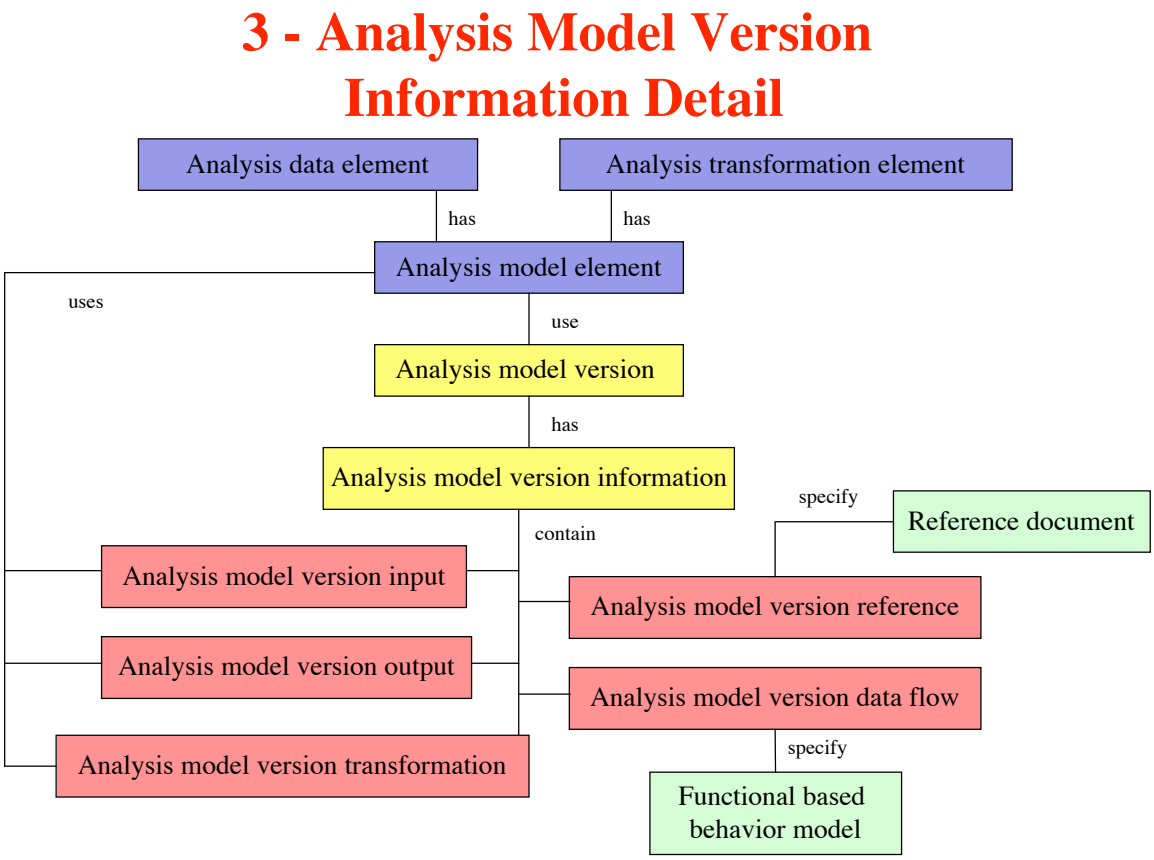
- **Analysis model version input** uses [is used by]
  Analysis model element
- **Analysis model version output** uses [is used by]
  Analysis model element
- **Analysis model version transformation** uses [is used by]
  Analysis model element
- **Analysis model version reference** is specified by [specifies]
  Reference document
- **Analysis model version data flow** is specified by [specifies]
  Function based behavior model

Design model version information subtypes (fig 2,4)

Design model version information
is generalization of [is specialization of]

- **Design model version input** uses [is used by]
  Analysis model element
- **Design model version output** uses [is used by]
  Analysis model element
- **Design model version transformation** uses [is used by]
  Analysis model element
- **Design model version reference** is specified by [specifies]
  Reference document
- **Design model version data flow** is specified by [specifies]
  Function based behavior model

Figure 3 presents the breakdown detail associated with Analysis model version. The analysis model version is assumed to be made up of a collection of analysis model elements that are linked together via a data flow diagram presented in manner consistent with AP233's Function based behavior module.

