NASA JPL Systems Environment

Jet Propulsion Laboratory, California Institute of Technology

Marie Piette

May 2018 – No Magic World Symposium - Allen, TX USA

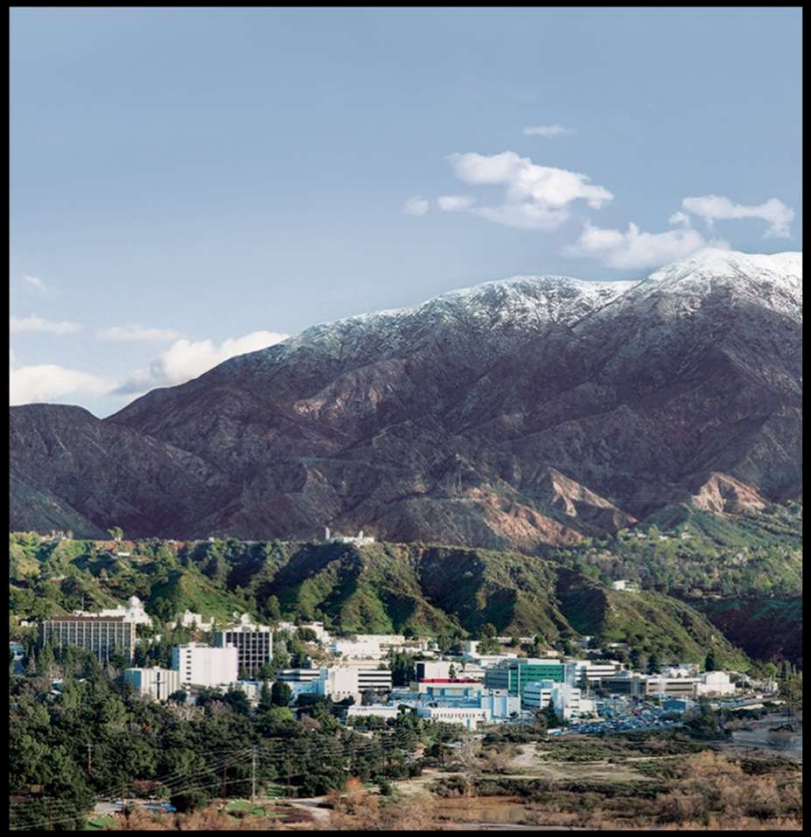
The cost information contained in this document is of a budgetary and planning nature and is intended for informational purposes only. It does not constitute a commitment on the part of JPL and/or Caltech. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

Agenda

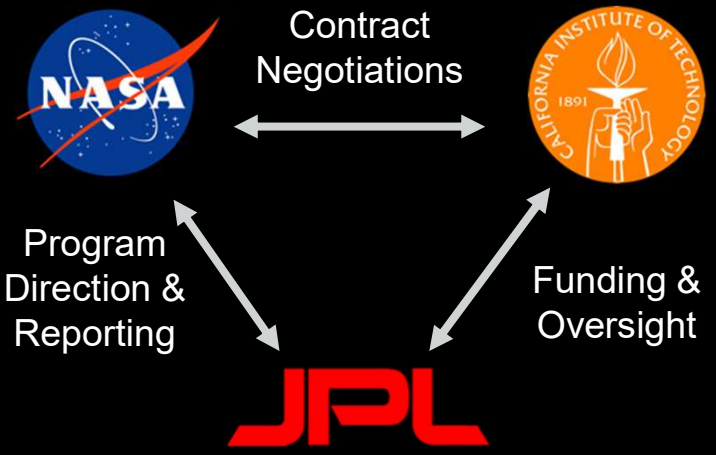
- Introduction
- OpenCAE Approach
- Open Source Contributions
- Questions

Jet Propulsion Laboratory
California Institute of Technology

NASA Jet Propulsion Laboratory (JPL)



- Located in Pasadena, CA
- NASA-owned *"Federally-Funded Research and Development Center"*
- University-operated
- ~5,000 employees



Computer Aided Engineering (CAE)

- Computer Aided Engineering provides the Laboratory's Engineering Staff and Scientific communities with tools and technical expertise
- Four Environments:
  - Systems Environment
  - Software Environment
  - Mechanical Environment
  - Electrical Environment

OpenCAE Vision

- Provide an open portfolio in a shared environment that seamlessly connects engineers developing missions and systems.
  - Open - Processes, code, apps, services and artifacts are accessible by JPL users as well as vendors and partners
  - Shared - The diverse community of users, developers partners and vendors are able to contribute
  - Connected - Collaboratively construct and analyze the same precision products needed to develop Missions and Systems at JPL using the CAE environment.

OpenCAE Mission

- Develop the CAE environment from a user centered architecture leveraging vendor partnerships using robust life cycle processes.
  - Vendor partnerships – Crucial feedback and insight into how Vendor products are serving the needs of engineers and developers
  - User centered – Architecture for CAE is driven by the needs of the practitioners and projects
  - Life-cycle process – Provide the integrity of the the applications services and support

Agenda

- Introduction
- OpenCAE Approach
- Open Source Contributions
- Questions

Jet Propulsion Laboratory
California Institute of Technology

Model-Based Engineering applied by Projects for delivering Engineering Products

Projects:

- Europa Clipper
- Europa Lander
- InSight
- Mars 2020
- Mars Sample Return (MSR)
- Thirty Meter Telescope
- Ground Data Systems
- Psyche
- MAIA

Products:

- MELs, PELs
- Resource allocation analysis
- System decomposition,
- Libraries / Reusable models

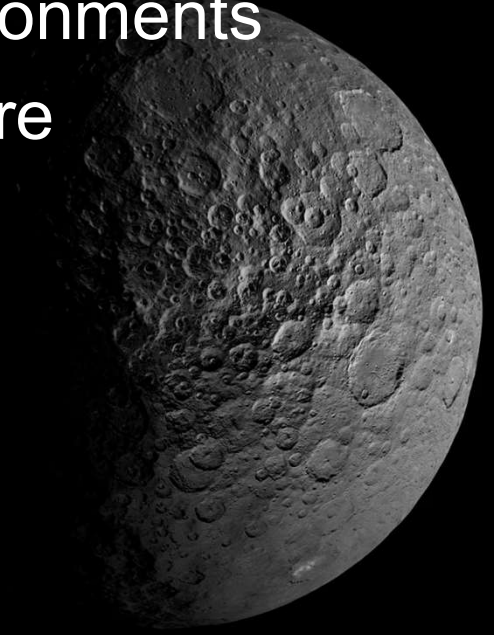
Not just spacecraft missions!
Not just early phases of design!

OpenCAE provides the engineering platform

- A platform for engineering tools to work together
- Incorporate tooling from systems, software, mechanical, and electrical domains
- Platform integrates heterogenous data sources
- Emphasize standards for data interchange
- Case studies inform the architecture of the engineering environments
- Multi-model environment

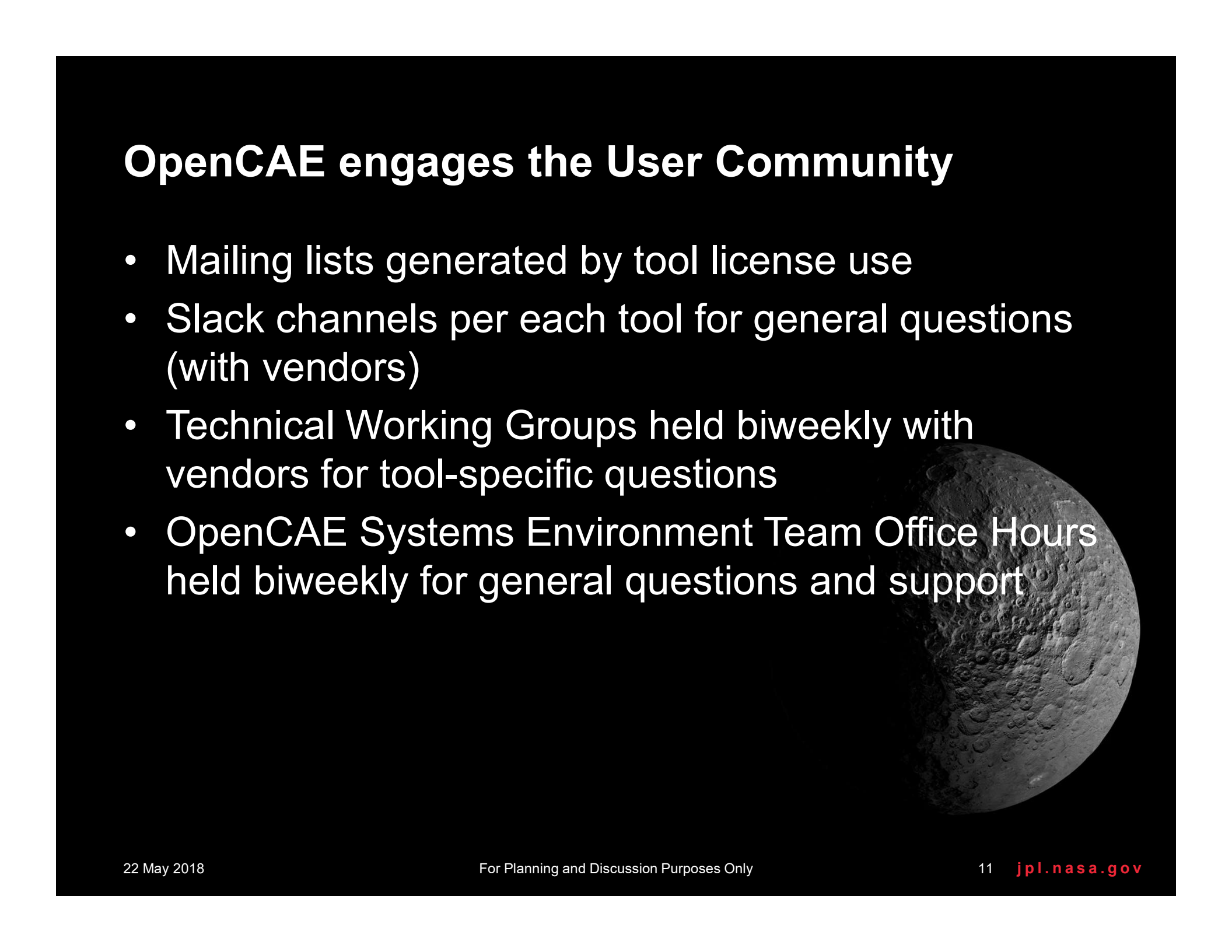
OpenCAE evolves through User Centered Design

- User Centered Design steers the development of the OpenCAE infrastructure
- Continuous communication with users to understand their experience in the OpenCAE environments
- Users evaluate solutions before they are implemented
- Following standard UX practices



OpenCAE engages the User Community

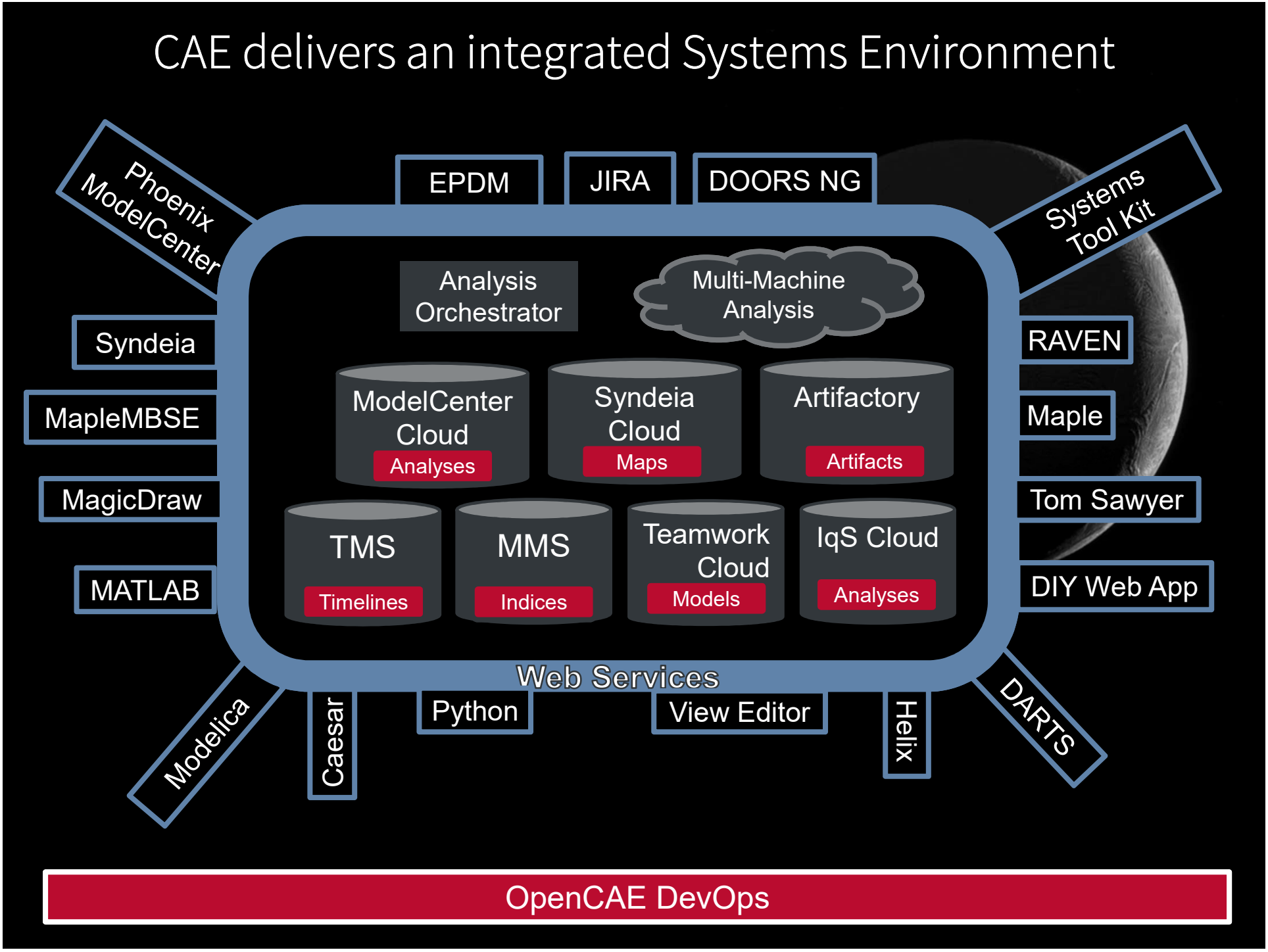
- Mailing lists generated by tool license use
- Slack channels per each tool for general questions (with vendors)
- Technical Working Groups held biweekly with vendors for tool-specific questions
- OpenCAE Systems Environment Team Office Hours held biweekly for general questions and support



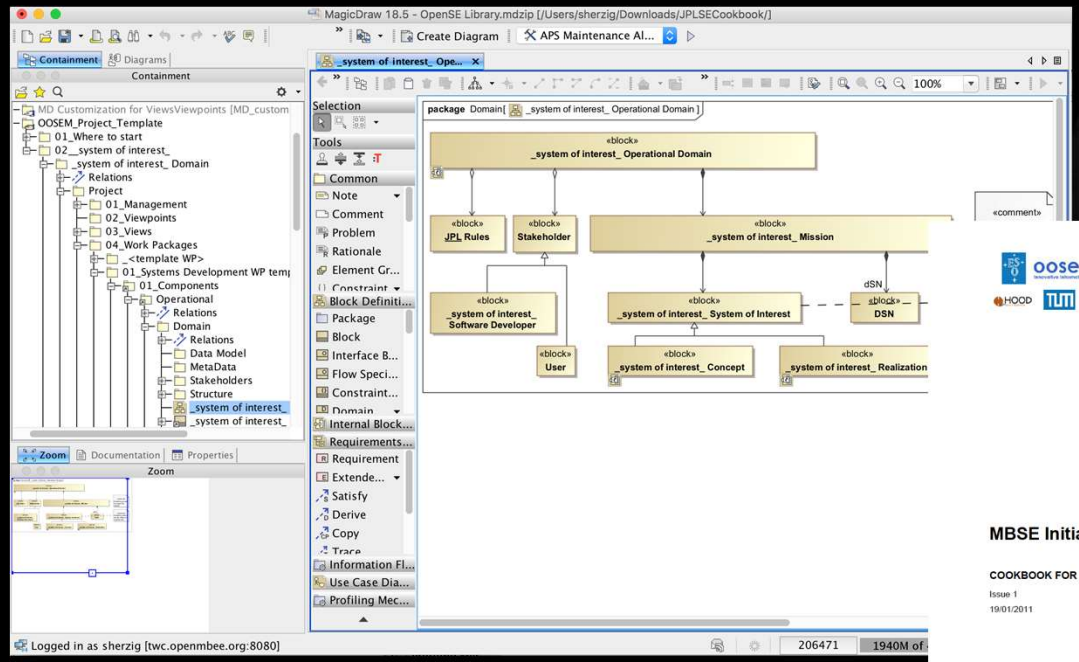
OpenCAE develops requirements through Case Studies

- Requirements Management
- Interface Management
- Design Management
- Trade Studies
- Interdisciplinary Integration
- Analysis Management
- Resource Management

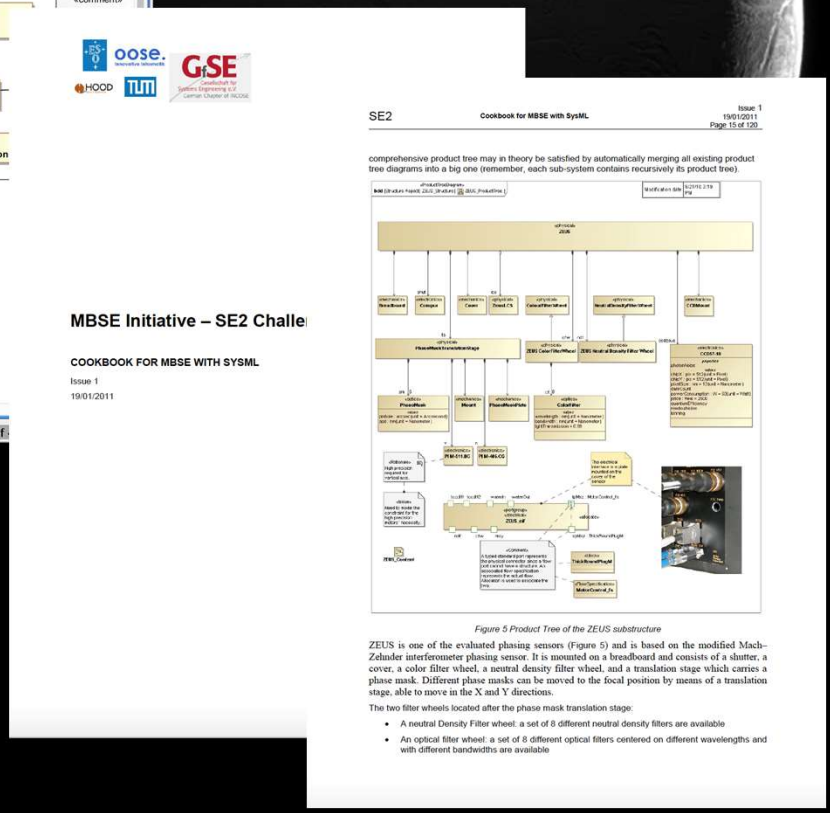
CAE delivers an integrated Systems Environment



CAE Systems Environment: Cookbook and Template Model



"Cookbook" for modeling methodology and patterns



Template models to be used by projects as a starting point, with recommended organization, model libraries, etc.

