# Model-based Systems Engineering
# MBSE 101

Moderated by
Elyse Fosse,
Jet Propulsion Laboratory,
California Institute of Technology

# Agenda

- Objective
- Acknowledgements
- Project Application Overview
- FAQ list
- FAQ answers
- References

Audience participation encouraged throughout

# Objective

- Provide answers to frequently asked MBSE questions
  - Answers are from a Jet Propulsion Laboratory (JPL)-centric view
  - Answers are software tool agnostic
- Will use a list of FAQs as a starting point
  - A real project application will be used to re-enforce responses throughout the presentation
- Audience provided questions/interactions are encouraged

# Acknowledgements

The FAQs and answers included have been collected from the presentations and experiences of:

- Todd Bayer
- Daniel Dvorak
- Sanford Friedenthal*
- Steve Jenkins
- Chi Lin
- Ann Devereaux
- Sanda Mandutianu
- Louise Anderson
- Elyse Fosse
- Bjorn Cole
- Chris Delp
- Robert Castillo
- Corey Harmon

*JPL, Lockheed

# Mars2020 MBSE Initiative Overview

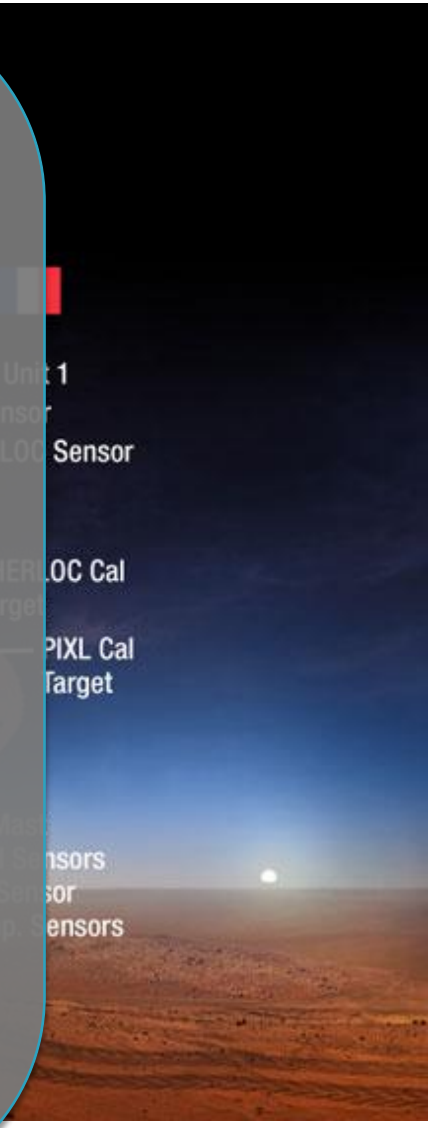
How Mars Science Laboratory evolved to Mars2020 and the motivation for infusing MBSE

# MSL Overview

- Mars Science Laboratory (MSL), Curiosity
- Landed August 6, 2012
- Pursues evidence of a past environment well-suited for microbial life
- Met it’s major science objective within a year of landing
- Still in operations

# Mars2020 Overview

- Inherits a majority of MSL's Flight System design
  - Cruise, Entry, Descent, and Landing systems are build to print
  - The Rover is a mixture of new and heritage designs
    - New **science instruments** seeking signs of past life on Mars
    - New **sample caching system** collecting and storing soil and rock samples for possible future return to Earth
    - New **technology demonstrations** benefiting future robotic and notional human exploration of Mars
    - Heritage hardware procured early in the project lifecycle
    - Heavy reuse of flight software
- Risk Areas
  - Heritage constraints
  - New arm and caching system
    - Cache compatibility with a potential future sample return mission
  - Payload scope creep
  - Parts/personnel obsolescence



# M2020 Flight System SE Goals

- Better awareness of the technical baseline
  - Capture the MSL as-flown design accurately
  - Move from “design by boxology” to an **integrated information technical baseline**
- Improve the communication, understanding, and visibility of the design
  - Provide an **authoritative source of information** that is easily accessible by the entire team
  - Create a **common language** to describe the system
  - Avoid information silos
  - Increase visibility and traceability of changes
- Focus on heritage design deviations
  - Rover first, then incorporate other stages
- Let FSSE products drive model implementation
  - Only model what is needed to create FSSE products
  - Add value at each increment

# M2020 Flight System SE Approach

**Model-Generated Visualizations**

- Prevents artifact discontinuity
- Aligns design artifacts
- Reduces information silos
- Increases visibility of design

**Model-Generated Documents**

- Prevents artifact discontinuity
- Aligns design artifacts with SE design work
- Reduces information silos

**FS Design Model**

- Authoritative source of info
- Technical baseline
- Reduces information silos by forcing integration

**Modeling Framework**

- Common language
- Consistency

# MBSE 101 FAQ

# FAQ

1. What is MBSE?
2. What SE problems does MBSE address?
3. How does MBSE compare to traditional SE?
4. What are the typical purposes of modeling?
5. What are the different types of models?
6. What is SysML
7. What is a System Model?
8. How can models help an SE effort?
9. What does MBSE mean for projects?
10. How has JPL infused MBSE?
11. How good is a model?
12. What is an ontology?
13. Why are ontologies relevant?

# FAQ 1

What is MBSE?

# MBSE Definition

| Final Report, Model-Based Engineering Subcommittee, NDIA, Feb. 2011 | INCOSE SE Vision 2020 (INCOSE-TP-2004-004-02, Sep 2007) |
| --- | --- |
| "Model-Based Engineering (MBE): An approach to engineering that ***uses models as an integral part of the technical baseline*** that includes the requirements, analysis, design, implementation, and verification of a capability, system, and/or product throughout the acquisition life cycle." | "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 throughout development and later life cycle phases." |

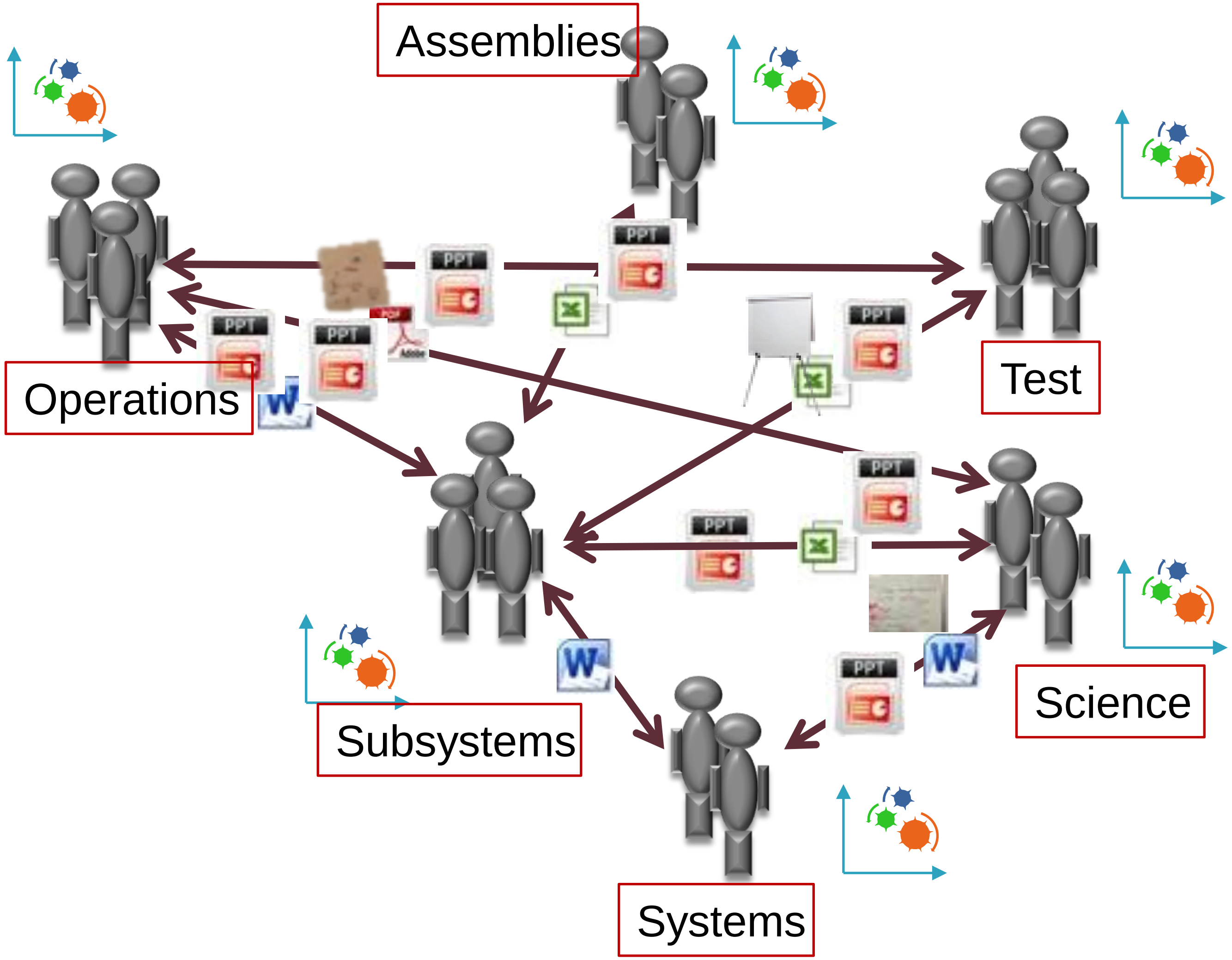
# MBSE Motivation

- Systems Engineering requires structural, behavioral, physics and simulation-based models representing the technical designs which evolve throughout the life-cycle, supporting trade studies, design verification and system V&V.

- Current practice tends to rely on standalone (discipline-specific) models whose characteristics are shared primarily through static documents.

- MBSE moves toward a shared system model with remaining discipline-specific models providing their characteristic information in a mathematically rigorous format. All disciplines “view” a consistent system model

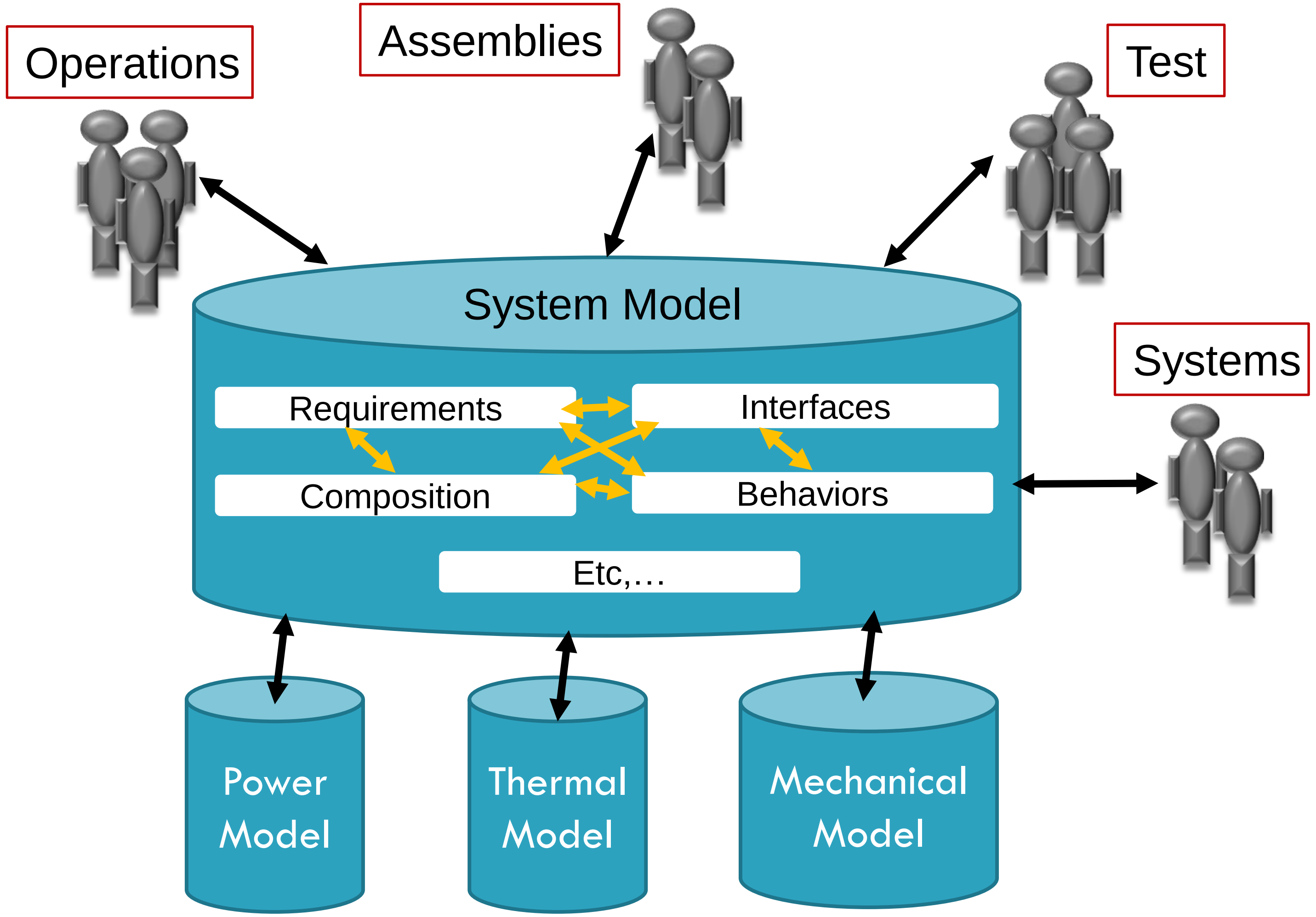
# Traditional SE Practice




Stand alone domain models/designs related via:

- Institutional life cycle **documents**
  - Operations Concepts
  - Requirements Documents
  - Interface Documents
  - Deployment Plans
- Formal review **presentations**
- **Informal communications**
  - White boards
  - Design Team Meeting Presentations
  - Email
  - Chat
  - Napkin

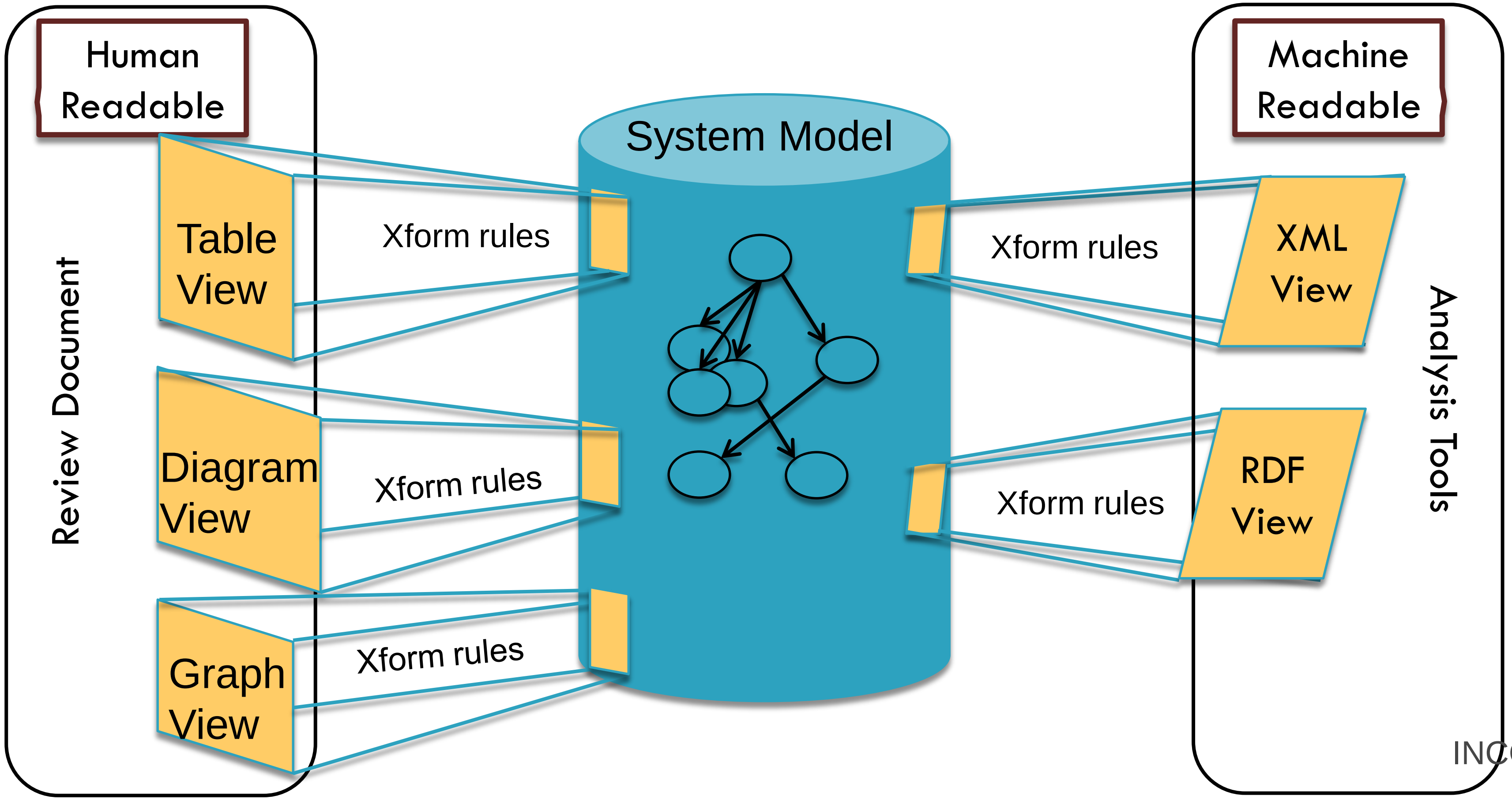
# Future MBSE Practice




**Integrated system model** with multiple views, connected to discipline models

- **Authoritative source** of information
- Exchanges information to/from analysis and stakeholders via **projections of model information**
- Information accessible to all members of the project

# System Model Projections

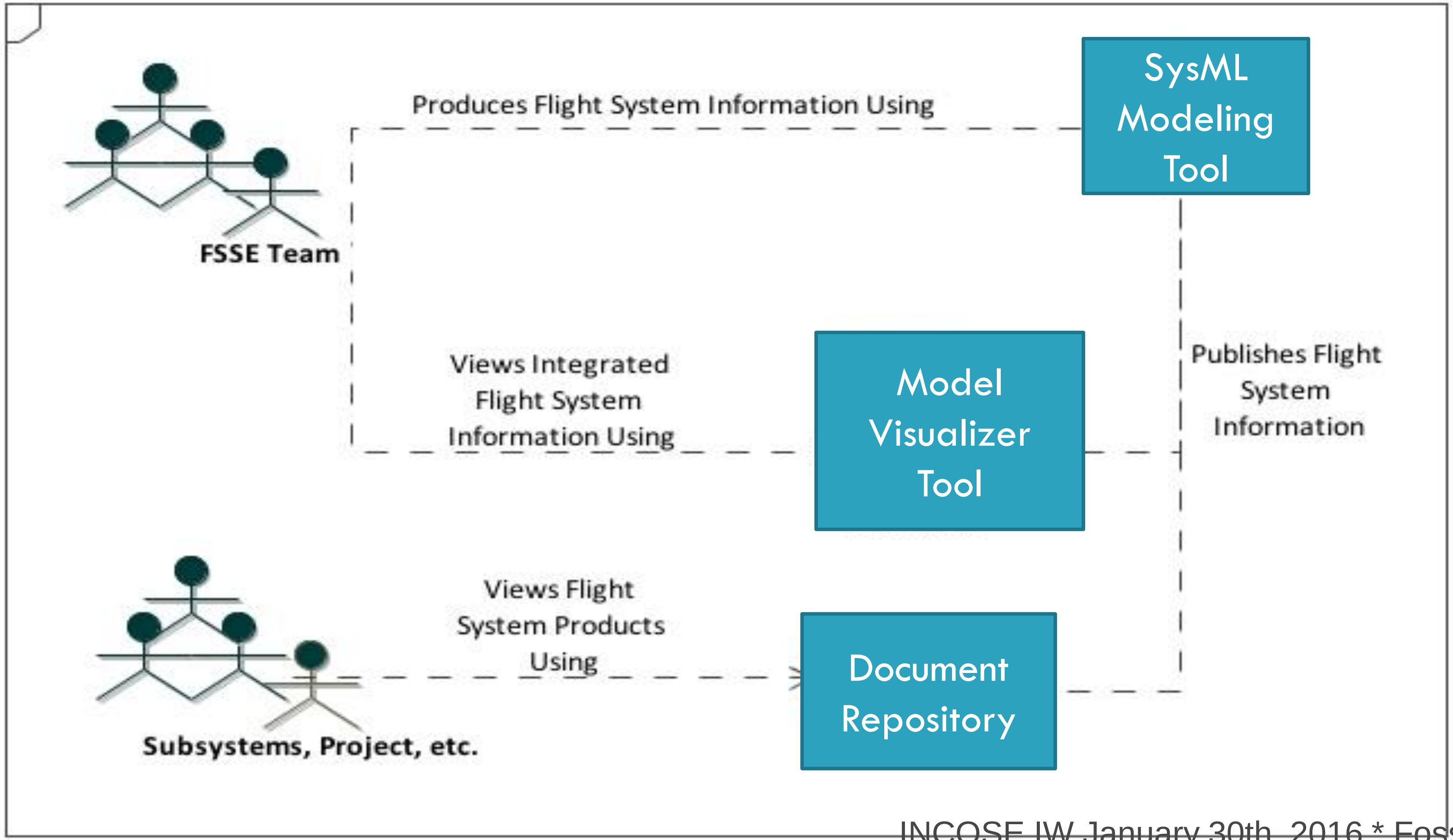



- Ability to **interrogate the the design information** and extract data into the format necessary for the given task
- Leverages **formalism**
- Transformation rules are **reusable**
- Provides machine and human readable formats

# M2020: Accessing Model Information

SysML model is the authoritative source

Source can be accessed via web-based visualizers and document repositories



# FAQ 2

What SE Problems does MBSE address?

