



Are we there yet? Assessing quality in MBSE models.

Matthew Hause – Atego

Agenda

- How do I know if my model is of good quality?
- What is quality?
- Model-Based Engineering
 - SysML and UML
- Examples:
 - Requirements traceability
 - Style, Standards and Visualization
 - Parametrics
 - Measures of Effectiveness
 - Trade-off Analysis
 - Model Execution
- Quality and Process
- Questions?



How do I know if my model is of good quality?

- A valid question whether for :
 - Yourdon, IDEF, UML, SysML, DoDAF, NAF, or Lego models.
- Can be both subjective and objective
- Assessment criteria is essential for project success
 - Without assessment criteria the model will become unfocussed
 - The most important criterion is whether or not :
it communicates its intent
 - Goes to the heart of why the model was created in the first place



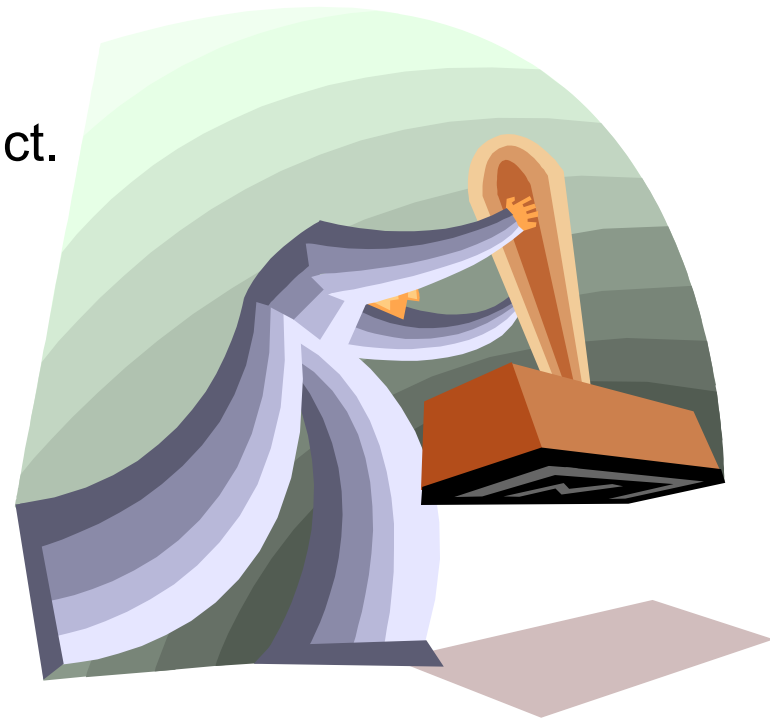
What was the question?

- “All models are *wrong*, some models are *useful*.”
Professor P.E. Box
- Models are an abstraction of the problem or solution space
 - Reflect an abstraction of one or more viewpoints
- A model should be created to answer one or more questions
 - Performance
 - Functionality
 - Timing
 - Structure
 - Usability
 - Project lifecycle
 - Product lifecycle
 - Etc.



What is quality?

- Dictionary definitions are neutral
 - 1. A distinguishing characteristic or attribute.
 - 2. The basic character or nature of something.
 - 3. A feature of personality.
 - 4. Degree or standard of excellence.
 - 5. High social status.
 - 6. Musical tone color.
 - 7. Excellent or superior; a quality product.
- Thesaurus synonyms are largely positive
 - Character
 - Sort
 - Tendency
 - Excellent
 - Goodness



Quality in Systems Development

- “A scalar attribute reflecting ‘how well’ a system functions.”
“Examples include Availability, Usability, Integrity, Adaptability, and many others.”
 - Quality levels are capable of being specified quantitatively (as are all scalar attributes)
 - Quality levels can be measured in practice
 - Quality levels can be traded off to some degree; with other system attributes valued more by stakeholders
 - When quality levels increase towards perfection, the resources needed to support those levels tend towards infinity



Planguage concept glossary, Tom Gilb

- The desired level of quality varies according to the model purpose
 - Project bid
 - Safety critical implementation
 - Brainstorming session
 - Etc.

- Model-based Systems Engineering (MBSE) is the formalized application of modeling to support system requirements, design, analysis, verification, and validation activities beginning in the conceptual design phase and continuing through-out development and later lifecycle phases.” (INCOSE, 2007).
- Modeling is at the heart of all aspects of the development effort
 - Covers the complete product and project lifecycle
 - Has a direct effect on any generated artifacts.
 - MBE encompasses architecture, systems and software development.

■ The Systems Modeling Language (SysML)

- “Supports modeling of a broad range of systems which may include hardware, software, data, personnel, procedures and facilities. “
- “Used to analyze, specify, design and verify complex systems, intended to enhance systems quality, improve the ability to exchange systems engineering information amongst tools and help bridge the semantic gap between systems, software and other engineering disciplines.” (OMG SysML, 2003).
- SysML added two new diagrams
 - parametric and requirements diagrams.
- Class and composite structure diagrams were modified
 - Include elements to model logical and physical block structures for systems engineers.
- In SysML they are called respectively:
 - Block Definition Diagram (BDD) and Internal Block Diagram (IBD)

Changes in Systems Engineering Practice

Change from Document centric to Model centric



Requirement Specifications
Interface Definitions
System Architecture
System Functionality
Trade-off Analysis
Test Specifications