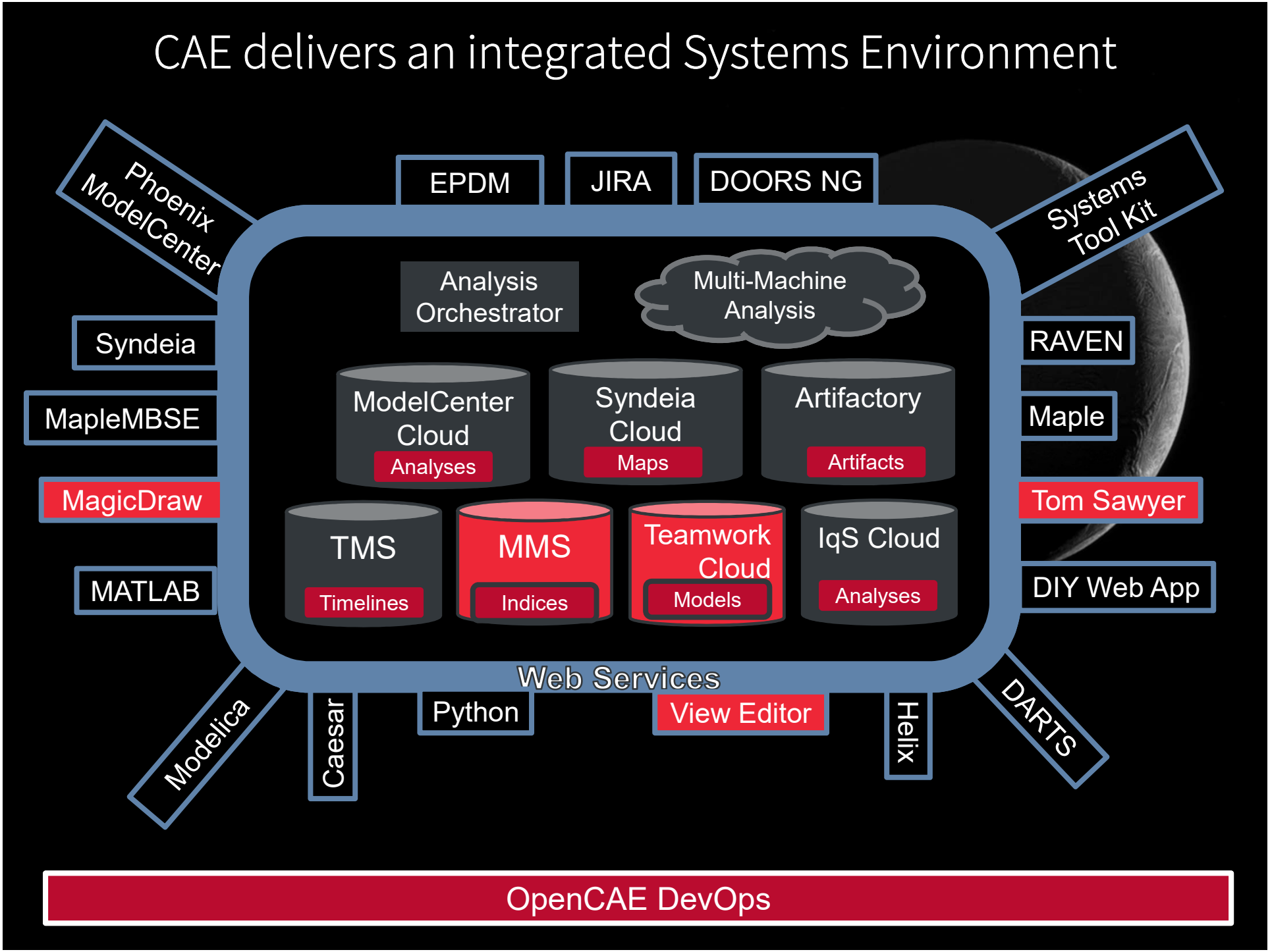
CAE offers project and domain specific adaptions

- CAE provides the same environment to all its customers (engineers and scientists)
- Embedded roles work directly on projects to adapt the standard environment specific to the project goals or methodology
- Embedded roles capture needs in general case studies which inform the CAE architecture

Mars2020 Embedded Role

- Ground Data System (GDS) Consolidation
- Flight System Model integrity
- Integration for generated artifacts
- Customized documents support
- Testing and Deploying various customizations of CAE Systems Environment

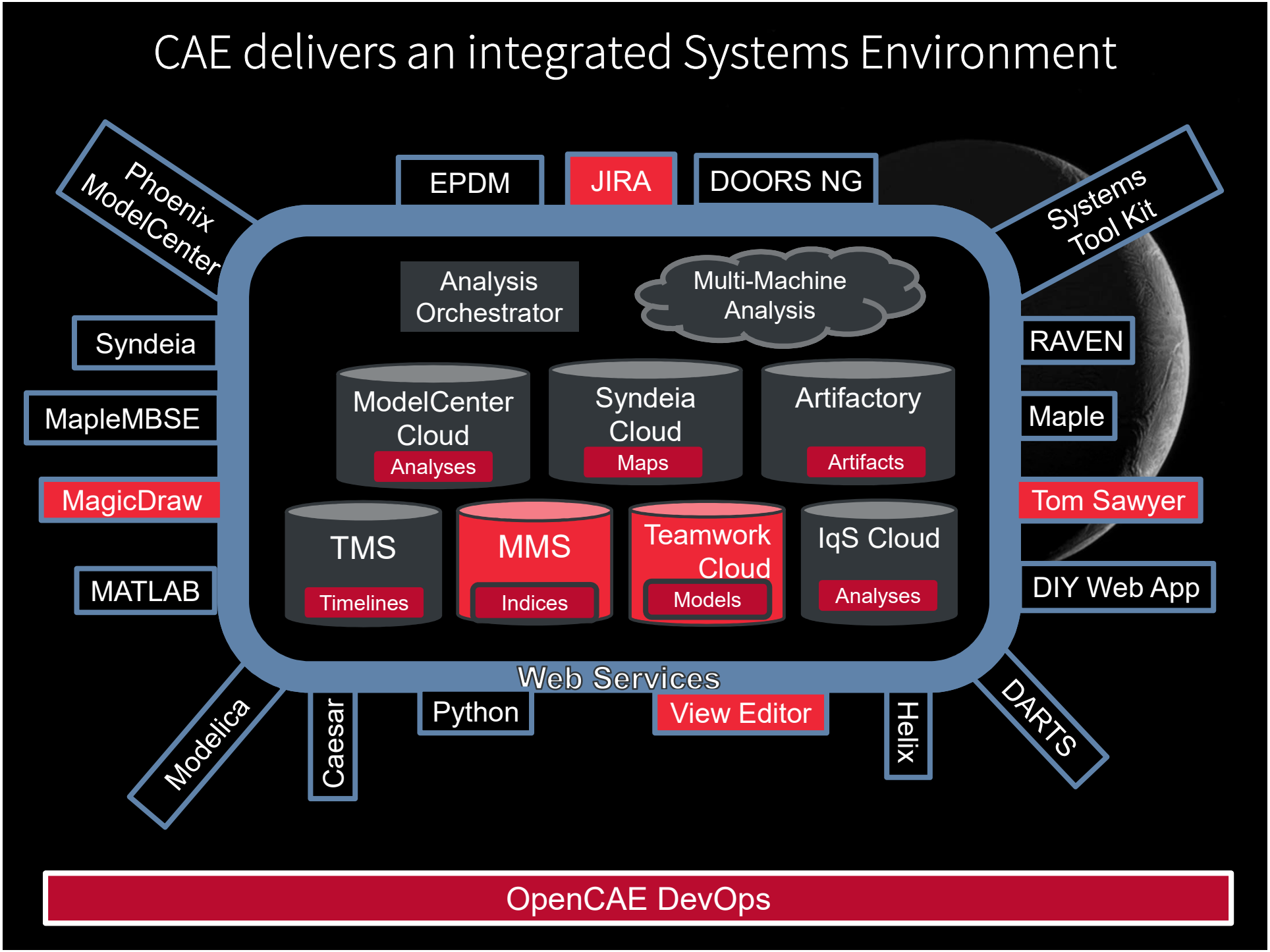
CAE delivers an integrated Systems Environment



NESC MBSE Pathfinder Embedded Role

- Apply Best Modeling Practices
- Investigate and use various workflows in environment in a restricted environment
- Specified Training and Tutorials
- Team meetings and evaluation support
- On Call Support

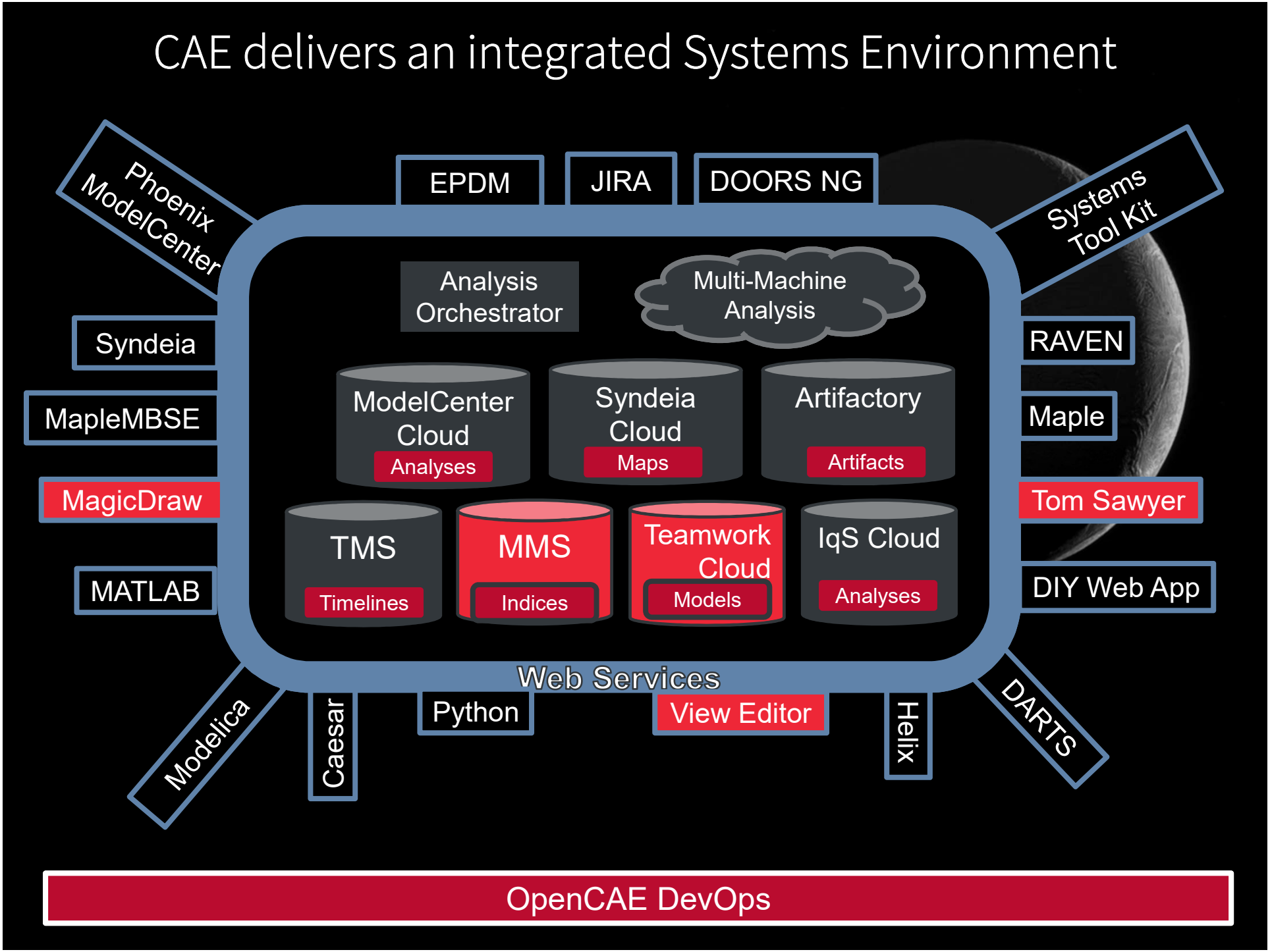
CAE delivers an integrated Systems Environment



Europa Lander Embedded Role

- Need:
  - Generate orderly and palatable diagrams from a system model describing the Lander
  - SE products should never be out of sync with the system model
- Approach:
  - Leverage Tom Sawyer plugin for MagicDraw development effort
  - Supply requirements directly from the project to the vendor
  - Coordinate with CAE development team on the use case for Tom Sawyer integration with DocGen and View Editor

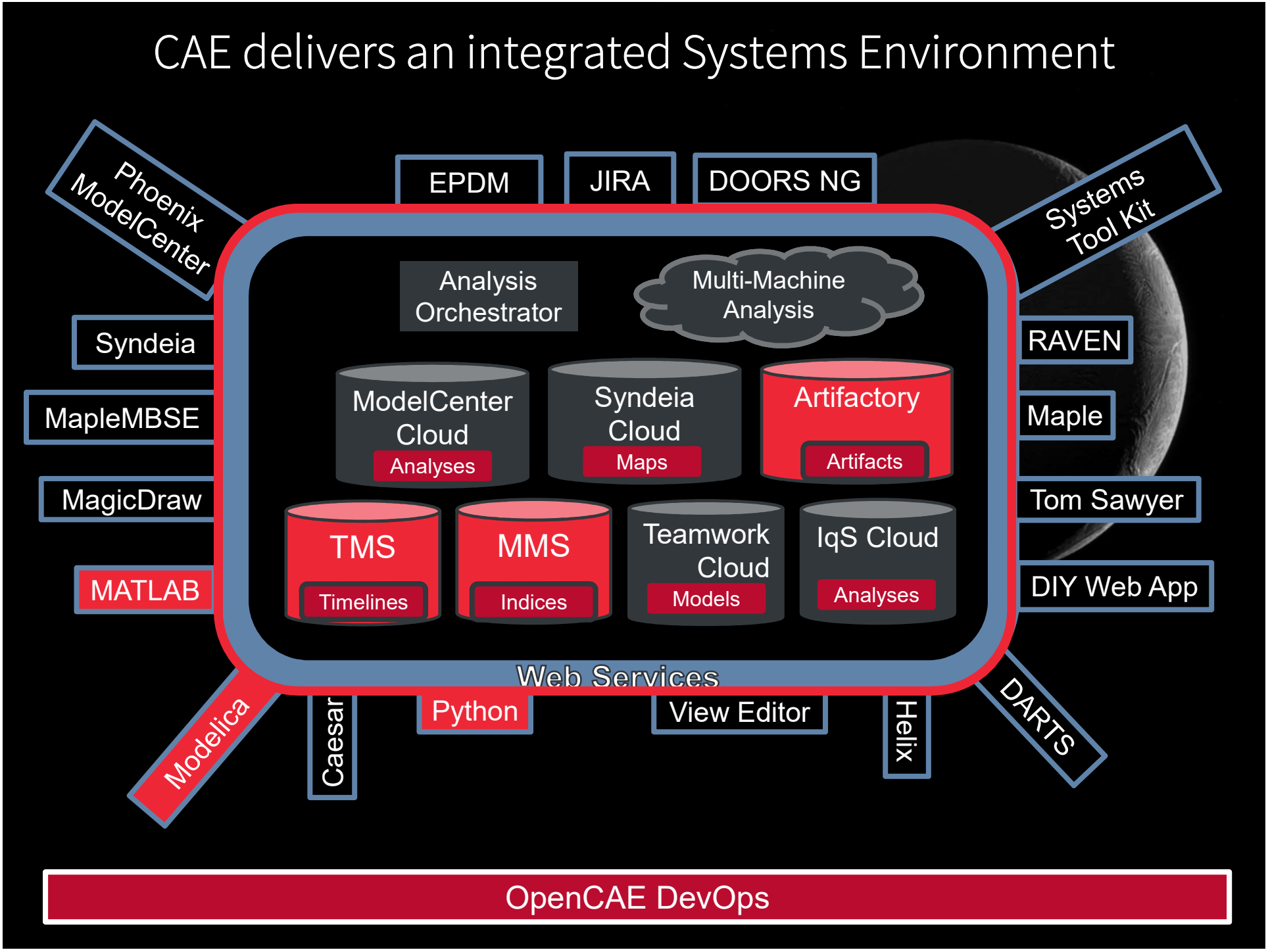
CAE delivers an integrated Systems Environment



Europa Clipper Embedded Role

- Need:
  - Publish artifacts to CAE services (MMS, TES, Artifactory)
- Approach:
  - Express the REST API endpoints of these servers in OpenAPI standard specification
  - Use Swagger codegen to generate clients for specific analysis environments
    - Mathematica, MATLAB, Python, Java
    - More than 20 other languages available