# Industry-Identified Problems in SE

- Poor integration of models across the life cycle*
  - …hard to get coherent, checkable model of the whole *system* (at some level of abstraction)
- Limited reuse of models between programs*
  - …paying for similar engineering work over and over
- Variation in modeling maturity and integration across Engineering Disciplines*
  - …Mechanical/Electrical CAD/CAE fairly *mature*
  - …Systems/Software/Test fairly *immature*

* NDIA Final Report, Model Based Engineering Subcommittee

# INCOSE SE Vision: SE Challenges

FIVE SYSTEMS ENGINEERING CHALLENGES

*Adapted from Todd Bayer, Jet Propulsion Laboratory*

1 | Mission complexity is growing faster than our ability to manage it . . . increasing mission risk from inadequate specifications and incomplete verification.

2 | System design emerges from pieces, rather than from architecture . . . resulting in systems that are brittle, difficult to test, and complex and expensive to operate.

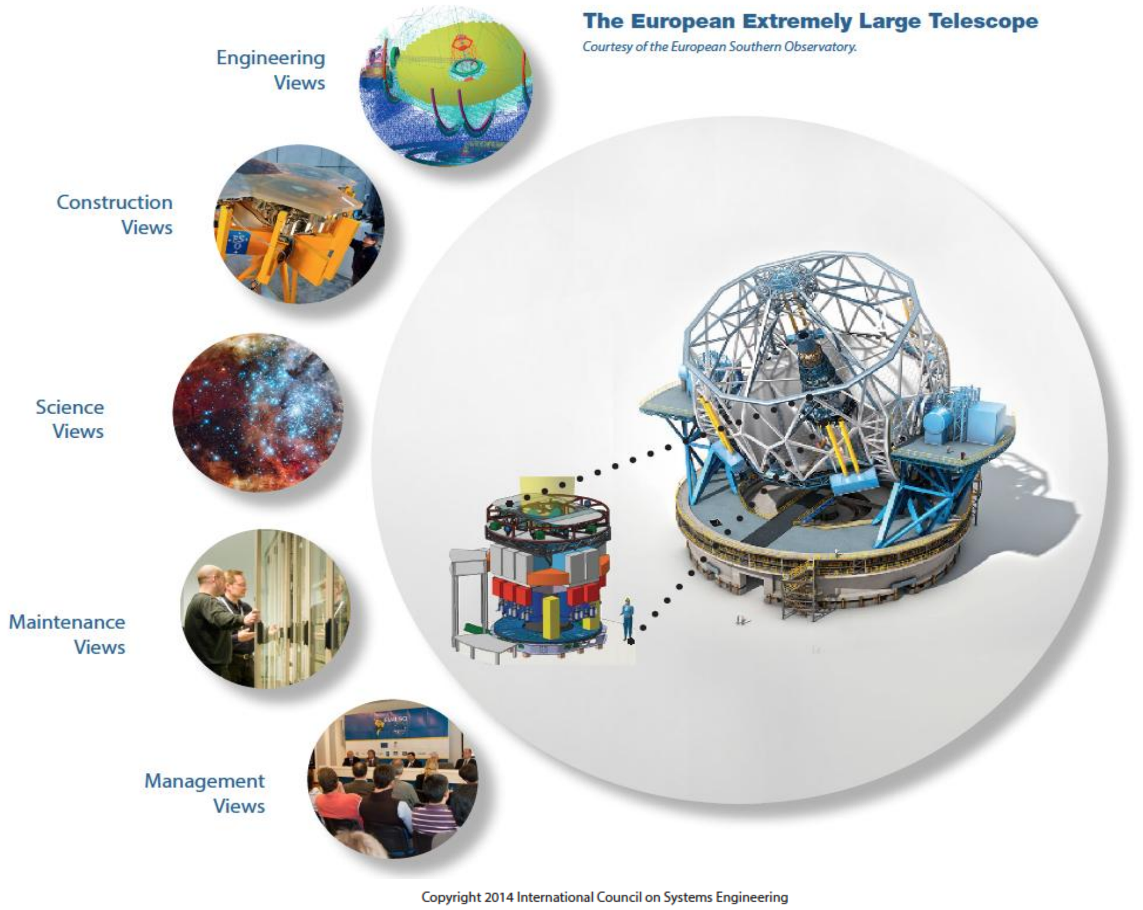
3 | Knowledge and investment are lost at project life cycle phase boundaries . . . increasing development cost and risk of late discovery of design problems

4 | Knowledge and investment are lost between projects . . . increasing cost and risk: dampening the potential for true product lines.

5 | Technical and programmatic sides of projects are poorly coupled . . . hampering effective project risk-based decision making.

6 | Most major disasters such as Challenger and Columbia have resulted from failure to recognize and deal with risks. The Columbia Accident Investigation Board determined that the preferred approach is an "independent technical authority".

# How MBSE Addresses Problems




A single, consistent, unambiguous system representation ensures integrity and traceability throughout the SE process

Provides the ability to codify institutional knowledge using formal methods, allowing for reuse and broad exposure

Captures information in a durable, evolvable format

Focusing on information integration rather than document generation allows for decimation of artifact inconsistency/staleness

# MSL to M2020: SE Challenges

**Technical baseline by "boxology"**

*Increased risk of late discovery of design problems*

**Trades/concurrent engineering**

*Varied design maturity made it hard to understand full of work scope*

**Information silos**

*S/C complexity resulted in mutually inconsistent, disparate sources of information*

**SE product alignment**

*Work had to slow to prepare for management/gate products*

**Personnel/artifact discontinuity**

*Increased risk due to knowledge and investment lost at launch slip*

**SE product relevance**

*Struggled to keep design documents reflective of as-built s/c*

# M2020 FS MBSE Goals

- Better awareness of the technical baseline
  - Capture the MSL as-flown design accurately
  - Move from “design by boxology” to an **integrated information technical baseline**
- Improve the communication, understanding, and visibility of the design
  - Provide an **authoritative source of information** that is easily accessible by the entire team
  - Create a **common language** to describe the system
  - Avoid information silos
  - Increase visibility and traceability of changes
- Focus on heritage design deviations
  - Rover first, then incorporate other stages
- Let FSSE products drive model implementation
  - Only model what is needed to create FSSE products
  - Add value at each increment

# M2020 MBSE Solution

**Model-Generated Visualizations**

- Prevents artifact discontinuity
- Aligns design artifacts

**Model-Generated Documents**

- Prevents artifact discontinuity
- Aligns design artifacts with SE design work
- Reduces information silos

**FS Design Model**

- Authoritative source of info
- Technical baseline
- Reduces information silos by forcing integration

**Modeling Framework**

- Common language
- Consistency

# FAQ 10

How does MBSE Compare to Traditional SE?

# A Historical Perspective on SE and MBSE

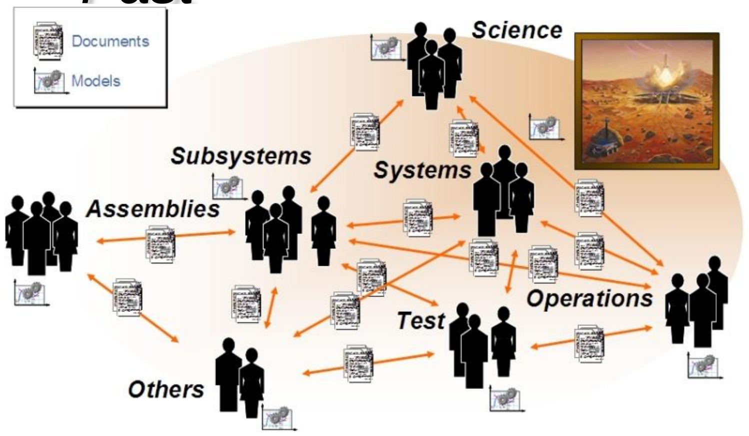
- Way back when, systems and subsystems were on equal footing
- Over time, computers allowed pretty much all the domains - except systems - to build rigorous modeling capabilities.
  - Systems is the last to get this rigor because it’s the broadest, most conceptual, and has by definition the full complexity of the system to deal with.  So it’s the hardest one to get rigorous about.
- SE has been at a disadvantage because of this
  - Lacking this quantitative rigor, SE has had to rely too heavily on intuition, overemphasizing the Art and neglecting the Science
- Systems needs to claim the rigor of its domain to restore balance
  - Otherwise our systems will continue to be assemblages of components whose performance and behavior must be discovered after assembly.
- MBSE provides this rigor

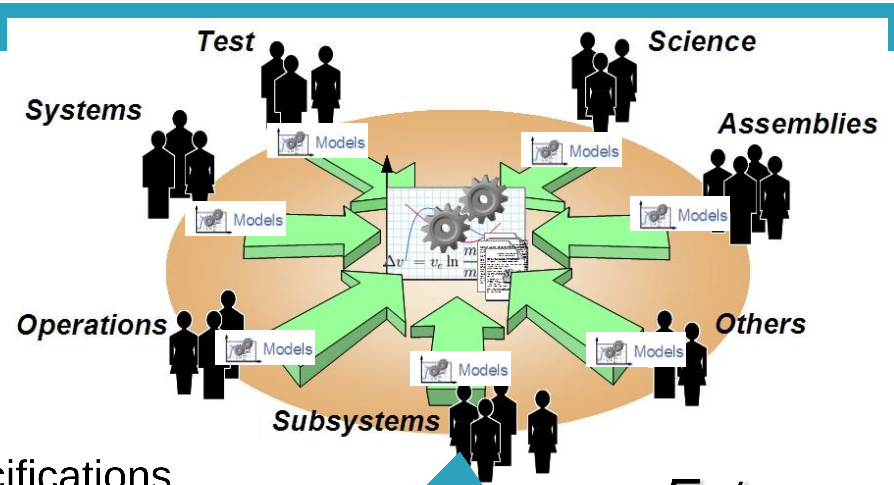
Source: Todd Bayer (JPL)

# MBSE: Consistency and Continuity

Transition to a rigorous system model ensures consistent modeling across disciplines and continuous access to this system model between system levels and across the life cycle

*Past*




*Future*

- Specifications
- Interface requirements
- System design
- Analysis & Trade-off
- Test plans

# Modeling in Traditional Systems Engineering

- “Models *have* been used as part of document-based systems engineering approach for many years, and include functional flow diagrams, behavior diagrams, schematic block diagrams, $N^2$ charts, performance simulations, and reliability models, to name a few.” *
- “However, the use of models has generally been <u>limited in scope</u> to support specific types of analysis or selected aspects of system design. The individual models have <u>not been integrated into a coherent model</u> of the overall system.” *

* *A Practical Guide to SysML*. Friedenthal, Moore and Steiner.

# What's New About MBSE

- Modeling is not new
  - Flight projects have a strong legacy of modeling (structural, thermal, circuit design, mission design…)
  - Systems Engineering uses models also, though typically limited in scope and duration. A set of requirements, an excel spreadsheet, and a PowerPoint drawing are all models.

- What is new is …
  - the availability of a formal modeling languages which can describe systems, and
  - the information engineering standards and tools which enable integration of a system model with existing discipline models.

# MBSE Benefits (1 of 2)

MBSE enables overall better quality, lower cost, and lower risk for the following reasons …

1. Modern modeling languages have clearer semantics than common boxes-and-lines diagrams, reducing miscommunication
2. Consistent, single source of information keeps team on same page and reduces time spent getting answers
3. Earlier detection of inconsistencies because models can be analyzed with respect to calculations, policies, rules, patterns, etc.
4. Documents are kept up-to-date because they can be auto-generated from the system model

# MBSE Benefits (2 of 2)

5. System model supports multiple views to address different stakeholder concerns, but all views refer to same model elements, so changes in an element appear on all views
6. Model-based simulation of state machines enables debugging and refinement of behavioral requirements
7. System model evolves across life-cycle *and* may guide similar future projects
8. System model helps to better manage complexity

**Summary**
Formal systems models offer these benefits because they introduce additional consistency and continuity, and because they are both human- and computer-understandable, and logically verifiable

# Comparison Summary

- MBSE doesn't replace traditional SE
  Rather, MBSE *formalizes* part of SE

- MBSE combines traditional methods and best practices with rigorous modeling techniques

- MBSE uses modeling languages that support rigorous modeling techniques and integration of various systems engineering disciplines (structural, electrical, mechanical, software, etc.) and stakeholders

# FAQ 5

What are typical purposes of modeling

# Model Purposes (1 of 2)

- To align interests and share understanding
  - The mission systems are usually created by people with different interests and skills who must work together.
  - A common model enables effective collaboration and helps to develop common understanding.
- To balance competing priorities to maximize stakeholder value
  - Offers a foundation for the systems to evolve, reuse, or integrate without substantial rework
  - Standard way of representing data across missions and it can be easily understood by the stakeholders
- The modeling offers process discipline by supporting small, iterative steps that can demonstrate incremental value and get early and continuous feedback
  - The model of a mission is built by composing common functions that can be re-used across missions
    - Increased efficiency of the system engineering process
    - Increased reliability (common modules already proven)
  - The model supports the addition of very specific functions without impact on the common modules

# Model Purposes (2 of 2)

- To describe a design in durable form
  - Almost anything can be used for that
- To communicate a design to a set of stakeholders
  - A common notation and familiar presentation idioms
  - Standards (e.g., SysML) cover most of that
- To Relate analyses to design
  - In general, a much harder problem
  - Largely outside the scope of SysML, except to provide language extension mechanisms that allow you to do this
  - If done automatically, software tools  to reason about models
    - This is also outside the scope of SysML, but some SysML modeling tools provide some help
- Modeling is supported by software tools that minimize the modeling efforts

# M2020 FS MBSE Goals

- Better awareness of the technical baseline
  - Capture the MSL as-flown design accurately
  - Move from “design by boxology” to an **integrated information technical baseline**
- Improve the communication, understanding, and visibility of the design
  - Provide an **authoritative source of information** that is easily accessible by the entire team
  - Create a **common language** to describe the system
  - Avoid information silos
  - Increase visibility and traceability of changes
- Focus on heritage design deviations
  - Rover first, then incorporate other stages
- Let FSSE products drive model implementation
  - Only model what is needed to create FSSE products
  - Add value at each increment

# FAQ 6

What are different types of models?

# Model Types (High Level)

- Different models render the system from different perspectives
  - Physical vs. abstract
  - Domain-specific vs. domain-independent

- And in different manners
  - Formal vs. informal
  - Descriptive vs. procedural

# Model Types (More Detailed)

- Models apply to a wide range of domains (e.g., systems, software, electrical, mechanical, human behavioral, logistics, manufacturing, business, socio-economic, regulatory)
- Computer-interpretable computational model
  - Time varying (e.g., performance simulations, structural dynamic analysis)
  - Static (e.g., reliability prediction model)
  - Deterministic or stochastic (e.g., Monte Carlo)
  - May interact with hardware, software, human, and physical environment
  - Includes input/output data sets
- Human-interpretable descriptive models (e.g., architecture/design such as UML, SysML, UPDM, IDEF, electrical schematic, 3D CAD geometry, DODAF 2.0)
  - Symbolic representation with defined syntax and semantics
  - Repository based (i.e., the model is stored in structured computer format)
- Supporting metadata about the models including assumptions, versions, regions of validity, etc.
- MBE can also include the use of physical models (e.g., scale models for wind tunnels or wave tanks)

# JPL Example Models

- Consumers of the M2020 FS design information are looking to extend the work into their domains:
  - End to End Information Systems
    - Enabling assessment of operational differences between MSL and Mars2020 due to communication relay asset obsolescence
  - Ground Data System
    - Leverage patterns and visualization products to populate a domain specific model
  - Thermal zone management

# FAQ 3

What is SysML?

# SysML Defined

- The OMG Systems Modeling Language (OMG SysML™) is a general-purpose graphical modeling language for specifying, analyzing, designing, and verifying complex systems that may include hardware, software, information, personnel, procedures, and facilities.
- It provides graphical representations with a semantic foundation for modeling system
  - Requirements
  - Behavior
  - Structure
  - Parametrics
- First version (1.0) released September 2007
- Current version (1.4) released September 2015

# What SysML Is Not

- SysML enables MBSE, but MBSE doesn't equal SysML; MBSE typically uses SysML as a standard visual modeling language and *lingua franca*, but is not limited to it

- SysML is *not* intended to replace current investment in modeling in the other engineering disciplines. (Nor could it.)