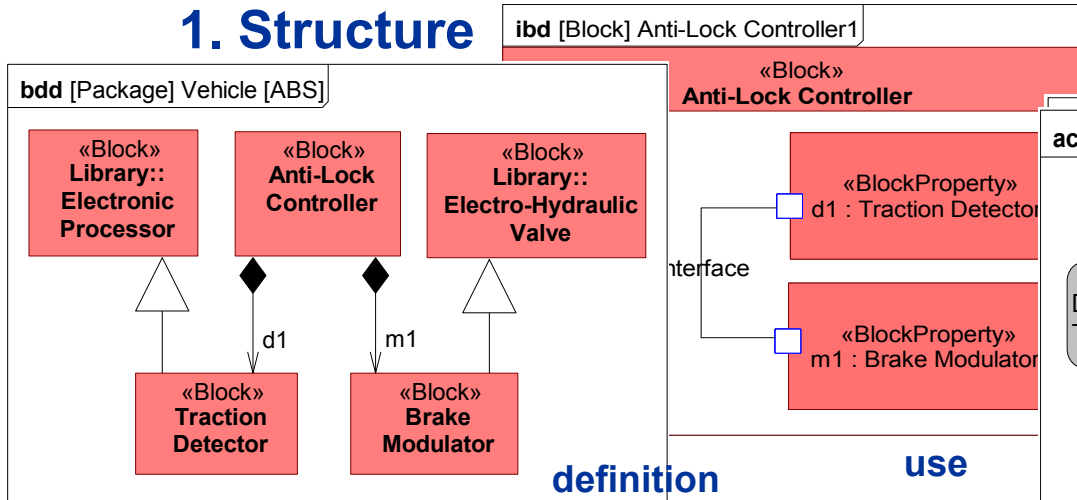


Old Approach

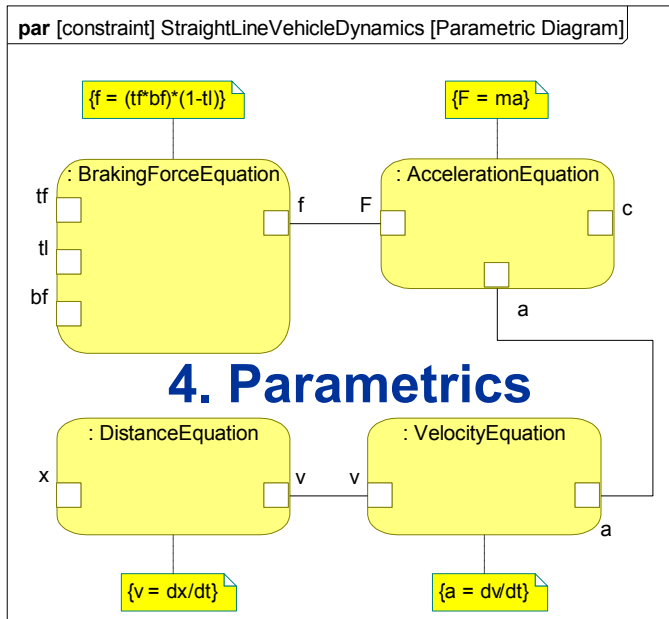
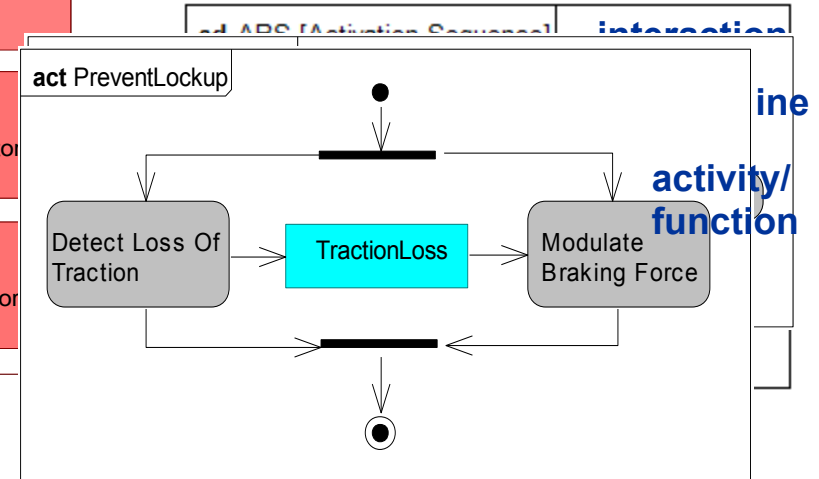
New Approach

The Four Pillars of SysML (ABS Example)

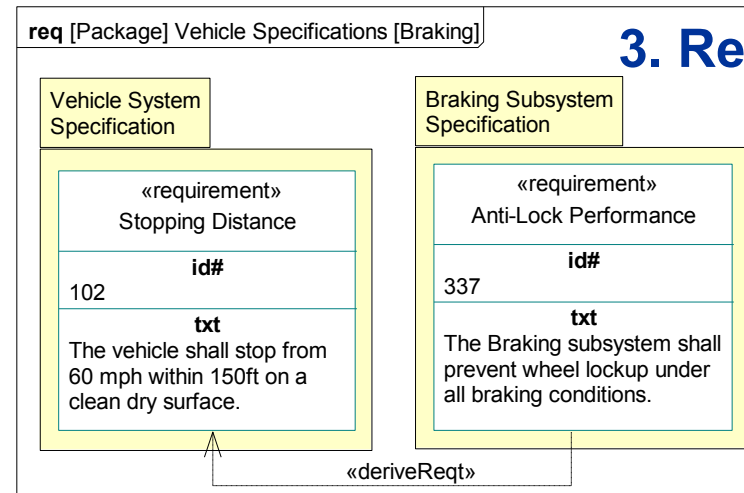
1. Structure



2. Behavior

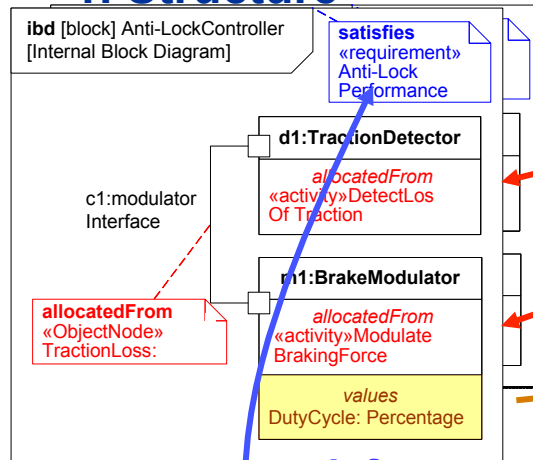


3. Requirements

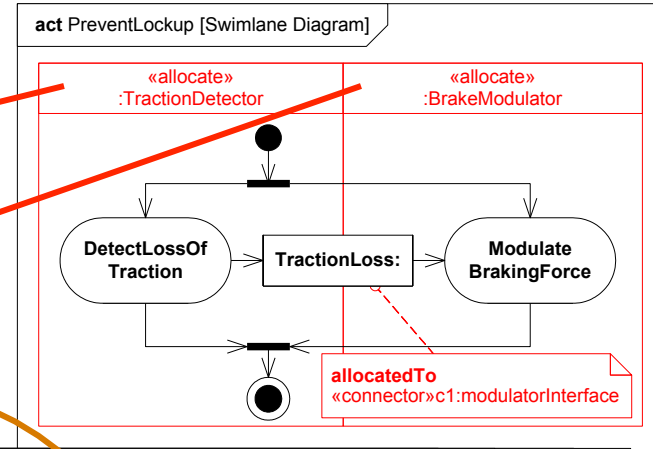


Cross Connecting Model Elements

1. Structure



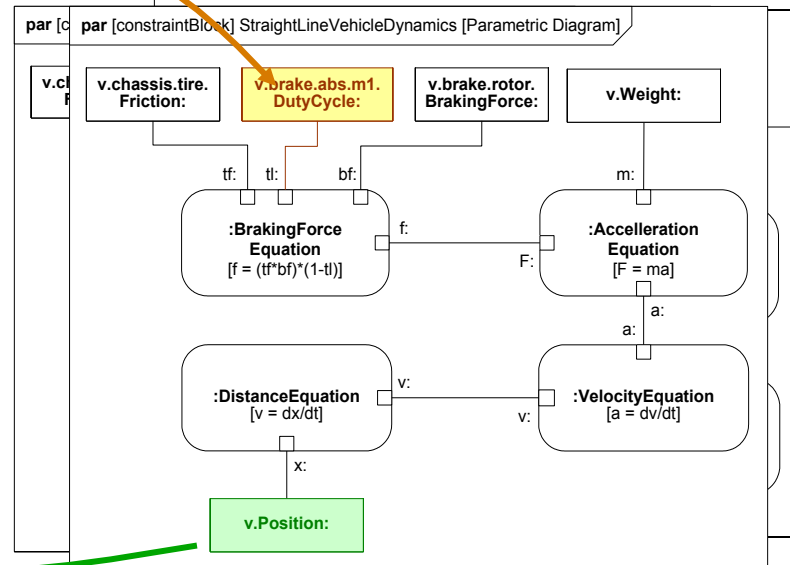
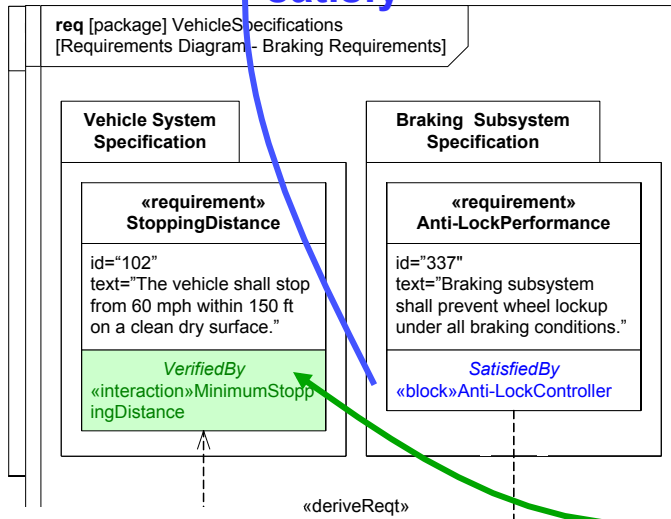
2. Behavior