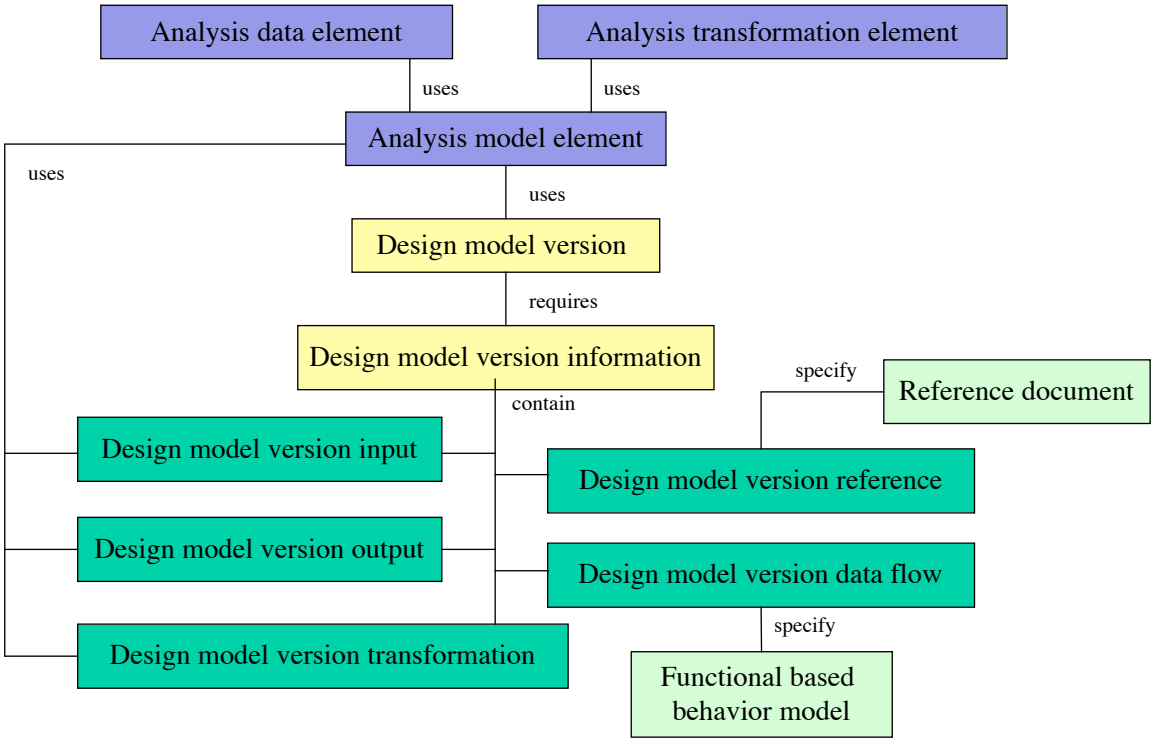


Analysis model version is built from elemental building blocks (fig 3)

Analysis model version uses [is used by] **Analysis model element**
Analysis model element has [is associated with] **Analysis data element**
Analysis model element has [is associated with] **Analysis transformation element**
Analysis model version has [is associated with] **Analysis model version information**

Figure 4 is a mirror reflection of figure 3. It presents the breakdown detail associated with design model version. The design model version is assumed to be made up of a collection of design model elements that are linked together via a data flow diagram presented in manner consistent with AP233's Function based behavior module.

4 - Design Model Version Information Detail



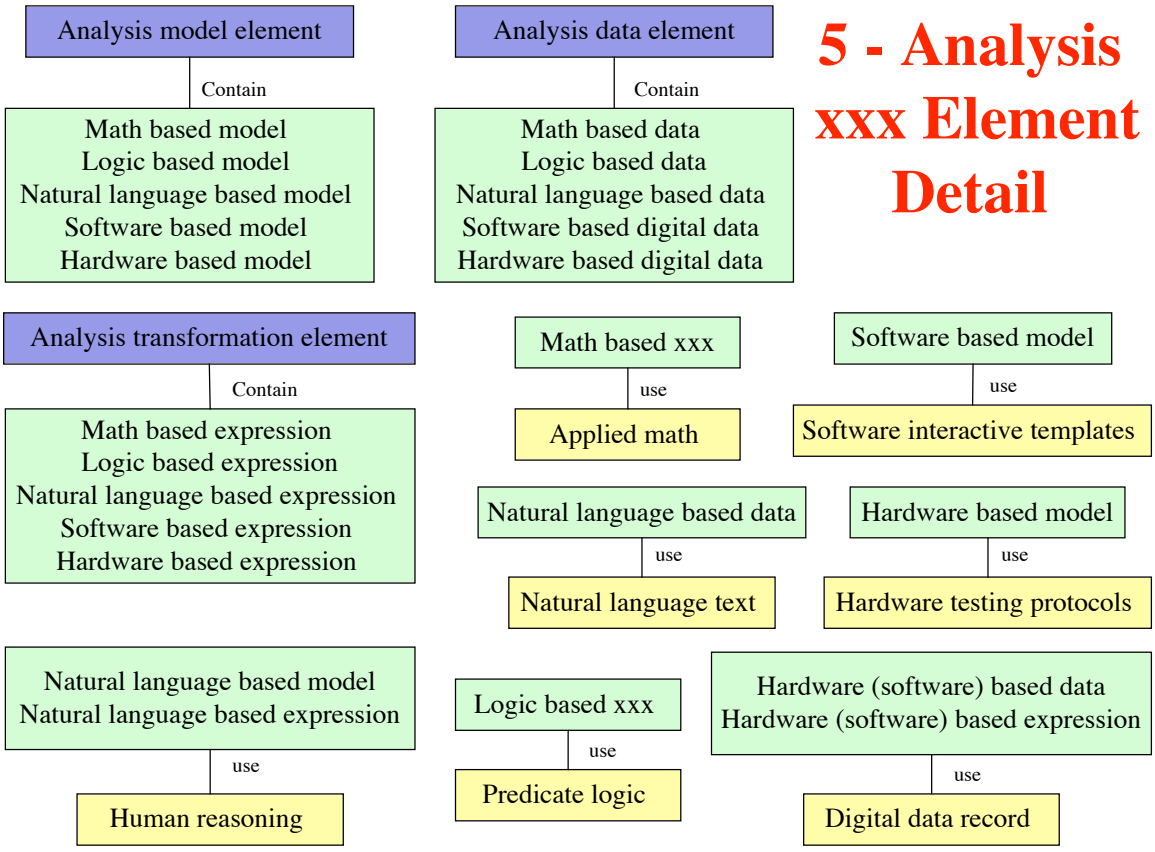
Design model version is built from elemental building blocks (fig 4)