- It *is* intended that SysML-based models be the framework for interoperating with these discipline models, thus enabling integrated model-centric engineering

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

# SysML Development

- The SysML specification is a product of the Object Management Group (OMG)
  - an international, open membership, not-for-profit computer industry consortium
- SysML development is performed under the auspices of a Revision Task Force
  - SysML v1.4 RTF has just concluded
  - SysML v2.0 RTF now under way
- NASA is a member of OMG
- JPL has strong influence on both RTFs
  - Nicolas Rouquette represents NASA at OMG and serves on UML and SysML RTFs
  - Sanford Friedenthal chairs the Systems Engineering Domain Special Interest Group and consults for JPL

# Four Pillars of SysML

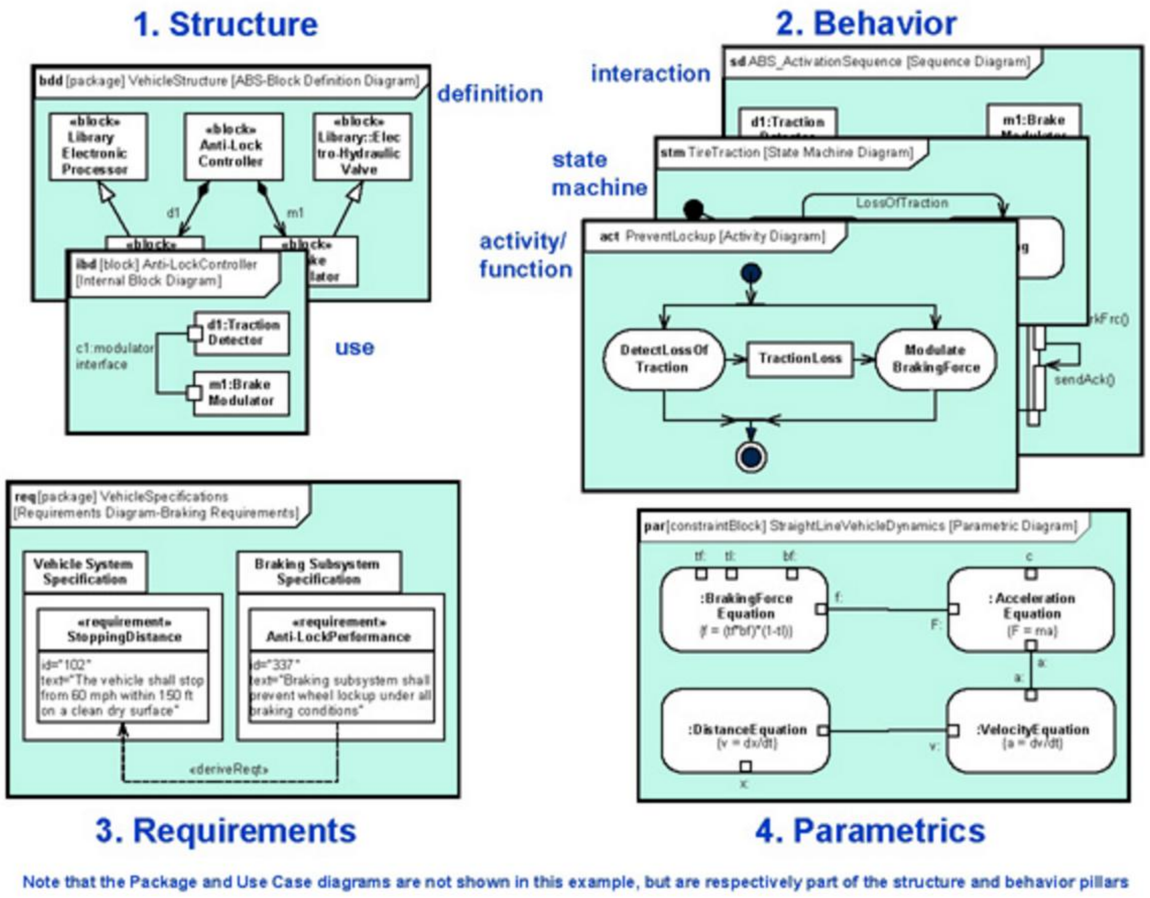



Figure 3. The Four Pillars of SysML

**Structure:** Specification of hierarchies, interconnection, model organization

**Behavior:** Specification of sequences of actions, life cycle of a block, message based behavior

**Requirements:** Specification of requirements and relationships among model elements

**Parametrics:** Expresses constraints, enables integration of engineering analysis and design models

www.omgsysml.org

# M2020 SysML Modeling Framework

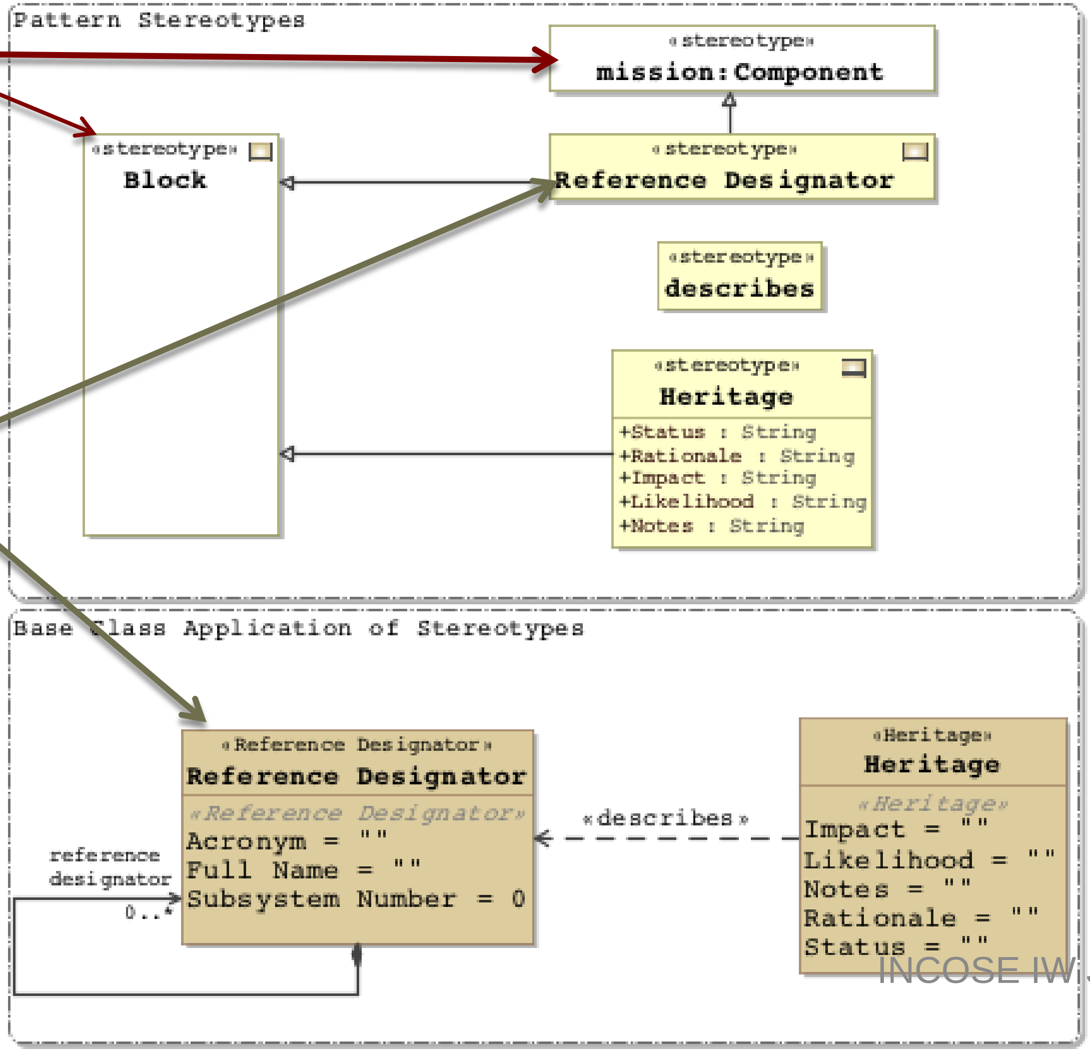
Extend SysML and JPL-provided stereotypes

Create M2020 specific terminology

Apply M2020 specific stereotypes to base elements

Relate base elements to create a re-usable pattern

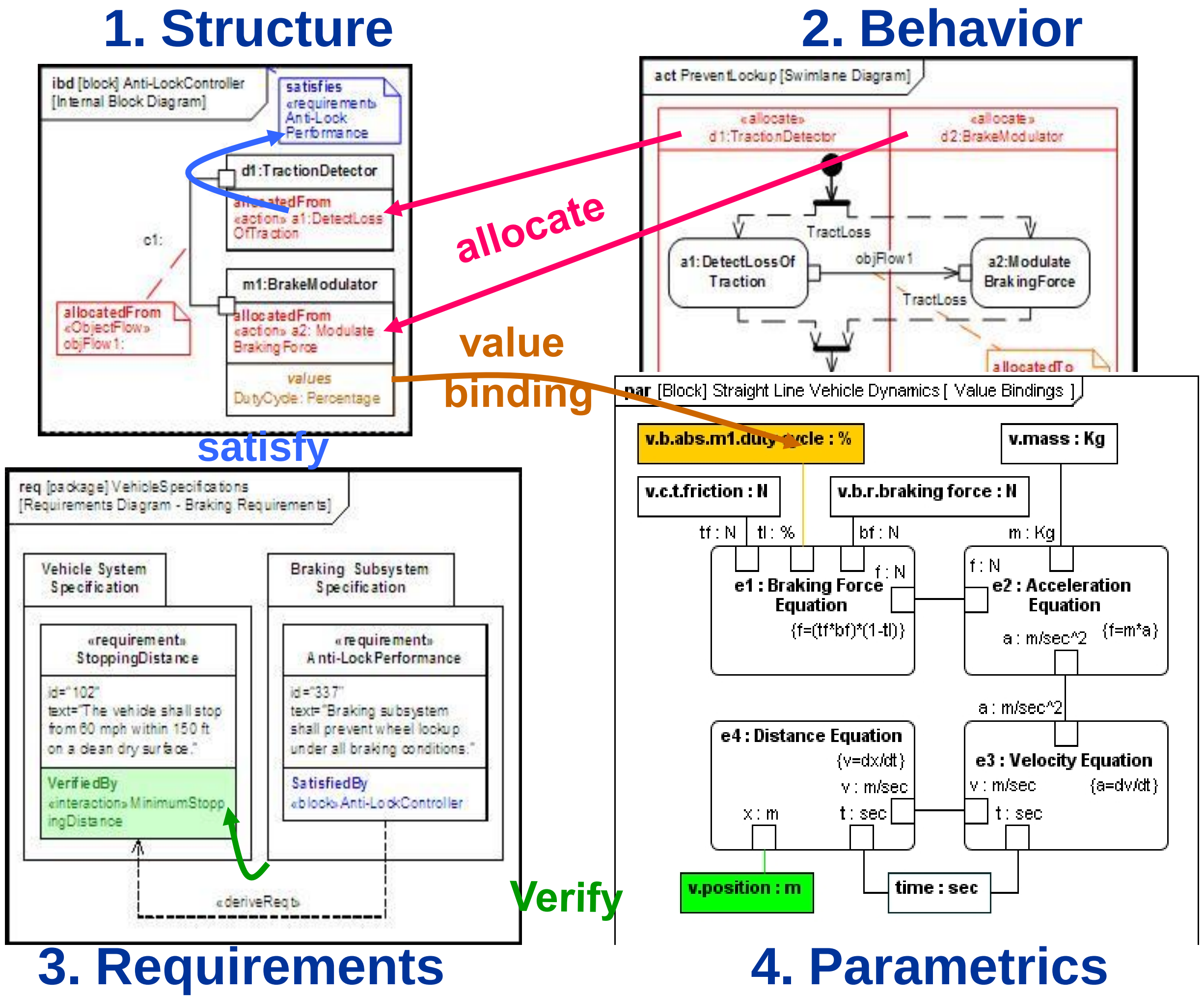
Utilizes SysML inheritance and redefinition concepts



# FAQ 4

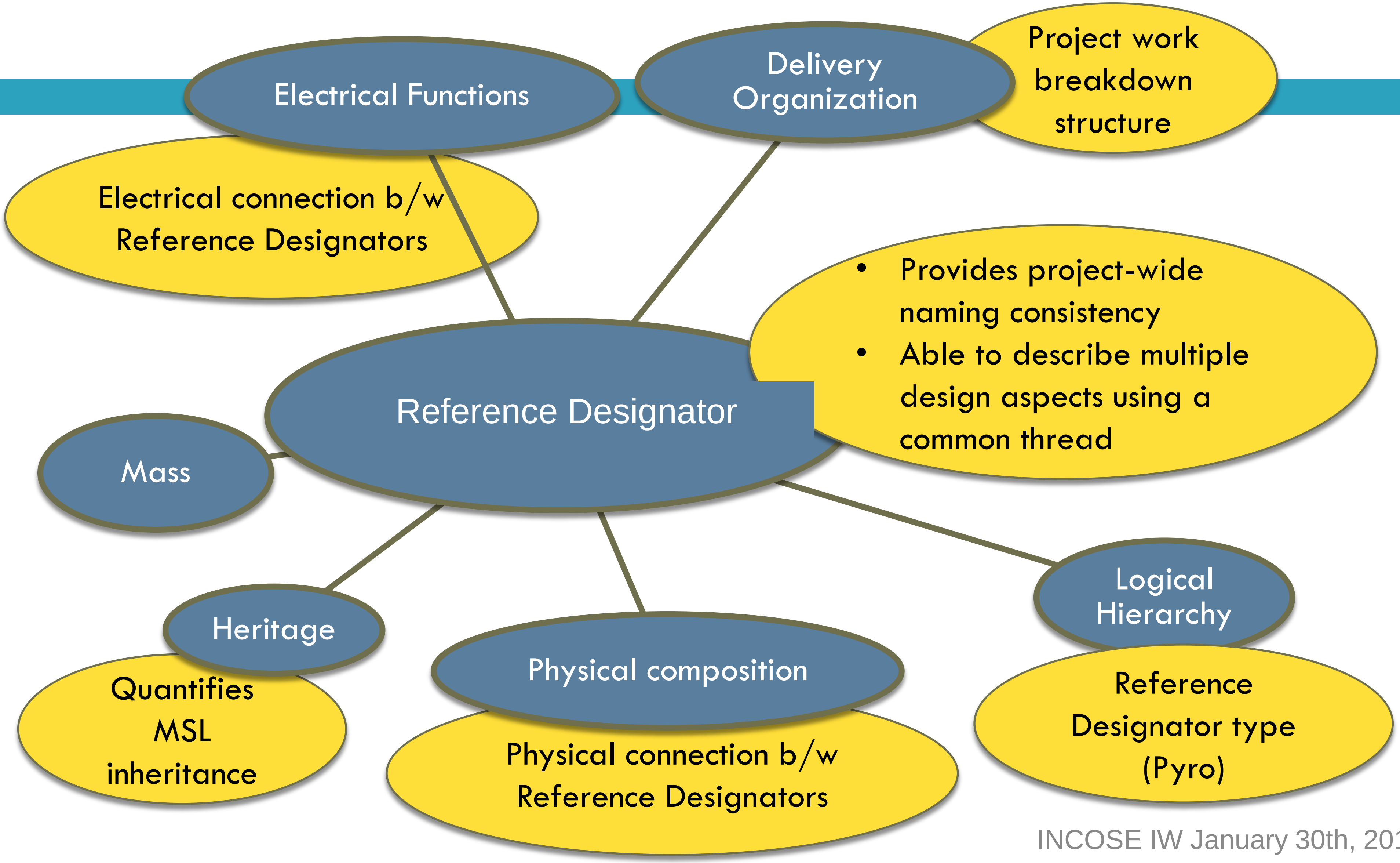
What is a System Model?

# System Model Defined




A system model is an interconnected set of model elements that represent key system aspects including structure, behavior, requirements, and parametrics

# M2020 Model System Model

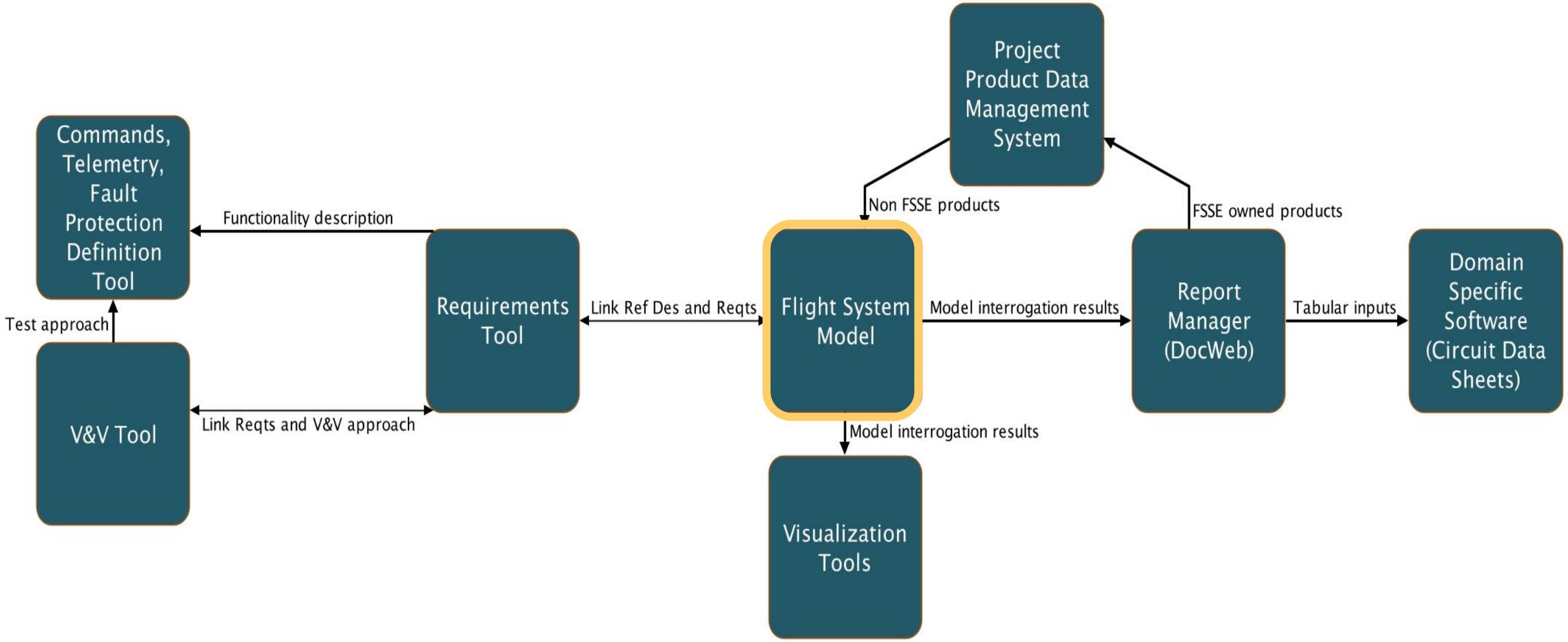


# M2020: Federated System Model

The system model exchanges information with discipline specific models to form the authoritative, up-to-date source of information at the system level

Institutional policy may prevent having one model to rule them all

Have to understand where all authoritative sources reside and have process/policy for integration



# FAQ 8

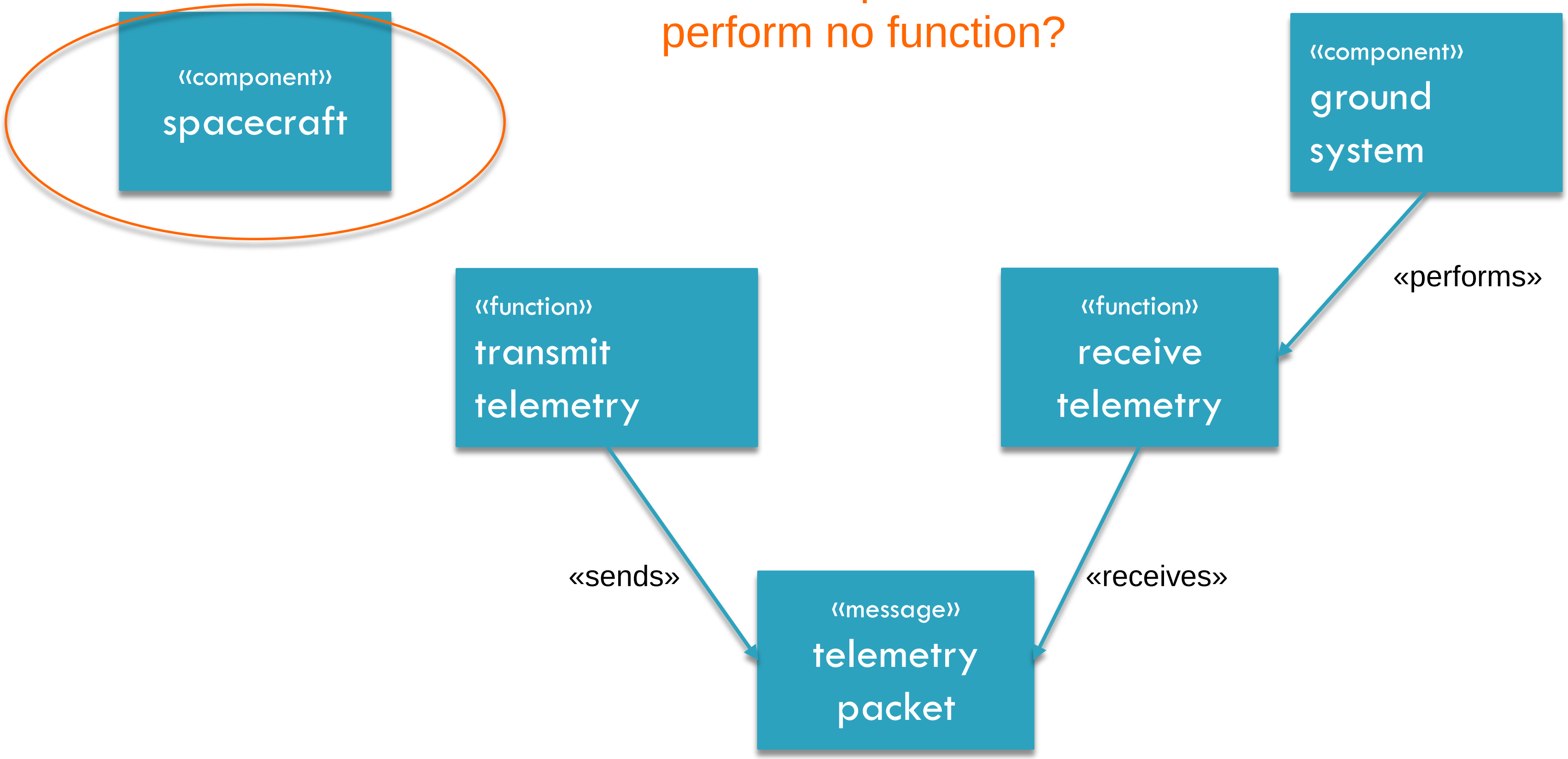
How Can Models Help an SE Effort?