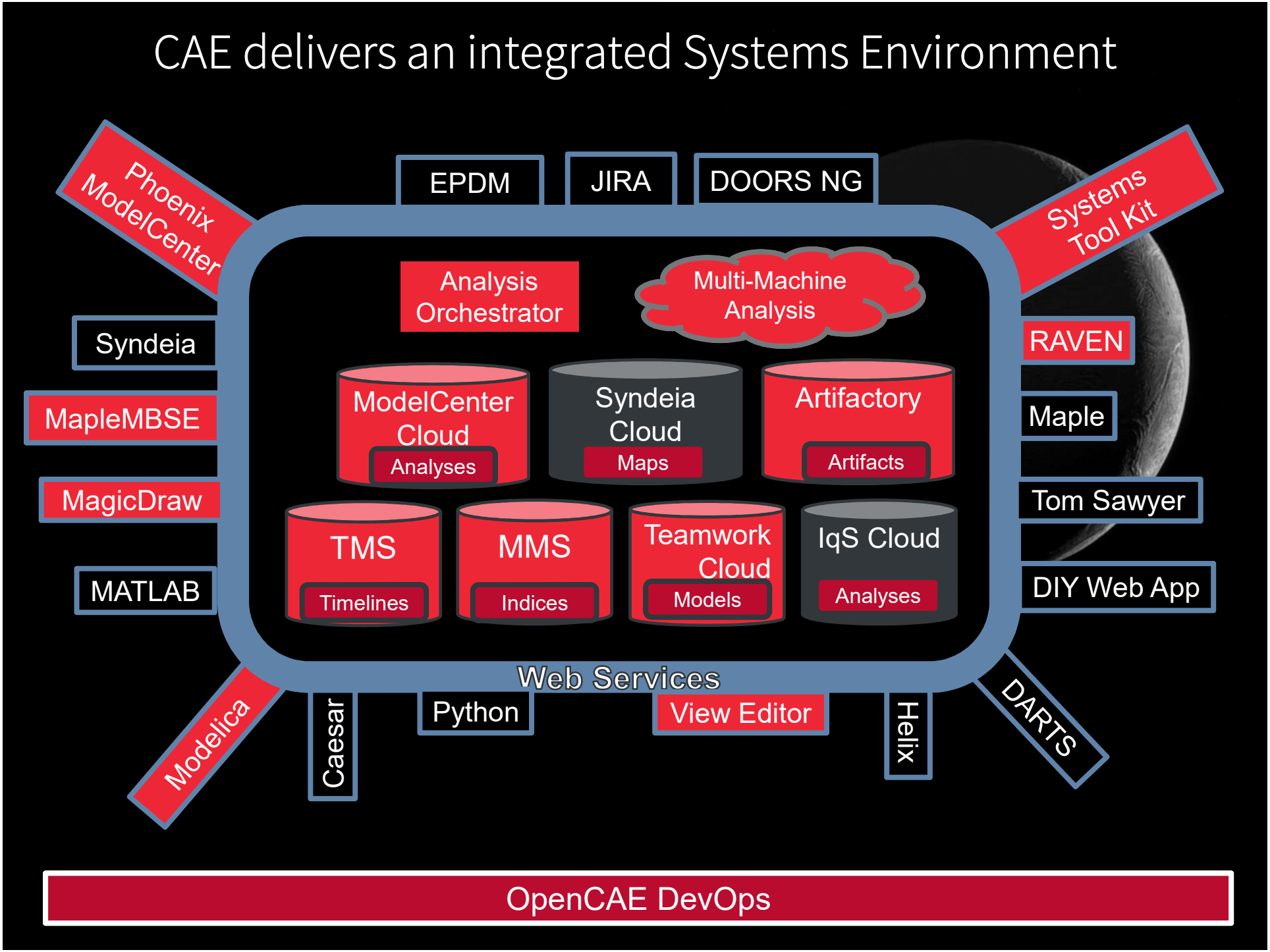
CAE delivers an integrated Systems Environment



Europa Clipper Embedded Role

- Need:
  - Formalize analysis workflows related to the Clipper Flight System
  - Want to capture the workflows in a model, but also want them to be executable
- Approach:
  - Use Phoenix MBSEPak plugin for MagicDraw to translate the workflow parameters into Phoenix ModelCenter
  - Configure ModelCenter to use shared components in the Analysis Library of ModelCenter Cloud

CAE delivers an integrated Systems Environment



OpenCAE: What Has Worked

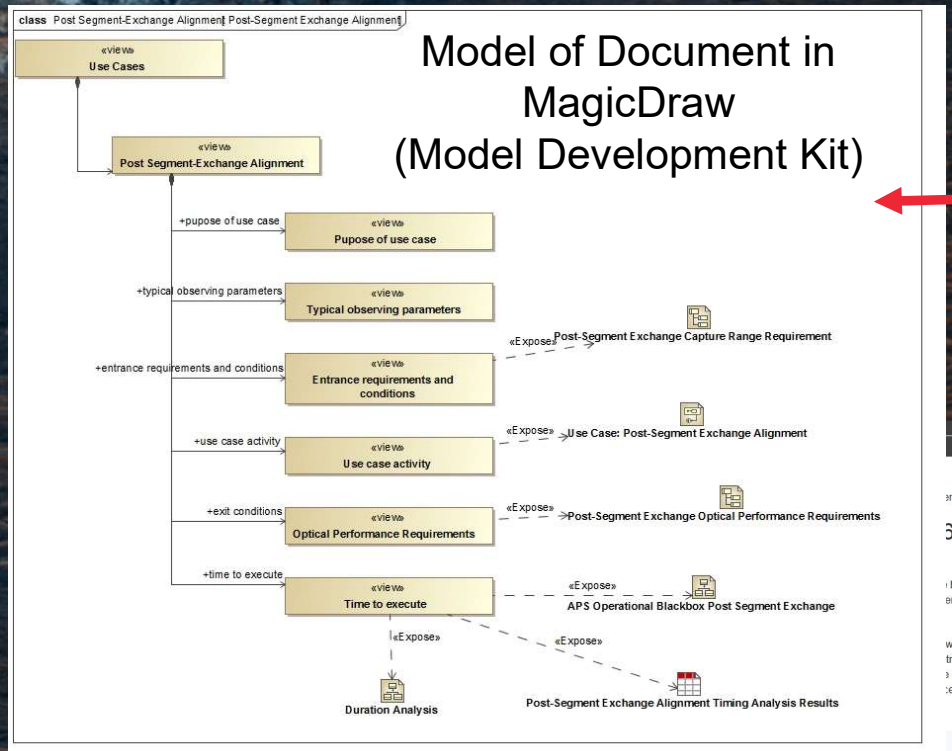
- Domain specific adaptations
  - Managed Services with vendors
  - Embedded roles
- Server-side operations preferred
  - Easier to update a server than many clients
  - COTS connections between services
- Speak the same language (SysML, FMI)
- OpenAPI REST specification
  - Generate clients for users' preferred languages
  - Enforces OpenAPI on environment services

Agenda

- Introduction
- OpenCAE Approach
- Open Source Contributions
- Questions

Jet Propulsion Laboratory
California Institute of Technology

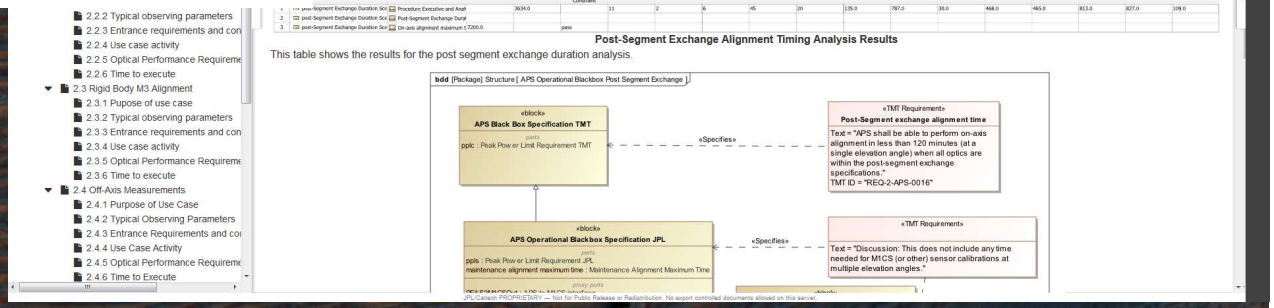
Core Integration of MMS, MDK, and VE



Model of Document in MagicDraw (Model Development Kit)

Model Repository (MMS)

Rendered and editable document in Web interface (View Editor)



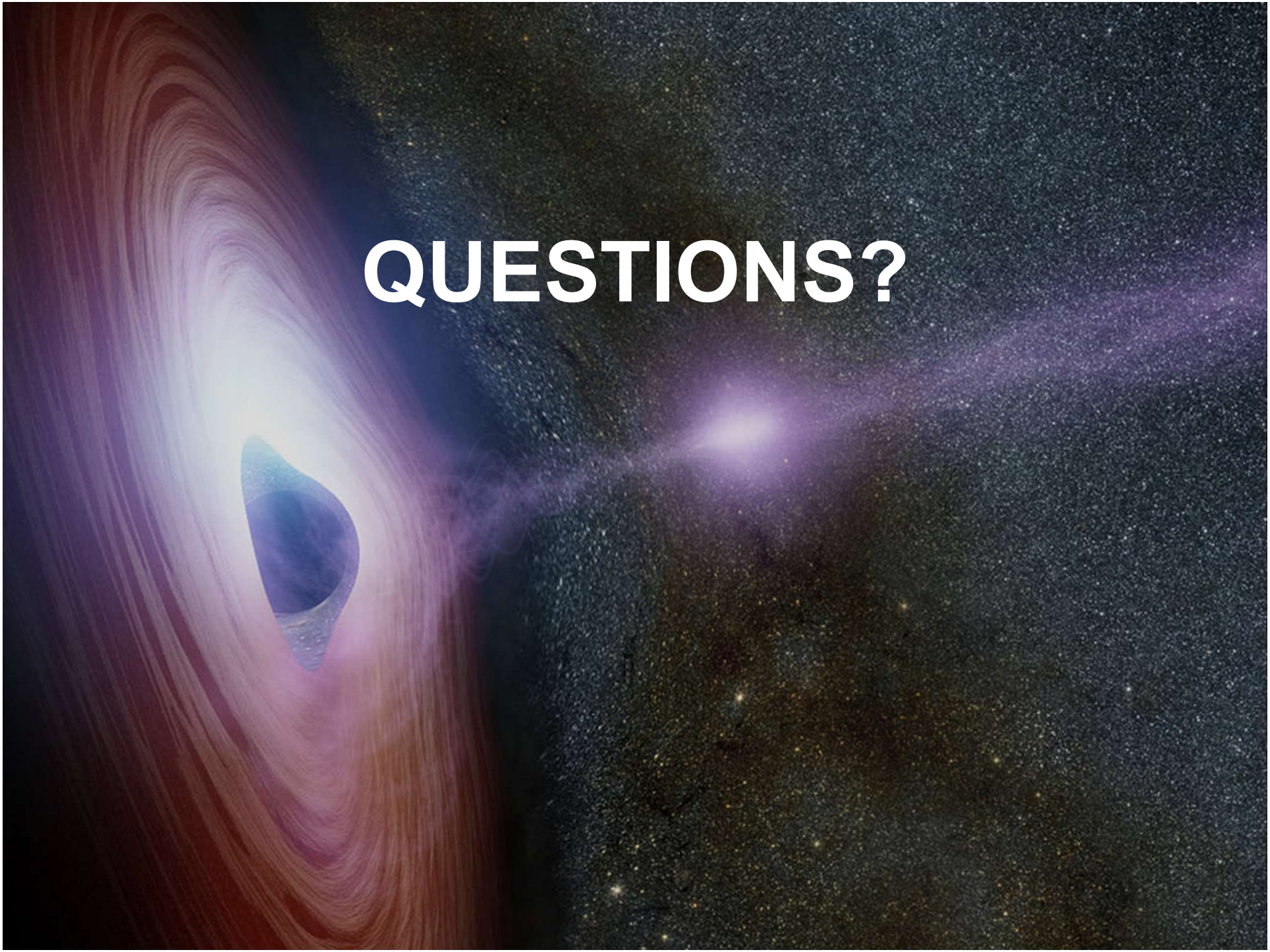
openmbee.org

Open Model Based Engineering Environment

- OpenMBEE is a community for open-source modeling software and models
  - Number of open source software activities
  - Number of open source models
- JPL is a participant and adopter of OpenMBEE software and models
- Along with Boeing, Lockheed, OMG, NavAir, Ford, Stevens, GaTech, ESO
- Vendor participants
- ~200 members

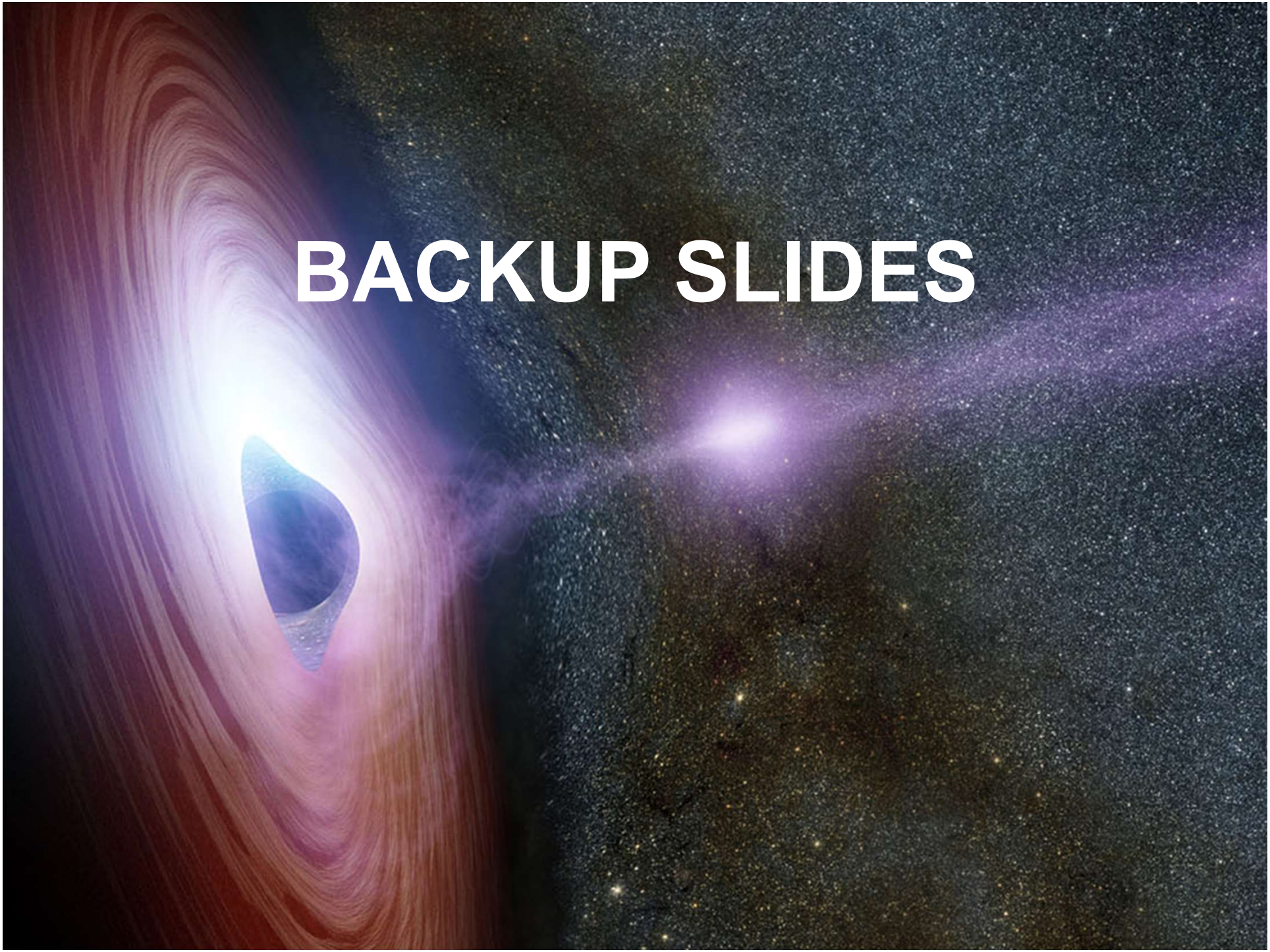
References

- Karban, R., Jankevičius, N., Elaasar, M. “ESEM: Automated Systems Analysis using Executable SysML Modeling Patterns”, (to appear in the proceedings of INCOSE International Symposium (IS), Edinburgh, Scotland, 2016.)
- Karban R., Dekens F., Herzig S., Elaasar M, Jankevičius N., “Creating systems engineering products with executable models in a model-based engineering environment”, SPIE, Edinburgh, Scotland, 2016
- Karban, R., “Using Executable SysML Models to Generate Systems Engineering Products”, NoMagic World Symposium, Allen, TX, 2016
- Open Source TMT model: https://github.com/Open-MBEE/TMT-SysML-Model
- Open Source Engineering Environment: https://open-mbee.github.io/
- Docgen, View&ViewPoints: https://github.com/Open-MBEE/mdk/tree/mdk-manual/src/main/dist/manual
- JPL Model-Based Systems Engineering Case Study: http://omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:incose_mbse_iw_2017:iw_2017_open_mbee.pdf
- A Practical Guide to SysML, 3rd Edition, Chapter 17 by Friedenthal, Moore, and Steiner
- Zwemer, D., “Connecting SysML with PLM/ALM, CAD, Simulation, Requirements, and Project Management Tools”, May 2016
- https://www.jpl.nasa.gov/spaceimages/

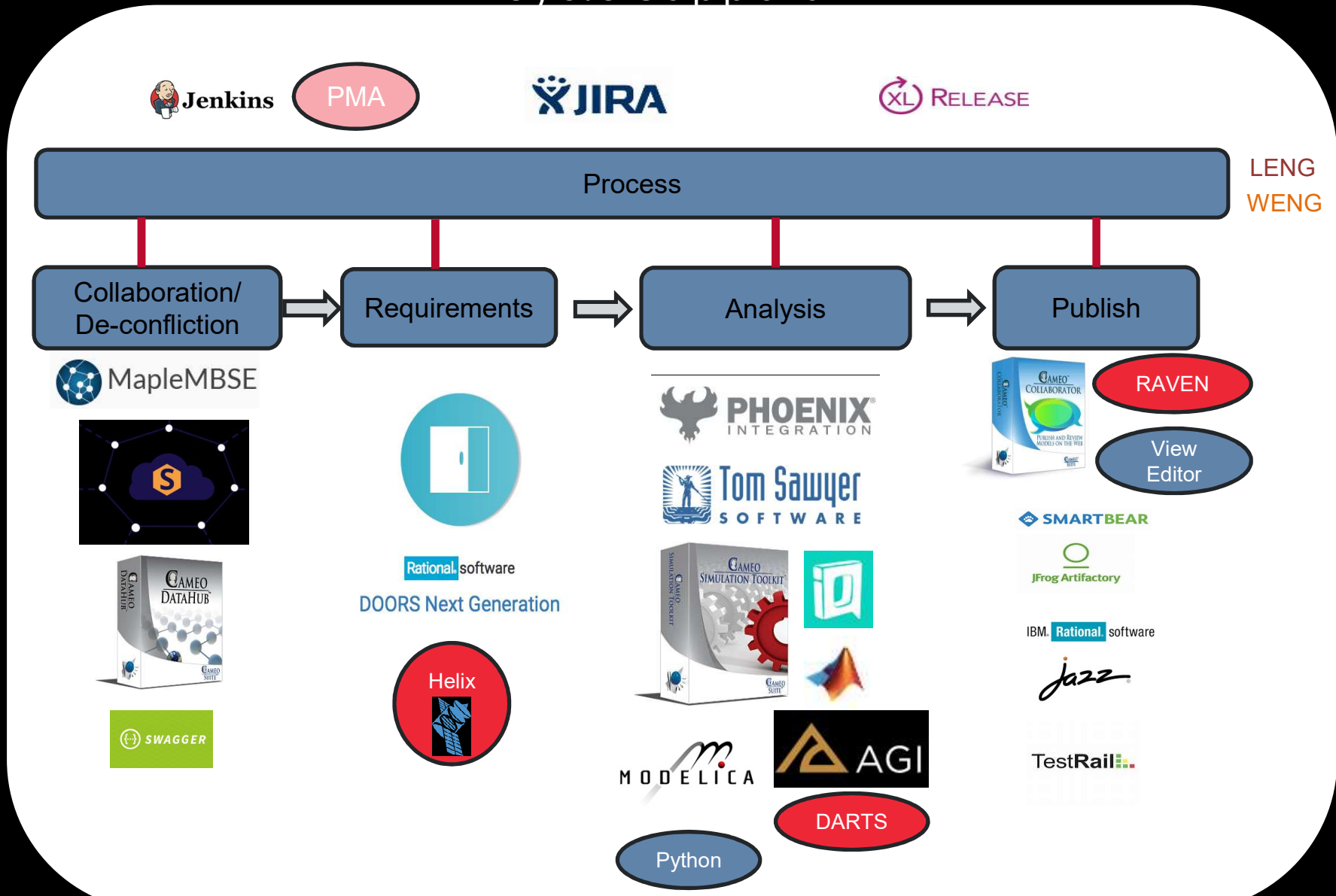

QUESTIONS?