Design model version uses [is used by] **Analysis model element**

Design model version has [is associated with]

Design model version information

Figure 5 presents the detail associated with analysis model, data and transformation elements. These all trace down to some math, logic, natural language, software or hardware representation. These end points are should be consistent with existing STEP module capability.



Analysis model element subtypes (fig 5)

Analysis model element is generalization of

- **Math based model** uses [is used by] **applied math**
- **Logic based model** uses [is used by] **predicate logic**
- **Natural language based model** uses [is used by] **human reasoning**
- **Hardware based model** uses [is used by] **hardware testing protocols**
- **Software based model** uses [is used by] **digital data record**

Analysis data element subtypes (fig 5)

Analysis data element is generalization of

- **Math based data** uses [is used by] **applied math**
- **Logic based data** uses [is used by] **predicate logic**

- **Natural language based data** uses [is used by] **natural language text**
- **Software based digital data** uses [is used by] **digital data record**
- **Hardware based digital data** uses [is used by] **digital data record**

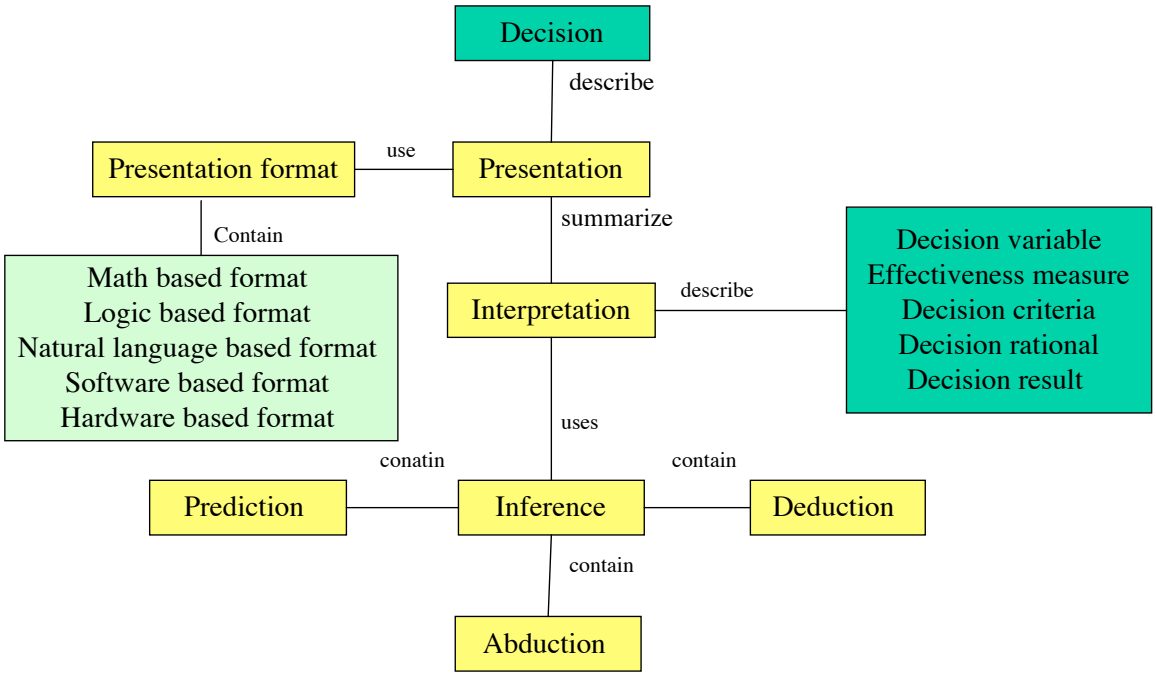
Analysis transformation element subtypes (fig 5)

Analysis transformation element is generalization of

- **Math based expression** uses [is used by] **applied math**
- **Logic based expression** uses [is used by] **predicate logic**
- **Natural language based expression** uses [is used by] **human reasoning**
- **Software based expression** uses [is used by] **digital data record**
- **Hardware based expression** uses [is used by] **digital data record**

Figure 6 presents the detail associated with decision interpretation and presentation. It is the presentation and all associated information that provides the desired record of decision rational.