# Reasoning About Models

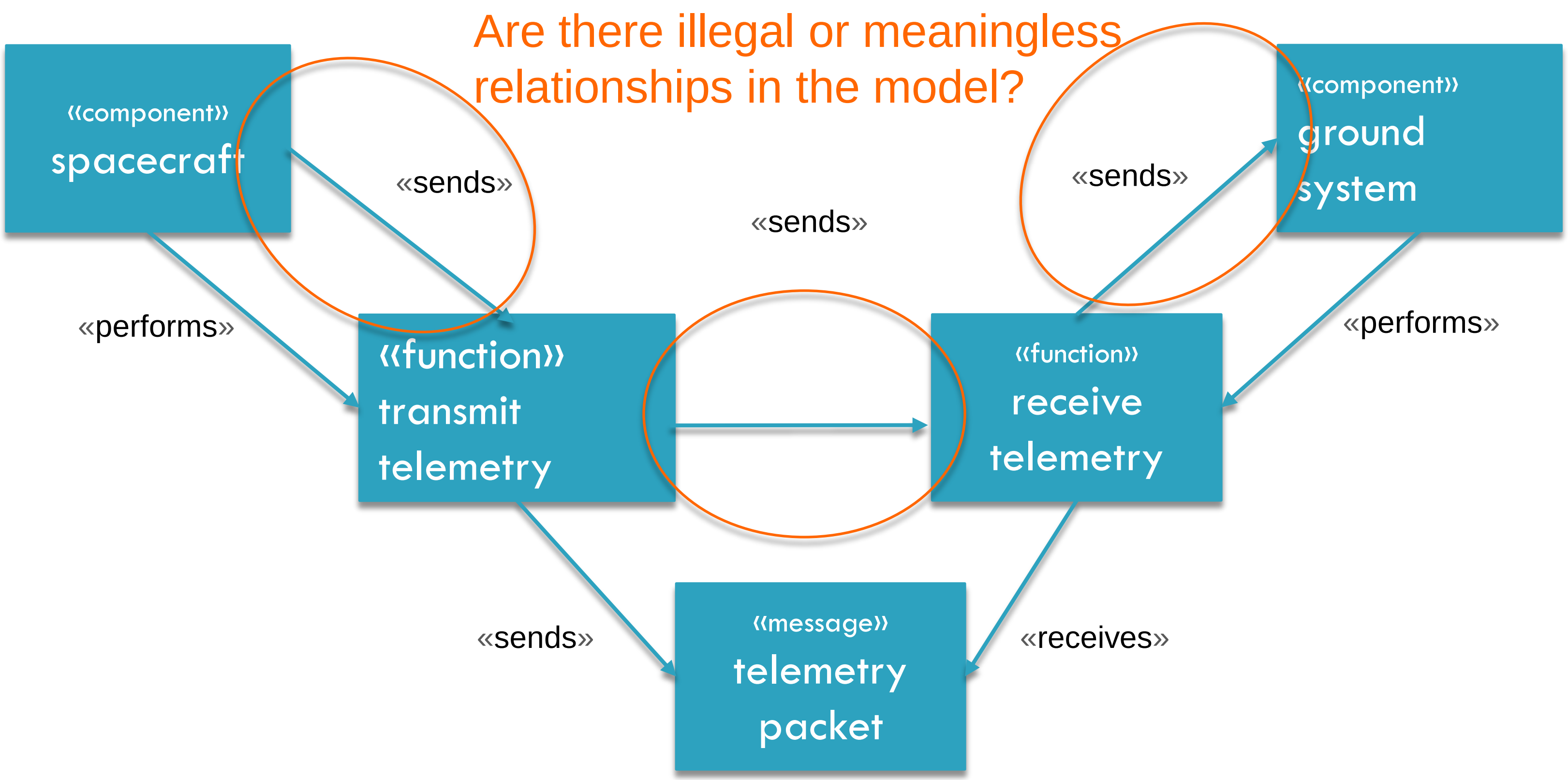
- Questions about the system itself
  - What is it?
  - How does it work?
  - Is the performance adequate?
  - What happens if something breaks?
- Questions about the model
  - Is it complete?
  - Is it consistent?
  - Does it support required analyses?
- Questions about the design artifacts
  - Are all required documents present?
  - Does each document contain all required content?
- We call answering these kinds of questions *reasoning*
  - It doesn’t necessarily mean exotic, artificial intelligence

# Reasoning About Completeness

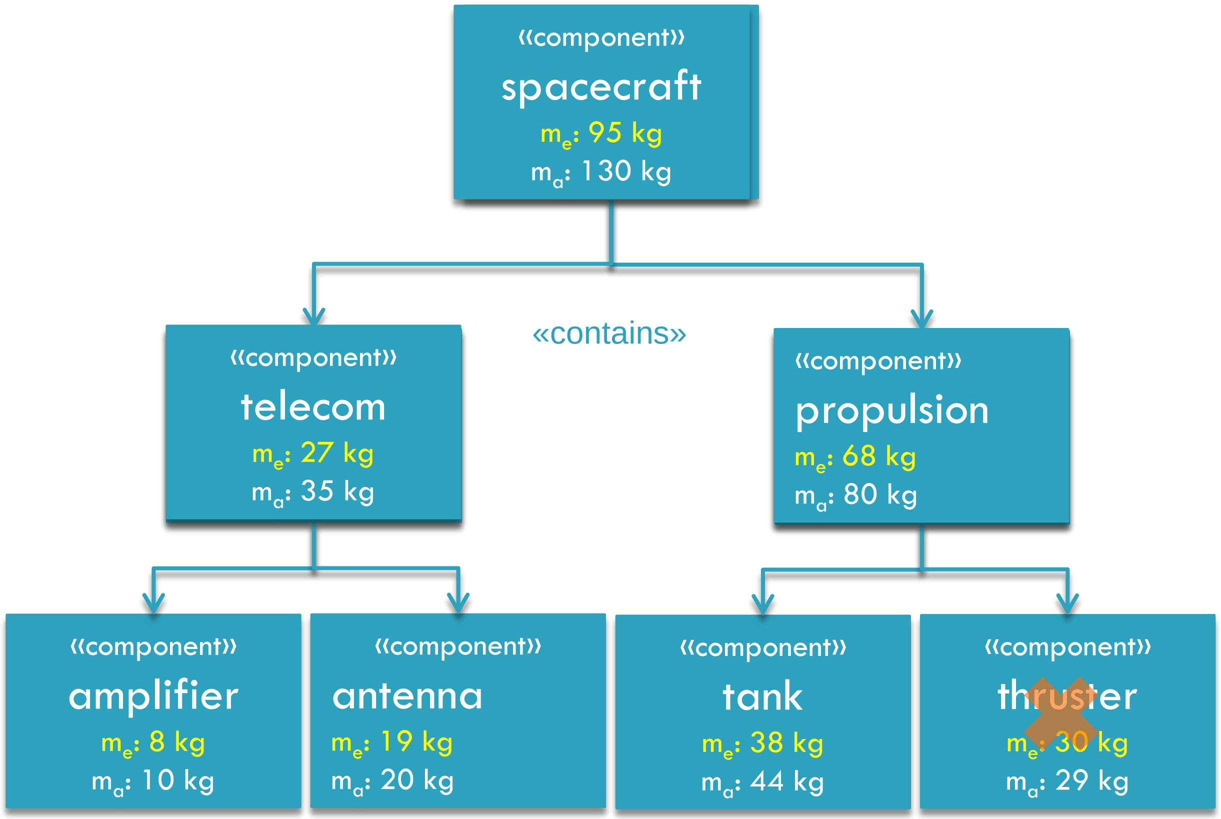
What components perform no function?



# Reasoning About Consistency



# Reasoning About Design




**Rule**: CBE mass $m_e$ of any component with parts is the sum of $m_e$ of its parts

**Policy**: $m_e < m_a$ for every component

$m_e$: estimated mass
$m_a$: allocated mass

# M2020 SysML Modeling Framework

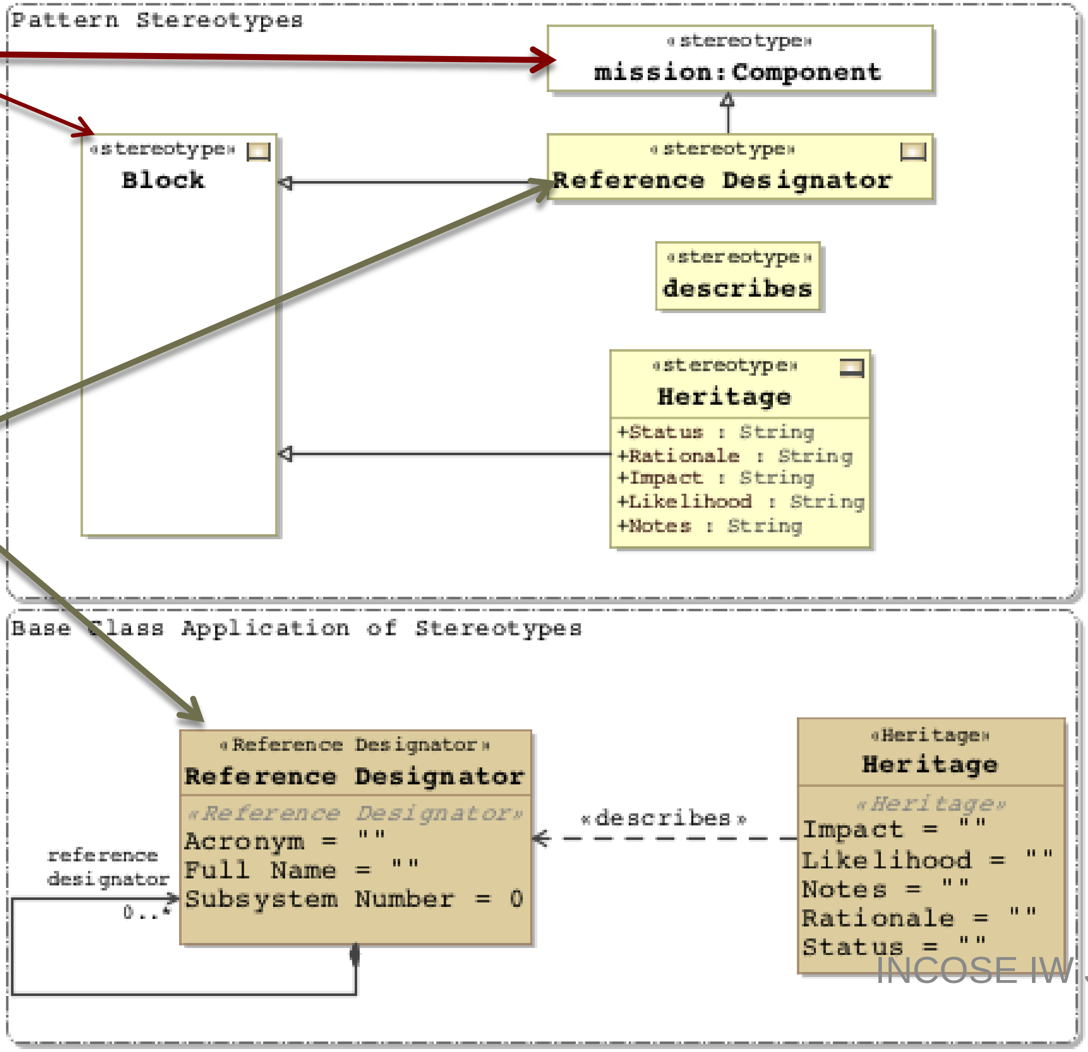
Extend SysML and JPL-provided stereotypes

Create M2020 specific terminology

Apply M2020 specific stereotypes to base elements

Relate base elements to create a re-usable pattern

Utilizes SysML inheritance and redefinition concepts

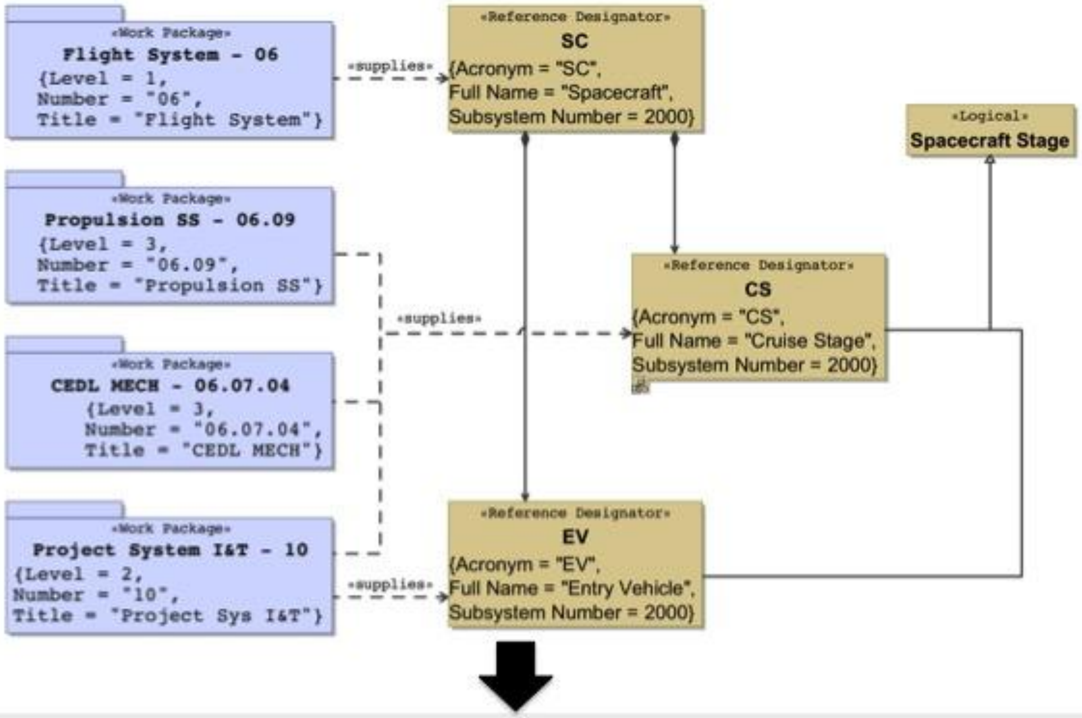



Identifies rules and policies

# Model Generated Products

- Transforms the FS model into documents
- SysML Viewpoint definition coupled with python scripts
  - Scripts collect, filter, and define how to display the model data
- Read only
- Web accessible

Reference Designator List
Electrical Function List
Reference Designator Heritage List
Mass Equipment List
Electrical Switch List
Mass Trending Report
Electrical Resource Metrics
Electrical Interface Derived Heritage




| | Subsys Number | ACRONYM | NAME | PHYSICAL HIERARCHY | LOGICAL HIERARCHY | DELIVERY ORGANIZATION |
|---|---|---|---|---|---|---|
| | 2000 | FS | Flight System | | | |
| Ref Des | 2000 | BS | Backshell | EV | Stage Element | CEDL MECH - 06.07.04 & Propulsion SS - 06.09 |
| Ref Des | 2000 | CS | Cruise Stage | SC | Spacecraft Stage | Project System I&T - 10 & CEDL MECH - 06.07.04 & Propulsion SS - 06.09 |
| Ref Des | 2000 | EV | Entry Vehicle | SC | Spacecraft Stage | Project System I&T - 10 |

# Leveraging Consolidated Information

## Electrical Function List

Table 1.1. Function List — COMMON FSSE PRODUCT

| Function Number | Function Name | Subsystem 1 | Assembly 1 | Flow | Subsystem 2 | Assembly 2 | SUB-SUB Name |
|---|---|---|---|---|---|---|---|
| 111-111-111 | RMCS-4-RLS-2 Position Sensors from SPAH | 2008 | RMCS-4 | > | 2014 | SCS | MCA-SCS (2008-2014) |
| 111-111-112 | RLCC-1B BusPwr: Driver to PAE-GUIDE-RLY-B-PYRO-B-ENABLE | 2004 | RLCC-1B | < | 2005 | PAE | PWR-PYRO (2004-2005) |
| 111-111-113 | RMCP-CSW-15 Position Sensors from SPAH | 2008 | RMCP | < | 2014 | SCS | MCA-SCS (2008-2014) |
| 111-111-114 | RTMC-A Analog: Voltage from RHPCU-B-TMC_+12V-MON | 2004 | RHPCU-B | > | 2006 | RTMC-A | PWR-CDH (2004-2006) |
| 111-111-115 | 28 V PWR to CIPA Liq Press Xdcr | 2004 | CLCC-1B | < | 2011 | CPLPX | PWR-THERM (2004-2011) |

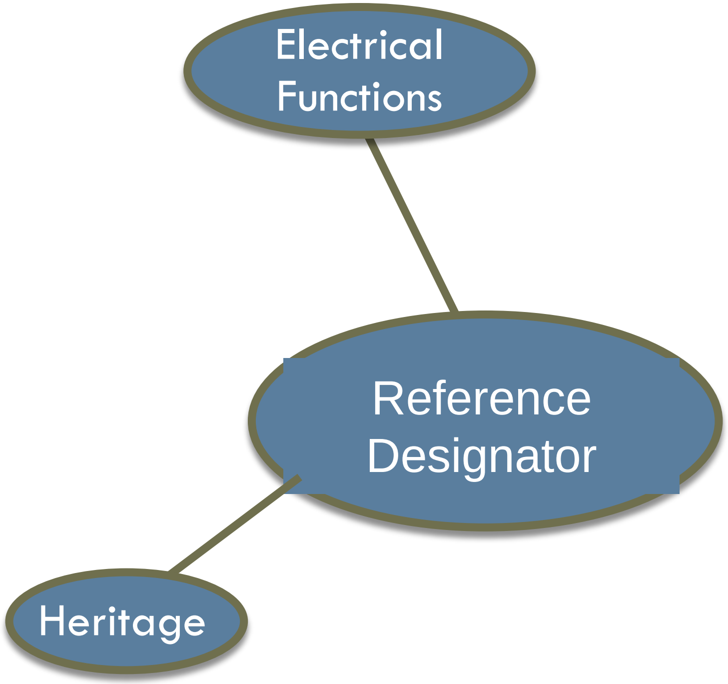
## Electrical Interface Resource Metrics

Table 1.1. Resource Metrics — COMMON FSSE PRODUCT

| | A-Side Eng | B-Side Eng | SCS A-Side Allocation | SCS B-Side Allocation | A-Side Open | B-Side Open | Total | Margin |
|---|---|---|---|---|---|---|---|---|
| RPAM Power Switches | | | | | | | | |
| 2A | 25 | 25 | 15 | 15 | 20 | 20 | 120 | 33.3% |
| 4A | 7 | 7 | 3 | 3 | 6 | 6 | 32 | 37.5% |
| 10A | 4 | 4 | 0 | 0 | 4 | 4 | 16 | 50.0% |
| PWR Analogs (RPAM) | | | | | | | | |

## Electrical Function Heritage Derivation — INFORMAL PRODUCT

| Function Number | Function Name | Subsystem 1 | Assembly 1 | Flow | Subsystem 2 | Assembly 2 | SUB-SUB Name | Notional? | Stages Involved | Heritage Designation |
|---|---|---|---|---|---|---|---|---|---|---|
| TEL-PWR (2002-2004) | | | | | | | | | | |
| 002-004-001 | 28 V PWR to TWTA | 2002 | TWTA | < | 2004 | DLCC-A1, DLCC-B1 | TEL-PWR (2002-2004) | | Descent Stage | Heritage |
| 002-004-002 | 28 V PWR to SSPA | 2002 | SSPA | < | 2004 | RLCC-A1, RLCC-B1 | TEL-PWR (2002-2004) | | Rover Internal | Heritage |



# Model Construction Validation

- Standardized language coupled with domain specific patterns gives greater situational awareness about the design
- Re-usable validation rules can be written against the model
- Can ask questions about completeness and work left to go in a quantifiable manner.

