System Analysis using SysML Parametrics: Current Tools and Best Practices

Chris Paredis
Model-Based Systems Engineering Center
Georgia Tech
Chris.paredis@me.gatech.edu

www.pslm.gatech.edu/courses
www.omg.org/ocsmp
Acknowledgments

Software Support
- No Magic, Inc
- InterCAX
- Phoenix Integration

Grad Students / Postdocs
- Aditya Shah
- Alek Kerzhner
- Axel Reichwein
- Ben Lee
- Kevin Davies
- Roxanne Moore
- Sebastian Herzig
- Wladimir Schamai

Collaborators
- Russell Peak \(^{(1)}\)
- Manas Bajaj \(^{(1)}\)
- Roger Burkhart
- Sandy Friedenthal
- Leon McGinnis

\(^{(1)}\) provided much of the materials on ParaMagic
The Systems Modeling Language (OMG SysML) is a general purpose modeling language that allows users to express a broad range of information and knowledge about systems and systems engineering projects: from requirements to functional architectures, and from behavior to structural specifications. An important set of modeling constructs in SysML are parametric constraints as used in parametric diagrams. SysML parametrics allow systems engineers not only to define parametric system specifications but also to express analysis models relating measures of effectiveness to system specifications. In this tutorial, the use of parametrics for analysis purposes will be reviewed in detail. To provide the appropriate context, we will start with a short overview of SysML, followed by a more detailed review of the syntax and semantics of parametric diagrams. We will then discuss different uses of parametrics, and provide best-practice examples, specifically focusing on analysis modeling. Parametric diagrams will also be compared with a new OMG specification for combining SysML and Modelica: The OMG SysML-Modelica Transformation Specification. A live demonstration will be included with the following two tools: The MagicDraw SysML Tool with the ParaMagic plug-in, and an open-source plug-in for solving parametric constraint models in ModelCenter.
Outline

- Short overview of SysML
- SysML Parametrics: Syntax and Semantics
- Best practices in Parametrics Modeling
  - Margin analysis – SimpleSat Satellite Example
  - Analysis context
  - Parameter aggregation
- Tools for solving Parametrics
  - ParaMagic
  - ModelCenter
- Parametrics versus SysML4Modelica
- Summary
Systems Engineering as Decision Making

- Our perspective:
  - Design involves deciding on the most preferred solution alternative
  - The most preferred alternative is the one that leads to the most preferred outcomes

We need a language to express this information and knowledge
What is SysML?

- The Systems Modeling Language (OMG SysML™) is a visual, general purpose modeling language

- **Is a** modeling language that provides
  - Semantics = meaning
  - Notation = representation of meaning

- **Is not a** methodology or a tool
  - SysML is methodology and tool independent

- Developed by the Object Management Group to support Model-Based Systems Engineering
What Can be Expressed in SysML?

- SysML is a language to express the information and knowledge generated and processed during the application of a systems development methodology

  - Specification
  - Analysis
  - Design
  - Verification
  - Validation

  - Hardware
  - Software
  - Data
  - Personnel
  - Procedures
  - Facilities
Think of SysML as an integrated collection of languages…

SysML Diagram Taxonomy

- **SysML Diagram**
  - **Behavior Diagram**
  - **Requirement Diagram**
  - **Structure Diagram**
    - **Activity Diagram**
    - **Sequence Diagram**
    - **State Machine Diagram**
    - **Use Case Diagram**
    - **Block Definition Diagram**
    - **Internal Block Diagram**
    - **Package Diagram**

- **Parametric Diagram**

Same as UML 2
Modified from UML 2
New diagram type
Some History…

  - v1.0: 2007-09
  - v1.1: 2008-11
  - v1.2: 2010-06
  - v2.x: RFI preparation workshop - 2008-12

- **Strong vendor support**
  - MagicDraw (No Magic), Artisan Studio (Atego), Enterprise Architect (Sparx Systems), Rhapsody (IBM),…

- **Good learning infrastructure**
  - Books, short courses, academic courses, INCOSE/OMG tutorial, public examples, etc.

- **OMG Certified Systems Modeling Professional**
Model-Based Systems Engineering in Industry

- Actively used in most large companies in Aerospace, Defense, Automotive:
  - In a recent SysML survey, 45 companies participated:
    - Space Systems: 23%
    - Aircraft: 20%
    - Defense: 20%
    - Automotive: 7%
    - Other: 30%

- No longer small pilot studies!