6 - Decision supported by Interpretation & Presentation



Elements of interpretation and presentation (fig 6)

Decision variable is described by [describes] **Interpretation**
Effectiveness measure is described by [describes] **Interpretation**
Decision criteria is described by [describes] **Interpretation**
Decision bound is described by [describes] **Interpretation**
Decision result is described by [describes] **Interpretation**
Interpretation uses [is used by] **Inference**
Inference is generalization of [is specialization of]

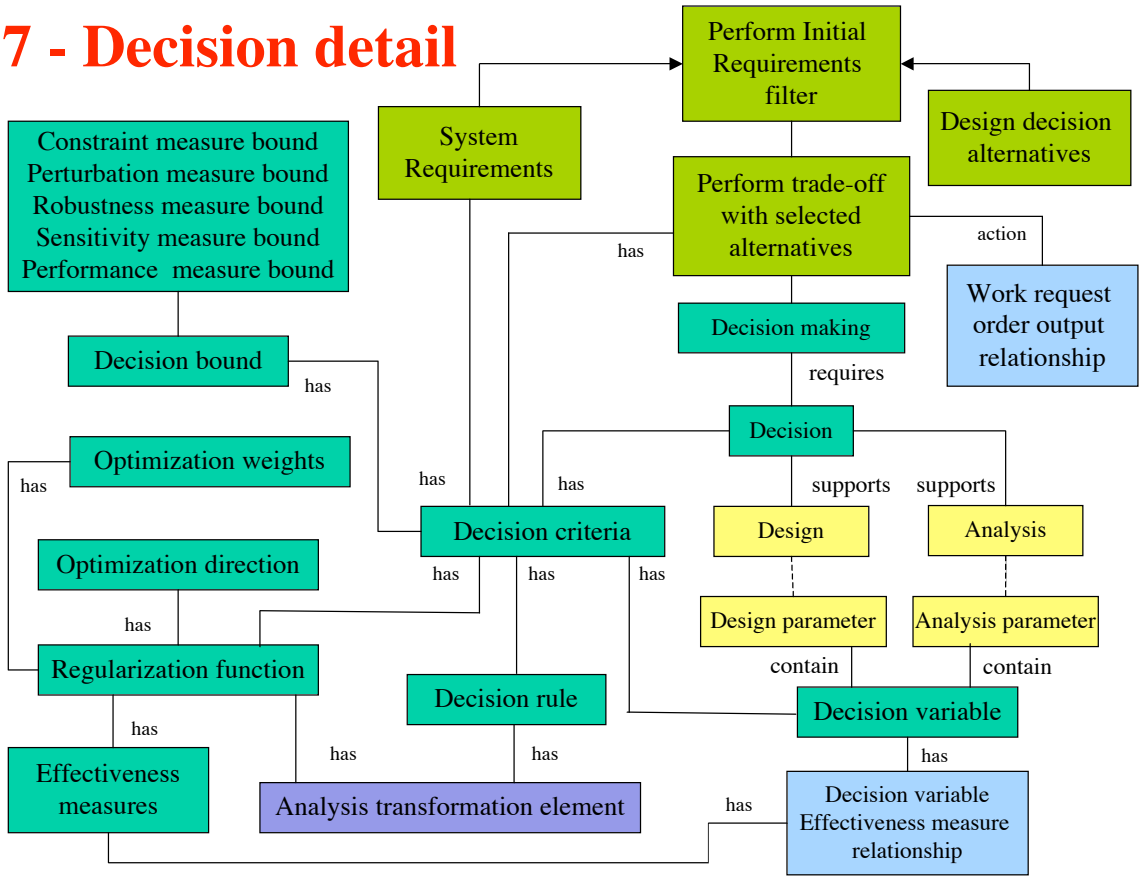
- **Prediction**
- **Deduction**
- **Abduction**

Interpretation summarizes [is summarized by] **Presentation**
Presentation uses [is used by] **Presentation format**
Presentation format is generalization of

- **Text based format**
- **Math based format**
- **Predicate logic based format**
- **Graphics based format**
- **Animation based format**

Decision is described by [describes] **Presentation**

Figure 7 presents the detail associated with a decision. Decision-making begins with a set of alternatives that pass some initial filtering process that discards obviously bad proposed alternative solutions. The actual decision is based upon some criteria that has embedded rules and takes into account initial system requirements along with all manner of constraint bounds and system optimization considerations embedded with the effectiveness measures and associated regularization functions.



Elements of a decision (fig 7)

Regularization function has [is associated with] **Analysis transformation element**

Decision rule has [is associated with] **Analysis transformation element**

Decision criteria has [is associated with] **Decision rule**

Decision criteria has [is associated with] **Regularization function**

Decision criteria has [is associated with] **Decision bound**

Decision criteria has [is associated with] **Decision variable**

Decision rule is defined by [defines] **Regularization function**

Decision variable is generalization of [is specialization of]

- **Design parameter**
- **Analysis parameter**

Decision bound is generalization of [is specialization of]

- **Constraint measure bound**
- **Perturbation measure bound**
- **Robustness measure bound**
- **Sensitivity measure bound**
- **Performance measure bound**

Decision variable effectiveness measure relationship
has [is associated with] **Decision variable**