## 1. Reference Designators (D-79381 Rev D)

**Table 1.1. Reference Designators**

| | Subsys Number | ACRONYM | NAME | PHYSICAL HEIRARCHY | LOGICAL HIERARCHY | DELIVERY ORGANIZATION |
|---|---|---|---|---|---|---|
| | **2000** | **FS** | **Flight System** | | | |
| RefDes | 2000 | BS | Backshell | EV | Reference Designator | Project System I&T - 10 |
| RefDes | 2000 | BUD | Bridle Umbilical Device | DS | Reference Designator | Mech SS - 06.07 |

Reference Designators should only have another Reference Designator identified as it's physical heir

## 2. Validation Suite Summary and Detail

**Table 2.1. M2020 Reference Designators Validation Suite Summary**

| Validation Rule | Description | Severity | Violations Count |
|---|---|---|---|
| Ref Des Parent | A Reference Designator should only be black diamond associated with another Reference Designator | ERROR | 0 |
| Ref Des Single Parent | A Reference Designator should have only 1 black diamond association point to it | ERROR | 0 |
| AOC Relation | Aggregators of Convenince (AOC) should only relate to other elements by white diamond associations | ERROR | 0 |
| Ref Des Not Associated to Itself | A Reference Designator should not be associated to itself | ERROR | 0 |
| AOC Not Associated to Itself | An Aggregator of Convenience (AoC) should not be associated to itself | ERROR | 0 |
| Ref Des Has WBS | All Reference Designators should be related to a WBS | ERROR | 663 |

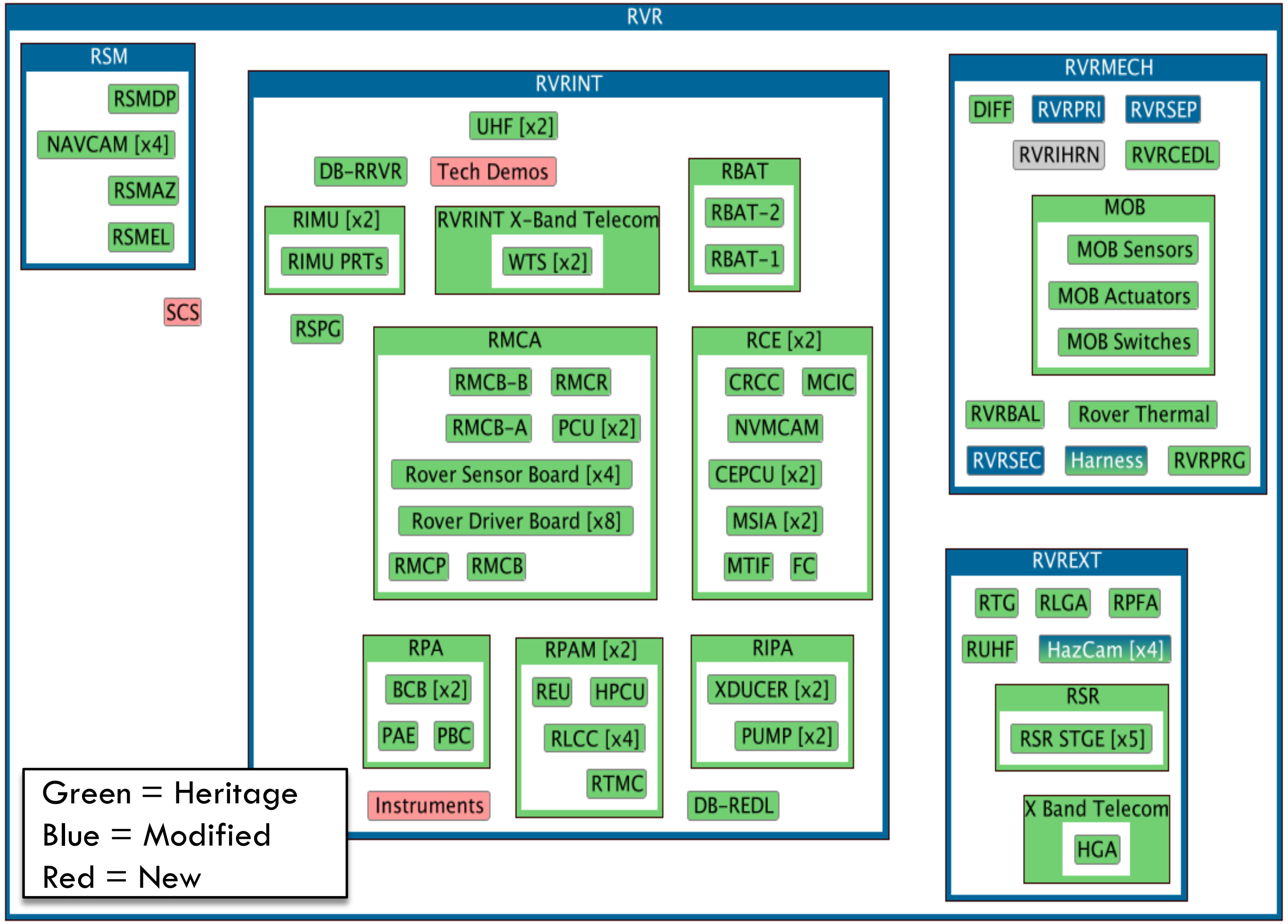
Validation results can be displayed as a report or live-linked within the modeling tool

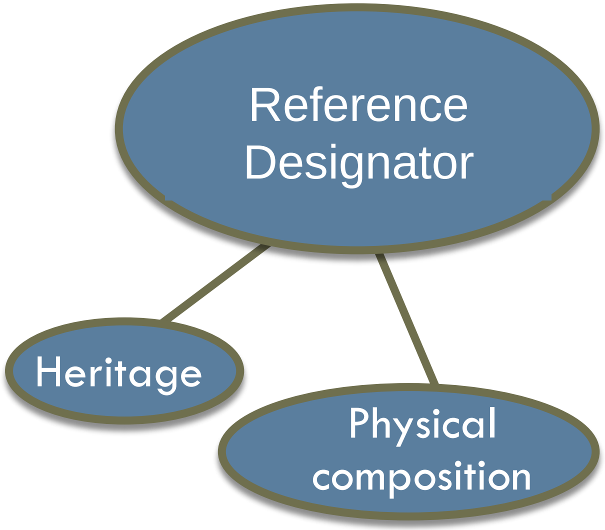
# Automatic Diagram Creation

- Visual representations of model info (IBDs, BDDs) are typically created and maintained by hand
  - Decays as information evolves
  - Technical information representations often communicate poorly to stakeholders/reviewers
  - No way to validate that what is being shown is complete
- Wanted to create rules that dictate the layout of elements on a diagram
  - *"All rover avionics boxes should be inside the rover structure"*
- Wanted to create presentation rules:
  - *"All 1553 Data connections are shown as dashed lines"*
- Deploy the visual representation as either static diagrams or interactive applications

# Rule Based Diagram

*What is the rover heritage story at the "box" level?*



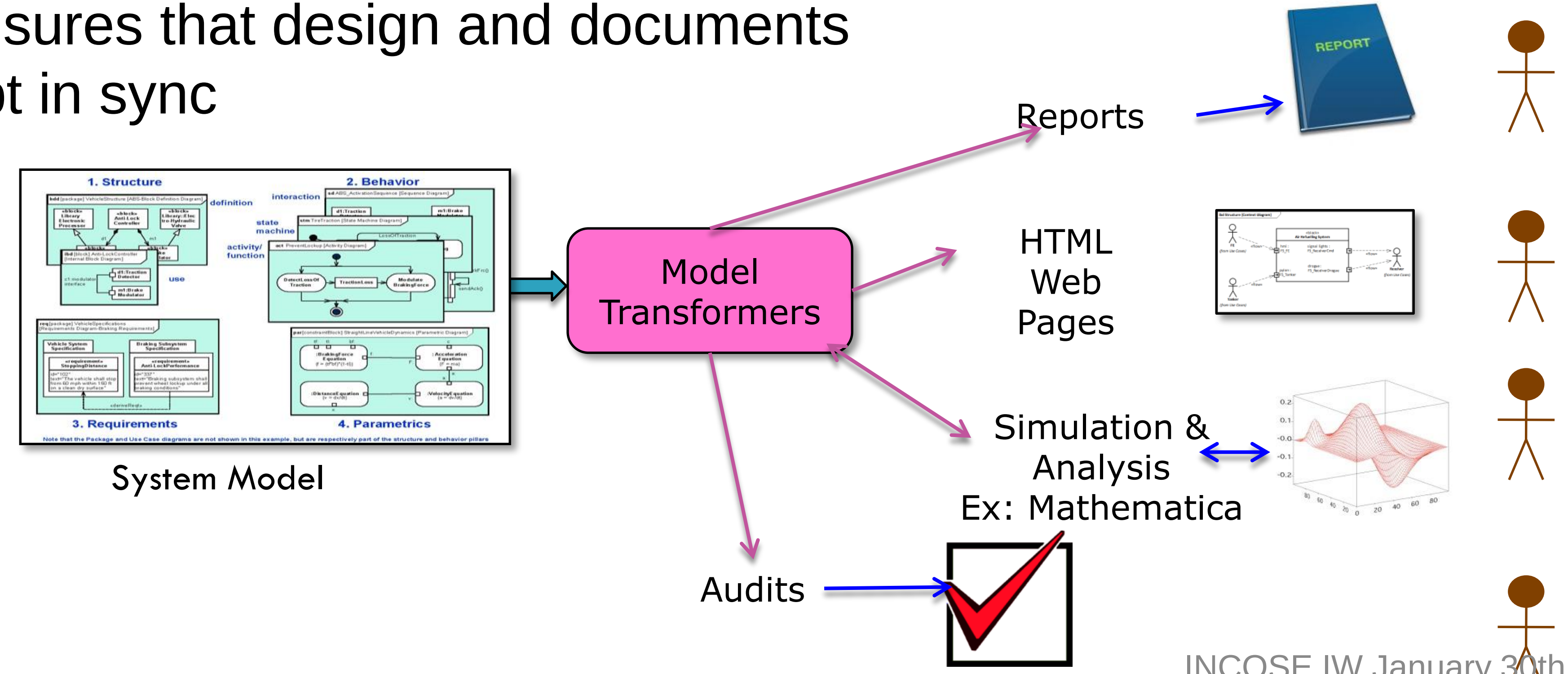

Rule based
- Color coding
- Element nesting
- Automatic Layout

# FAQ 9

What Does MBSE Mean For Projects?

# MBSE and Deliverables

- Project still has to produce deliverables for each review
- Some documents may be generated automatically from system model
  - This ensures that design and documents are kept in sync

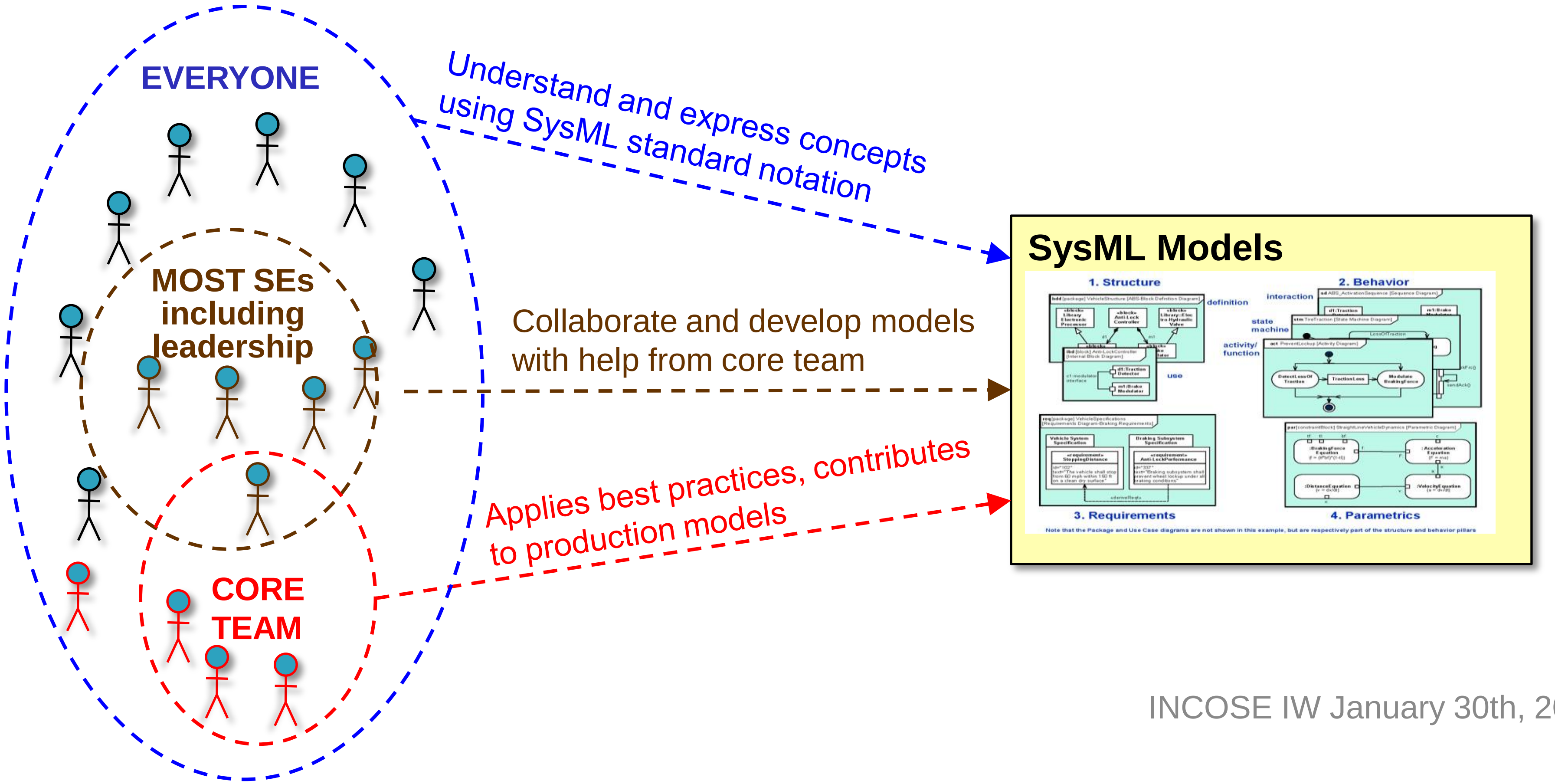


# MBSE and Schedule

- Projects will need to schedule time and resources to deploy infrastructure and train workforce
- Model development becomes infused within the product development schedule

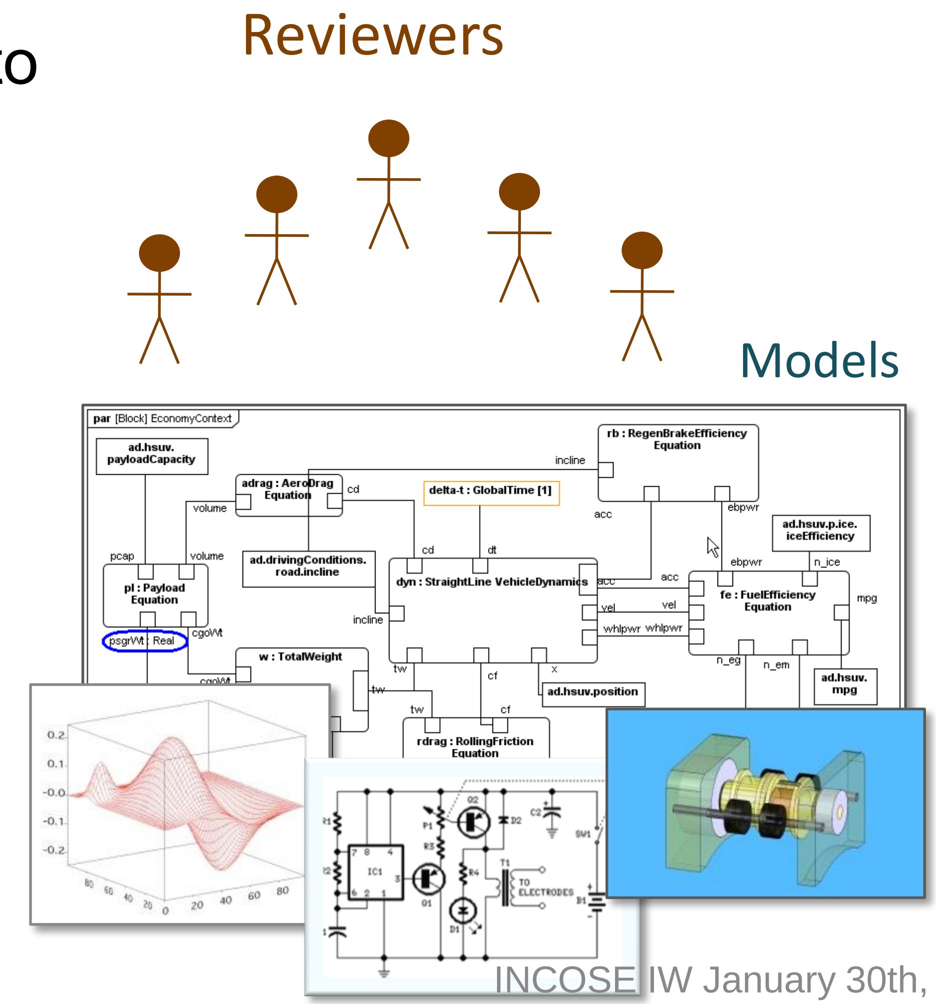
# MBSE and Project Organization

- Everyone needs training, but not to the same depth
- Different levels of training for different levels of modeling



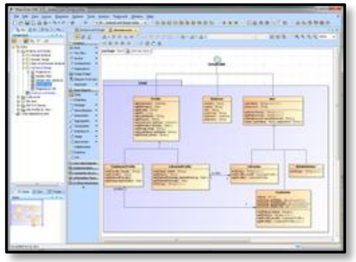
# MBSE and Project Reviews

- It *can* affect reviews, but doesn't have to
- Leverage the model by reviewing *the model itself*
- Stakeholders focus on the views of the system model that address *their* concerns



# MBSE and Infrastructure

- Need:
  - System modeling tool(s)
  - Training (in modeling and in tool usage)
  - Standards (modeling style guide, model management)
  - Methodology*



**SysML Modeling Guide**

IMCE Team
Version: 9/24/11

* See "Survey of Model-Based Systems Engineering (MBSE) Methodologies", J. A. Estefan, 2008, INCOSE. http://www.omgsysml.org/MBSE_Methodology_Survey_RevA.pdf

# MBSE and Project Metrics

- Easier to get data from models and update metrics
- Example metrics
  - Quality of design
    - Mass margin, power margin, data margin, cost, …
  - Progress of design and development effort
    - Completeness of component specs, # use case scenarios, …
  - Estimated effort to complete design and development
    - Constructive Systems Engineering Cost Model (COSYSMO) gets inputs from system model (# requirements, # use cases, etc.)
  - Others:
    - Number of critical TBDs
    - Stability of requirements and design changes over time
    - Potential defect rates

# M2020 MBSE Benefits

- FS model forced the integration of disparate MSL as-flown information
  - Drove out inconsistencies early
  - Allows team to focus on what's new and likely to change
- Inclusion of the less mature design aspects in the model allows the FSSE's to work in the same environment/use the same patterns to describe both new and old
- Consumers of the FS design information are looking to extend the work into their domains:
  - End to End Information Systems
    - Enabling assessment of operational differences between MSL and Mars2020 due to communication relay asset obsolescence
  - Ground Data System
    - Leverage patterns and visualization products to populate a domain specific model

# FAQ 14

How has JPL infused MBSE?

# Lessons Learned* (1 of 2)

1. Investment is crucial
   - Project investment in tools, modeling environment, and training
2. Unity of leadership is essential
   - Management must be willing to pay the startup costs and give time for the effort to pay dividends
3. Best way to start modeling is to hire people who already know how to do it
   - Later infusions will benefit from an experienced pool of engineers
4. Team organization matters
   - 3-tiers: small set of core modelers, larger set of modeling-savvy SEs, within larger set of project personnel
5. Everyone needs training, but not to the same depth

* Source: Todd Bayer (JPL)

# Lessons Learned* (2 of 2)

6. Best way to figure out how to apply MBSE: do it for real
   - "Shadow pilots" would not have been helpful
   - Pressure to deliver real engineering products forces discovery and resolution of problems not likely encountered in a shadow
7. Keep the focus on project deliverables, and model only as far as you need to answer the questions
   - This may need to be constantly reinforced
8. Description first, then analysis
   - Just describing something in a formal model language immediately improves communication and understanding
9. Separate models from analyses
   - Mass analysis script is independent of system details
10. Real examples are powerful
    - Much more effective at conveying understanding and building support

* Source: Todd Bayer (JPL)

# Lessons Learned: Strategy

- Young engineers are already attuned to system modeling
  - the transition to MBSE is happening whether we help or not
  - pair young engineers (modelers) with veterans (subject matter experts)
- Collaboration has been essential
  (industry, other space agencies, academia)
- Infusion can be gradual, both in time, and in project space
  - benefits are evident even when systems modeling is used in a modest way on a single subsystem
  - the simple act of creating a formalism is by itself a significant help in communication and understanding
- Culture change does not follow a project life-cycle

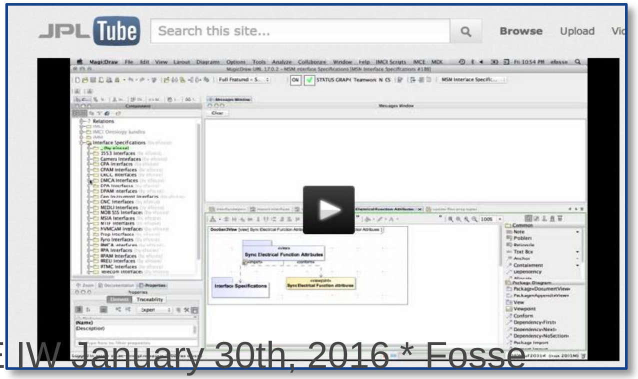
# Lessons Learned: Just Modeling Isn't Enough

…we also have to agree on…

- Ontology
  - What concepts are important to us?
  - What properties and relationships do those concepts have?
  - How do we name concepts and properties?
- Notation and exchange syntax
- Tools and model repository
  - How do I create models?
  - Where can I store my models?
  - Where can I find other models and relate to their content?
- Model data access mechanisms
- Model validation rules and constraints
- Configuration management procedures
- Relationships between engineering deliverables and model content

# M2020 MBSE Infusion

- June 2013-December 2013: Initial FS MBSE effort
  - FSSE team members were unfamiliar with SysML and MBSE methodology
  - Six MBSE practitioners were added to the team
  - Significant portion of MSL as flown design information was imported
  - Preliminary document views were created
  - Framework was defined
- December 2013-present: FS MBSE Infusion
  - FSSE information owners use the model to maintain design information
  - Two MBSE practitioners remain (<1 FTE total)
    - Framework extension
    - Validation rules
    - Report and visualization generation
    - SysML software usability
- Documented process and created M2020-specific video tutorial

# M2020 MBSE Infusion Barriers

- Difficult to support “sandbox” type work/analysis
  - Mass Equipment List (MEL) is maintained in the model
  - Not conducive to “what-if” analysis during meetings
    - Opening up the SysML model, navigating to the mass information, re-calculating the MEL
  - Instead the MEL owner duplicates information and performs such work outside of the model
- Attention must be paid to communicating usefulness of new tools
  - Training for Tom Sawyer, how to interpret the deployed visualization
  - Training for M2020-specific framework
- Ability to create useful views within the modeling environment for editing
  - Tables v. specification windows

# Benefits

- FS model forced the integration of disparate MSL as-flown information
  - Drove out inconsistencies early
  - Allows team to focus on what’s new and likely to change
- Inclusion of the less mature design aspects in the model allows the FSSE’s to work in the same environment/use the same patterns to describe both new and old
- Consumers of the FS design information are looking to extend the work into their domains:
  - End to End Information Systems
    - Enabling assessment of operational differences between MSL and Mars2020 due to communication relay asset obsolescence
  - Ground Data System
    - Leverage patterns and visualization products to populate a domain specific model

# FAQ 11

How good is a model?