<table>
<thead>
<tr>
<th>Project Duration</th>
<th>1 mo – 1 year: 20%</th>
<th>1 year – 3 years: 35%</th>
<th>&gt; 3 years: 45%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Size</td>
<td>&lt; 10 people: 28%</td>
<td>10 – 100: 40%</td>
<td>100 – 1000: 22%</td>
</tr>
</tbody>
</table>

- MBSE is becoming part of day-to-day engineering practice

What is SysML?
(www.omgsysml.org)

1. Structure

2. Behavior

3. Requirements

4. Parametrics
What is SysML?
(www.omgsysml.org)

1. Structure

- Object Diagrams:
  - VehicleSystemSpecification
  - BrakingSubsystemSpecification

- Relationships:
  - Satisfies: id="102" text="The vehicle shall stop from 60 mph within 150 ft on a clean dry surface."
  - Allocate: objectFlow: Anti-LockController

2. Behavior

- Block Diagrams:
  - Anti-LockController
  - TractionDetector
  - BrakeModulator

- Dynamic Behavior:
  - BrakingSubsystemSpecification

3. Requirements

- Requirements Diagram:
  - Braking Requirements
  - Stopping Distance
  - Anti-Lock Performance

- Verify:
  - MinimumStoppingDistance

4. Parametrics

- Parametric Diagram:
  - Vehicle Dynamics
  - Equations:
    - Braking Force Equation
    - Distance Equation
    - Velocity Equation
Systems Engineering as Decision Making

- **Our perspective:**
  - Design involves deciding on the most preferred solution alternative
  - The most preferred alternative is the one that leads to the most preferred outcomes

```
Generate Alternatives | Evaluate Alternatives | Select Alternative
----------------------|-----------------------|----------------------
Formulation           | Analysis              | Interpretation
BDD, IBD, ACT, STM, SD, (PAR) → Specify the alternatives
PARAMETRIC, executable (ACT, STM, SD) (analyses external to SysML) → Predict outcomes
```
Escaping “Spreadsheet Sabotage”

“The use of spreadsheets to perform complex business functions exposes a business to a number risks…”

“A detailed study by Panko (2000) revealed that many companies are using highly error-prone spreadsheets to perform numerous functions. According to research over a long period of time, Panko has proven that spreadsheet errors are common and result in meaningful, harmful business impacts.”

“According to Panko (2005), real life audits found errors in 94 percent of the spreadsheets examined. In summary, it is fair to say that the use of spreadsheets to make business decisions is potentially dangerous.”

Discussion

- What are the advantages of representing Systems Engineering in a (formal) model?
Outline

- Short overview of SysML
- SysML Parametrics: Syntax and Semantics
- Best practices in Parametrics Modeling
  - Margin analysis – SimpleSat Satellite Example
  - Analysis context
  - Parameter aggregation
- Tools for solving Parametrics
  - ParaMagic
  - ModelCenter
- Parametrics versus SysML4Modelica
- Summary
Parametric Diagrams in SysML

- **Behavior Diagram**
- **Requirement Diagram**
- **Structure Diagram**

- **Activity Diagram**
- **Sequence Diagram**
- **State Machine Diagram**
- **Use Case Diagram**
- **Block Definition Diagram**
- **Internal Block Diagram**
- **Package Diagram**

- Same as UML 2
- Modified from UML 2
- New diagram type

Expressing Constraints in SysML

- Need to represent mathematical relationships or constraints in SysML

- Equation = mathematical constraint (or relationship) between parameters (or variables)

- Equation versus assignment:
  - What is the difference between:
    
    \[ F = m \cdot a \]
    
    \[ m \cdot a = F \]
    
    \[ F - m \cdot a = 0 \]
    
    \[ F / m = a \]

In SysML constraints are modeled *declaratively*:

- state what must be true
- not how to compute it
Defining a Constraint in SysML

- **Constraint Block**
  - Constraint (optional: specify language in which constraint is expressed)
  - Constraint parameters

- **Compare to Block:**
  - Block ~ Constraint Block
  - ValueProperty ~ Constraint Parameter
Using a Constraint Block: Constraint Property

- Constraint Property is owned by a Block
- Constraint Property is typed to a Constraint Block
Using a Constraint Block: Parametric Diagram

- Constraint Property constrains the value properties of the block that owns it.
- The value properties are *bound* to constraint parameters using binding connectors.
  - Meaning of binding connector: bound values are equal.
  - Be careful: only values that are type compatible can be bound.
How is a PAR different from an IBD?