Jet Propulsion Laboratory
California Institute of Technology

jpl.nasa.gov


BACKUP SLIDES

CAE Systems Environment Provides Integrated Life-Cycle Support



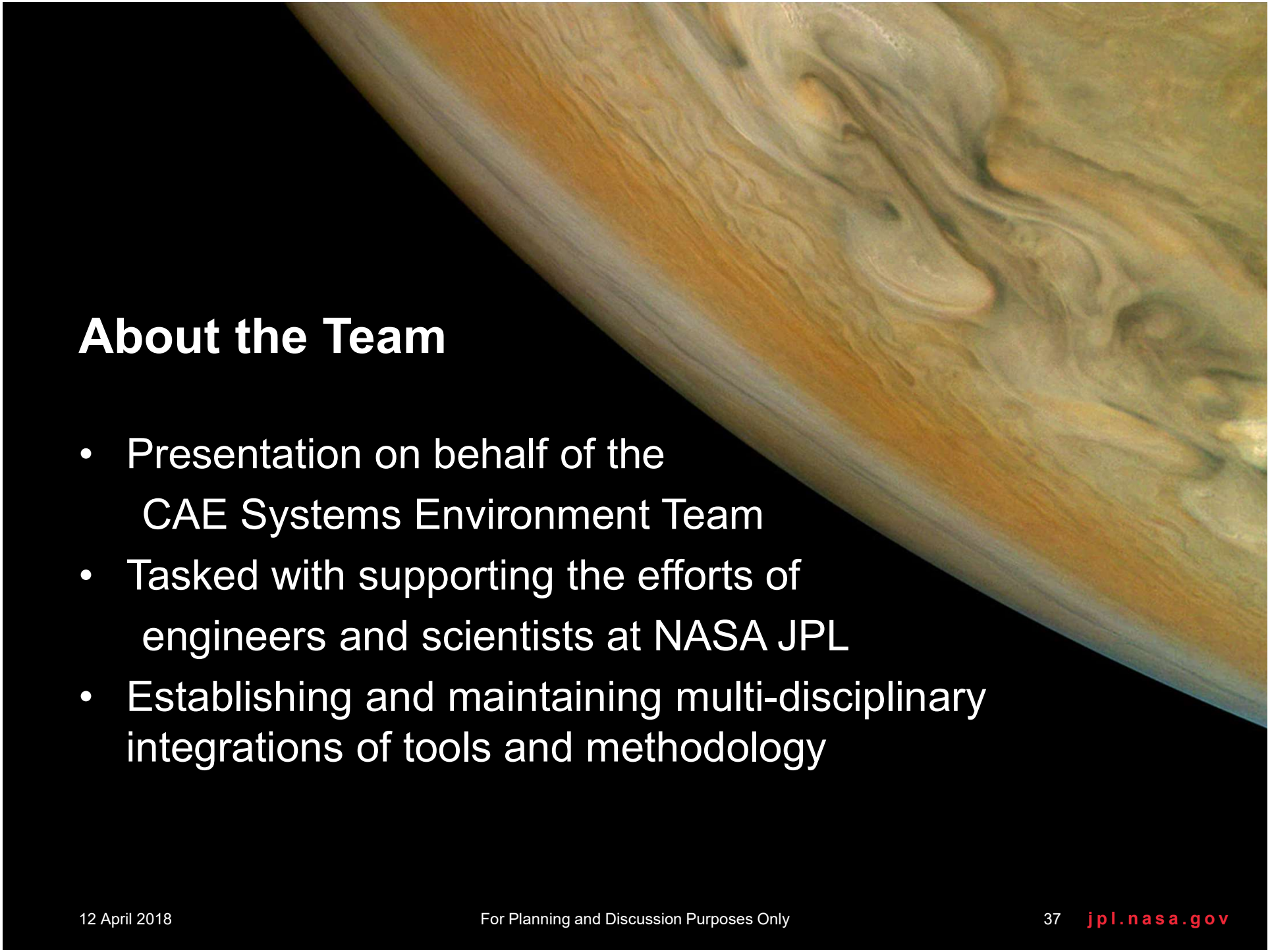
Acknowledgements

This work was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

The TMT Project gratefully acknowledges the support of the TMT collaborating institutions. They are the Association of Canadian Universities for Research in Astronomy (ACURA), the California Institute of Technology, the University of California, the National Astronomical Observatory of Japan, the National Astronomical Observatories of China and their consortium partners, and the Department of Science and Technology of India and their supported institutes. This work was supported as well by the Gordon and Betty Moore Foundation, the Canada Foundation for Innovation, the Ontario Ministry of Research and Innovation, the National Research Council of Canada, the Natural Sciences and Engineering Research Council of Canada, the British Columbia Knowledge Development Fund, the Association of Universities for Research in Astronomy (AURA) and the U.S. National Science Foundation.

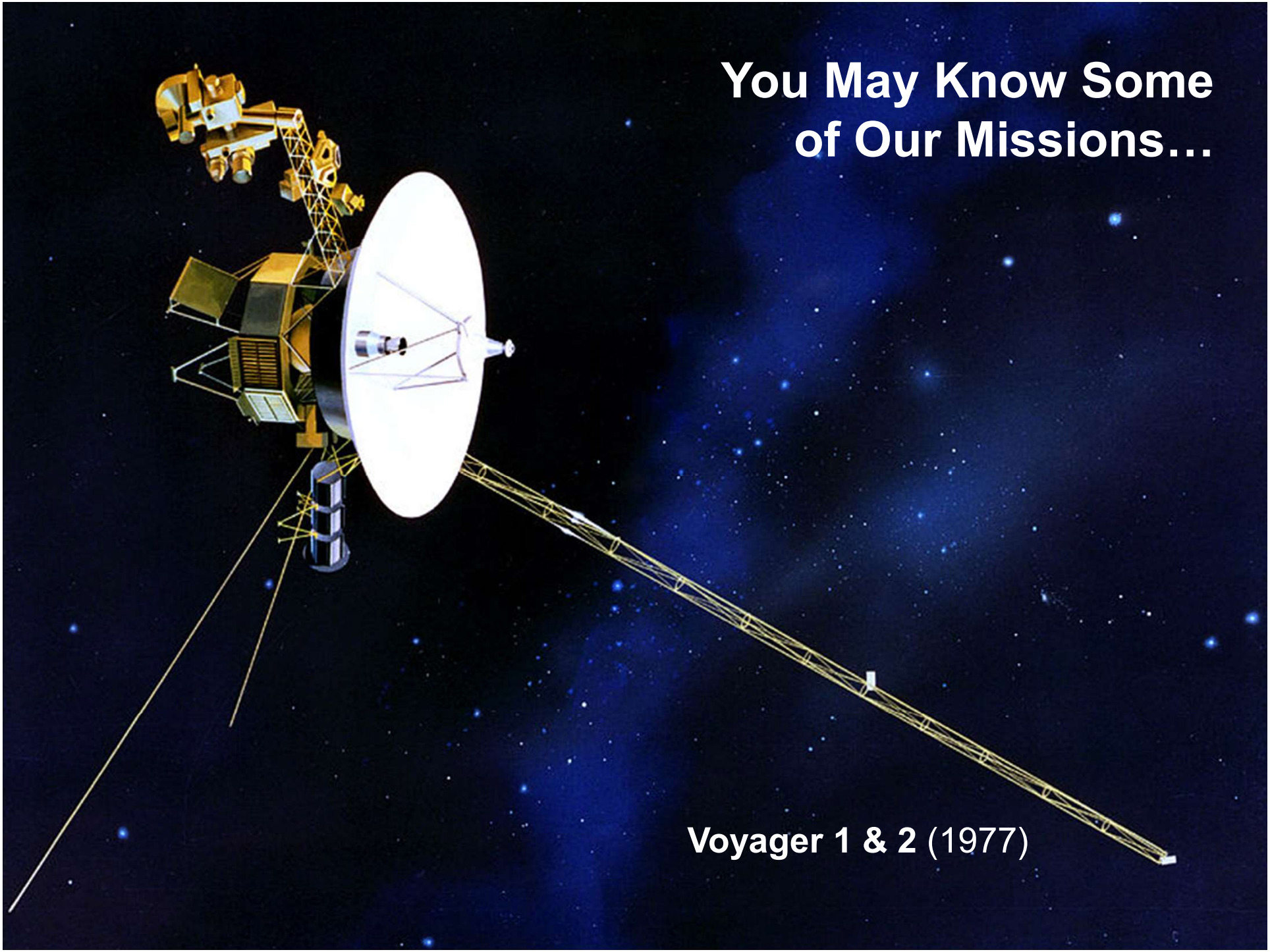
Conclusions and Summary

- JPL is successfully applying Model-Based Engineering over numerous projects
- There has been tremendous progress in tools and methodology in recent years
- The paradigm shift is manifesting in a vibrant open-source community of practitioners from around the world



About the Team

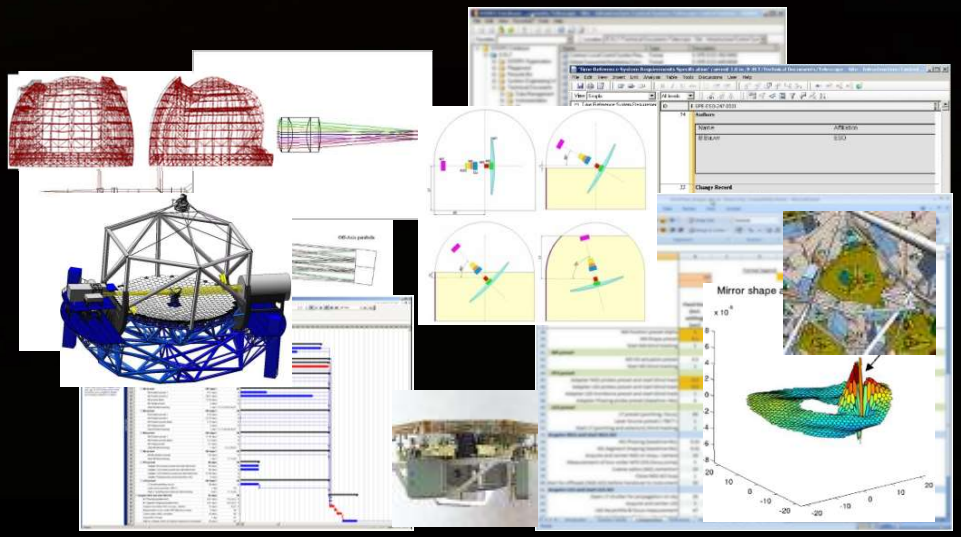
- Presentation on behalf of the CAE Systems Environment Team
- Tasked with supporting the efforts of engineers and scientists at NASA JPL
- Establishing and maintaining multi-disciplinary integrations of tools and methodology


You May Know Some
of Our Missions…
Voyager 1 & 2 (1977)

JPL's Mission is Robotic Exploration

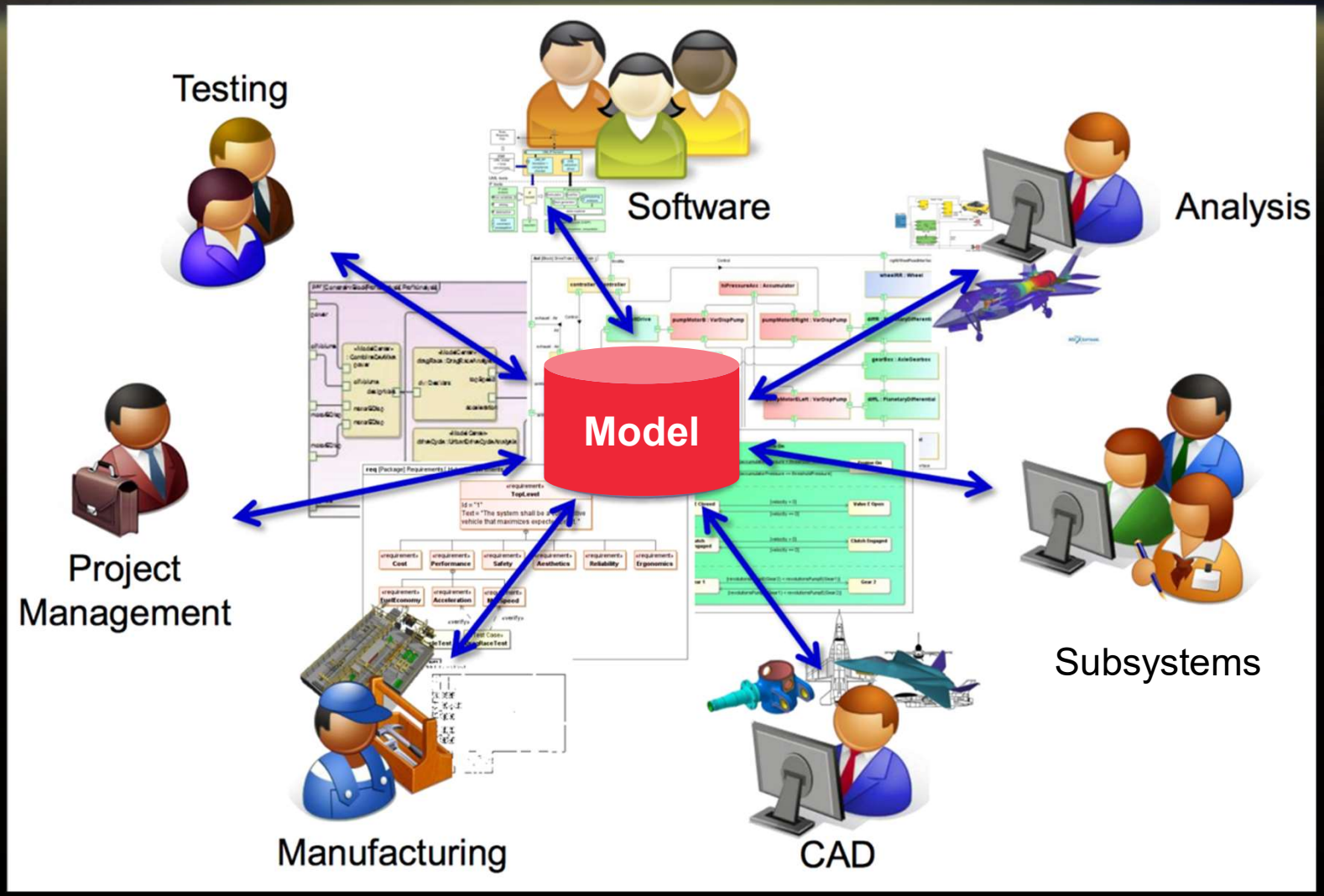
CAE Problem Statement

- Current State of Practice
  - Dispersed domain specific modeling (CAD, FEA, MATLAB)
  - Document-based artifacts related to models, but not connected
- Need for a Model-Based Engineering Environment
  - Tie system level models into existing models and modeling tools
  - Provide methods and tooling environment to support the effort



Systems Environment: Model-Based Approach

Information Management Across All Disciplines and the Life Cycle



Systems Engineering: Executable Approach

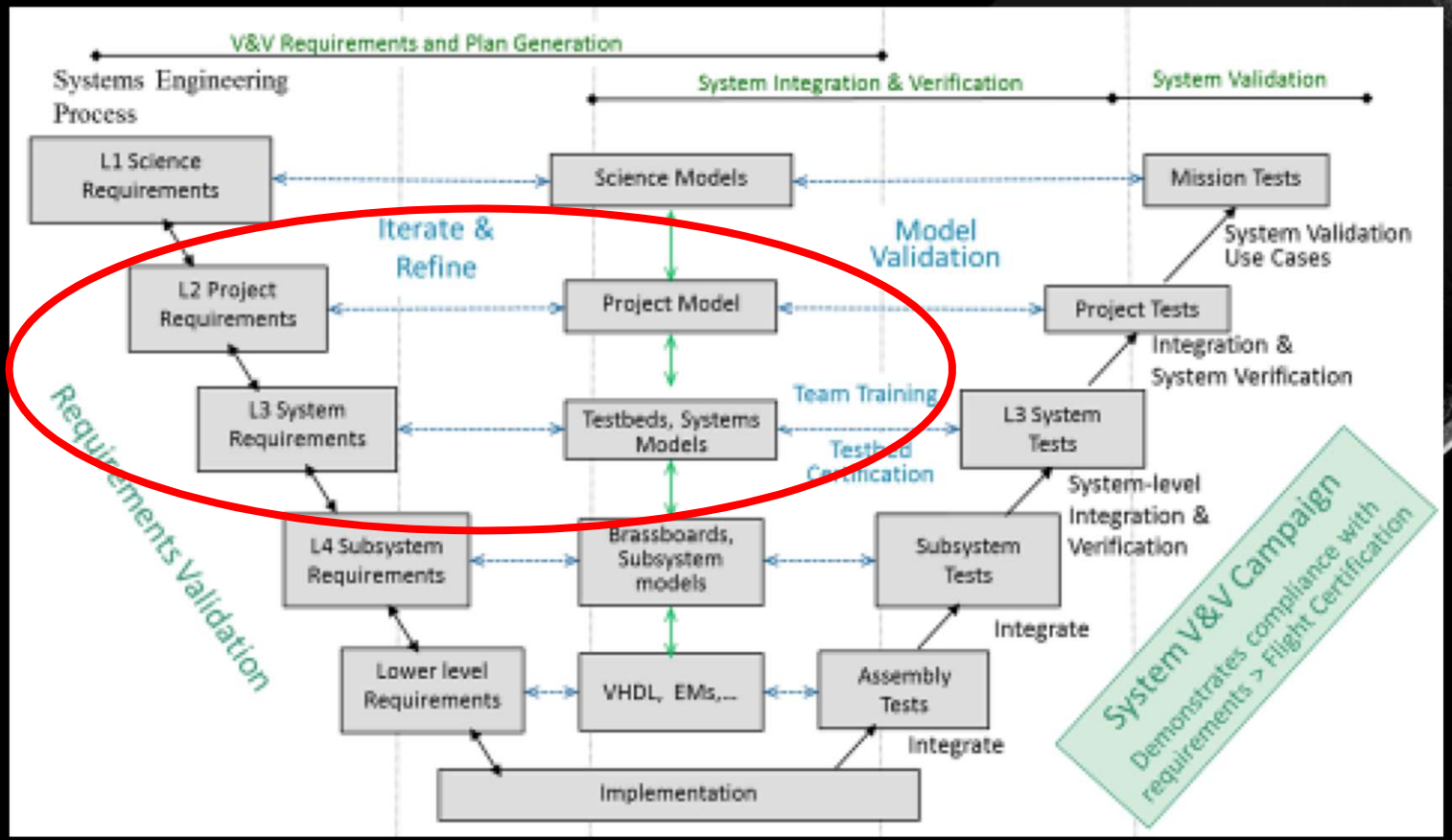
- Next phase of modeling emphasizes executable models to enhance understanding, precision, and verification of requirements
- Executable Systems Engineering Method (**ESEM**) augments the OOSEM activities by enabling executable models
  - ESEM defines executable SysML models that verify requirements
  - Includes a set of analysis patterns that are specified with various SysML structural, behavioral and parametric diagrams
  - Also enables integration of supplier/customer models and analysis