Decision variable effectiveness measure relationship
has [is associated with] **Effectiveness measure**

Definitions for classes and properties (Concept model boxes and lines)

Most definitions are from One Look Online dictionary, see, http://www.onelook.com/

Class definitions:

- **Abduction or abductive reasoning or abductive inference -** The process of reasoning to the best explanations. In other words, it is the reasoning process that starts from a set of facts and derives their most likely explanations. The term abduction is sometimes used to mean just the generation of hypotheses to explain observations or conclusions, but the former definition is more common both in philosophy and computing. Deduction and abduction differ in the direction in which a rule like "a entails b" is used for inference (see also logical reasoning for a comparison with induction): The term is sometimes used to mean just the generation of hypotheses to explain observations or conclusions, but the former definition is more common both in philosophy and computing.
- **Analysis -** noun: an examination of a complex, its elements, and their relations; an investigation of the component parts of a whole and their relations in making up the whole; the abstract separation of a whole into its constituent parts in order to study the parts and their relations; a detailed examination of the elements or structure of something; a method in philosophy of resolving complex expressions into simpler or more basic ones; clarification of an expression by an elucidation of its use in discourse.
- **AP233 System Engineering – Ref. AP233 working draft #1 at:** https://sourceforge.net/project/showfiles.php?group_id=27487&package_id=154135&release_id=332007
- **Applied math –** noun: the branches of mathematics that are involved in the study of the physical or biological or sociological world. Implicit is the need to record this information in a computer sensible manner consistent with recall and reuse. For example: text based documents, MathML, Mathematica scripts, etc.
- **Approval – Ref. STEP foundation modules**
- **Bound –** noun: a limitation or restriction.

- **Constraint –** noun: a limitation or restriction.
- **Constraint bound –**a quantifiable limitation or restriction applied to something.
- **Data flow –** noun: The path taken by a message from origination to destination. Data flow is one kind of data driven architecture. It is a technique for specifying parallel computation at a fine-grain level, usually in the form of two-dimensional graphs.
- **Decision -** noun: the act of making up your mind about something; a position or opinion or judgment reached after consideration.
- **Decision making -** noun: the cognitive process of reaching a decision
- **Decision alternative -** one of a number of things from which only one can be chosen
- **Decision criteria -** a basis for comparison; a reference point against which other things can be evaluated
- **Decision rational -** consistent with or based on or using reason; having its source in or being guided by the intellect; of or associated with or requiring the use of the mind
- **Decision result -** a phenomenon that follows and is caused by some previous phenomenon; that which is derived from the cognitive process of reaching or drawing conclusions; the opinion formed by judging something; the document stating the conclusion reached
- **Decision rule -** a principle or condition or standard procedure or prescribed guide that governs the cognitive process of reaching or drawing conclusions; a basic generalization that is accepted as true and that can be used as a basis for reasoning.
- **Decision variable –** Relative to the "decision making process" a quantity that can assume any of a set of values; a symbol (like x or y) that is used in mathematical or logical expressions to represent a variable quantity; something that is likely to vary; something that is subject to variation
- **Deduction or deductive inference or deduction -** Is inference in which the conclusion is of lesser or equal generality than the premises, as opposed to inductive reasoning, where the conclusion is of greater generality than the premises. Other theories of logic define deductive reasoning as inference in which the conclusion is just as certain as the premises, as

opposed to inductive reasoning, where the conclusion can have less certainty than the premises. In both approaches, the conclusion of a deductive inference is necessitated by the premises: the premises can't be true while the conclusion is false. (In Aristotelian logic, the premises in inductive reasoning can also be related in this way to the conclusion.)

- **Design -** noun: the act of working out the form of something; a preliminary sketch indicating the plan for something; something intended as a guide for making something else; The arrangement of parts or the form of construction; The approach that engineering (and some other) disciplines use to specify how to create or do something. A successful design must satisfy a (perhaps informal) functional specification (do what it was designed to do); conforms to the limitations of the target medium (it is possible to implement); meets implicit or explicit requirements on performance and resource usage (it is efficient enough).
- **Digital data record –** noun: a computer sensible information file
- **Direction –** noun: a general course along which something has a tendency to develop; the act of setting and holding a course.
- **Documentation – Ref. STEP foundation modules**
- **Element –** noun: an abstract part of something; an artifact that is one of the individual parts of which a composite entity is made up; especially a part that can be separated from or attached to a system; a distinct group within a larger group; a basic member of a mathematical or logical class or set; one of a number of distinct groups composing a larger group; one of the necessary data or values on which calculations or conclusions are based; a member of a set or class; a constituent part
- **Effectiveness measure - Ref. AP233**
- **Expression -** noun: a group of symbols that make a mathematical statement; a group of words that form a constituent of a sentence and are considered as a single unit. **NOTE: These definitions are inadequate. The concept model must span expressions in the domains of math, logic, natural language, software and hardware. Each provides computer sensible representation of the transformation that exists between input and output.**
- **Format –** noun: the general appearance of a publication; the organization of information according to preset specifications (usually for computer processing)