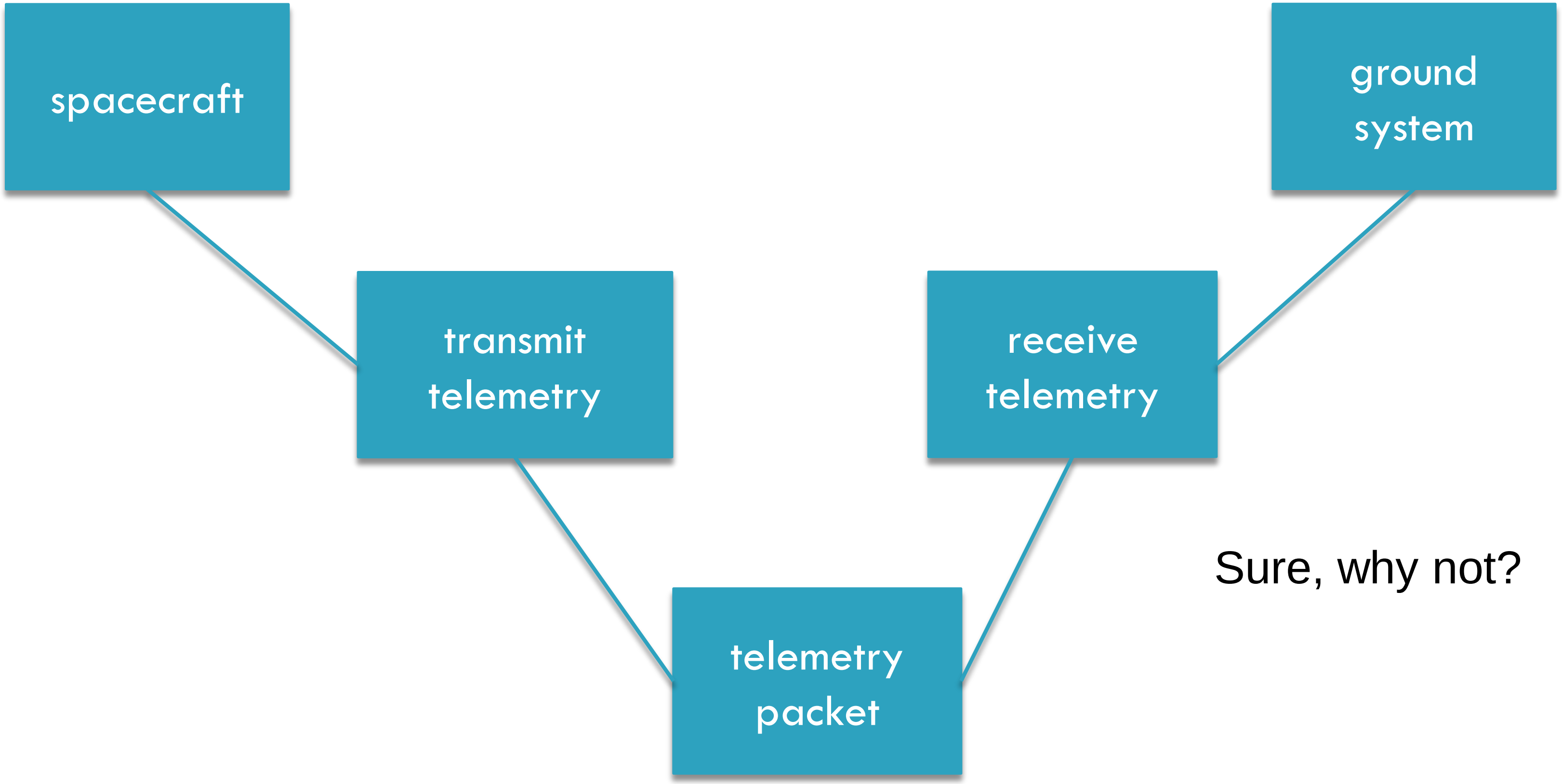
# Some Objectives of Modeling

- To describe a design in durable form
  - You can use almost anything for that
- To communicate a design to a set of stakeholders
  - Now you need (at least) a common notation and familiar presentation idioms
  - Standards (e.g., SysML) cover most of that
- To organize and relate analyses of a design
  - This is, in general, a much harder problem
  - You have to make sure that every element that could affect an analysis is present, properly identified, and consistently related to appropriate other elements
    - This is largely outside the scope of SysML, except to provide extension mechanisms that allow you to define the rules
  - You also need software to reason about your models
    - This is also outside the scope of SysML, but some tools do
  - Analysis operates on facts

# Questions to Ask When Evaluating a System Model

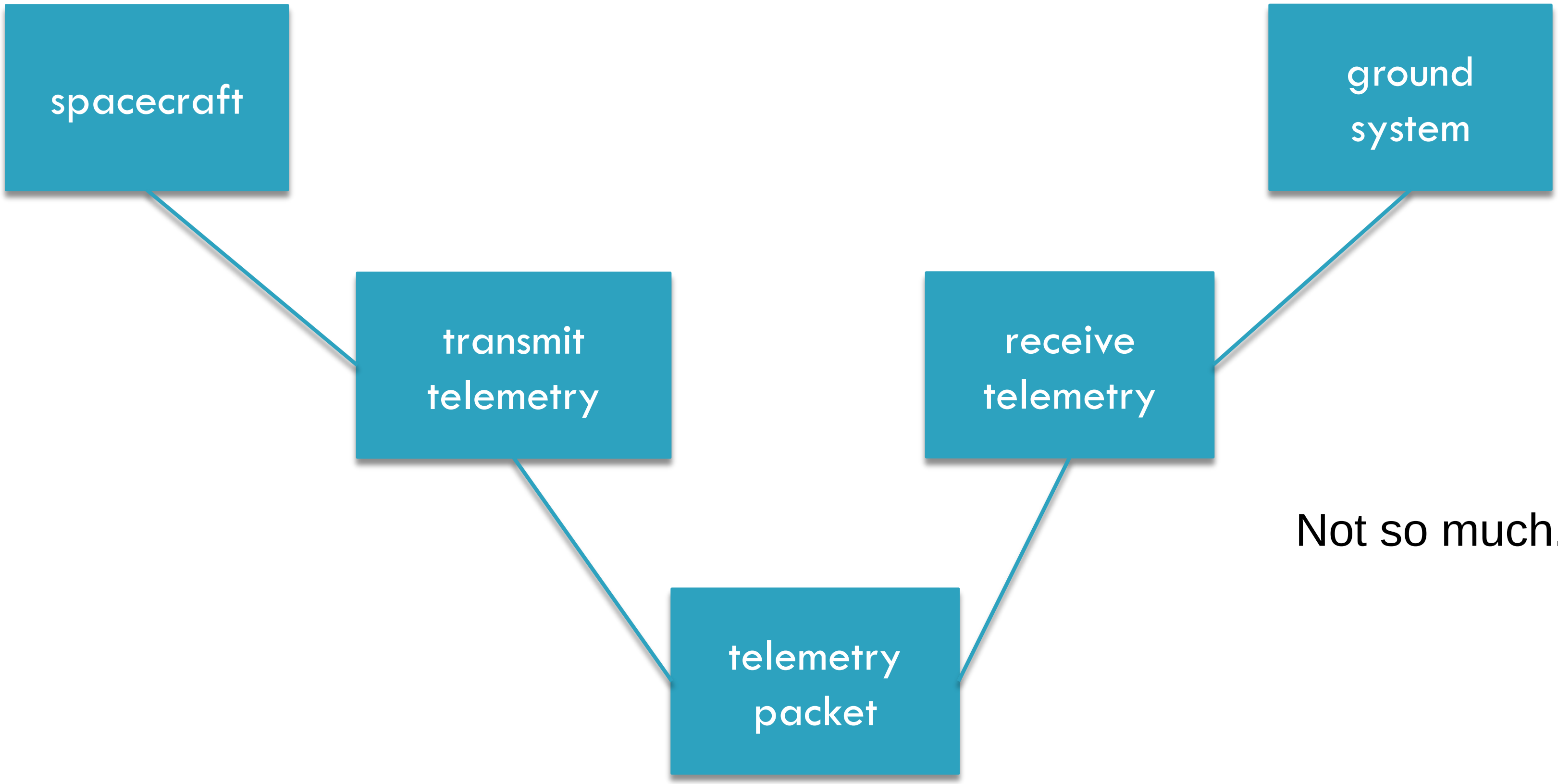
- Meaning of the model
  - Is the modeling notation expressive enough for the domain?
  - Does it convey the conventional domain wisdom?
  - Is the semantics of the model elements unambiguous?
  - Can pertinent questions about the domain be answered?
- Model generic logical correctness
  - Does it support "reasoning" about the model?
  - Is the model complete?
  - Does it support the required analyses?
  - Does it support reasoning about the design?
  - Does it support reasoning about the programmatic aspects?

# Is This A Model?



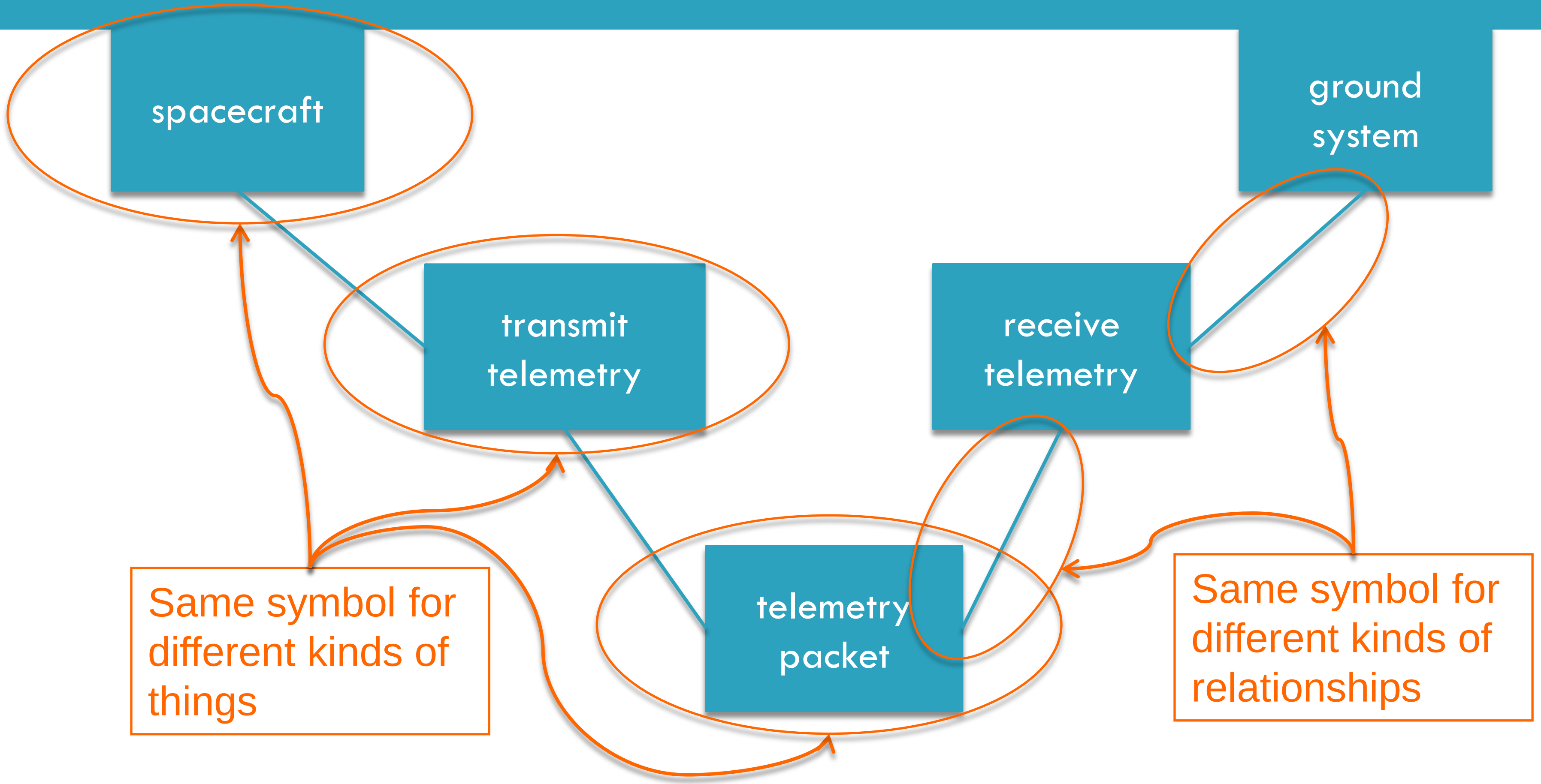

Sure, why not?

# Is It A Good Model?

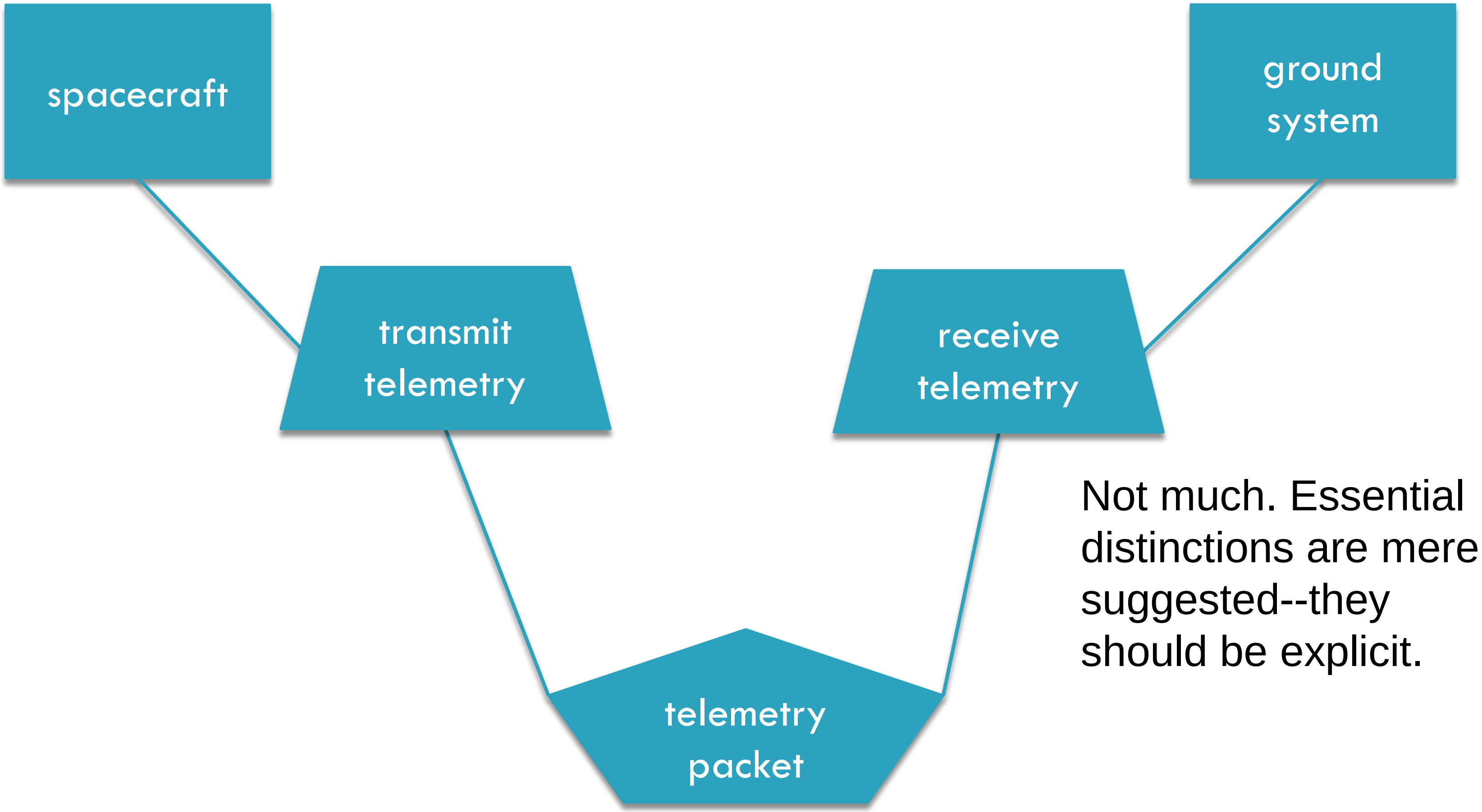



Not so much.

# What's Wrong With It?



# Better?

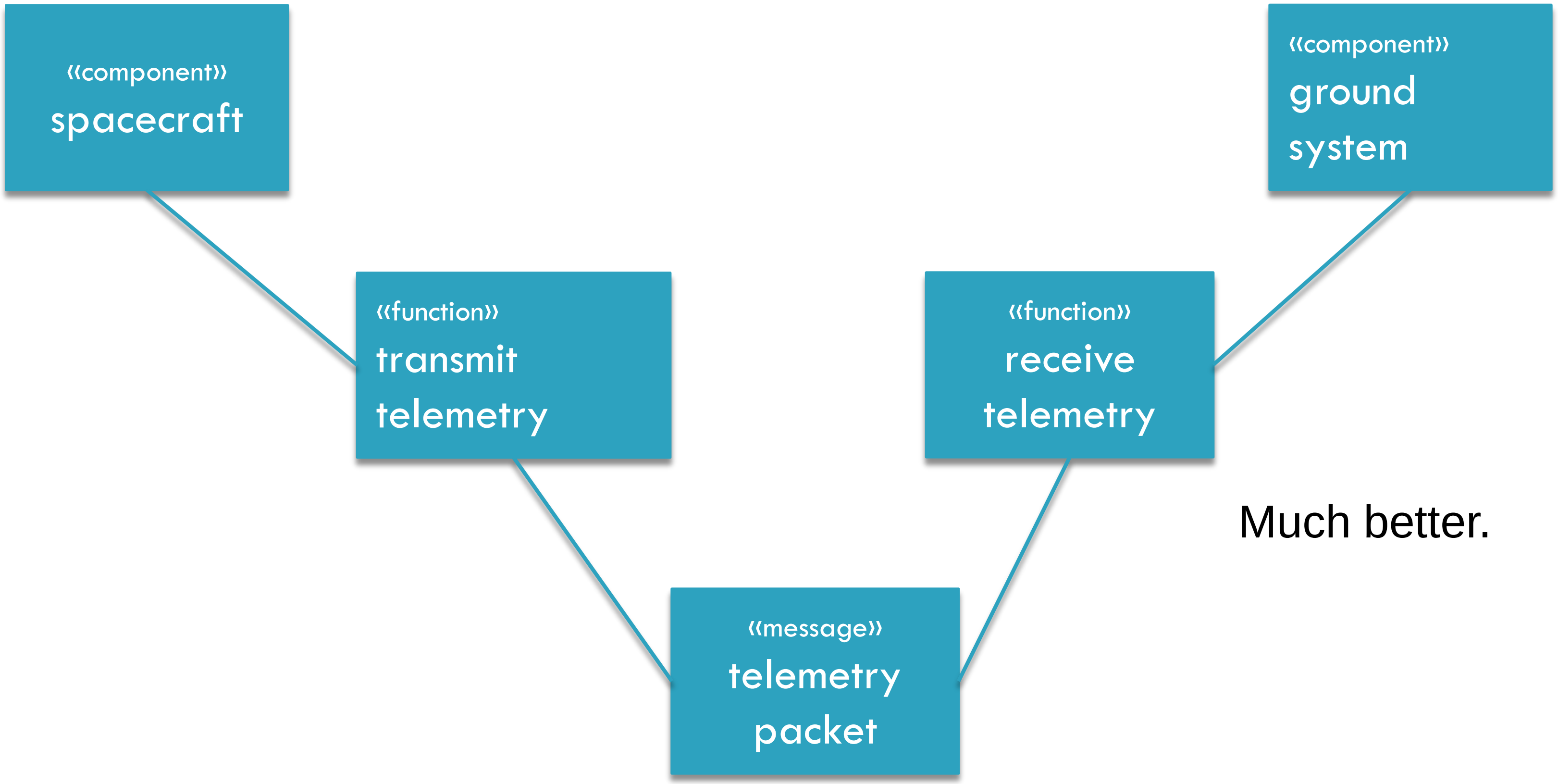



Not much. Essential distinctions are merely suggested--they should be explicit.