Agenda

- Introduction
- Current State
- OpenCAE Approach
- Open Source Contributions
- Example Application at JPL
- Conclusions and Summary

Jet Propulsion Laboratory
California Institute of Technology

Scope of the CAE Systems Environment



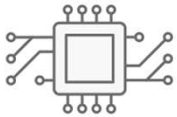
CAE Systems Environment

Case Studies	Realization
Systems Design Management	MagicDraw, View Editor, MapleMBSE
Systems Resource Management	Phoenix ModelCenter, Cameo Simulation Toolkit, Systems Tool Kit
Interdisciplinary Integration	Syndeia, Cameo Datahub
Viewing and Reporting	Tom Sawyer, View Editor
Systems Analysis Management	Phoenix ModelCenter, Platform for Modeling Analysis (PMA)

OpenCAE Environments and Technology Portfolio

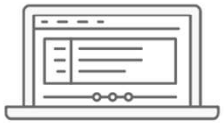
CAE Disciplines

Collections of tools and resources for engineering disciplines

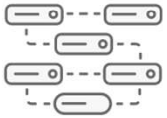


Electrical

Mechanical

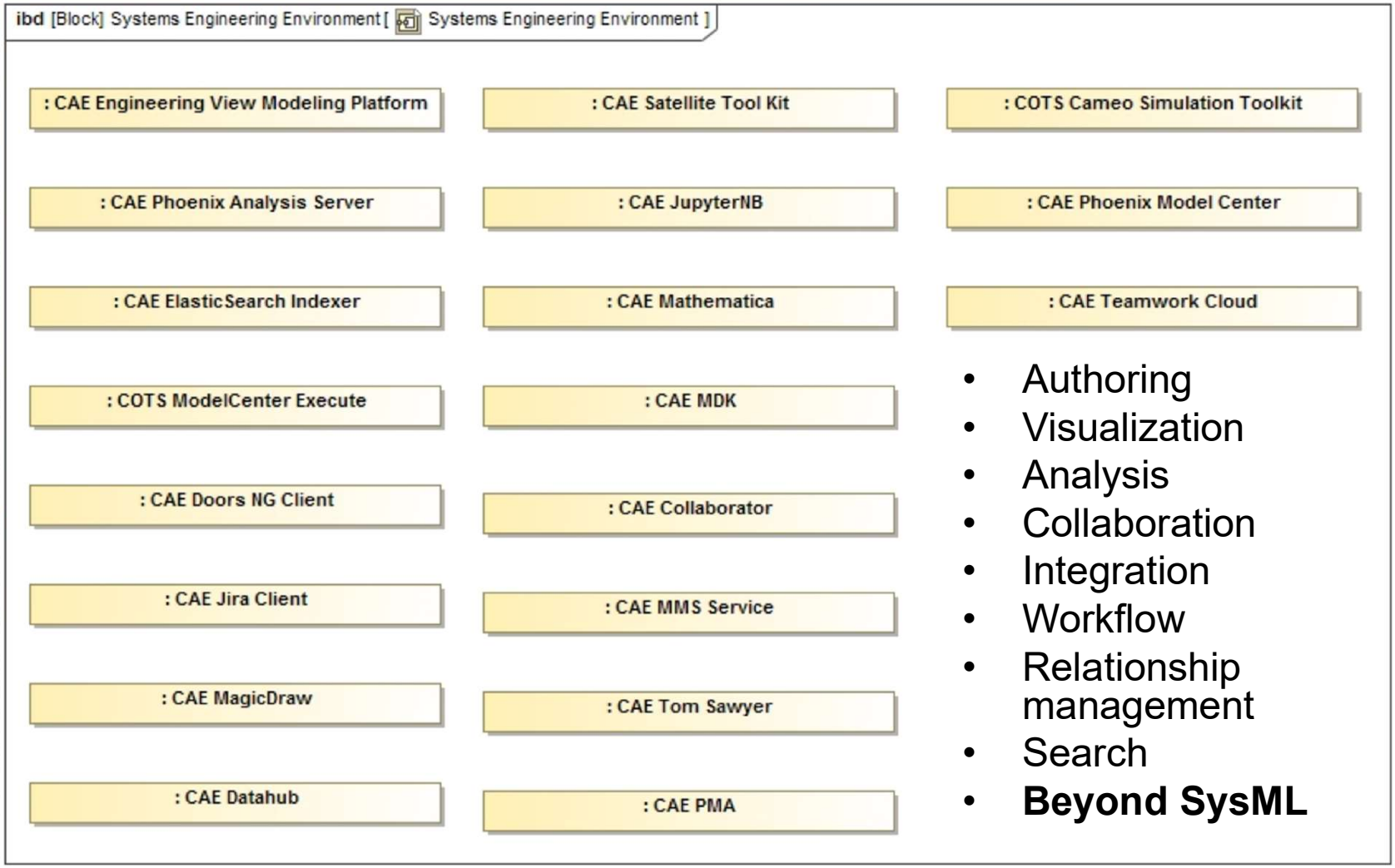


Software



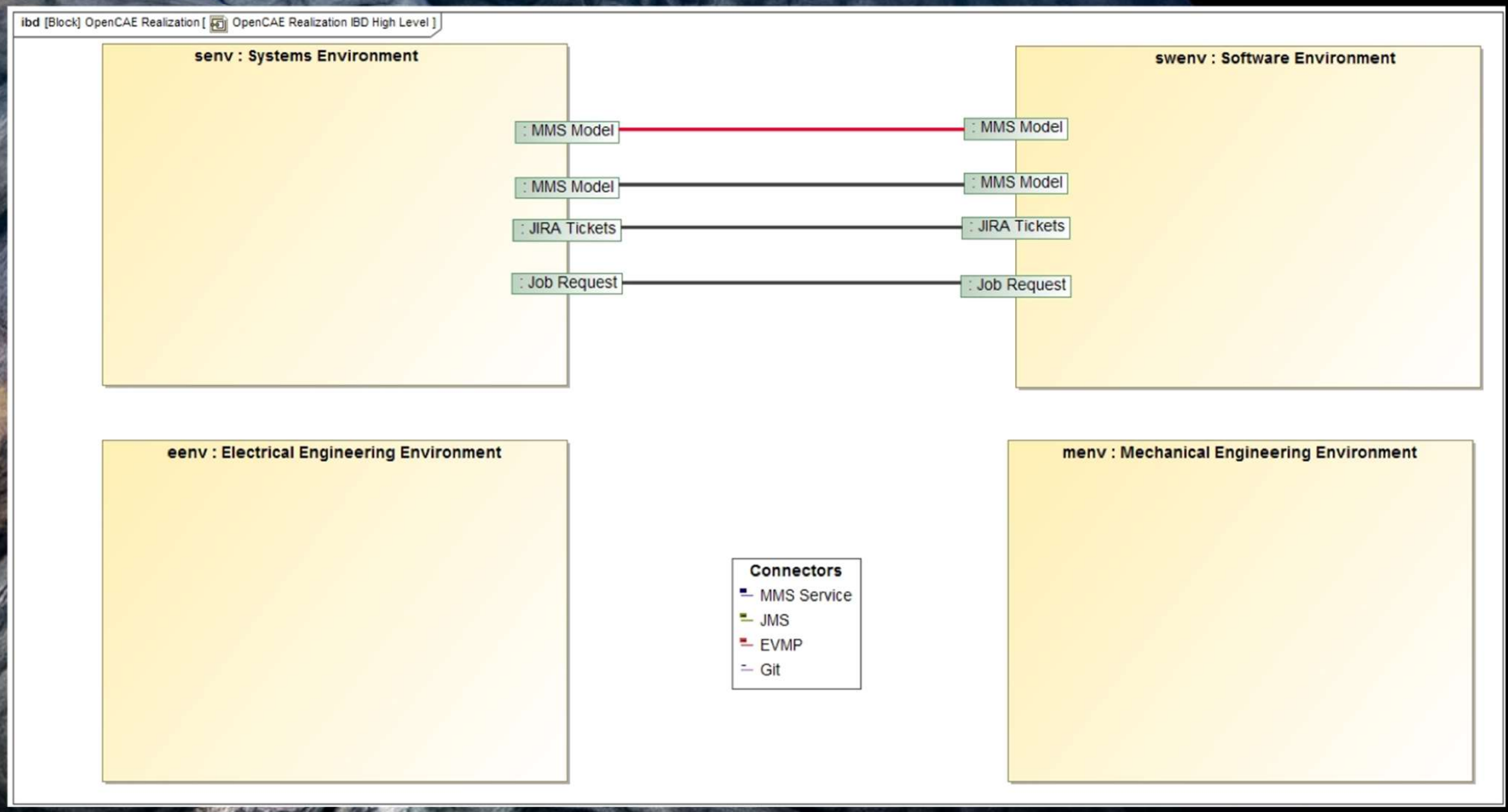
Systems

Systems Environment Tools

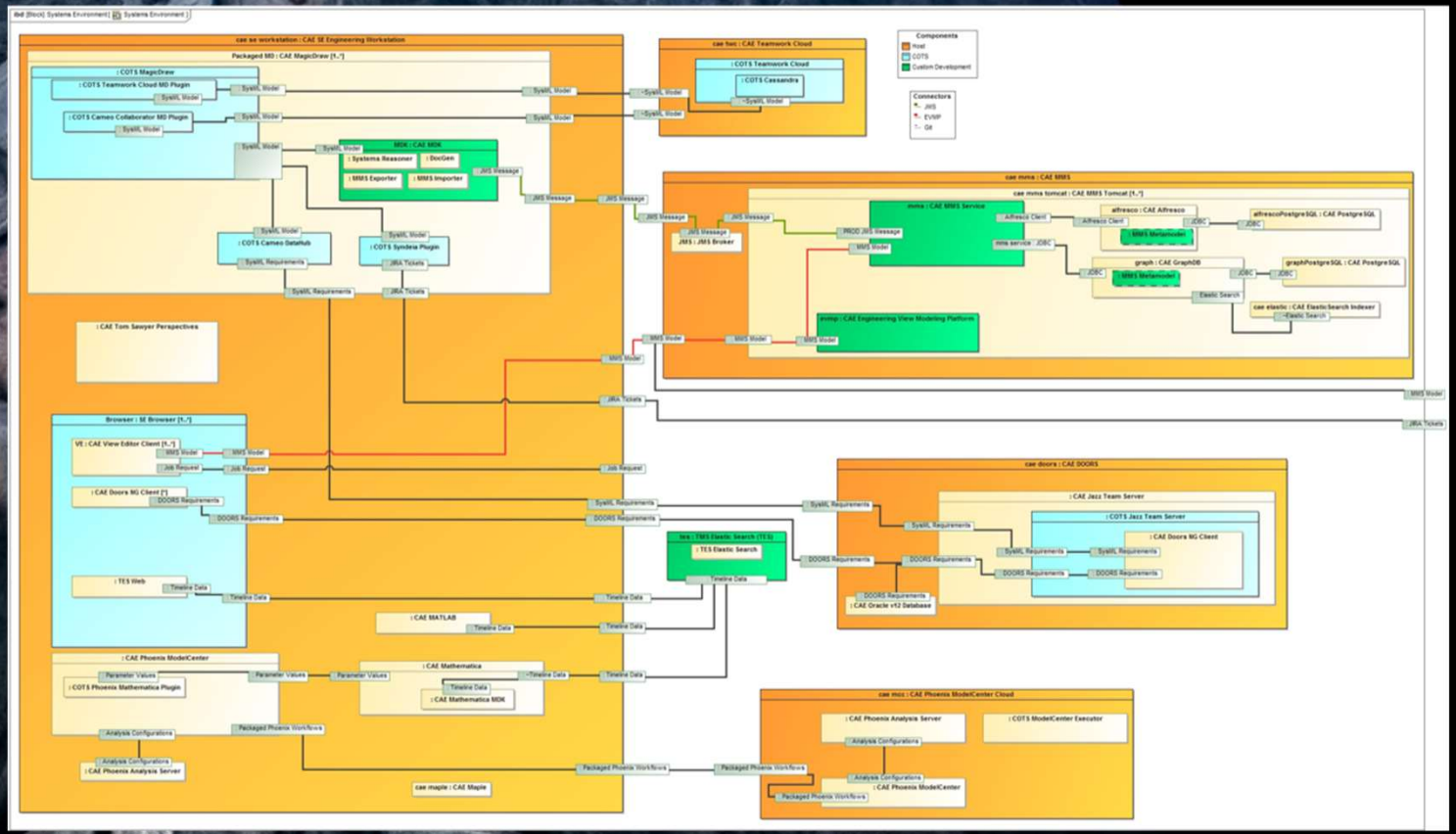


- Authoring
- Visualization
- Analysis
- Collaboration
- Integration
- Workflow
- Relationship management
- Search
- **Beyond SysML**

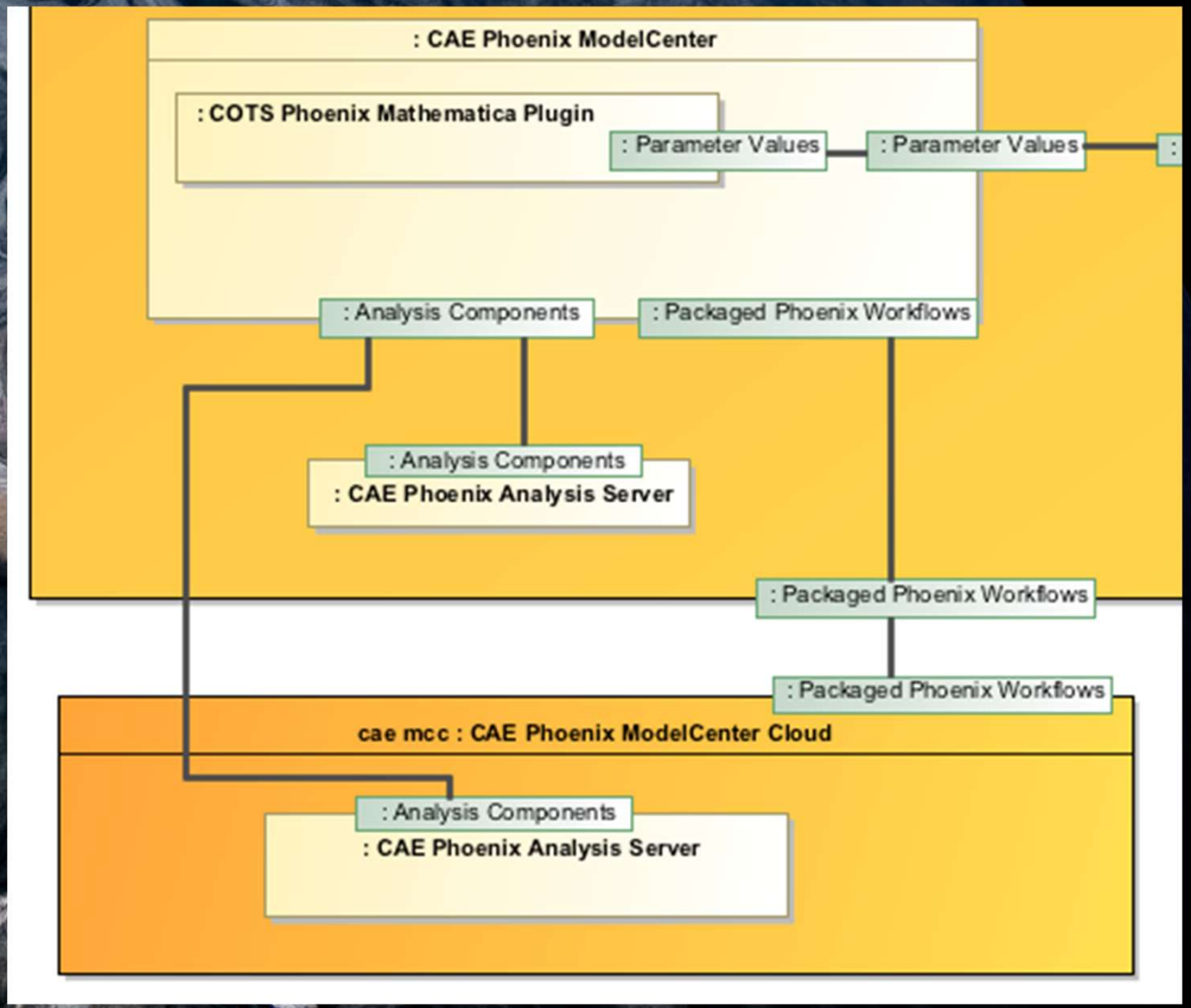
Interactions Between CAE Environments



Interactions Within CAE Systems Environment



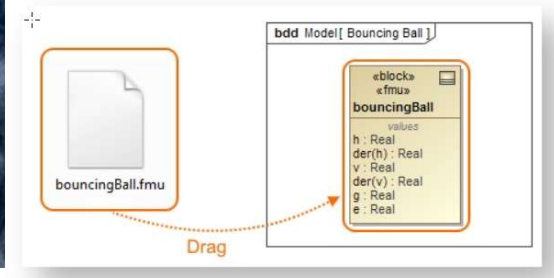
Vendor Spotlight: Phoenix Integration

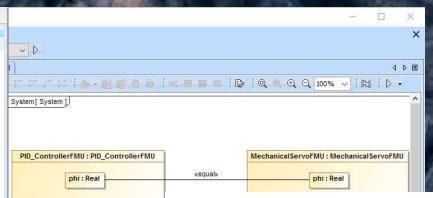


Standardized Co-Simulation

- The Functional Mock-up Interface (or FMI) defines a standardized interface to be used in computer simulations to develop complex cyber-physical systems
- Integration with System Level behavior model

Tools supporting FMI	FMI Version	ModelExchange		CoSimulation		Notes
		Export	Import	Slave	Master	
Cameo Simulation Toolkit	FMI_1.0		Available		Available	FMUs can be imported, represented, connected and co-simulated in SysML models.