allocate

value binding

satisfy



3. Requirements

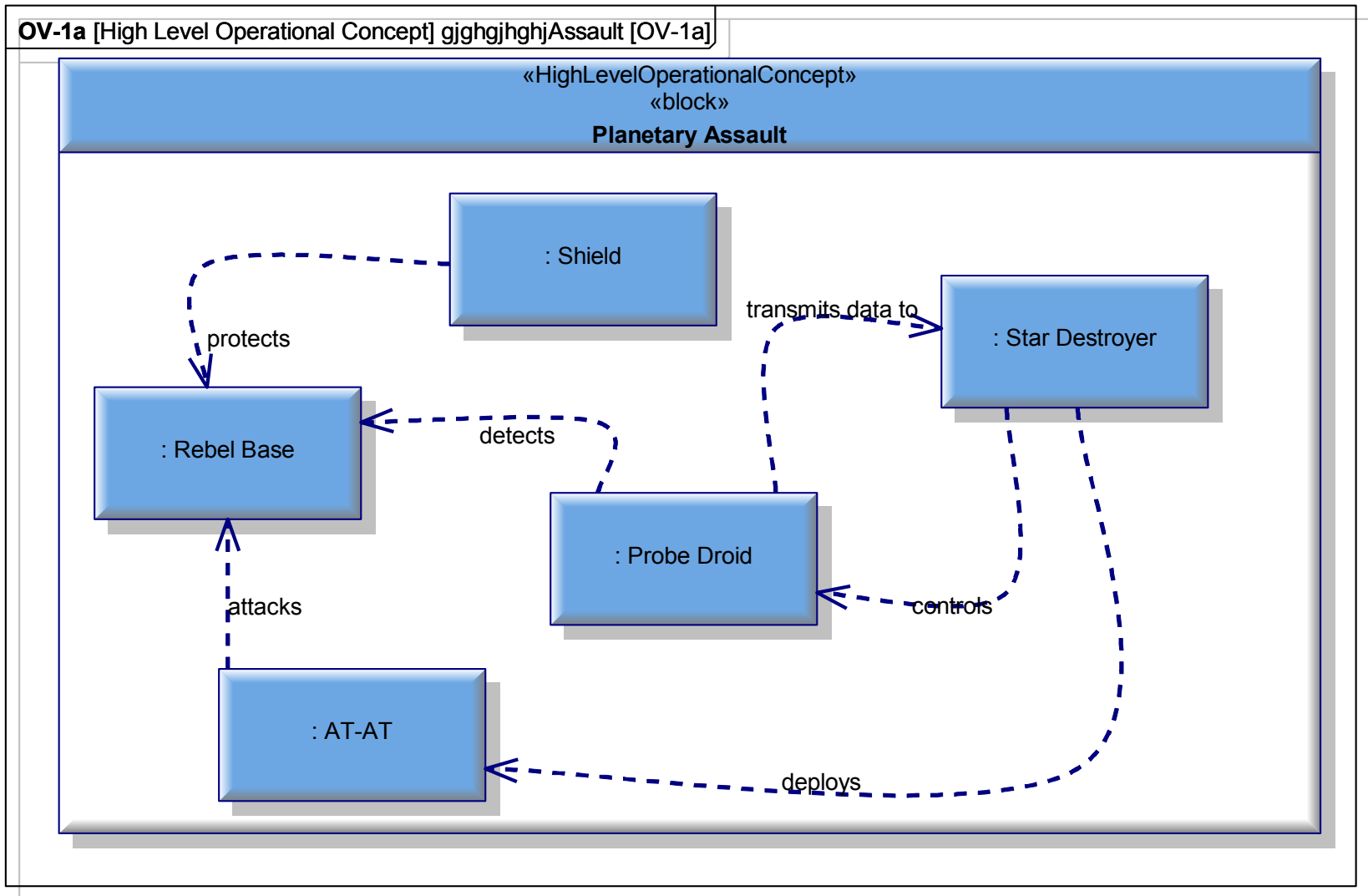
verify

4. Parametrics

Examples for estimating quality

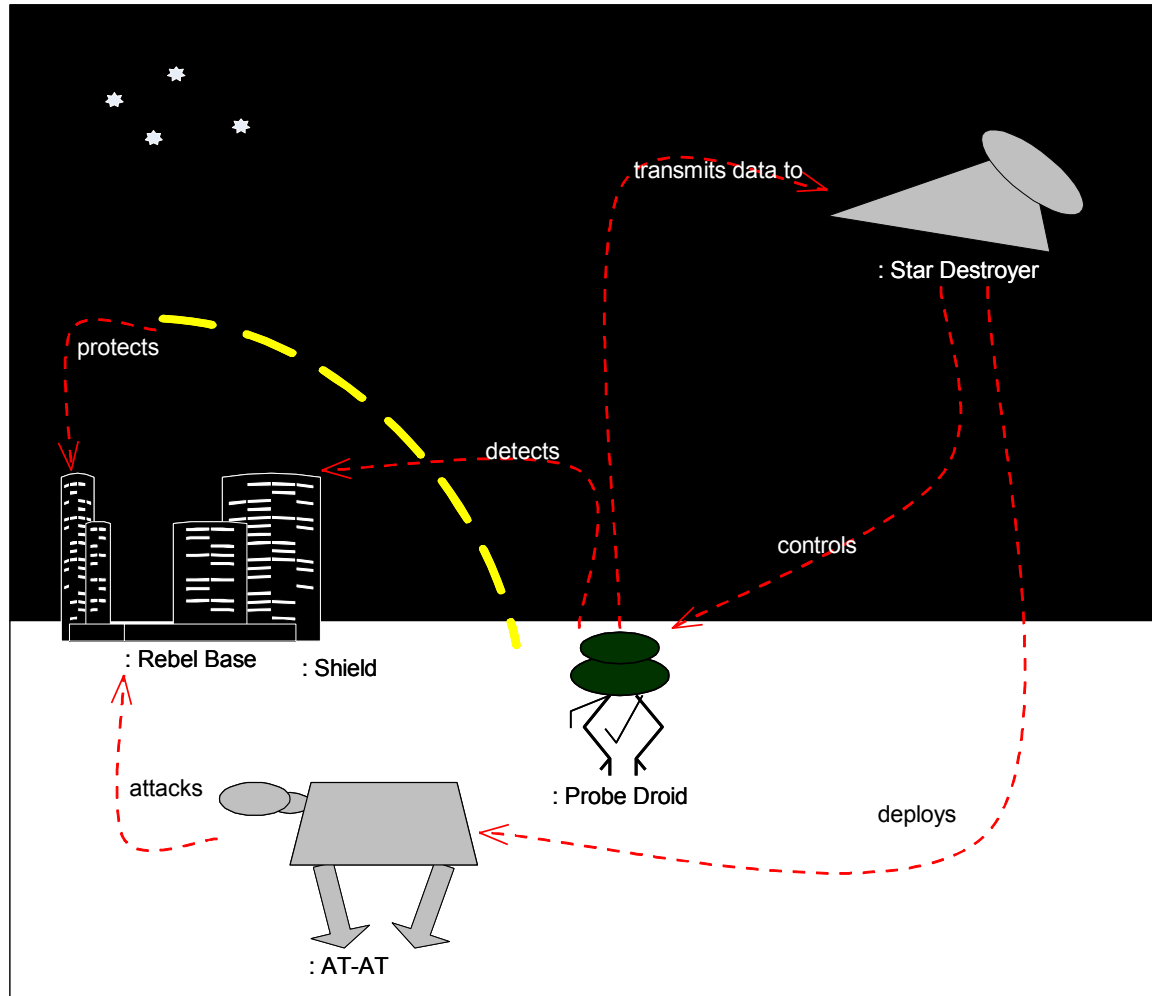
- Communication
- Requirements traceability
- Style, Standards and Visualization
- Parametrics
- Measures of Effectiveness / Trade-off Analysis
- Model Execution