PAR
- Connect constraint parameters / value properties
- Binding connectors
- Constraint properties

IBD
- Connect ports / flow ports / parts
- Connectors
- Part properties

---

2008-2011 Copyright © Georgia Tech. All Rights Reserved.
Nested Constraint Blocks

**bdd [Package] Parametric Example**

```
```

**constraint**
K

parameters
- a : Real
- b : Real
- c : Real
- d : Real
- K : Real

```

**eq1**
```
```

**constraint**
K1

constraints
{K1 = a*b}

parameters
- a : Real
- b : Real
- K1 : Real

```

**eq2**
```
```

**constraint**
K1*K2

constraints
{K = K1*K2}

parameters
- K1 : Real
- K2 : Real
- K : Real

```

**eq3**
```
```

**constraint**
K2

constraints
{K2 = c*d}

parameters
- c : Real
- d : Real
- K2 : Real

```
Nested Constraint Blocks

\[ \text{par} \ [\text{ConstraintBlock}] \ K \]

- eq1 : K1
  \{K1 = a\cdot b\}

- eq2 : K1\cdot K2
  \{K = K1\cdot K2\}

- eq3 : K2
  \{K2 = c\cdot d\}
### Review of PAR Graphical Notation

<table>
<thead>
<tr>
<th>Diagram Element</th>
<th>Notation</th>
<th>Description</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constraint Parameter</td>
<td><code>&lt;Name&gt;: </code>&lt;Type&gt;[&lt;Multiplicity&gt;]</td>
<td>A constraint parameter is a special kind of property that is used in the constraint expression of a constraint block. Constraint parameters do not have direction.</td>
<td>7.3</td>
</tr>
<tr>
<td>Parameter Node</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Constraint Property</td>
<td>`&lt;Name&gt;::&lt;ConstraintBlock&gt;[&lt;Multiplicity&gt;]</td>
<td>Constraint properties are defined by constraint blocks and used to bind (i.e., connect) parameters. This enables complex systems of equations to be composed from more primitive equations, and for the parameters of the equations to explicitly constrain properties of blocks.</td>
<td>7.4</td>
</tr>
<tr>
<td>Node</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Value Binding Path</td>
<td><code>&lt;Multiplicity&gt; &lt;Multiplicity&gt;</code></td>
<td>Binding connectors connect constraint parameters to each other and to value properties. They express an equality relationship between their bound elements.</td>
<td>7.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Outline

- Short overview of SysML
- SysML Parametrics: Syntax and Semantics
- Best practices in Parametrics Modeling
  - Margin analysis – SimpleSat Satellite Example
  - Analysis context
  - Parameter aggregation
- Tools for solving Parametrics
  - ParaMagic
  - ModelCenter
- Parametrics versus SysML4Modelica
- Summary
SimpleSat Parametrics Example

bdd depicting model structure
SimpleSat Parametrics Example

req diagram showing requirements verification pattern

- The requirements here are shown as being verified by a margin block in the satellite and power system blocks. After executing the corresponding parametrics model, if each margin > 0, then the system meets the requirements.
- It may be better to use an explicit test case (as recommended per the SysML spec), but the less complicated approach employed here may be sufficient in many cases.
SimpleSat Parametrics Example
par structure of building blocks and subsystems

concept (generic): (a) expanded/flattened view

(b) encapsulated view

<comment>
MarginBlock overview (aka margin of safety)
margin >= 0 means it passes.
margin value * 100% = what percentage of the determined value is left before you reach failure.
i.e., margin = (alw - det) / det

Example:
alw = 200
det = 150
margin = 0.333
(i.e., you can increase det by 33% = 50 before you reach margin = 0 = failure).

This form assumes you want determined < allowable, (i.e., allowable is a max Allowable). It could be generalized to handle min Allowable case also.

See also:
SimpleSat Parametrics Example
SysML parametrics diagram defining margin analysis

---

**Diagram Notes:**
- **r1:** Mass Balance
  \[ m = m_1 + m_2 + m_3 + m_4 \]
  - \( m_1, m_2, m_3, m_4 \)
- **r2:** Power Balance
  \[ p = p_1 + p_2 + p_3 \]
  - \( p_1, p_2, p_3 \)
- **r3:** CtrlPwrEqn
  \[ pwrctrl = 0.2 \times m_{\text{mass}} \]
  - \( m_{\text{mass}}, pwrctrl \)

**Rationale:**
This captures a rule-of-thumb knowledge regarding the relationship between total mass and controller power. It represents that knowledge as an equation.