- **Function based behavior – Ref. AP233**
- **Hardware based -** noun: written protocols or procedures or rules and associated documentation pertaining to the operation of a physically realizable hardware system.
- **Hardware testing protocols -** a human written document in any language that defines all that is needed to obtain reproducible test results with some physical system and test facility.
- **Human reasoning –** the cerebrally stored algorithm owned by a specific person used to transform inputs from any source to a rational output (opinion) expressed in natural language.
- **Information -** noun: a collection of facts from which conclusions may be drawn; the communication or reception of knowledge or intelligence; the attribute inherent in and communicated by one of two or more alternative sequences or arrangements of something; something (as a message, experimental data, or a picture) which justifies change in a construct (as a plan or theory) that represents physical or mental experience or another construct; a word which has many different meanings in everyday usage and in specialized contexts, but as a rule, the concept is closely related to others such as data, instruction, knowledge, meaning, communication, representation, and mental stimulus. Information is a message, something to be communicated from the sender to the receiver. A concept from the ontology of a modeler, problem solver, or decision-maker. In this view, information is data describing a property of an object or entity of interest.
- **Input – Ref. AP233**
- **Issue -** noun: an important question that is in dispute and must be settled; some situation or event that is thought about; a phenomenon that follows and is caused by some previous phenomenon; A technical problem, also referred to as a "known issue," an "intermittent issue," a "design side effect," or "undocumented behavior."
- **Inference -** noun: the reasoning involved in drawing a conclusion or making a logical judgment on the basis of circumstantial evidence and prior conclusions rather than on the basis of direct observation
- **Interpretation –** noun: an explanation of something that is not immediately obvious
- **Justification – Ref. STEP foundation modules**
- **Logic based -** noun: the branch of philosophy that analyzes

inference; the formal mathematical study of the methods, structure, and validity of mathematical deduction and proof. In mathematical logic the predicate calculus, predicate logic or calculus of propositional functions is a formal system used to describe mathematical theories. The predicate calculus is an extension of propositional calculus, which is inadequate for describing more complex mathematical structures. Grammatically speaking the predicate calculus adds a predicate-subject structure and quantifiers on top of the existing propositional calculus. A subject is a name for a member of a given group of individuals (a set) and a predicate is a relation on this group.

- **Math based –** noun: the branch of science concerned with number, quantity, and space, either as abstract ideas (pure mathematics) or as applied to physics, engineering, and other subjects (applied mathematics).
- **Measure –** noun: how much there is of something that you can quantify.
- **Natural language based -** noun: a human written or spoken language used by a community; opposed to e.g. a computer language
- **Natural language text -** noun: a human written document in any language. Implicit is the assumption that it can be referenced for future reuse.
- **Output – Ref. AP233**
- **Optimization –** noun: an act, process, or methodology of making something (as a design, system, or decision) as fully perfect, functional, or effective as possible;
- **Parameter –** noun: a secondary influence on a system that causes it to deviate slightly
- **Performance –** noun: process or manner of functioning or operating
- **Performance bound –**a quantifiable limitation or restriction applied relative to the process or manner of functioning or operating.
- **Perturbation –** a secondary influence on a system that causes it to deviate slightly
- **Perturbation bound –**a quantifiable limitation or restriction applied relative to a secondary influence on a system that causes it to deviate slightly.
- **Predicate logic –** In mathematical logic the predicate calculus,

predicate logic or calculus of propositional functions is a formal system used to describe mathematical theories. The predicate calculus is an extension of propositional calculus, which is inadequate for describing more complex mathematical structures. Grammatically speaking the predicate calculus adds a predicate-subject structure and quantifiers on top of the existing propositional calculus. A subject is a name for a member of a given group of individuals (a set) and a predicate is a relation on this group. In mathematical logic, second-order logic is an extension of either propositional logic or first-order logic which contains variables in predicate positions (rather than only in term positions, as in first-order logic), and quantifiers binding them. Implicit is the need to record this information for future recall and reuse.

- **Presentation –** noun: a show or display; the act of presenting something to sight or view; the activity of formally presenting something; the act of making something publicly available; presenting news or other information by broadcasting or printing it.
- **Rational approximation –** an approximation of something: having its source in or being guided by the intellect; consistent with or based on or using reason; of or associated with or requiring the use of the mind.
- **Regularization function – Ref. AP233**
- **Relationship – Ref. STEP foundation modules**
- **Reference – Ref. STEP foundation modules**
- **Risk – Ref. AP233**
- **Robustness –** having or exhibiting operational strength relative to system input uncertainty; strong and unlikely to break or fail.
- **Robustness bound –**a quantifiable limitation or restriction applied relative to system input uncertainty.
- **Sensitivity –** the degree to which something responds to stimuli
- **Sensitivity bound –**a quantifiable limitation or restriction on the degree to which something responds to stimuli.
- **Software based –** noun: (computer science) written programs or procedures or rules and associated documentation pertaining to the operation of a computer system and that are stored in read/write memory; a computer sensible product, tool used for computer aided work.
- **Software based –** noun: (computer science) written programs or

procedures or rules and associated documentation pertaining to the operation of a computer system and that are stored in read/write memory; a computer sensible product, tool used for computer aided work.