# Making Distinctions Explicit

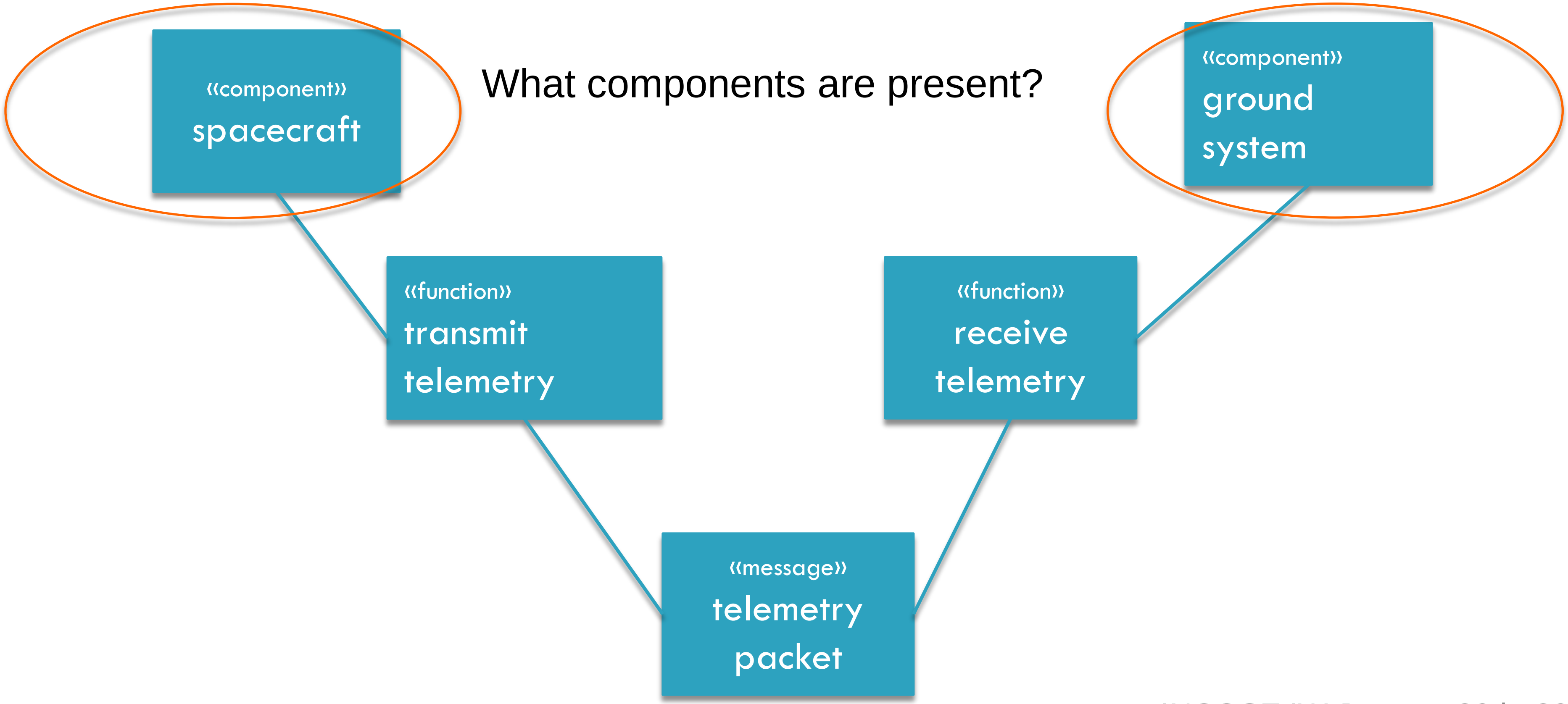
- Rather than merely hinting at distinctions with shapes or colors, we could devise a set of types or classes to be applied to model elements
- The set of types is application-dependent
  - Systems engineers talk about different things from chefs
  - The distinctions are whatever matters for your application
  - Is red wine a different type from white, or is is merely a property of wine?
    - It depends on what you want to say about wine
- What kinds of things do systems engineers talk about?
  - Component, Interface, Function, Requirement, Work Package, Product, Process, Objective, Message, etc.
- Let's apply some classes to our model
- For now, every element has
  - one type, denoted like this: «type»
  - one name, which identifies an individual of that type

# Model With Typed Elements



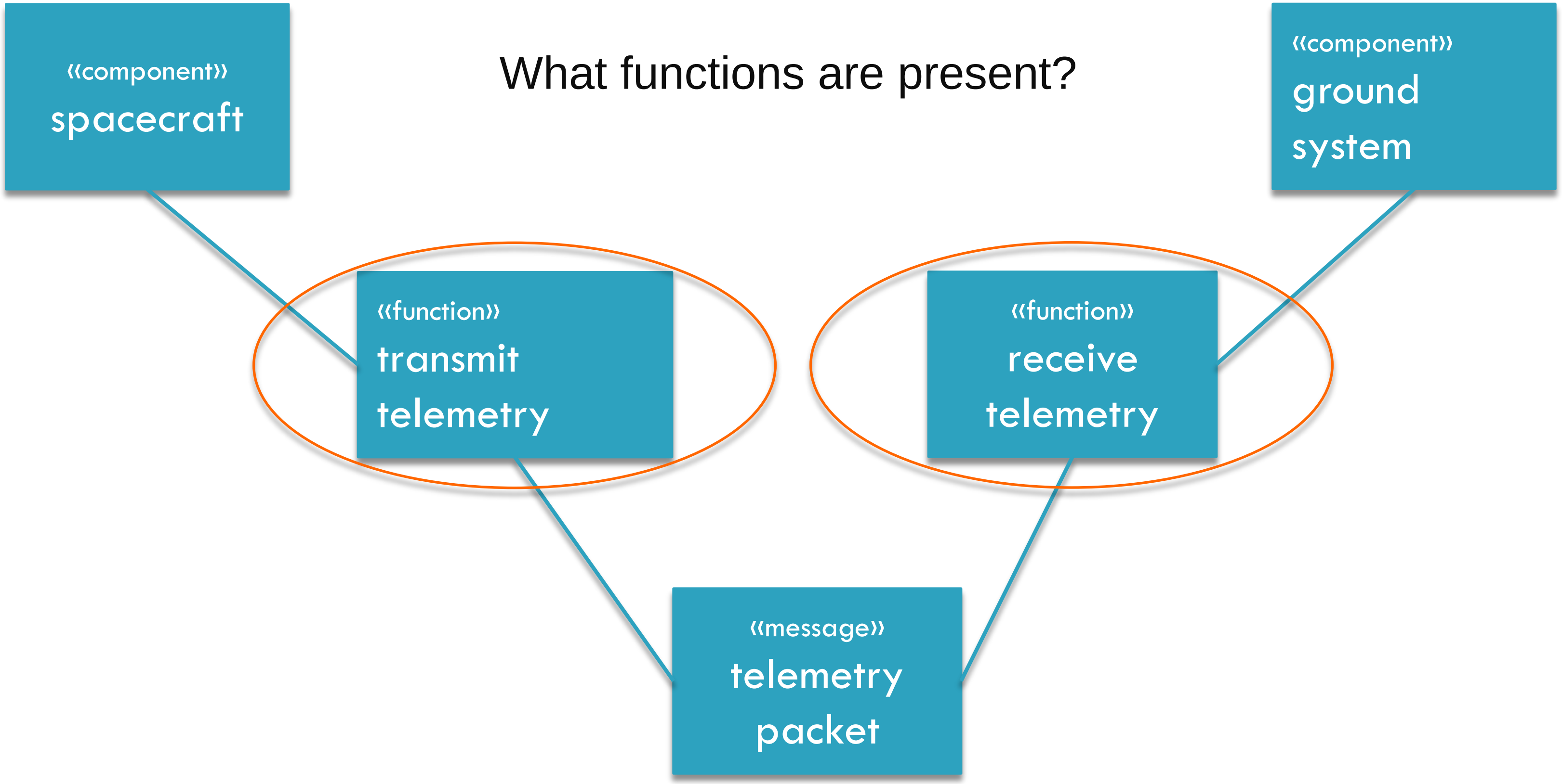

Much better.

# Answering Questions (1 of 2)

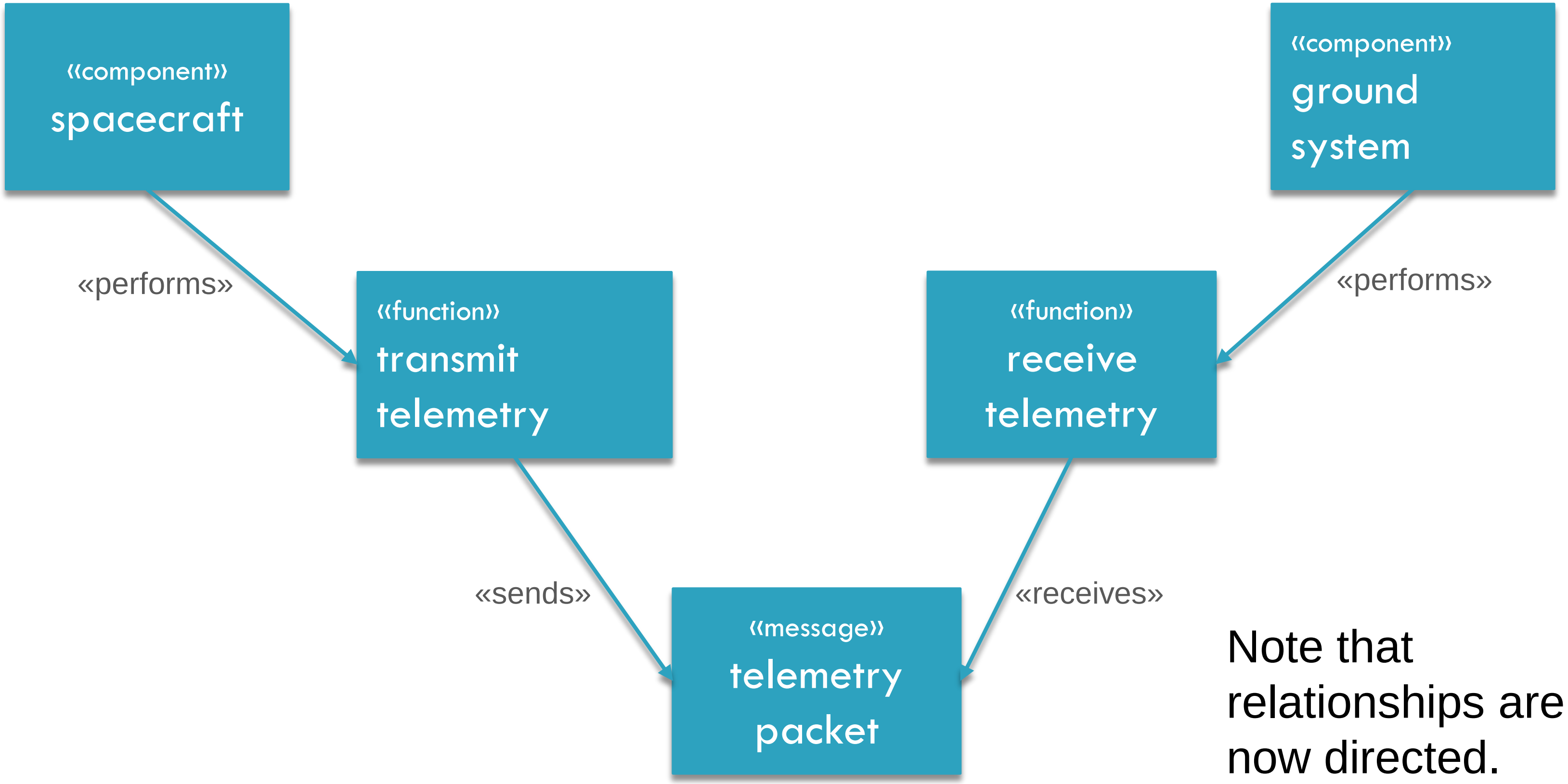


# Answering Questions (2 of 2)

What functions are present?



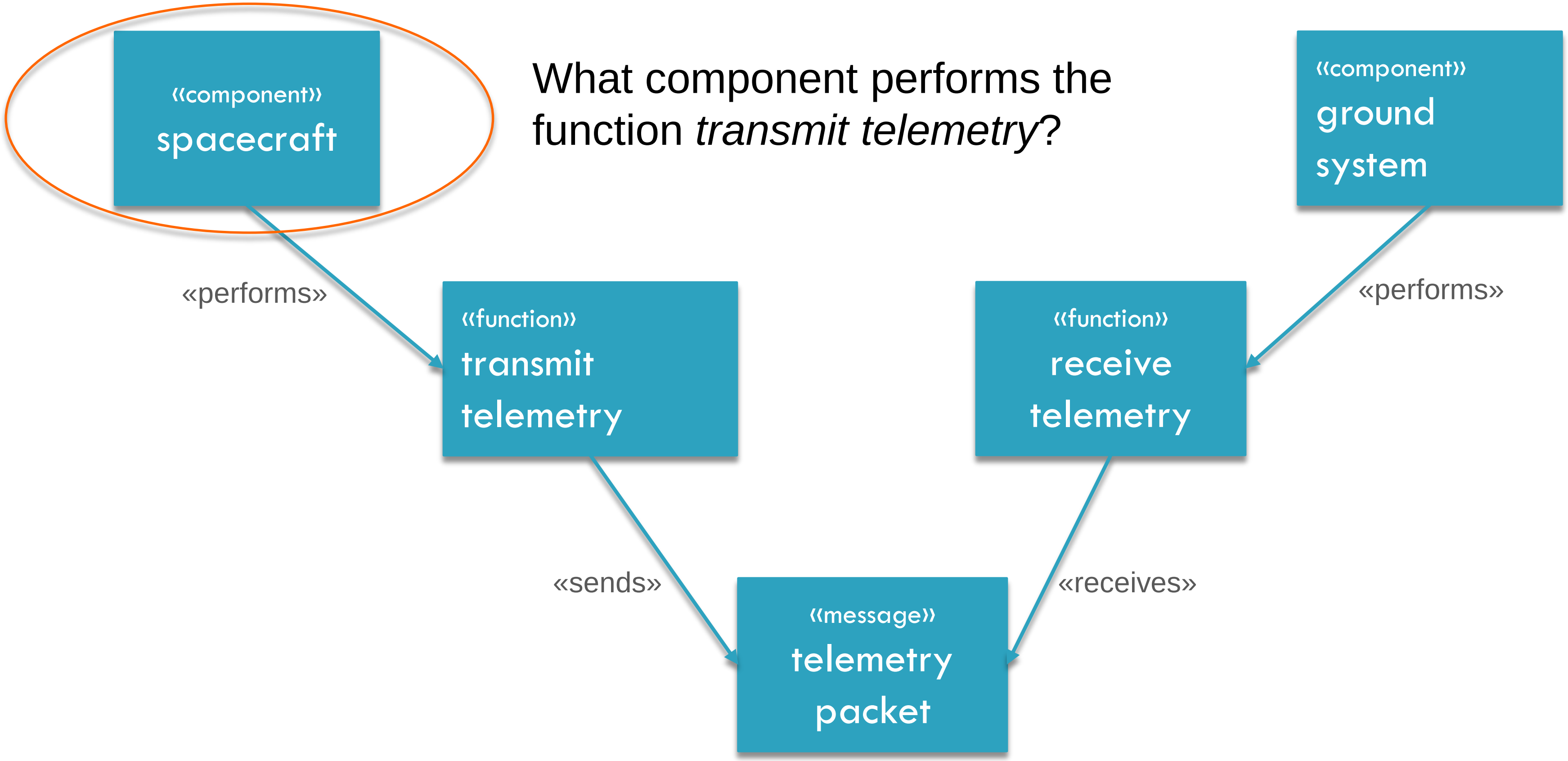
# Add Typed Relationships




Note that relationships are now directed.

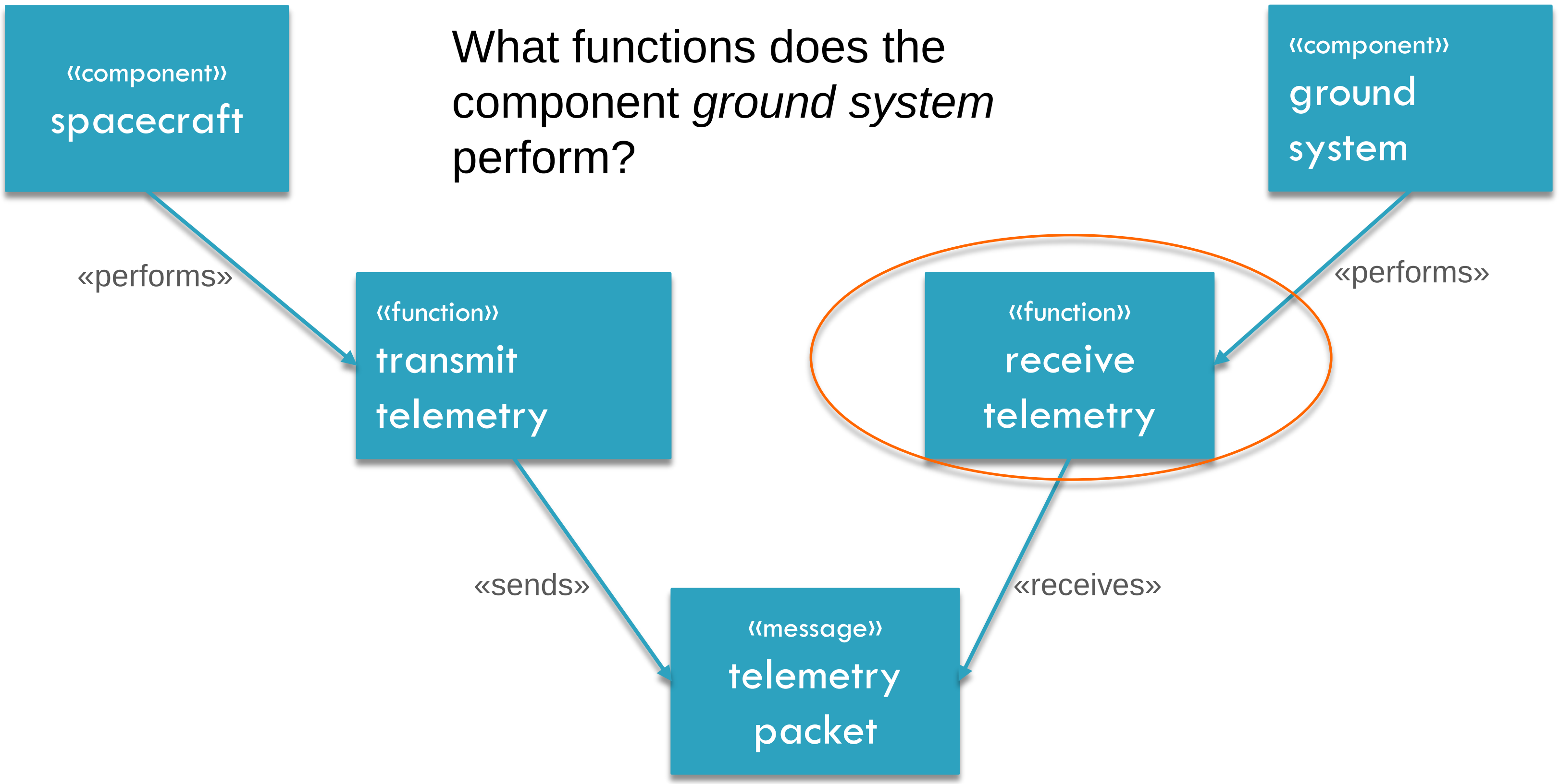
# More Questions and Answers (1 of 4)

What component performs the function *transmit telemetry*?