Models as a means of communication - Boxes



Models as a means of communication - Graphics

OV-1a [High Level Operational Concept] Planetary Assault - Graphic Version [OV-1a]



For a non-graphic version see [\[High Level Operational Concept\] gjghgjghjAssault \[OV-1a\]](#)

Requirements Traceability

- The degree to which the inherent characteristics of process, product or system meet "The Requirements" is the quality of process, product or system, irrespective of the sub-classification or sub-categorization of "The Requirements". ISO 9000
- Quality is, therefore, a question of degree.
 - How well does this set of inherent characteristics comply with this set of requirements?
 - The quality of something depends on a set of inherent characteristics and a set of requirements and how well the former complies with the latter.
 - The quality of something cannot be established in a vacuum.
 - For ISO 9000 quality is always relative to a set of requirements.

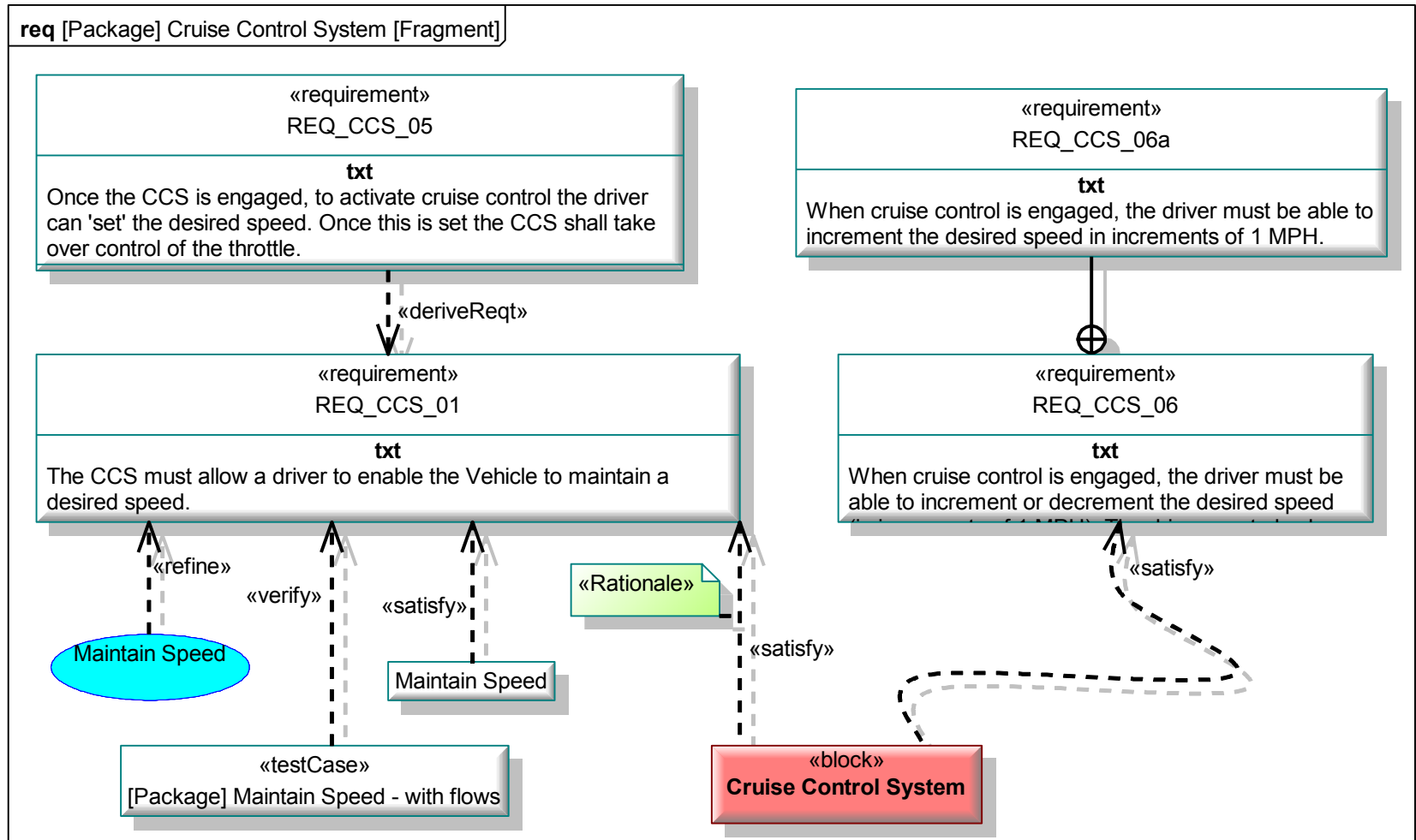
Requirements Traceability

- CMMI Level 2 also defines Requirements Management:
- **Requirements Management (REQM)**
 - **Purpose**
 - The purpose of **Requirements Management** (REQM) is to manage the requirements of the project's products and product components and to identify inconsistencies between those requirements and the project's plans and work products.
 - **Specific Practices by Goal**
 - SG 1 Manage Requirements
 - SP 1.1 Obtain an Understanding of Requirements
 - SP 1.2 Obtain Commitment to Requirements
 - SP 1.3 Manage Requirements Changes
 - SP 1.4 Maintain Bidirectional Traceability of Requirements
 - SP 1.5 Identify Inconsistencies Between Project Work and Requirements

The SysML Requirements Diagram

- Captures requirements hierarchies and the derivation, satisfaction, verification, copy, trace, and refinement relationships.
 - Relate requirements to
 - one another
 - system design model elements
 - test cases.
 - The «rationale» concept used to annotate any model element to identify supporting rationale including :
 - analysis and trade studies
 - derived requirement
 - Design decision, etc.
- The requirement diagram provides a bridge between typical requirements management tools and the system models.
- Reports and analysis can be generated to show traceability completeness, traceability trees, etc.