functional mockup interface for model exchange and tool coupling

courtesy Daimler

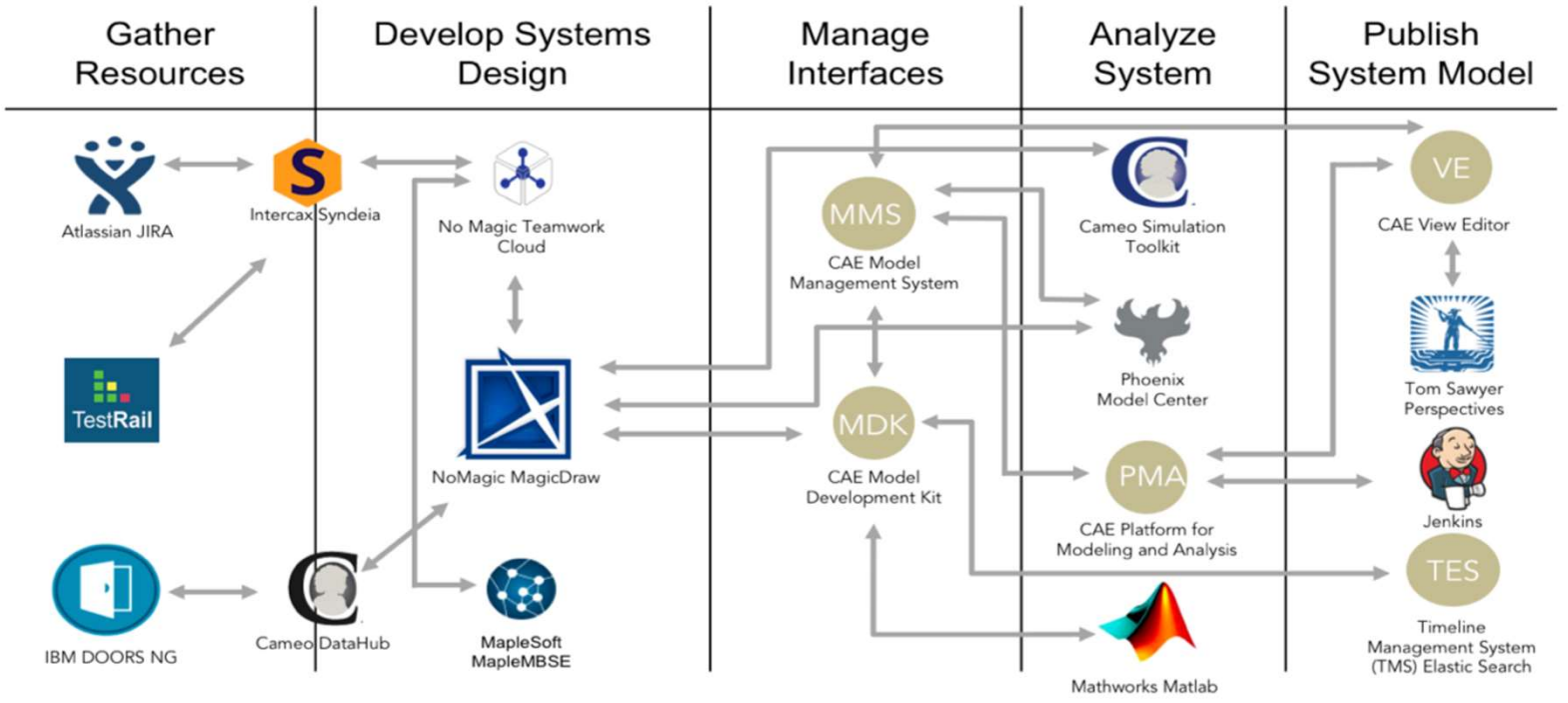
Systems Environment Use Case

National Aeronautics and Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

Use Case: Preliminary Design Review

cae.jpl.nasa.gov



The CAE Systems Environment supports systems engineering activities from requirements capture and management, traceable to architectural and design, and finally a wide range of analysis capabilities, data search, and integration capabilities.

3

Agenda

- Introduction
- Current State
- OpenCAE Approach
- Open Source Contributions
- Example Application at JPL
- Conclusions and Summary

Jet Propulsion Laboratory
California Institute of Technology

MMS, MDK, and View Editor

- The MMS model repository supports the following features:
  - Basic Infrastructure for Version, Workflow, Access Control
  - Flexibility of model content
  - Support for Web Applications and Web-based API access
  - Integration across engineering and management disciplines
- MMS is accessible from:
  - Rich SysML desktop clients like MagicDraw (via MDK)
  - Light-weight web-based clients like View Editor
  - Mathematical computation programs like Mathematica
  - Any tool that can utilize RESTful web services
- View Editor enables users to interact with SysML models within a web-based environment
  - System models are constructed, queried and rendered following the view and viewpoint paradigm
- View Editor implements the MMS REST API to provide a web environment to create, read, and update model elements

Document Generation Results on View Editor

View Editor | TMT-org

Secure https://mms.openmbee.org/alfresco/mmsapp/mms.html#/projects/PROJECT-d94630c2-576c-4edd-a8cd-ae3ecd25d16c/master/documents/_18_0_2_b4c02e1_143517683123...

VE **TMT-org** Switch Org

Search selected project

Help

Project: **TMT** › TMT-APS-SE › TMT-APS Use Cases › TMT-APS Use Cases

Branch: **master**

Filter items in the tree

- TMT-APS Use Cases
  - 1 Use Case Introduction
    - 1.1 Purpose
    - 1.2 Overview
    - 1.3 Modeling Approach and
  - 2 Use Cases
    - 2.1 Post Segment-Exchange
      - 2.1.1 Pupose of use case
      - 2.1.2 Typical observing p
      - 2.1.3 Entrance requireme
      - 2.1.4 Use case activity
      - 2.1.5 Optical Performanc
      - 2.1.6 Time to execute
    - 2.2 Maintenance Alignment
    - 2.3 Rigid Body M3 Alignmer
    - 2.4 Off-Axis Measurements
    - 2.5 On-Sky Measurement of
    - 2.6 Self Test
    - 2.7 M1CS Sensor Calibratio
    - 2.8 M2 and M3 Rigid Body
    - 2.9 Wavefront Calibration
    - 2.10 Pupil Registration Relat
    - 2.11 Reference Point Trackir
  - 3 High Level Activities
    - 3.1 Acquisition & Guiding
    - 3.2 Coarse Tilt Alignment
    - 3.3 Rigid Body and Segmen
    - 3.4 Broad Band Phasing

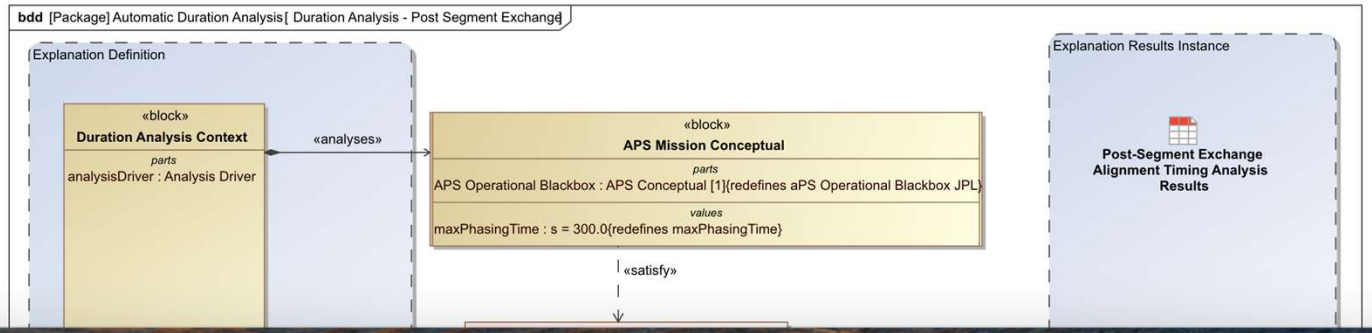
DOCLIB

EXPORT

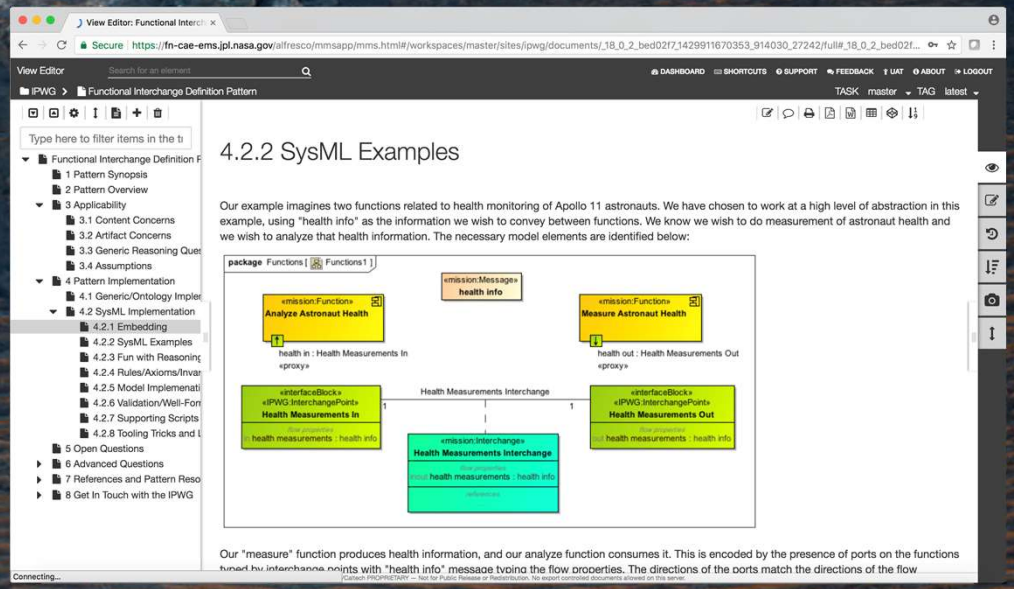
2.1.6 Time to execute

The table below shows our current bottom-up time estimate for each of the activities that make up this use case. The total time estimate is ~75 (TBR) minutes, which is to be compared with our requirement of 120 min (as shown in the figure below).

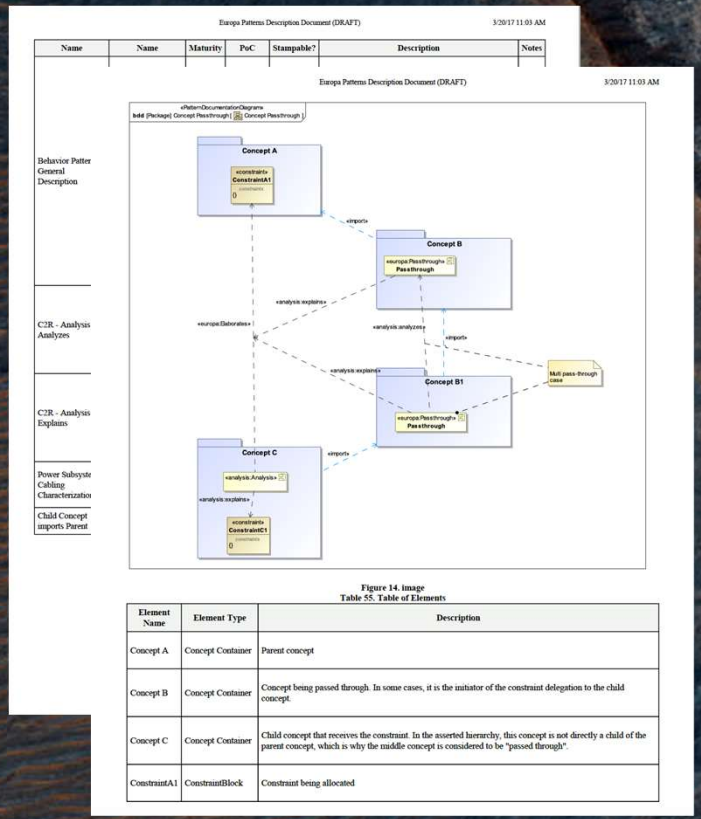
At Keck, we routinely perform post-segment exchange alignment in 120 minutes or less. However, at Keck the segment shapes are measured in a separate test, with each segment measured separately, but adjustment of the segment warping harnesses is manual and occurs the next day. We will measure the TMT segment shapes in parallel as part of the rigid body and segment figure activity and imidiatly adjust the segment shapes during the night via the motorized warping harnesses and iterate the control at least once. In addition the CCD read out time for APS is significantly faster then at Keck, ~10 vs ~55 seconds, given the post-segment exchange alignment takes ~60 frames, this accounts for 45 minutes. Given our bottom up estimate and our Keck experience we have a high degree of confidence we can met the 120 minute requirement.



SysML Modeling Patterns Development



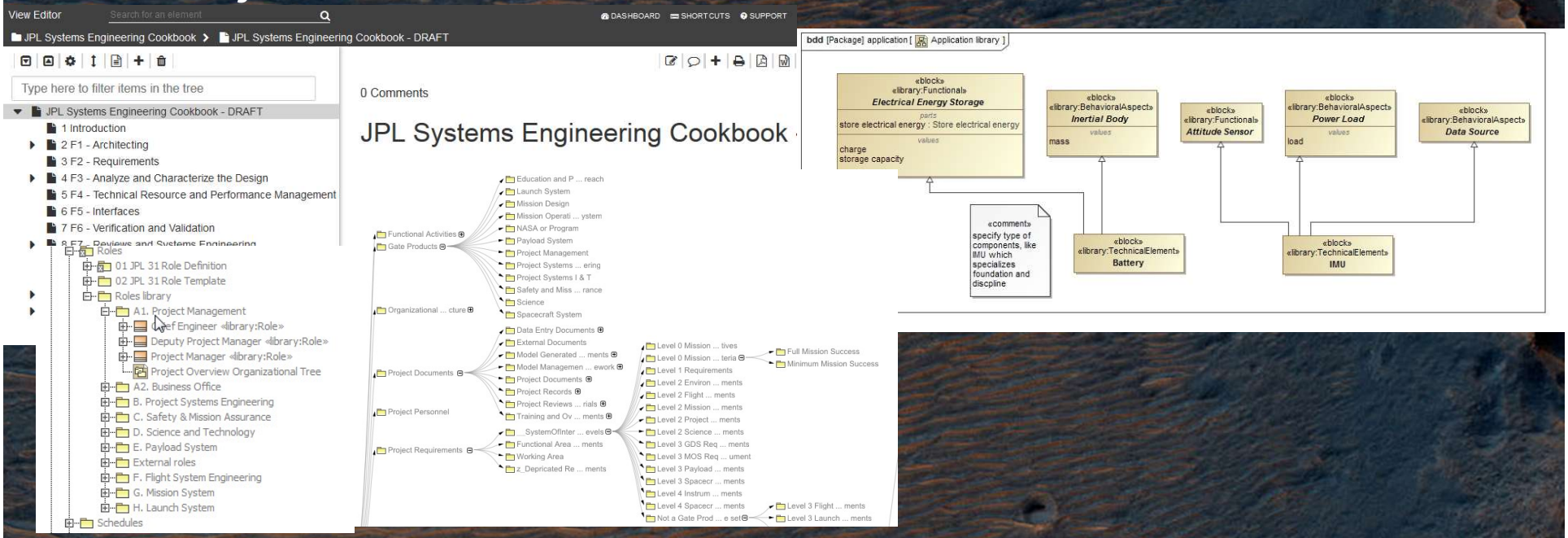
Project-specific modeling patterns for common modeling tasks



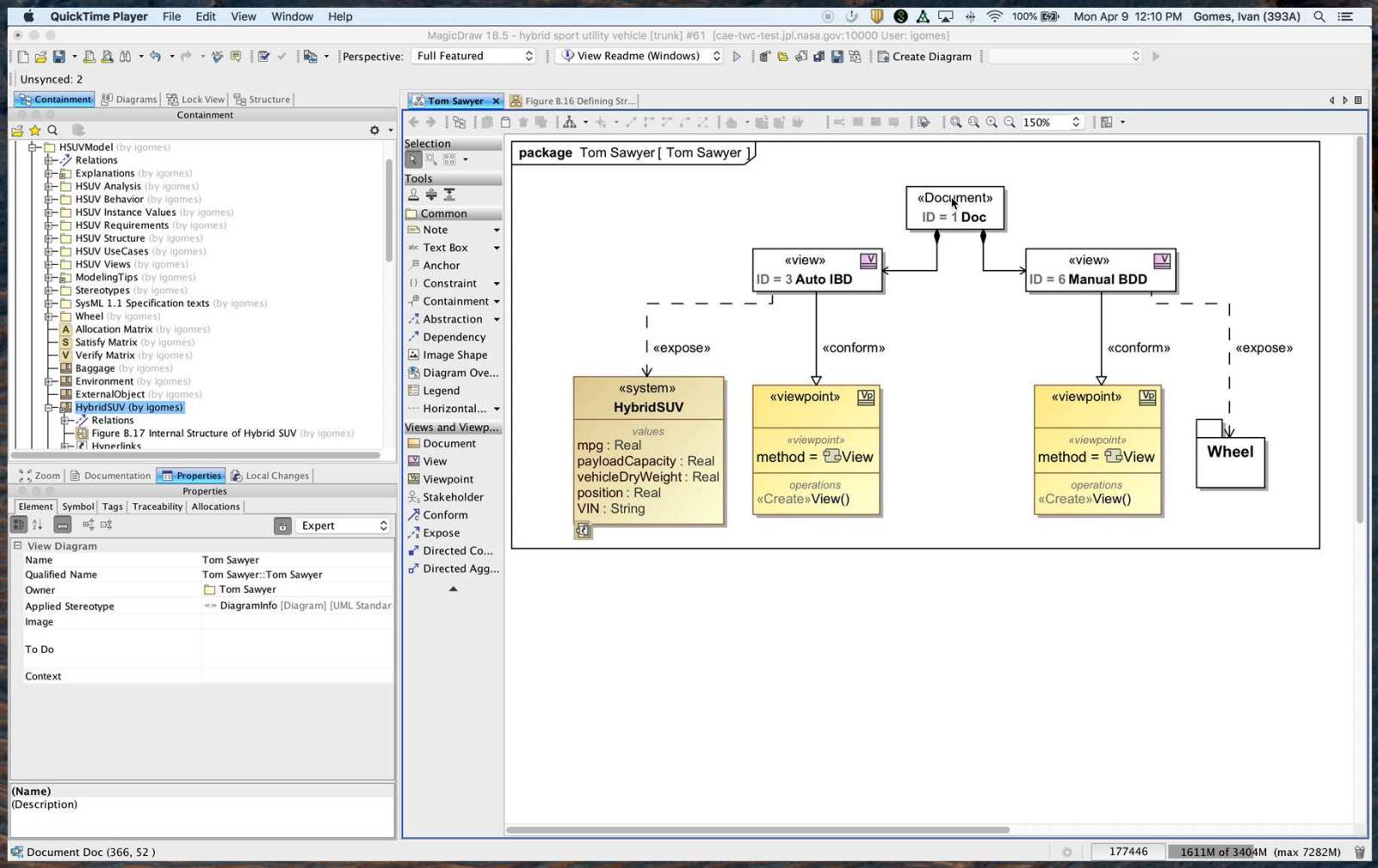
Project-independent modeling patterns as guidelines from overarching line organization

JPL SE Cookbook

- Collection of processes, practices, patterns to support Systems Engineering with model based techniques specific to JPL
- Organized according to 10 JPL SE functions
- Provides a set of SysML libraries, e.g. WBS Elements, Project Roles, Functional Elements, Model structure