---

2008-2011 Copyright © Georgia Tech. All Rights Reserved.
SimpleSat Parametrics Example
SysML par view and ParaMagic tool for execution
SimpleSat Parametrics Example
Analysis Results: a Satellite Instance

---

```
controllerSubSys = controller1
instruments = instruments1
mass = ""{unit = Kilogram}
powerSubSys = power1
propulsionSubSys = propulsion1
reqVerifierMass = margin1
```

```plaintext
propulsion1 : PropulsionSystem
mass = "5000.0"{unit = Kilogram}
power = "5000.0"{unit = Watt}
```

```plaintext
power1 : PowerSystem
mass = "1500.0"{unit = Kilogram}
power = ""{unit = Watt}
reqVerifierPower = margin2
```

```plaintext
controller1 : ControlSystem
mass = "1400.0"{unit = Kilogram}
power = ""{unit = Watt}
powerRating = "1800.0"{unit = Watt}
reqVerifierRating = margin3
```

```plaintext
margin1 : MarginBlock
allowable = "10000.0"
determined = ""
margin = ""
```

```plaintext
margin2 : MarginBlock
allowable = "10000.0"
determined = ""
margin = ""
```

```plaintext
margin3 : MarginBlock
allowable = ""
determined = ""
margin = ""
```

---

This margin block is like an implied requirement which states that the power used by the controller should not exceed its rating.
SimpleSat Parametrics Example
Analysis Results: a Satellite Instance

- **propulsion1**: Propulsion System
  - mass = "5000.0" (unit = Kilogram)
  - power = "5000.0" (unit = Watt)

- **power1**: Power System
  - mass = "1500.0" (unit = Kilogram)
  - power = "8980.0" (unit = Watt)

- **instruments1**: Instruments
  - mass = "2000.0" (unit = Kilogram)
  - power = "2000.0" (unit = Watt)

- **margin1**: Margin Block
  - allowable = "10000.0"
  - determined = "9900.0"
  - margin = "0.01010101010102"

- **controller1**: Control System
  - mass = "1400.0" (unit = Kilogram)
  - power = "1980.0" (unit = Watt)
  - powerRating = "1800.0" (unit = Watt)
  - reqVerifierRating = margin3

- **margin3**: Margin Block
  - allowable = "1800.0"
  - determined = "1980.0"
  - margin = "-0.0909090909090909"

- **ModelOverview - Exercise 1**

- **requirement**: PowerReq
  - Id = "2.0.1"
  - Text = "The system shall support a peak power demand up to 10,000 watts."

This margin block is like an implied requirement which states that the power used by the controller should not exceed its rating.
Outline

- Short overview of SysML
- SysML Parametrics: Syntax and Semantics
- Best practices in Parametrics Modeling
  - Margin analysis – SimpleSat Satellite Example
  - Analysis context
  - Parameter aggregation
- Tools for solving Parametrics
  - ParaMagic
  - ModelCenter
- Parametrics versus SysML4Modelica
- Summary
Two Different Uses of Parametrics:

1. Parametric Specification

- A constraint is used to define a parametric relationship between value properties of a system alternative

- Constraint Property is owned by the Specified Block
Two Different Uses of Parametrics:

2. Parametric Analysis

- A constraint is used to analyze an alternative
- Constraint Property is owned by an Analysis Context
Two Different Uses of Parametrics: 2. Parametric Analysis

- A constraint is used to analyze an alternative
- Different analyses are possible for the same system
- Constraint Property is owned by Analysis Context
Why Use an Analysis Context?

- Many different analyses for a single system
  - Different views
  - Different levels of abstraction
- Analysis models often reused from libraries

One Context for each view / abstraction
Outline

- Short overview of SysML
- SysML Parametrics: Syntax and Semantics
- Best practices in Parametrics Modeling
  - Margin analysis – SimpleSat Satellite Example
  - Analysis context
  - Parameter aggregation
- Tools for solving Parametrics
  - ParaMagic
  - ModelCenter
- Parametrics versus SysML4Modelica
- Summary
Complex Aggregates

\[ \text{mass} = \sum_{i=1}^{n} \text{mass}_i \]

\[ \text{cost} = \sum_{i=1}^{n} \text{cost}_i \]
Complex Aggregates

\[ n = 10 \]

\[ \text{mass} = \sum_{i=1}^{n} \text{mass}_i \]

\[ \text{cost} = \sum_{i=1}^{n} \text{cost}_i \]
Outline

- Short overview of SysML
- SysML Parametrics: Syntax and Semantics
- Best practices in Parametrics Modeling
  - Margin analysis – SimpleSat Satellite Example
  - Analysis context
  - Parameter aggregation

Tools for solving Parametrics
- ParaMagic
- ModelCenter

Parametrics versus SysML4Modelica

Summary
Enabling Executable SysML Parametrics
Commercialization by InterCAX LLC in Georgia Tech VentureLab incubator program

Advanced technology for graph management and solver access via web services.