- **Software interactive template –** noun: the real time computer aid interactive interface between a human and a software capability.
- **Stakeholder –** The people who have a vested interest in the outcome of the project. A person, group, or community who has an interest (has a stake in or may be impacted by a given approach) in the system. A person with an interest or concern in something. A person or group who can affect or is affected by an action. Responsible decision-making requires consideration of the effects on all stakeholders.
- **Stakeholder need -** The business or operational problem (opportunity) that must be fulfilled in order to justify purchase or use.
- **Systems Engineering – Ref. AP233**
- **Trade off – Ref. AP233**
- **Transformation – Ref. AP233**
- **Validation plan – Ref. AP233**
- **Validation requirement – Ref. AP233**
- **Validation result – Ref. AP233**
- **Verification plan – Ref. AP233**
- **Verification requirement – Ref. AP233**
- **Version – Ref. STEP foundation modules**
- **View – Ref. STEP foundation modules**
- **Weight - Ref. AP233**

Property definitions:

- Contain - verb: have as a component, have or hold within;
  - Is generalization of
  - Is specialization of
- Define - verb: determine the nature of; delineate the form or outline of; give a definition for the meaning of a word; determine the essential quality of
  - Defines
  - Is defined by
- Describe - verb: give a detailed account in words of
  - Describe

    - Is described by
- Have -possess - verb: possess, own, or hold; to be able to make use of; to stand in a certain relationship to; to acquire or get possession of; to be marked or characterized by (a quality, attribute, or faculty)
    - Has
    - Is associated with
- Relate - verb: have to do with or be relevant to; make a logical or causal connection; intention is to mirror STEP EXPRESS entities that define relating-related relationships.
    - Has relating
    - Is relating to
    - Has related
    - Is related to
- Require - verb: consider obligatory; request and expect; have need of; require as useful, just, or proper.
    - requires
    - is required by
- Specify - verb: determine the essential quality of ; decide upon or fix definitely; select something or someone for a specific purpose; specify as a condition or requirement in a contract or agreement; make an express demand or provision in an agreement; define clearly.
    - specifies
    - is specified by
- Summarize - verb: to express the most important facts or ideas about something or someone in a short and clear form
    - summarizes
    - is summarized by
- Use - verb: take or consume; avail oneself to; take, hold, or deploy as a means of achieving something; consume or expend the whole of; consume or expend the whole of; to put into service; make work; make use of; employ for a particular purpose:
    - uses
    - is used by

Lambda Calculus view of Ontology theory

The following just collects some ideas that I don't want to forget. They may or may not prove useful. I have just cut and pasted here some old work with the germs of some ideas that I may or may not follow up with at the end.

Extract from File: Parameter, property and Lambda calculus.doc

On Property & Parameter

Harold P. Frisch

02, June 2002

Extract from local file Lambda calculus.pdf obtained from:
http://citeseer.nj.nec.com/barendregt94introduction.html

Some history

Leibniz had as ideal the following.

(1) Create a `universal language' in which all possible problems can be stated.
(2) Find a decision method to solve all the problems stated in the universal language.

If one restricts oneself to mathematical problems, point (1) of Leibniz' ideal is fulfilled by taking some form of set theory formulated in the language of first order predicate logic.

This was the situation after Frege and Russell (or Zermelo).

Point (2) of Leibniz' ideal became an important philosophical question. `Can one solve all problems formulated in the universal language?' It seems not, but it is not clear how to prove that. This question became known as the Entscheidungsproblem.

In 1936 the Entscheidungsproblem was solved in the negative independently by Alonzo Church and Alan Turing. In order to do so, they needed a formalisation of the intuitive notion of `decidable', or what is equivalent `computable'. Church and Turing did this in two different ways by introducing two models of computation.

(1) Church (1936) invented a formal system called the ***lambda calculus*** and defined the notion of ***computable function*** via this system.
(2) Turing (1936/7) invented a class of machines (later to be called ***Turing machines***) and defined the notion of ***computable function*** via these machines.

Also in 1936 Turing proved that both models are ***equally strong*** in the sense that ***they define the same class of computable functions*** (see Turing (1937)). Based on the concept of a Turing machine are the present day Von Neumann computers. Conceptually these are Turing machines with random access registers. Imperative programming languages such as Fortran, Pascal etcetera as well as all the assembler languages are based on the way a Turing machine is instructed: by a sequence of statements.

Functional programming languages, like Miranda, ML etcetera, are based on the lambda calculus. An early (although somewhat hybrid) example of such a language is Lisp. Reduction machines are specifically designed for the execution of these functional languages.