DocGen – Tom Sawyer Integration for Query-Based Visualization

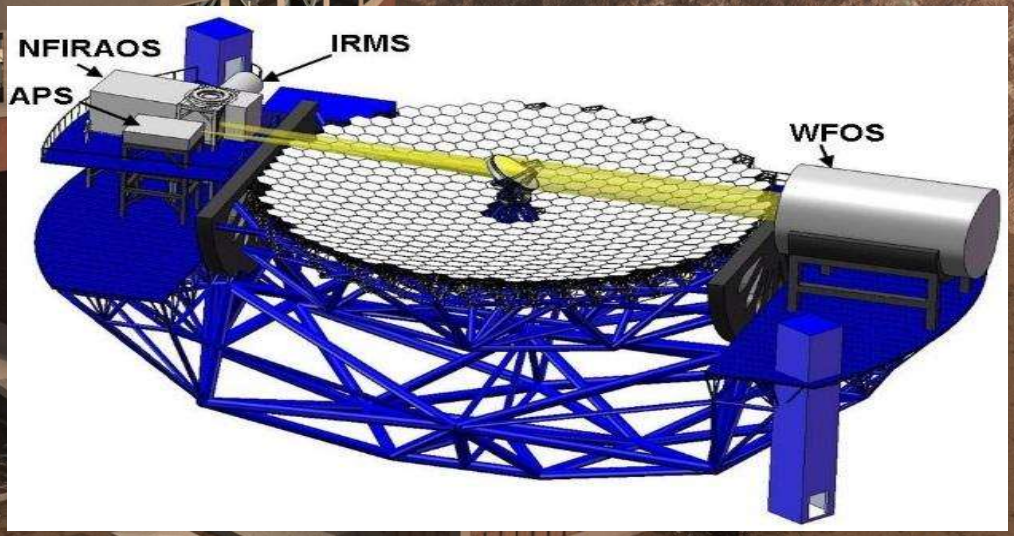


Agenda

- Introduction
- OpenCAE Vision
- Open Source Contributions
- Conclusions and Summary

Jet Propulsion Laboratory
California Institute of Technology

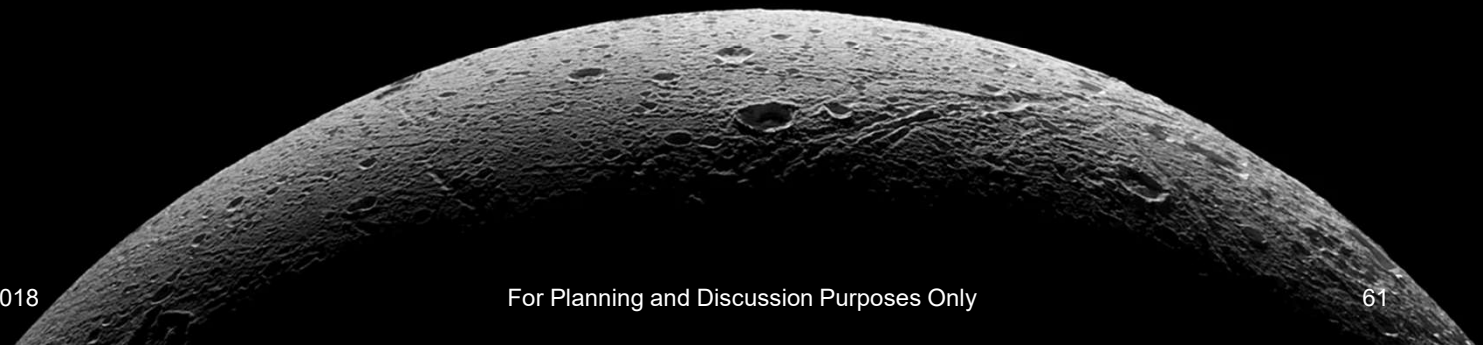
The Thirty Meter Telescope (TMT) Project



www.tmt.org

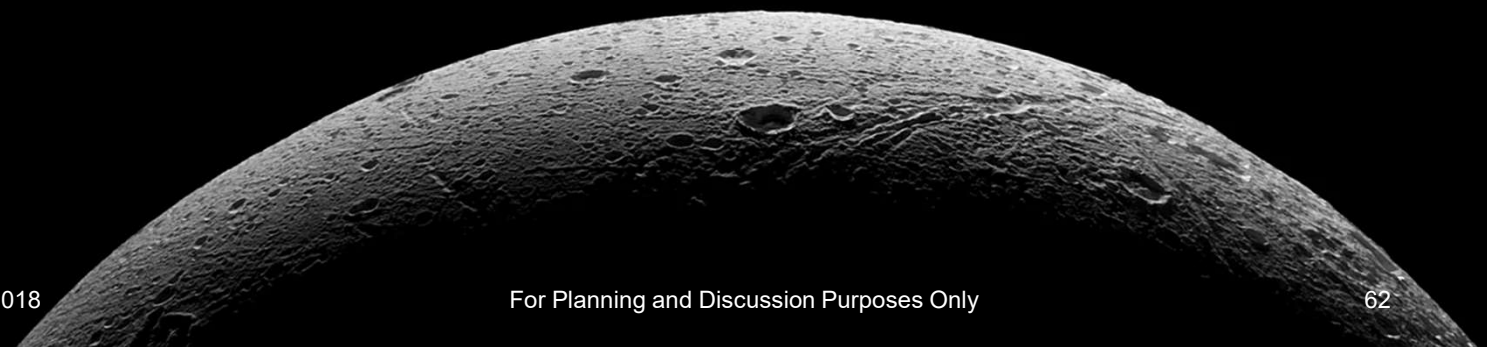
TMT MBSE Objectives

- Define an **executable SysML model**
- Use the model to **analyze the system design and verify requirements** on power consumption, mass, duration, pointing errors, etc.
- Produce **engineering documents**
  - Requirement Flow Down Document
  - Operational Scenario Document
  - Design Description Document
  - Interface Control Documents
- Use **standard languages and techniques, and COTS tools where practical** to avoid custom software development



Modeling Approach

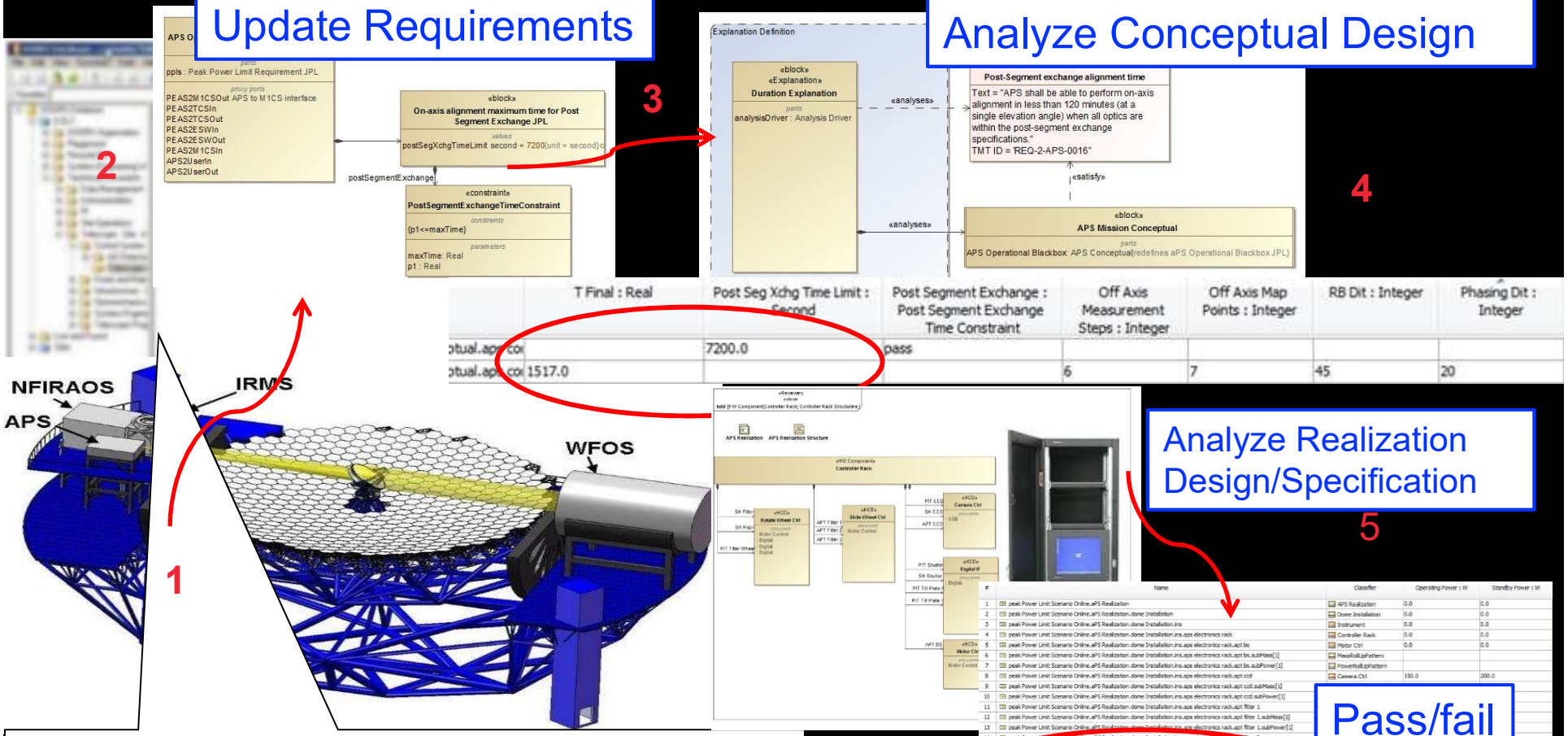
- **Object-Oriented Systems Engineering Methodology (OOSEM)**, but with additional activities focusing on building an executable model
- Use case driven model development
- Challenges:
  - JPL is a **supplier** for a number of subsystems of the TMT (the **customer**)
  - Model is used by a number of teams, including TMT



Analysis of Architecture and Design

Update Requirements

Analyze Conceptual Design



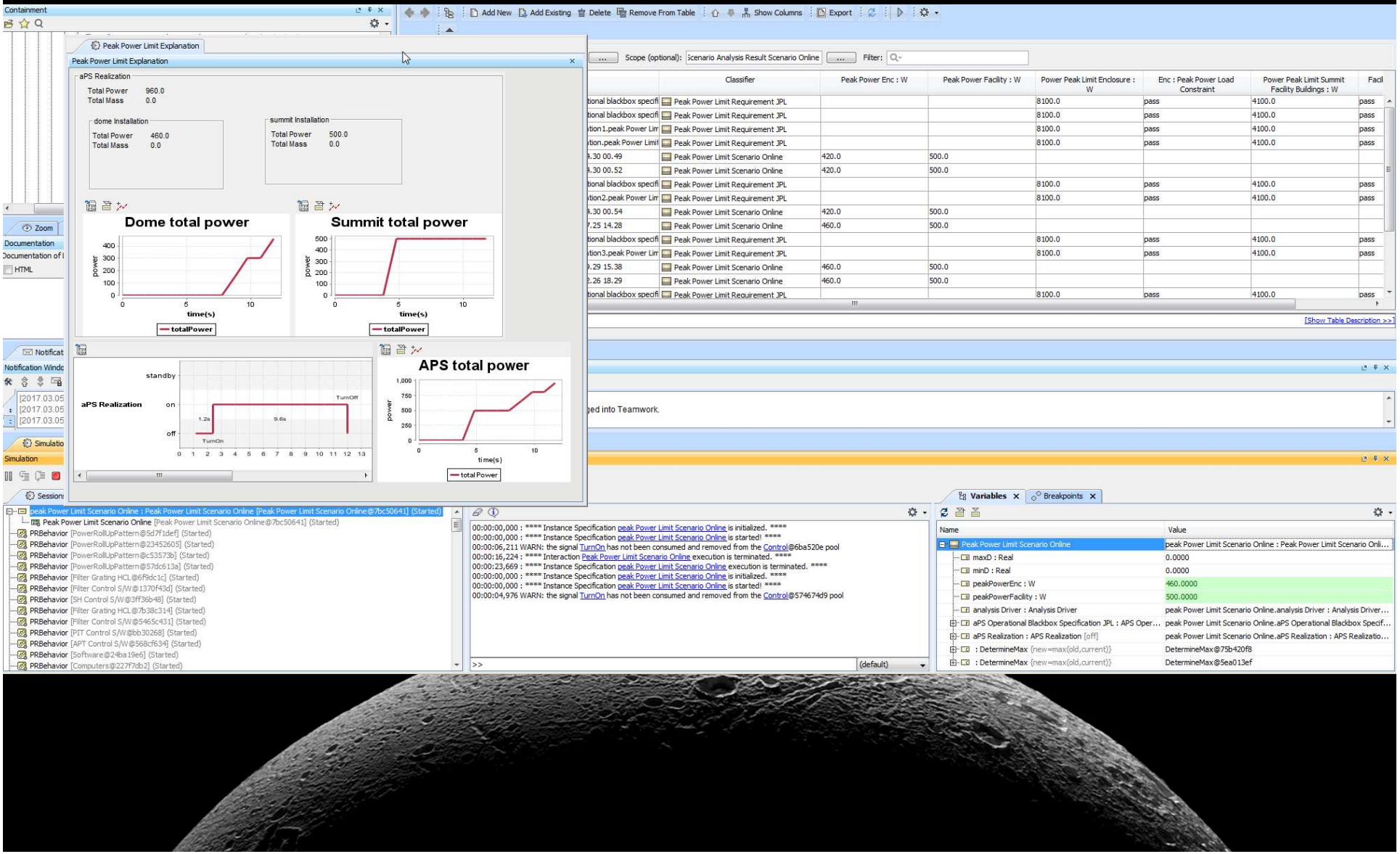
Analyze Realization Design/Specification

Pass/fail

Max duration Post-segment exchange: ~~7200s~~ 5000s
Number of exposures of 45s 4 ~~6~~
Max peak power consumption in dome: 8.5kw ~~8.1kw~~
Number of motors with 50W 10 ~~12~~

OCD, Requirements, ICD, DDD

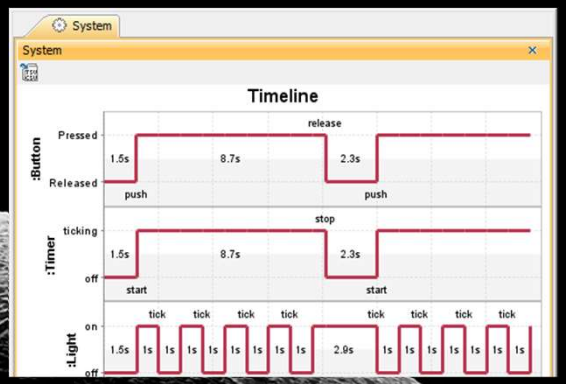
Power Analysis



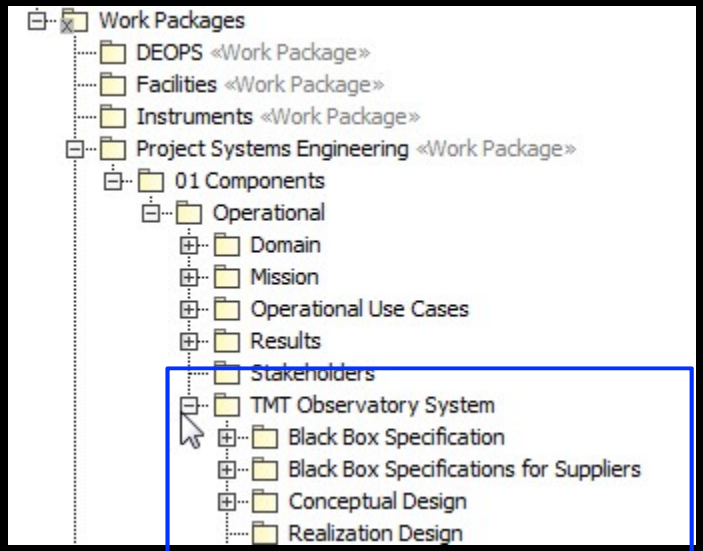
Run Analysis

- Run a configured analysis with a simulation engine on the initial conditions to get the final conditions
- Produce the following views on final conditions
  - **Table** showing final analysis values (e.g., peak power) and the constraint's pass/fail status for each scenario
  - **Timelines**: state changes for components over time
  - **Value profiles**: total rolled up values over time

#	Name	Classifier	T Final : Real	Ph
1	calibrations Duration S	Calibrations Duration S		
2	calibrations Duration S	APS Conceptual		
3	calibrations Duration S	Procedure Executive ar	8466.0	11