The SysML Requirements Diagram



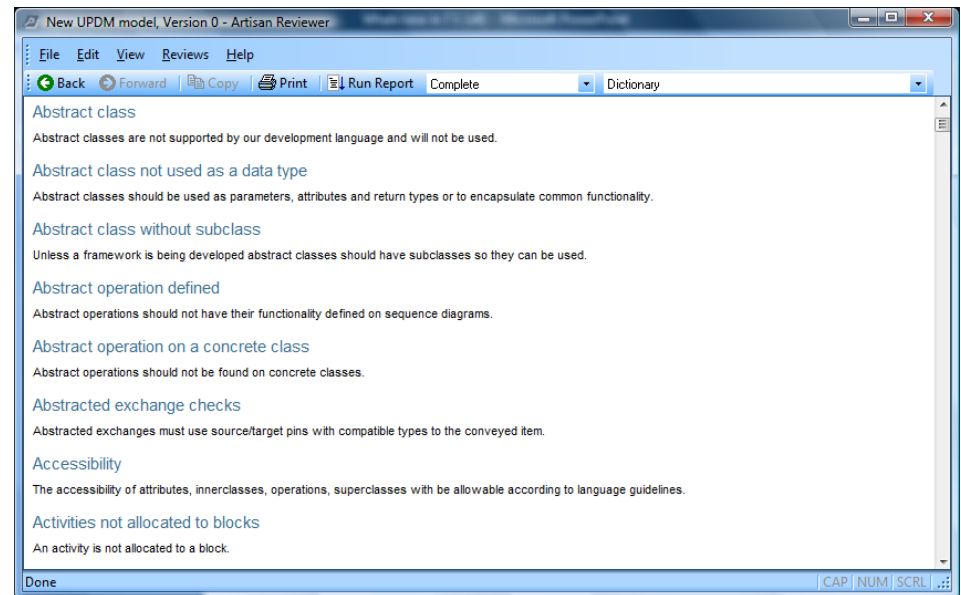
Standards, Style and Visualization

- As there are coding and documentation standards, there needs to be modeling standards.
 - Completeness and global checks
 - SysML models
 - Quality checks for model



Standards, Style and Visualization

- As there are coding and documentation standards, there needs to be modeling standards.
- Simple completeness checks
 - All required fields have been filled in
 - Use cases should contain a full use case description, pre and post conditions, intent and alternate courses.
- More complex tests
 - Use cases have been elaborated to sequence diagrams
 - Trace to or refine functional requirements
 - Use case text follows the company standard PDL.



Standards, Style and Visualization

- As there are coding and documentation standards, there needs to be modeling standards
- SysML models, ensure that
 - all activities have been allocated to structural elements,
 - item flows and port types are consistently types,
 - logical or abstract elements have been allocated to concrete or physical ones.

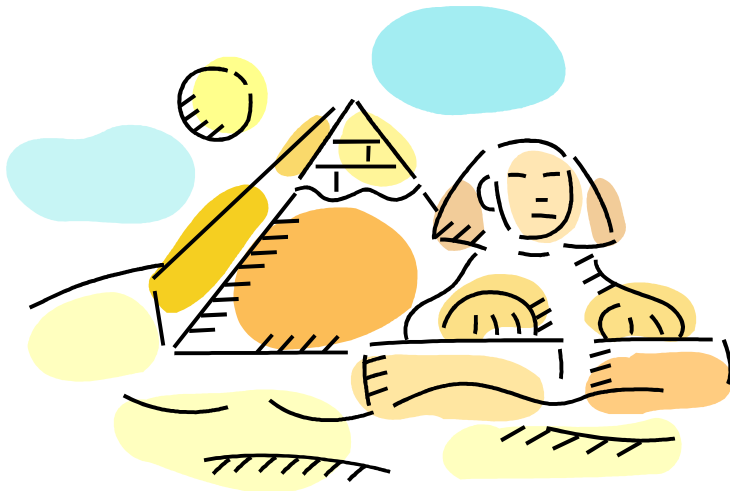


Standards, Style and Visualization

■ Generic quality checks

- Diagram complexity such as the “7 plus or minus 2” rule, (originally described by George A. Miller) to ensure that diagrams are readable.
- Ensure that state diagrams do not contain dead-end states and activity diagram paths can be executed in a deterministic fashion.

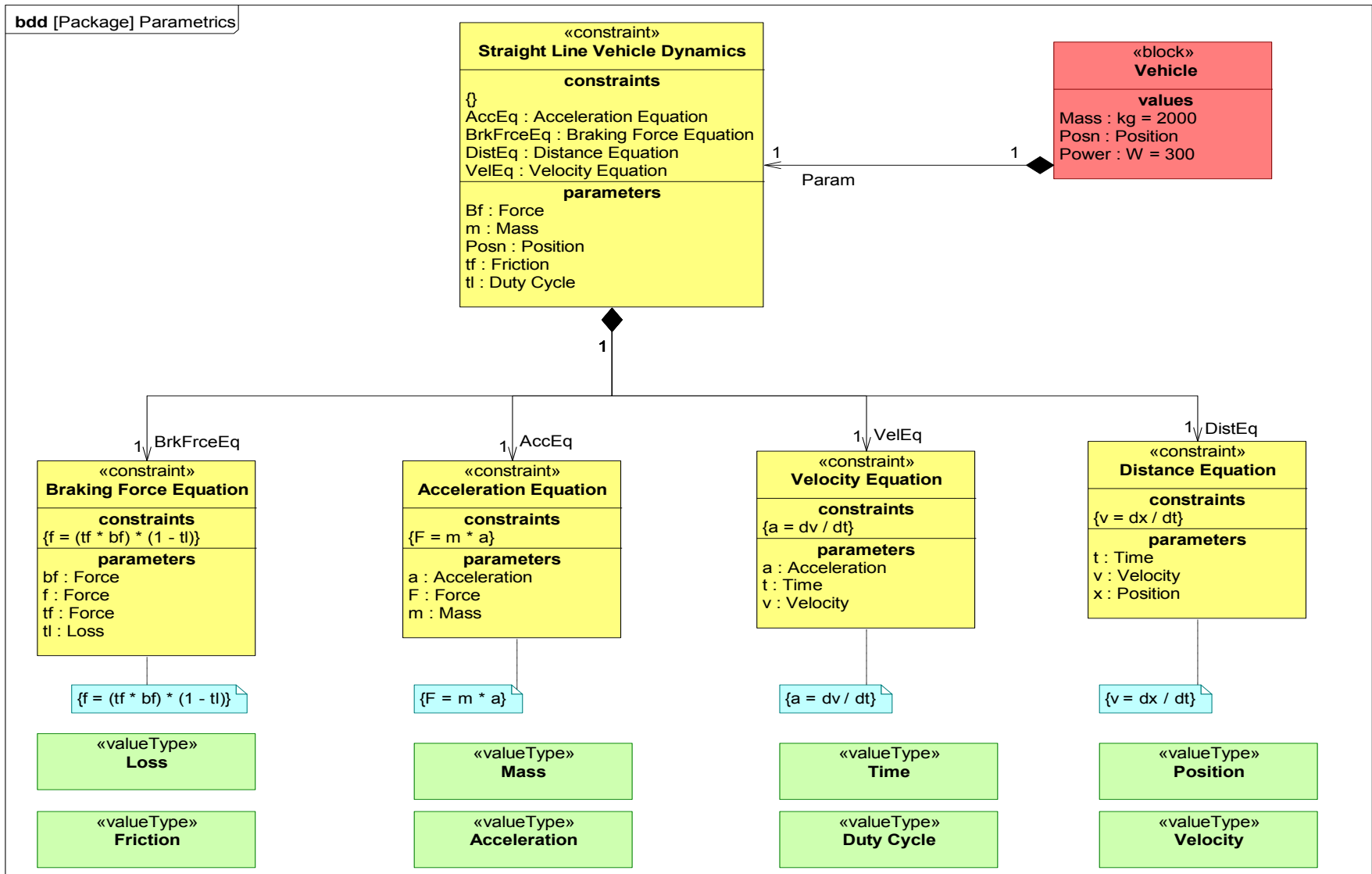
■ As part of the adoption of MBE, a style guide should be produced by the process owner with examples to ensure a consistent approach to MBE.