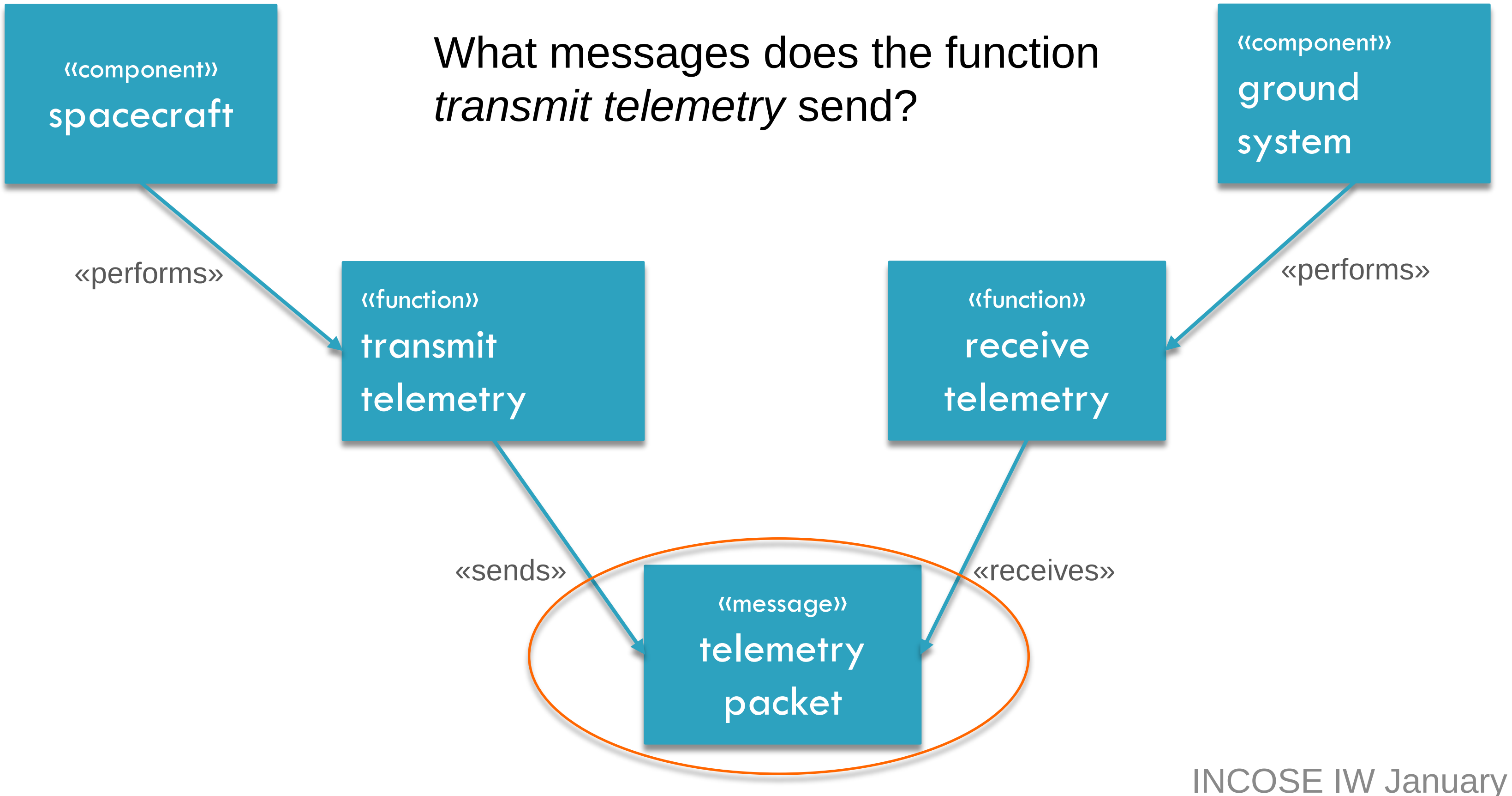
# More Questions and Answers (2 of 4)

What functions does the component *ground system* perform?



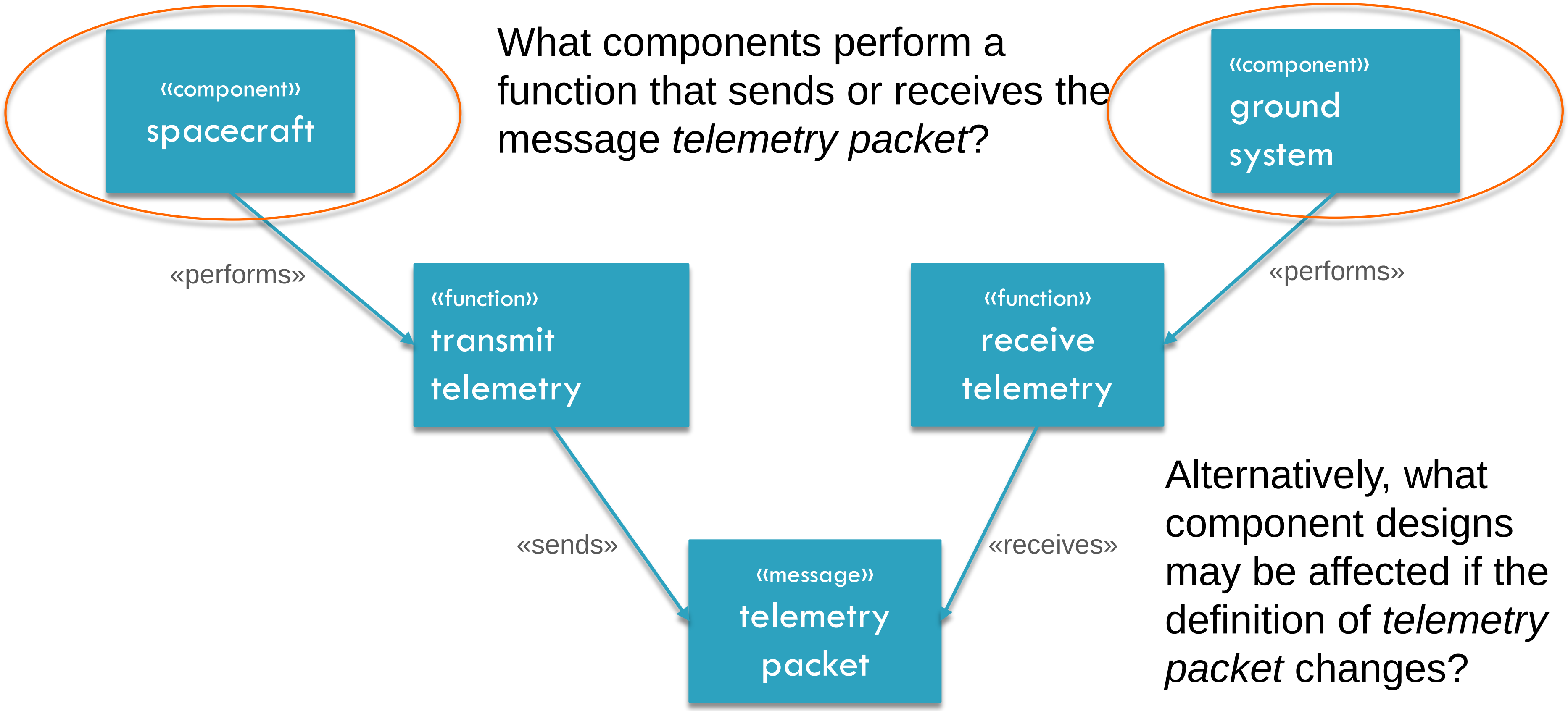
# More Questions and Answers (3 of 4)

What messages does the function *transmit telemetry* send?



# More Questions and Answers (4 of 4)

What components perform a function that sends or receives the message *telemetry packet*?




Alternatively, what component designs may be affected if the definition of *telemetry packet* changes?

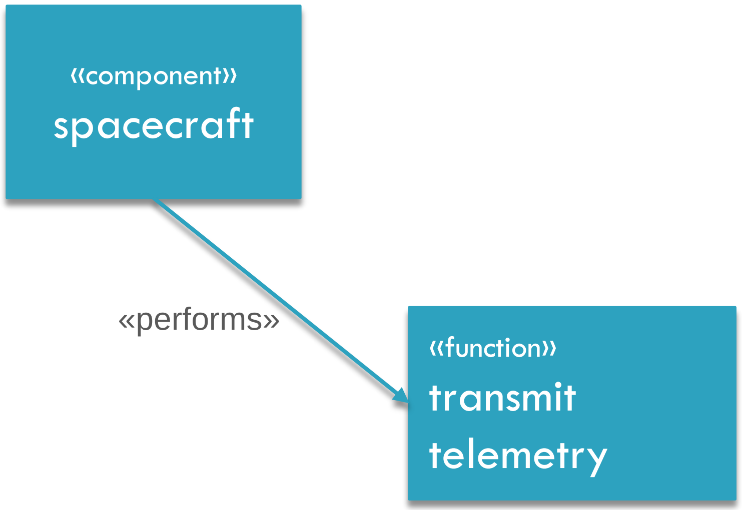
# A Good Model

- Describes a system design clearly and transparently
  - In particular, it shows a flawed design to be flawed
- Communicates a system design effectively to an incompletely bounded audience
  - In particular, it uses standard, well-defined vocabulary and notation
- Lends itself to automated reasoning for
  - validation in modeling domain (well-formedness, etc.)
  - validation in engineering domain (performance, etc.)
  - validation in programmatic domain (cost, schedule, etc.)
- Lends itself to automated transformation into
  - analysis specifications for discipline-specific tools
  - documents or other information products
- Narrows the risky and expensive gap between describing a design and analyzing it

# FAQ 12

What is an ontology

# Presentations Versus Facts

| Presentation | Facts |
|---|---|
|   | □ *spacecraft* is a ‹‹component››<br>□ *transmit telemetry* is a ‹‹function››<br>□ *spacecraft* ‹‹performs›› *transmit telemetry* |
| **SysML is (among other things) a presentation standard** | **We need other standards for our facts** |

# Facts and Ontologies

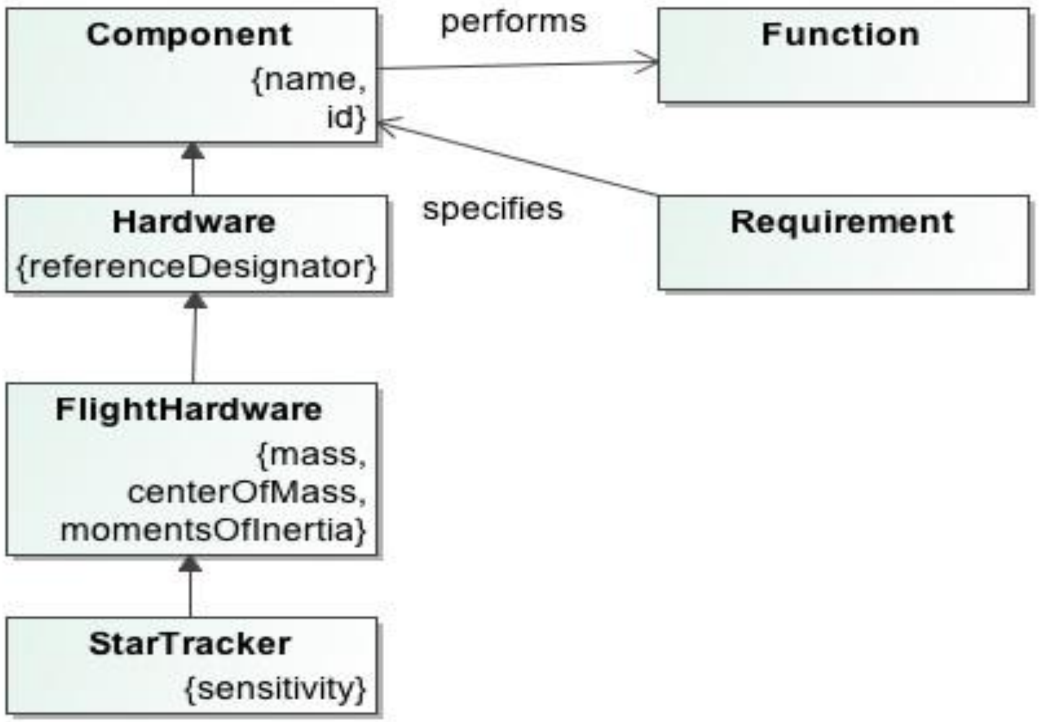
- The field that deals with facts and reasoning is *logic*
- The subset of logic that deals with facts and their meaning is *ontology*
- Ontologies contain *axioms*:
  - Definitions of concepts and their specializations
    - e.g., a *Spacecraft* is a *Flight Component*, which is a *Component*
    - These are sometimes called *classes*
  - Definitions of attributes of individuals of a class
    - e.g., *mass* is a property of *Flight Component*
    - These are sometimes called *data properties*
  - Definitions of relationships among individuals
    - e.g., a *Component performs* a *Function*
    - These are sometimes called *object properties*
  - Restrictions
    - e.g., a *Function isPerformedBy* at most one *Component*
  - Facts about individuals using these concepts and properties

# Ontology Definition

Ontologies provide descriptions of concepts and their relationships for a domain of interest

Ontologies are agreements on usage, more than a dictionary or a taxonomy

Formal ontology standards provide powerful mechanisms for automati domain specific reasoning



# Some Simple Ontology Reasoning Examples

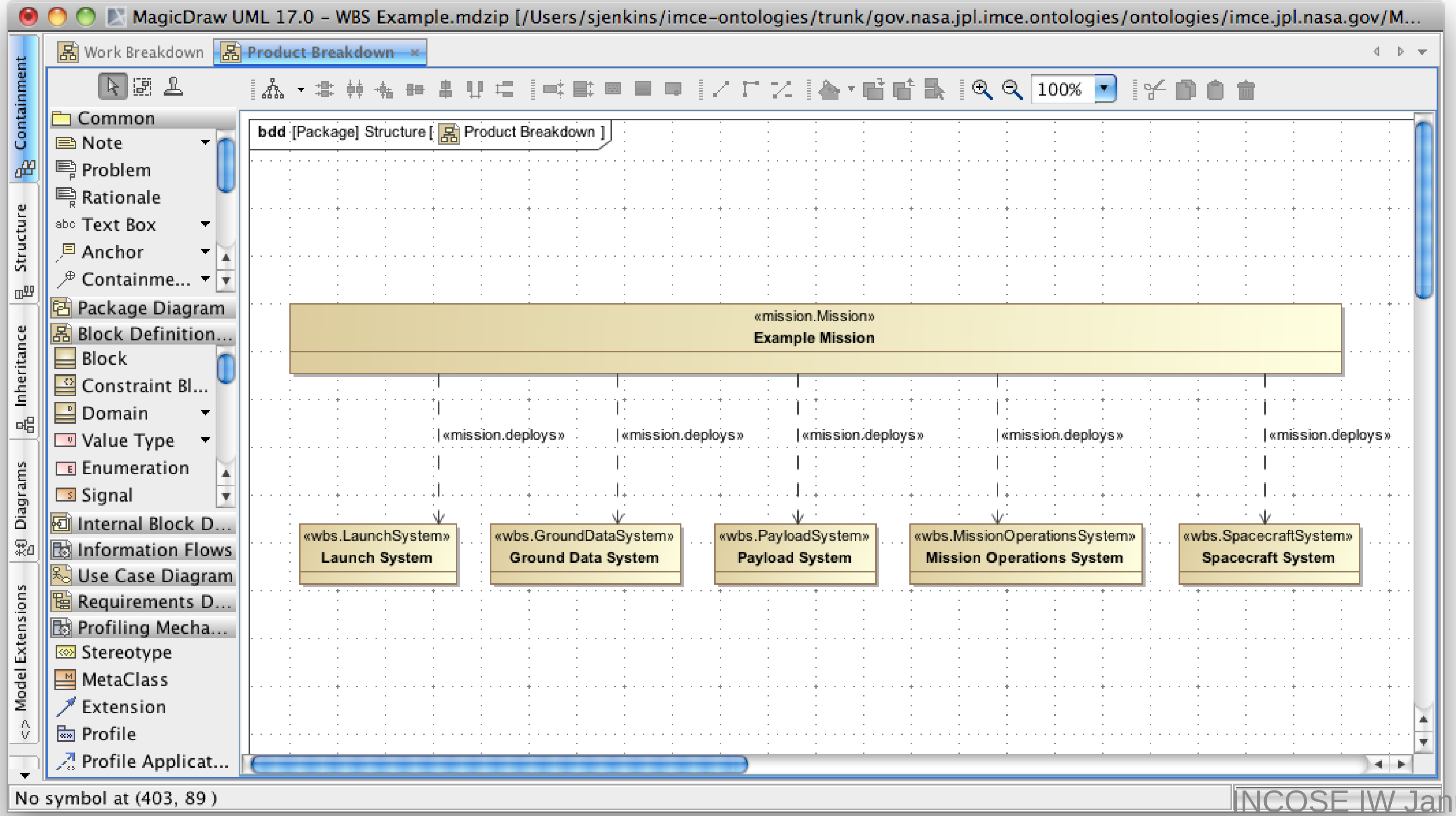
| Type | Given this input | A reasoner concludes |
| --- | --- | --- |
| **Consistency**<br>Determine whether individuals do not violate descriptions and axioms | "has mass" is a functional property. Curiosity is a HardwareComponent. Curiosity has mass 850 kg. Curiosity has mass 900 kg. | Inconsistent: at least two facts are mutually contradictory. |
| **Rules Entailment**<br>(satisfiability, subsumption) | Every Spacecraft is a Component. Every Orbiter is a Spacecraft. | Every Orbiter is a Component. (therefore, all Component rules apply to Orbiter.) |
| **Facts Entailment**<br>(satisfiability, subsumption) | Every Spacecraft is a Component. MSL Rover (an individual, not a class) is a Spacecraft. | MSL Rover is a Component. |

These examples are given in "equivalent" natural language, not OWL. The purpose is to show the kinds of problems for which reasoning is useful, not to demonstrate the mechanics.

# Ontologies as Integrating Standards

- We use a lot of discipline-specific tools and terminology in space flight systems engineering
  - e.g., trajectory synthesis, radiation effects modeling
  - SysML supports the broad discipline of systems engineering, but we need a unifying vocabulary that can relate these disciplines to each other
- This problem is not unique to space flight (nor to systems engineering)
  - Lots of people have been working on it for years.
- There is a set of international (W3C) standards for defining and using ontologies
  - All related to the Web Ontology Language (OWL)
- Can build OWL ontologies for disciplines of interest

# Example of SysML Profile Application



# FAQ 13

Why are ontologies relevant?

# Why Do We Care about Ontology?

- There is a well-developed body of theory that can
  - help us avoid undecidable questions
    - i.e., not solvable in principle
  - help us avoid intractable questions
    - i.e., solvable in principle but not in practice
- There is a body of tools that can
  - help us edit our ontologies
  - validate our ontologies
    - i.e., tell us if they're well-formed, consistent, and satisfiable
  - compute inferences
    - i.e., *Blah* is a *Spacecraft* and *Spacecraft* is a *Component* implies *Blah* is a *Component*
    - these are sometimes called *entailments*
  - answer a large class of questions about facts
    - i.e., What *Components perform* a *Function* that *sends* or *receives* the particular *Message*?

# Ontologies and SysML – which one?

- Ontology languages can be used to validate extensions to SysML to address ambiguities
- SysML and its ontology extensions can be translated to ontology languages.
  - Disambiguate semantics of the modeling language
  - Support automated checking and reasoning

Ontology languages enable **modelers** to design better and more reliable models

This doesn’t mean that the **systems engineers** need to learn ontology languages

# Appendix

# References

- SysML: Official Specification
- Books:
  - Friedenthal, Moore, Steiner, A Practical Guide to SysML, 2nd edition, Morgan Kaufmann, 2011
  - Wielkiens, Systems Engineering with SysML/UML, 1st edition, Morgan Kaufmann, 2008
- SysML Tools:
  - OMG list of vendors: http://www.omgsysml.org/#Vendors
  - SysML Forum list of vendors: http://www.sysmlforum.com/tools/
  - Recommendations for tool comparison: http://www.sysmltools.com/article/selecting-a-sysml-tool/