Plugins Prototyped by Git (to SysML vendor tools)
1) Artisan Studio [2/06]
2) EmbeddedPlus [3/07]
3) NoMagic [12/07]
Spring System Example

\[ r_1 : L = x_2 - x_1 \]
\[ r_2 : \Delta L = L - L_0 \]
\[ r_3 : F = k\Delta L \]

(a) Analytical springs tutorial block definition diagram.

(b) LinearSpring parametric diagram.

(c) TwoSpringSystem parametric diagram.

Reference:
http://eislab.gatech.edu/pubs/conferences/2007-incose-is-1-peak-primer/
Example Instance: Two_Spring_System

(a) Lexical COB instance as XML (CXI)

```
<linear_spring loid="_15">
  <undeformed_length causality="given">8.0</undeformed_length>
  <spring_constant causality="given">5.5</spring_constant>
</linear_spring>
<linear_spring loid="_25">
  <undeformed_length causality="given">8.0</undeformed_length>
  <spring_constant causality="given">6.0</spring_constant>
</linear_spring>
<two_spring_system loid="_3">
  <spring1 ref="_15"/>
  <spring2 ref="_25"/>
  <deformation1 causality="target"/>
  <deformation2 causality="target"/>
  <load causality="given">10.0</load>
</two_spring_system>
```

(b) Parametrics execution in XaiTools / ParaMagic

Example 2, state 1.0 (unsolved)

Example 2, state 1.1 (solved)
ParaMagic Core Solver: Mathematica
Mathematica Job — SpringSystems

(a) Input script
(auto-generated from ParaMagic)
example 2, state 1.0 (unsolved)

... solutions = Solve[ 
  q16==k10,
  q16==o14*5.5,
  o14==n13,
  i8==j9-h7,
  10==k10,
  p15==g6-0,
  l11==m12+n13,
  g6==h7,
  k10==m12*6,
  m12==i8-8,
  o14==p15-8
} ];

WriteString[ output,
  ToString[ CForm [N [ solutions ] ] ] ];
Close[output];
Exit[];

(b) Output script (results)
(auto-imported back into ParaMagic)
example 2, state 1.1 (solved)

List (List( 
  ... 
  Rule(g6,9.8181818181818181818181818181818),
  Rule(h7,9.8181818181818181818181818181818),
  Rule(i8,9.6666666666666666666666666666666),
  Rule(j9,19.4848484848484848484848484848484),
  Rule(k10,10.),
  Rule(m12,1.6666666666666666666666666666666),
  Rule(l11,3.4848484848484848484848484848484),
  Rule(n13,1.81818181818181818181818181818183),
  Rule(o14,1.81818181818181818181818181818183),
  Rule(p15,9.8181818181818181818181818181818),
  Rule(q16,10.))) ... 
))

Note: ParaMagic supports either of these as a core solver (in production releases): Mathematica and OpenModelica. Support for Matlab Symbolic Math Toolbox (SMT) as a core solver is WIP.
ParaMagic Core Solver: OpenModelica
OpenModelica Job — SpringSystems

(a) Input script
(auto-generated from ParaMagic)

example 2, state 1.0 (unsolved)

class SpringSystems991034
    Real e4;    Real i8;
    Real l11;   Real a0;
    Real k10;   Real m12;
    Real b1;    Real d3;
    Real p15;   Real f5;
    Real o14;

    equation
        10.0=l11;
        p15=m12-8.0;
        l11=p15*6.0;
        i8=f5-8.0;
        b1=p15+a0;
        m12=k10-o14;
        f5=d3-0.0;
        i8=a0;
        e4=l11;
        e4=i8*5.5;
        d3=o14;

end SpringSystems991034;

(b) Output script (results)
(auto-imported back into ParaMagic)

example 2, state 1.1 (solved)

...    
DataSet: a0
0, 1.81818181818182
DataSet: k10
0, 19.48484848484849
DataSet: m12
0, 9.66666666666667
DataSet: b1
0, 3.48484848484849
DataSet: p15
0, 1.66666666666667
DataSet: o14
0, 9.81818181818182
DataSet: e4
0, 10
...