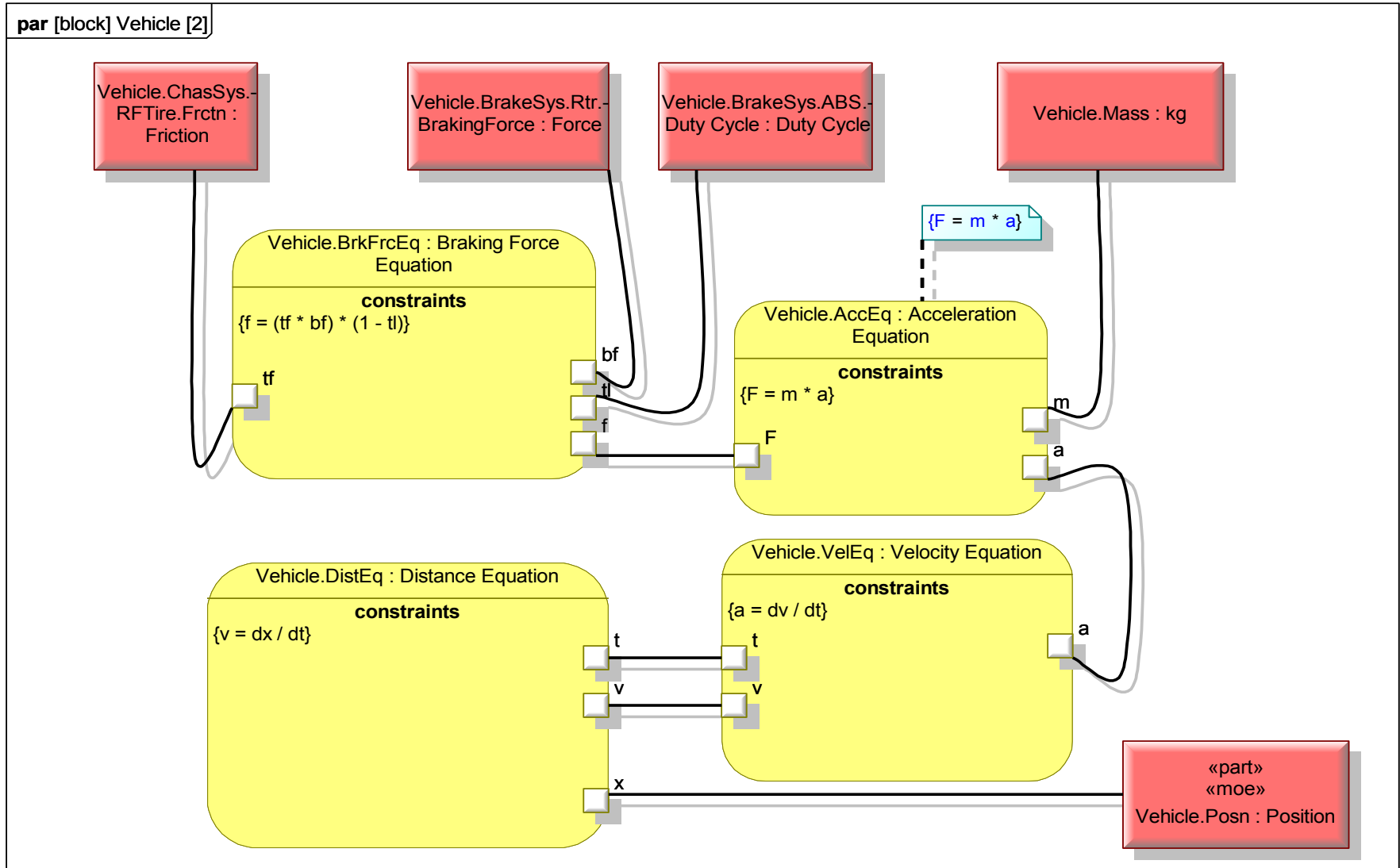
Parametrics

- Used to express constraints (equations) between value properties
 - Provides support to engineering analysis
 - e.g. performance, reliability, etc
- Constraint block captures equations
 - Expression language can be formal
 - e.g. MathML, OCL ...
 - or informal
 - Computational engine is defined by applicable analysis tool
 - and not by SysML
- Parametric diagram represents the usage of the constraints in an analysis context
 - Binding of constraint usage to value properties of blocks
 - e.g. vehicle mass bound to $F = m * a$