Quoted from Turing's paper, obtain @ http://www.abelard.org/turpap2/tp2-ie.asp

> The "computable" numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means. Although the subject of this paper is ostensibly the computable numbers, it is almost equally easy to define and investigate computable functions of an integral variable or a real or computable variable, computable predicates, and so forth. The fundamental problems involved are, however, the same in each case, and I have chosen the computable numbers for explicit treatment as involving the least cumbrous technique. I hope shortly to give an account of the relations of the computable numbers, functions, and so forth to one another. This will include a development of the theory of functions of a real variable expressed in terms of computable numbers. According to my definition, a number is computable if its decimal can be written down by a machine.
>
> In a recent paper Alonzo Church[2] has introduced an idea of "effective calculability", which is equivalent to my "computability", but is very differently defined. Church also reaches similar conclusions about the Entscheidungsproblem.[3] The proof of equivalence between "computability" and "effective calculability" is outlined in an appendix to the present paper.

Turing Machine vs. Lambda Calculus Information Storage Views

The key to seeing view difference starts with the output, then the input and then the executable in the classic block diagram:

Input → Executable → Output

From a Turing machine perspective information gets packaged as

(Input) → (Executable) → (Output)

From a Lambda calculus perspective information gets packaged as

(Input → Executable) → (Output)

Where (…) implies information package

If it is feasible to store information ***non-redundantly*** then the Turing machine approach allows for the creation of a neutral format of all associated information. This enables the ability to PDM and to insure data integrity via the mapping algorithms that are used to read and write information to and from the neutral format. This is done by STEP AP's 203, 209, 210, 214, etc. in a manner that services their respective domains of focus.

If the information to be stored is fundamentally ***redundant*** then the Lambda calculus approach has advantages.

It follows from the definition of a parameter that the set of parameters can be used as the quantifiers of requirements. These also can be used to define the set of all system characterization information that will be fed forward and fed back between system engineer and system stakeholders.

From the perspective of systems engineering - and hence AP233 - this is the information set that the systems engineer will use during attempts to optimize design relative to the needs of all stakeholders.

Redundancy of information in the design process stems from the many stakeholder views of the same parameter; e.g., cost, weight, power, behaviour, etc. That is, from the many sets of different dependency variables and associated executables that stakeholders will use to quantify their requirements identify their allowable variance ranges and define their validation and verification methods.

The information is redundant from the perspective that all stakeholders are quantifying the exact same parameter relative to different use perspectives, different determination procedures, different variance needs and different methods to be used for validation and verification.

These stakeholder perspectives are all encapsulated within the Lambda calculus information package view

$$(\text{Input} \rightarrow \text{Executable})$$

This can be place in pure Lambda calculus terms by the letting:

- x - set of dependency variables
- M(x) - executable expression to model, measure or observe
- N - a set of values assigned to the set of input dependency variables
- M(N) - computable number

In λ-calculus notation this is written as:

$$(\lambda x \bullet M[x])N = M[N]$$

All that one needs to see from this expression is that input (x =N) and executable M(x) are bound together in one information package that yields the output M(N).

In a redundant information domain, if the one attempts to decompose this package and store information separately as is done with the Turing machine approach one immediately looses traceability.

In a redundant information domain, inputs and executables cannot be separated from each other. The separation destroys stakeholder source and property value rational information that is critical to the system optimization process. If this metadata is lost the output becomes ambiguous relative to use by the system's engineer and reuse by any other stakeholder or engineering specialty domain.

This is unacceptable within an information model.

I find it interesting to observe that the Lambda calculus approach to information packaging allows SE's to work where they should. That is; at the, what influences what and why level, while still providing the clean interface into the specialty domains – if needed.

Some more extractions from previous work

"Parameter" Placed in a λ-Calculus Context

Harold P. Frisch

18 May 2002

λ-Calculus Foundation

The foundational definition of a functional program is:

"A functional program consists of an expression E (representing both algorithm and the input)."

"A functional program transforms data into other data according to a certain algorithm"

From Barendregt chapter 7 "Functional programming and Lambda calculus" in Handbook of theoretical computer science, edited by J. van Leeuwen, Elsevier, 1990.

For the purposes of AP233 I define parameter as:

A Parameter is a functional program that can be evaluated to provide a numeric or non-numeric parameter property value.

To transform this definition into λ-calculus notation, define:

- M[x] to be an expression which is dependent upon the set of variables x,
- N to be a particular set of values for the set of variables x,
- M[N] to be the value of the expression M[x] when x is defined to be N

In λ-calculus notation this is written as:

$$(\lambda x \bullet M[x])N = M[N]$$

The functional program (parameter) consists of both the expression M[x] and definitions for the set of variables x which map into M[x].

Note: For AP233 application the expression M[x] is generic. M[x] may be representative of an analytical math function, a CAE software tool, an expert's opinion, a trade-off metric, a test procedure, etc. In all cases the variables x identify all items of information that need to be known, relative to the expression, to insure that results are reproducible.

New stuff – Some quick thoughts

Extension to Ontology Theory as embedded in Protégé

Start with:

From a Lambda calculus perspective information gets packaged as

(Input → Executable) → (Output)

This is the doublet representation used in ontology theory. By doublets I mean class or subclass + assertion where "assertion" is formed by a single box and a one directional line. From above:

1. **(Systems Engineering** requires) → **(Decision making)**
2. **(Decision making** is required by) → **(Systems Engineering)**

These Lambda calculus doublets appear to provide the underlying theoretical basis for a STEP consistent view of this concept model.