Note: ParaMagic supports either of these as a core solver (in production releases): Mathematica and OpenModelica. Support for Matlab Symbolic Math Toolbox (SMT) as a core solver is WIP.
ParaMagic Capabilities

InterCAX

Modeling & Simulation Technology

Products

SysML Parametric Solvers
- Execute SysML parametric models using
  - ParaMagic® for MagicDraw
  - Melody™ for Rational Rhapsody
  - ParaSolver™ for Artisan Studio
- Use Mathematica or OpenModelica (free) as a core solver
- Link to Excel spreadsheets and perform batch trade studies
- Link to existing MATLAB/Simulink and Mathematica functions

SysML Integration Tools
- SysML and Math tools (Mathematica, MATLAB, Excel)
- SysML and M&S tools (Simulink, STK, OpenModelica)
- SysML and CAD tools (NX, STEP MCAD/AP203, ECAD/AP210)
- SysML and CAE tools (ANSYS, ABAQUS)
- Plus tailored interfaces (Teamcenter/PLM, in-house tools)
Outline

- Short overview of SysML
- SysML Parametrics: Syntax and Semantics
- Best practices in Parametrics Modeling
  - Margin analysis – SimpleSat Satellite Example
  - Analysis context
  - Parameter aggregation
- Tools for solving Parametrics
  - ParaMagic
  - ModelCenter
- Parametrics versus SysML4Modelica
- Summary
What is ModelCenter?

- Process Integration and Design Optimization (PIDO) framework
- Integration with most Engineering Analysis tools
- Supports grid computing

Objectives:
- Maximize Efficiency
- Minimize Cost

\[ \text{Modelica} \rightarrow \text{Simulation} \]

\[ \text{optimizer} \leftarrow \text{Latin Hypercube + Response Surface} \]
ModelCenter Model as Constraint Block

- Constraint block stereotyped to «ModelCenter»
- ModelCenter variables = SysML constraint parameters
- MagicDraw plug-in supports import-export
- Additional access to optimization, DOE, UQ, etc.
Nested Constraints in ModelCenter

```
par [ModelCenter][EngineAnalysis][EngineAnalysis]
torque : Real

feasibilityScript : FeasibilityModel
{feasibility = 1
if torque > maxTorque then
feasibility = 0
end if}

analysis : EngineAnalysis
id : Real
omega : Real
maxTorque : Real

engineDB : EngineDB
a0
a1
a2

torqueScript : TorqueModel
{torque = a0+a1*omega+a2*omega^2}
```

2008-2011 Copyright © Georgia Tech. All Rights Reserved.
ModelCenter Advantage / Disadvantages

◆ Pro
  – One simple and convenient interface from MagicDraw to/from ModelCenter
  – Access to all the tools supported by ModelCenter
    ◆ Math: Matlab, Excel,
    ◆ Analysis: many CAE and CFD tools, Simulation tools,
    ◆ Optimization: Dakotah, Design Explorer, …
    ◆ CAD tools: CATIA, Pro/ENGINEER, NX
    ◆ Specialized tools: STK, SEER, easy custom wrappers
  – Scalable: model management, grid computing, etc.

◆ Con
  – More expensive
  – Only causal models
Outline

- Short overview of SysML
- SysML Parametrics: Syntax and Semantics
- Best practices in Parametrics Modeling
  - Margin analysis – SimpleSat Satellite Example
  - Analysis context
  - Parameter aggregation
- Tools for solving Parametrics
  - ParaMagic
  - ModelCenter
- Parametrics versus SysML4Modelica
- Summary
Modeling System-Level Trade-offs

- Must consider ALL objectives and requirements ... not just functional requirements
Modeling System-Level Trade-offs

- Must consider ALL objectives and requirements... not just functional requirements
Modeling the Physics of Systems — Differential Algebraic Equations in Modelica
OMG SysML-Modelica Transformation Specification
OMG SysML-Modelica Transformation Specification
SysML Parametrics Discussion ...

- Advantages — disadvantages?
- Compared with Excel?
- Compared with Modelica?
Summary

- Short overview of SysML
- SysML Parametrics: Syntax and Semantics
- Best practices in Parametrics Modeling
  - Margin analysis – SimpleSat Satellite Example
  - Analysis context
  - Model reuse
- Parameter aggregation
- Tools for solving Parametrics
  - ParaMagic
  - ModelCenter
- Parametrics versus SysML4Modelica
- Summary