Vehicle Parametrics BDD



Parametrics – Straight Line Vehicle Dynamics

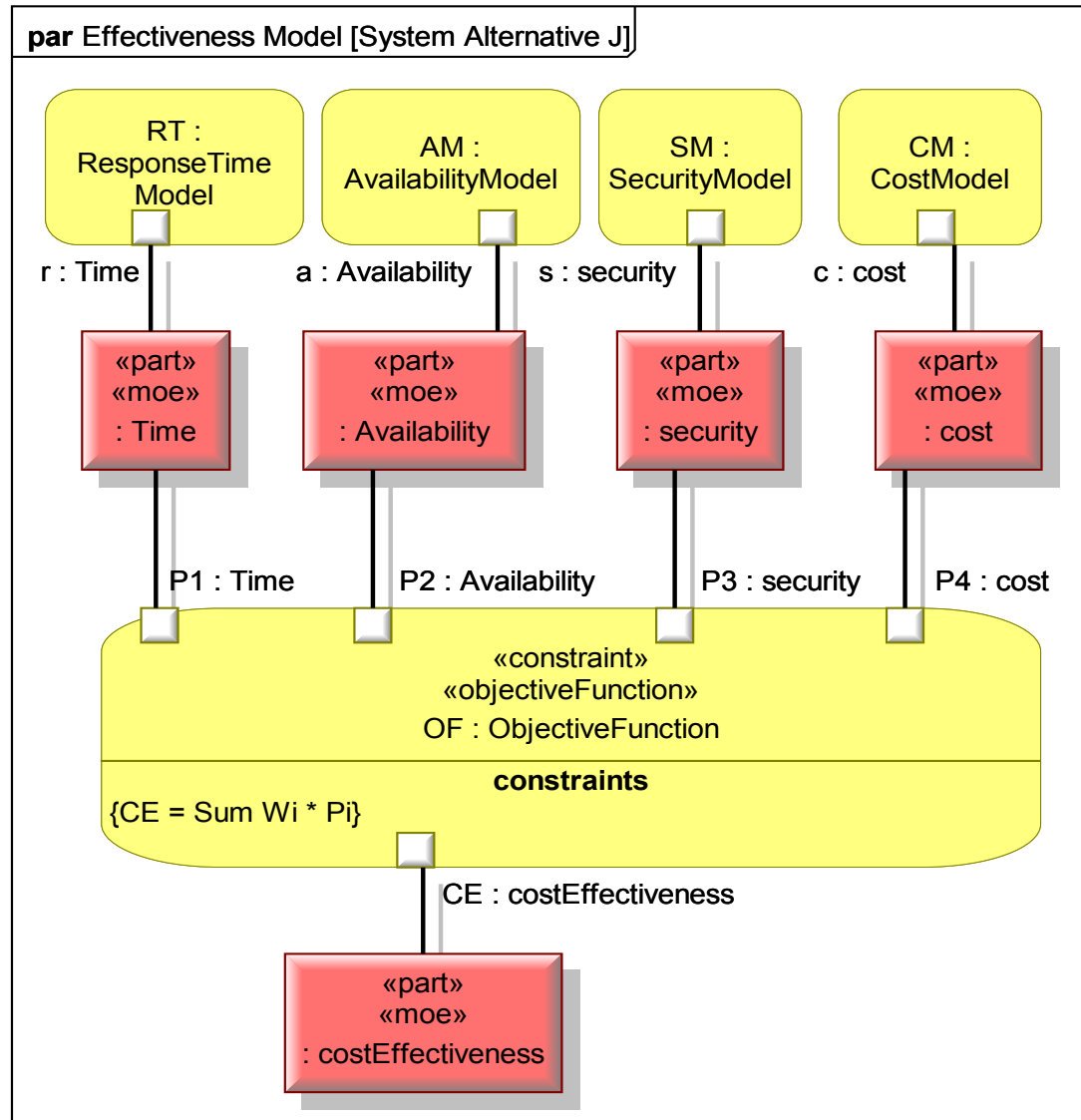


Measure of Effectiveness / Trade-Off Analysis

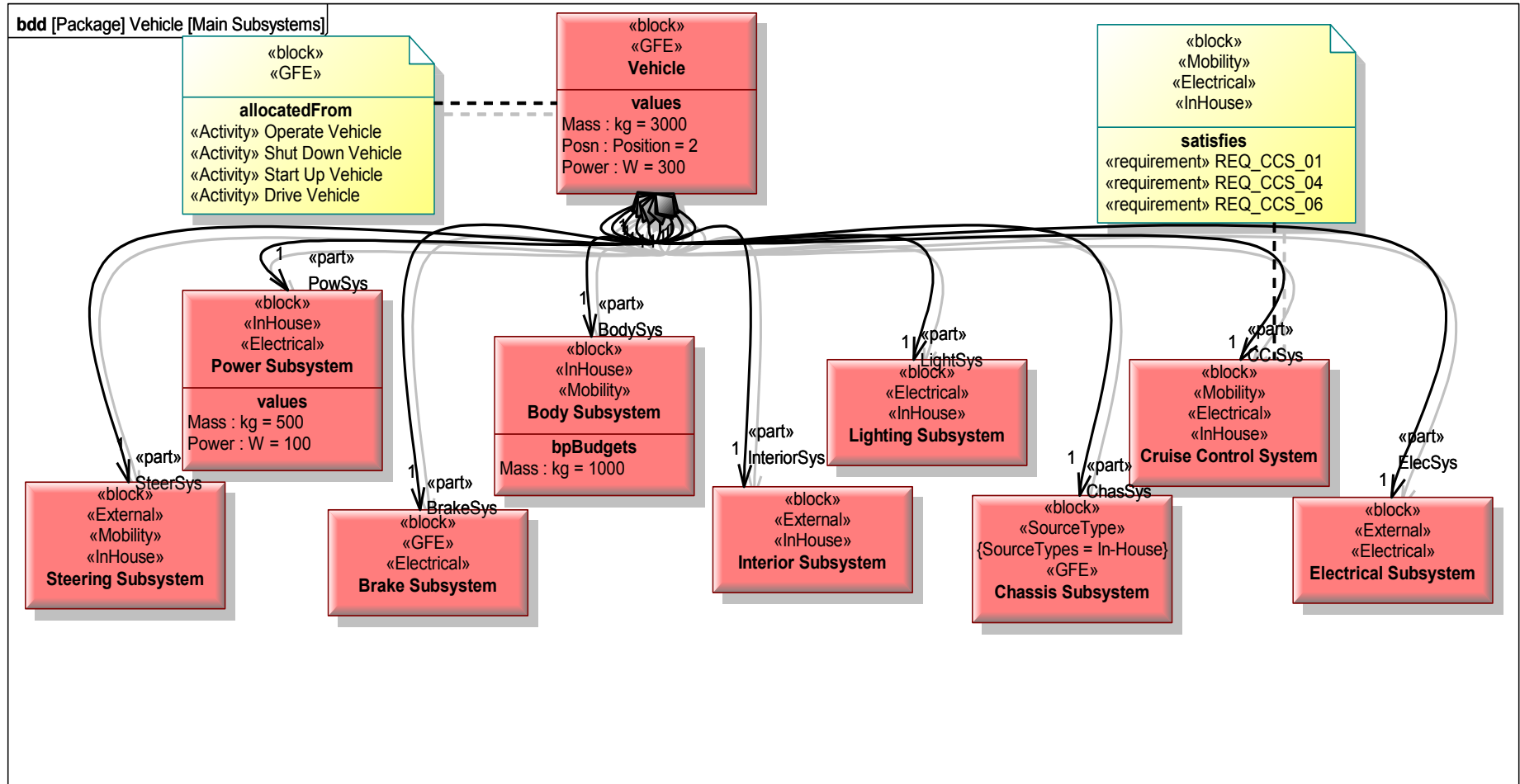
- A measure of effectiveness (moe) represents a parameter whose value is critical for achieving the desired mission requirements
 - Cost effectiveness
 - Performance
 - Communication
 - Etc.
- On the following slide, the overall cost effectiveness for each alternative is defined by an objective function that represents a weighted sum of their moe values.
 - For each moe, there is a separate parametric model to estimate the value of operational availability, mission response time, security effectiveness and life cycle cost to determine an overall cost effectiveness for each alternative. It is assumed that the moe's refer to the values for system alternative



Measure of Effectiveness / Trade-Off Analysis



System Budget Analysis



System Budget Analysis

Name	Quantity	Mass (in kg)					Power (in W)							
		Estimate	No Margin	Margin (%)	With Margin	Budget	Quantity On	Estimate	No Margin	Margin (%)	With Margin	Duty Cycle (%)	Average Power	Budget
Vehicle (CC Model::Analysis::Structure::Vehicle)	n/a	1000	2152	10.00%	2367.2	3000	1	300	379	10.00%	416.9	50.00%	208.45	300
Accelerator Pedal (CC Model::Analysis::Structure::Vehicle)	1	0	0	0.00%	0	0	0	0	0	0.00%	0	0.00%	0	0
Body Subsystem (CC Model::Analysis::Structure::Vehicle)	1	1000	1000	5.00%	1050	1000	0	0	0	0.00%	0	0.00%	0	0
Brake Subsystem (CC Model::Analysis::Structure::Vehicle::Brake)	1	0	0	0.00%	0	0	0	0	0	0.00%	0	0.00%	0	0
Anti-Locking Braking System (CC Model::Analysis::Structure::Vehicle::Brake)	1	0	0	0.00%	0	0	0	0	0	0.00%	0	0.00%	0	0
Brake Pedal (CC Model::Analysis::Structure::Vehicle::Brake)	1	0	0	0.00%	0	0	0	0	0	0.00%	0	0.00%	0	0
Brake Assembly Sensor (CC Model::Analysis::Structure::Vehicle::Brake)	1	0	0	0.00%	0	0	0	0	0	0.00%	0	0.00%	0	0
Rotor (CC Model::Analysis::Structure::Vehicle::Brake)	1	0	0	0.00%	0	0	0	0	0	0.00%	0	0.00%	0	0
Cruise Control System (CC Model::Analysis::Structure::Vehicle::CC System)	1	0	0	0.00%	0	0	0	0	0	0.00%	0	0.00%	0	0
Cruise Control IO (CC Model::Analysis::Structure::Vehicle::CC System)	1	0	0	0.00%	0	0	0	0	0	0.00%	0	0.00%	0	0
CC Display Panel (CC Model::Analysis::Structure::Vehicle::CC System)	1	0	0	0.00%	0	0	0	0	0	0.00%	0	0.00%	0	0
Dec Speed (CC Model::Analysis::Structure::Vehicle::CC System)	1	0	0	0.00%	0	0	0	0	0	0.00%	0	0.00%	0	0
Engage/Disengage CC (CC Model::Analysis::Structure::Vehicle::CC System)	1	0	0	0.00%	0	0	0	0	0	0.00%	0	0.00%	0	0
Inc Speed (CC Model::Analysis::Structure::Vehicle::CC System)	1	0	0	0.00%	0	0	0	0	0	0.00%	0	0.00%	0	0
Set Speed (CC Model::Analysis::Structure::Vehicle::CC System)	1	0	0	0.00%	0	0	0	0	0	0.00%	0	0.00%	0	0
Suspend/Resume CC (CC Model::Analysis::Structure::Vehicle::CC System)	1	0	0	0.00%	0	0	0	0	0	0.00%	0	0.00%	0	0
Cruise Control Unit (CC Model::Analysis::Structure::Vehicle::CC System)	1	0	0	0.00%	0	0	0	0	0	0.00%	0	0.00%	0	0
CAN (CC Model::Analysis::Structure::Vehicle::CC System)	1	0	0	0.00%	0	0	0	0	0	0.00%	0	0.00%	0	0
CC IO Card (CC Model::Analysis::Structure::Vehicle::CC System)	1	0	0	0.00%	0	0	0	0	0	0.00%	0	0.00%	0	0

Model Execution

- UML and SysML models can be executed.
 - Previously, execution semantics were not-standard
 - Often done by embedding code into state charts
 - Acceptable for code specific models and programmers, but not for systems engineers
- ‘foundational UML’ or fUML defines an execution semantic for both activity and state diagrams.
 - fUML specifies a language independent means of executing models
 - More ideal solution for systems engineers and system architects who may not be familiar with programming languages.
- Execution of the model against a pre-defined set of criteria can determine correct functionality, performance, timing, error handling and can help to validate use interfaces.
 - The extent to which they can do this effectively determines the level of quality of the model

Integrating Quality into the Process

- Imperative that a well defined process be specified elaborating how quality checks fit into the overall process
 - Suggested vs. mandatory, and
 - How updates, modifications, variations, dispensations, etc will be handled.
- Start with your existing process, figure out where you would like to be, and determine how you are going to arrive at your destination incrementally whilst ensuring that improvement can be measured.
- Object Oriented Systems Engineering Methodology (OOSEM).
 - A good starting point for defining a process or integrating these concepts into an existing process
 - Successfully adopted by several major companies.
 - More information is available at the OOSEM website <http://syseng.omg.org>.

QA and CMMI

■ Process and Product Quality Assurance (PPQA)

– Purpose

- The purpose of **Process and Product Quality Assurance** (PPQA) is to provide staff and management with objective insight into processes and associated work products.

– Specific Practices by Goal

- SG 1 Objectively Evaluate Processes and Work Products
 - SP 1.1 Objectively Evaluate Processes
 - SP 1.2 Objectively Evaluate Work Products and Services
- SG 2 Provide Objective Insight
 - SP 2.1 Communicate and Ensure Resolution of Noncompliance Issues
 - SP 2.2 Establish Records

Quality Assurance

- Tom Gilb describes Quality Assurance (QA) as “the generic name for any set of activities, which have as their primary or partial intent, or effect, to influence (‘assure’) the quality levels of a product or process.”
 - Assumes that modifying the process will affect the quality of the product that is being delivered.
 - For MBE projects it is essential that quality criteria for models be included in the process.
- Jim McCarthy states that “QA’s principal function is to continually assess the state of the product so that the rest of the team’s activities can be properly focused.”
 - Time-consuming
 - Potentially error-prone

Automated Quality Assurance

- QA checks and criteria need to be as automated, transparent, and painless as possible.
 - No C programmer should submit his or her code for review without having it go through Lint,
 - No system designer should submit a design for review without submitting it to a quality check.

For example, the following program, `Test1.c`, contains an error.

```
1 #include <string.h>
2 static void cpv(char *s, char* v, unsigned n)
3 { int i;
4   for (i=0; i<=n; i++)
5       *v++ = *s++;
6 }
7 void main(int argc, char* argv[])
8 {
9     if (argc != 0)
10        cpv(argv[0], argv, strlen(argv[0]));
11}
```

Using `lint` on `Test1.c` with the option:

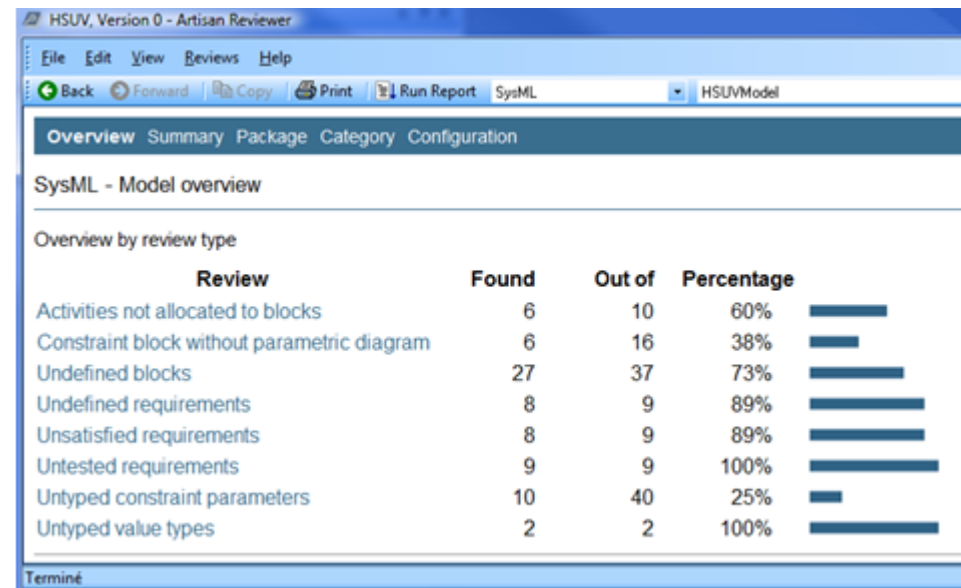
```
% lint -errfmt=src -Nlevel=2 Test1.c
```

Automated Quality Assurance

- To work effectively, QA needs to be automated as much as possible.
 - Manually checking all of the tests provided by Lint would take a very long time, be error-prone, and suffer from subjectivity.
 - Integrated Model-Based Quality Assurance requires integrated tools.

These should provide:

- Summary (often called dashboards) and detailed views
- Auto-correction of specific types of errors
- Configurable modelling standards, visualization of errors, and configurable and user-defined reviews to ensure that your model is complete, consistent, and correct.



Conclusions

- The purpose of a model is to answer one or more questions
 - Before you start, agree on the question
- Assessment criteria should be agreed prior to starting the model
- Determine the right level of quality before starting
- A well-documented process is essential for success
 - Documentation standards
 - Style standards
 - Etc.
- Automation of assessment checks will greatly improve productivity and ensure a consistent implementation
 - Without this level of automation, it will be difficult to enforce quality standards across an organisation.

Questions, Comments, Discussion