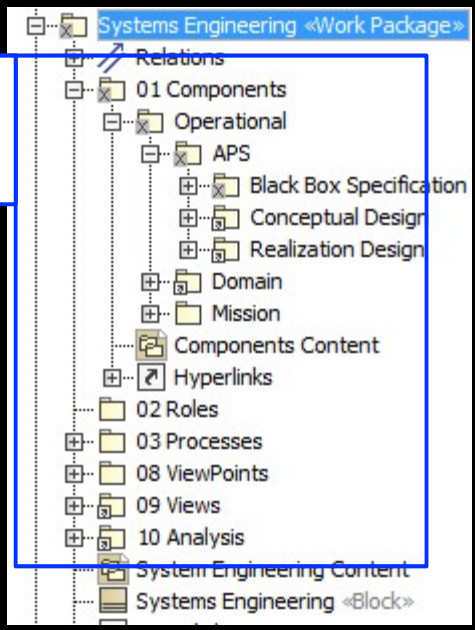


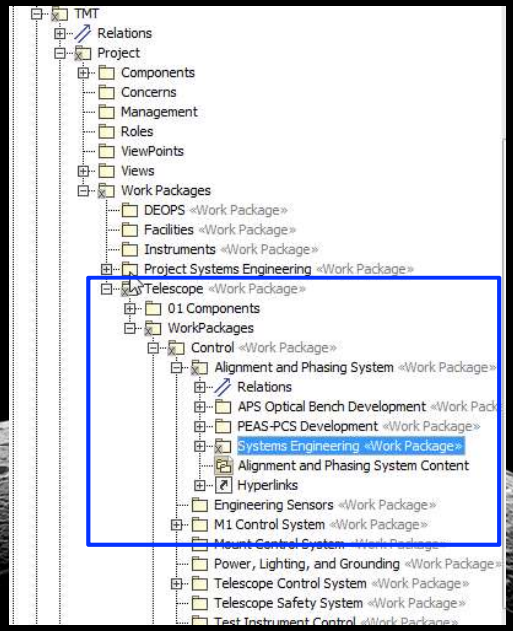
Package Organization



Customer / supplier relationship

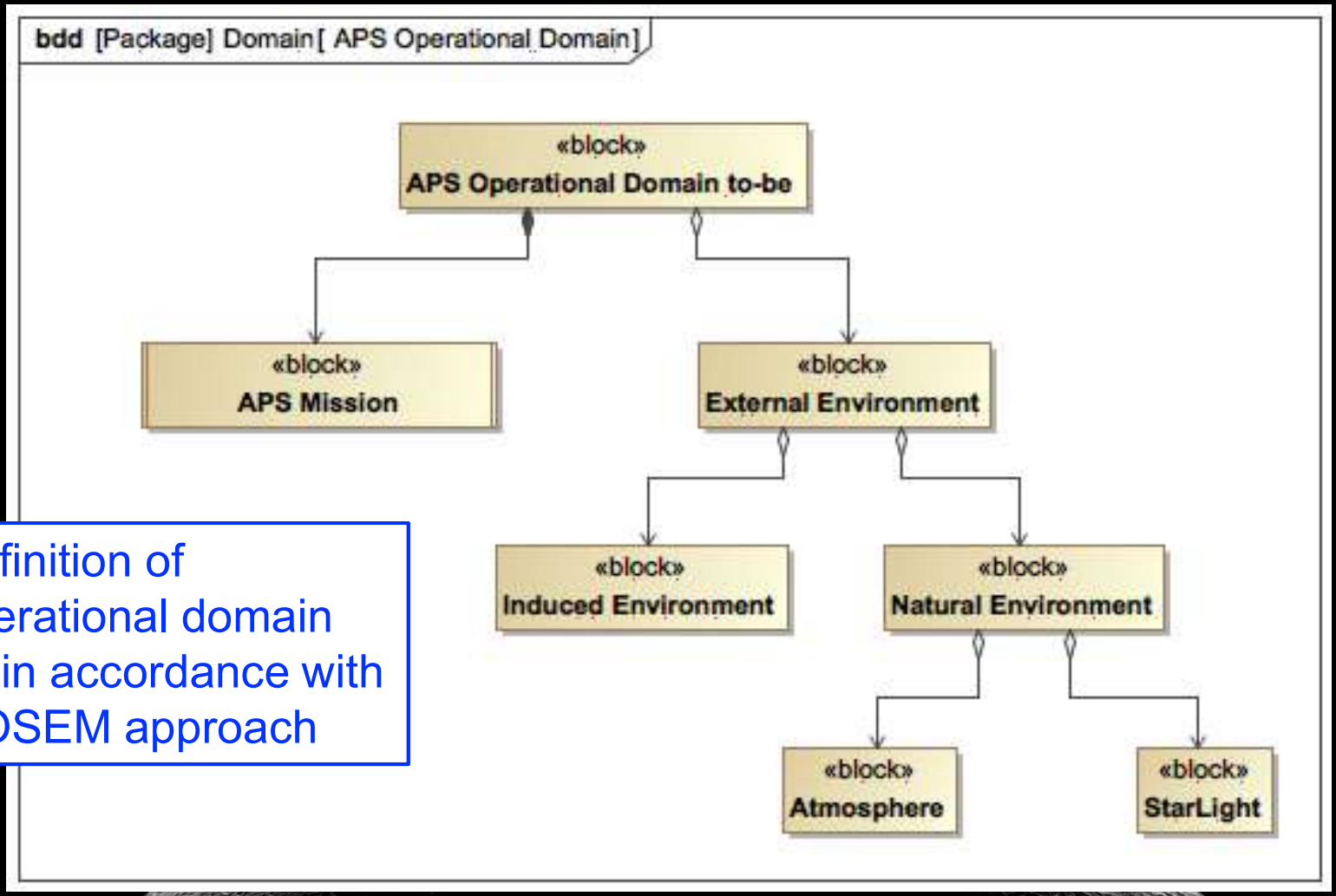
OOSEM abstraction layers





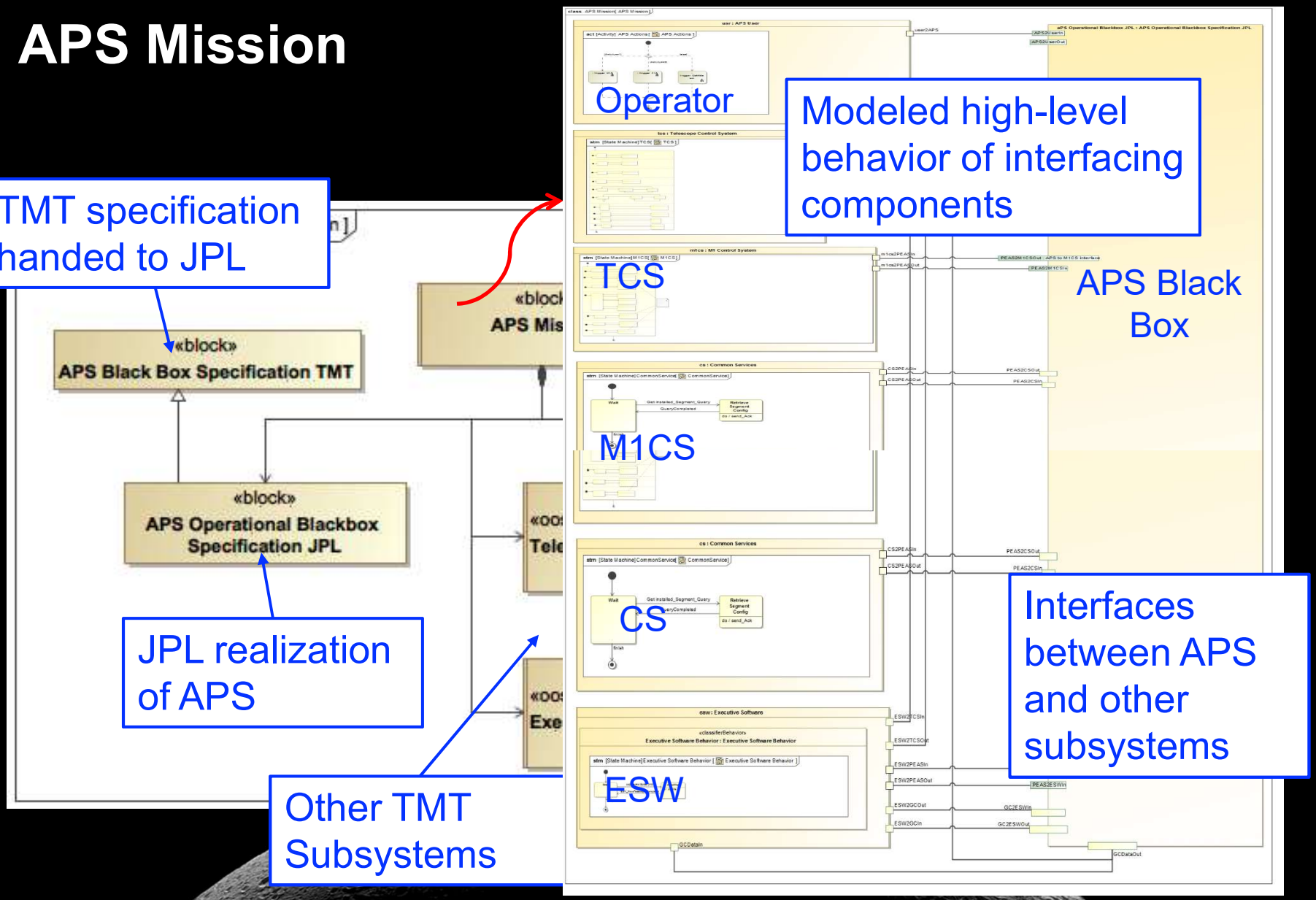
Work breakdown structure

Operational Domain

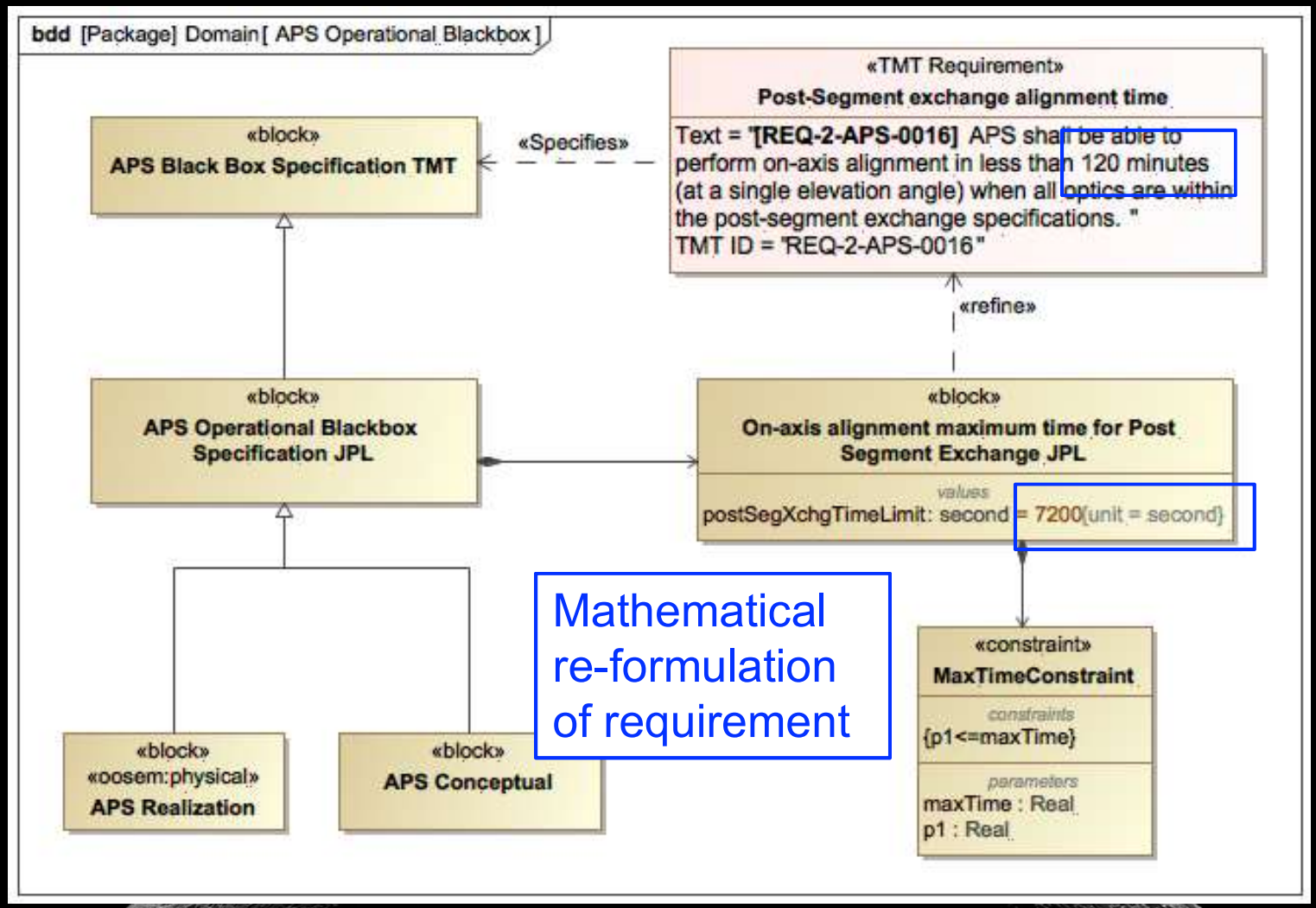


Definition of operational domain as in accordance with OOSEM approach

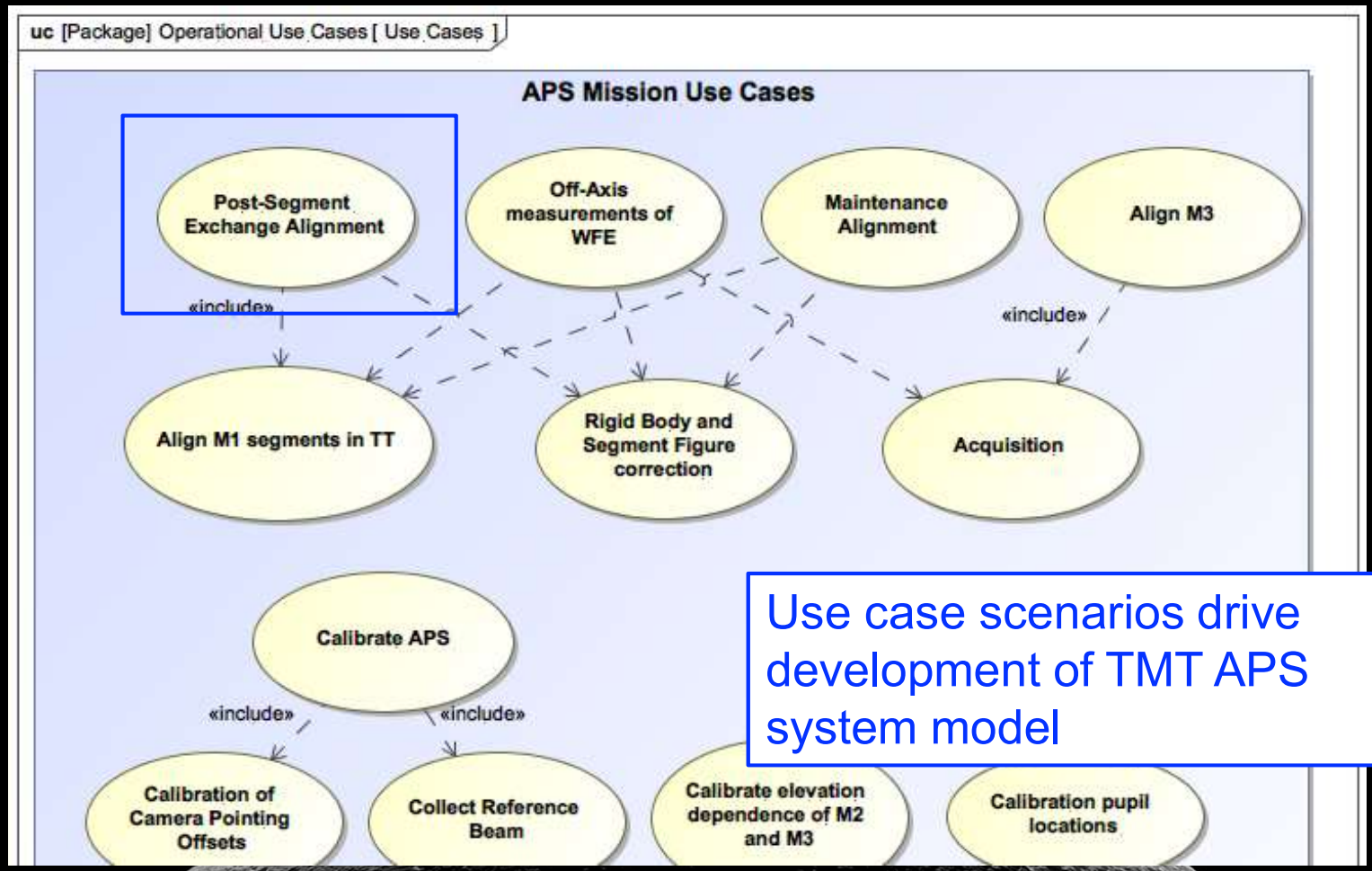
APS Mission


TMT specification handed to JPL
«block»
APS Black Box Specification TMT
«block»
APS Operational Blackbox Specification JPL
JPL realization of APS
Other TMT Subsystems
Operator
Modeled high-level behavior of interfacing components
TCS
APS Black Box
M1CS
CS
ESW
Interfaces between APS and other subsystems

Formalizing Requirements



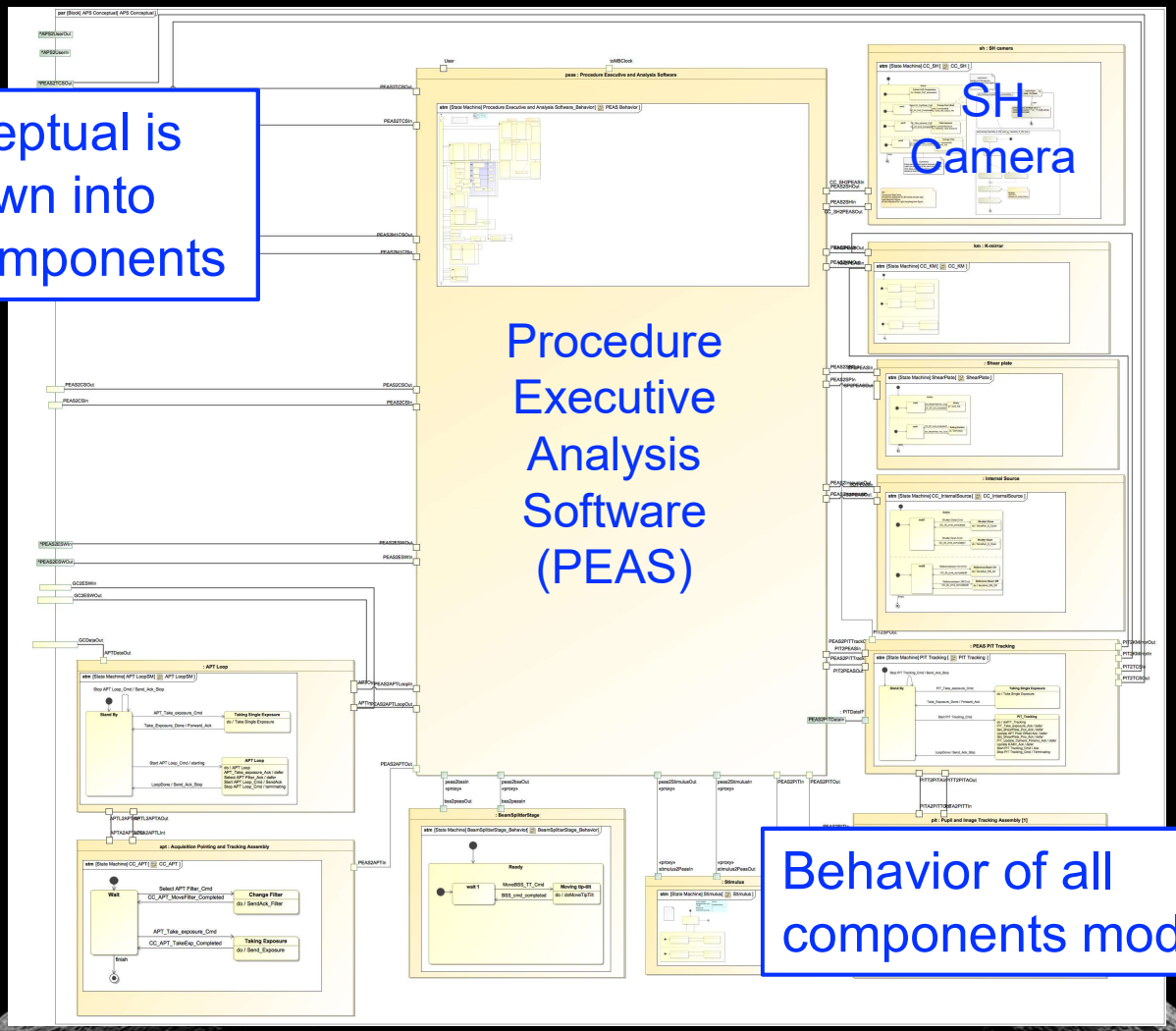
Use Cases



Use case scenarios drive development of TMT APS system model

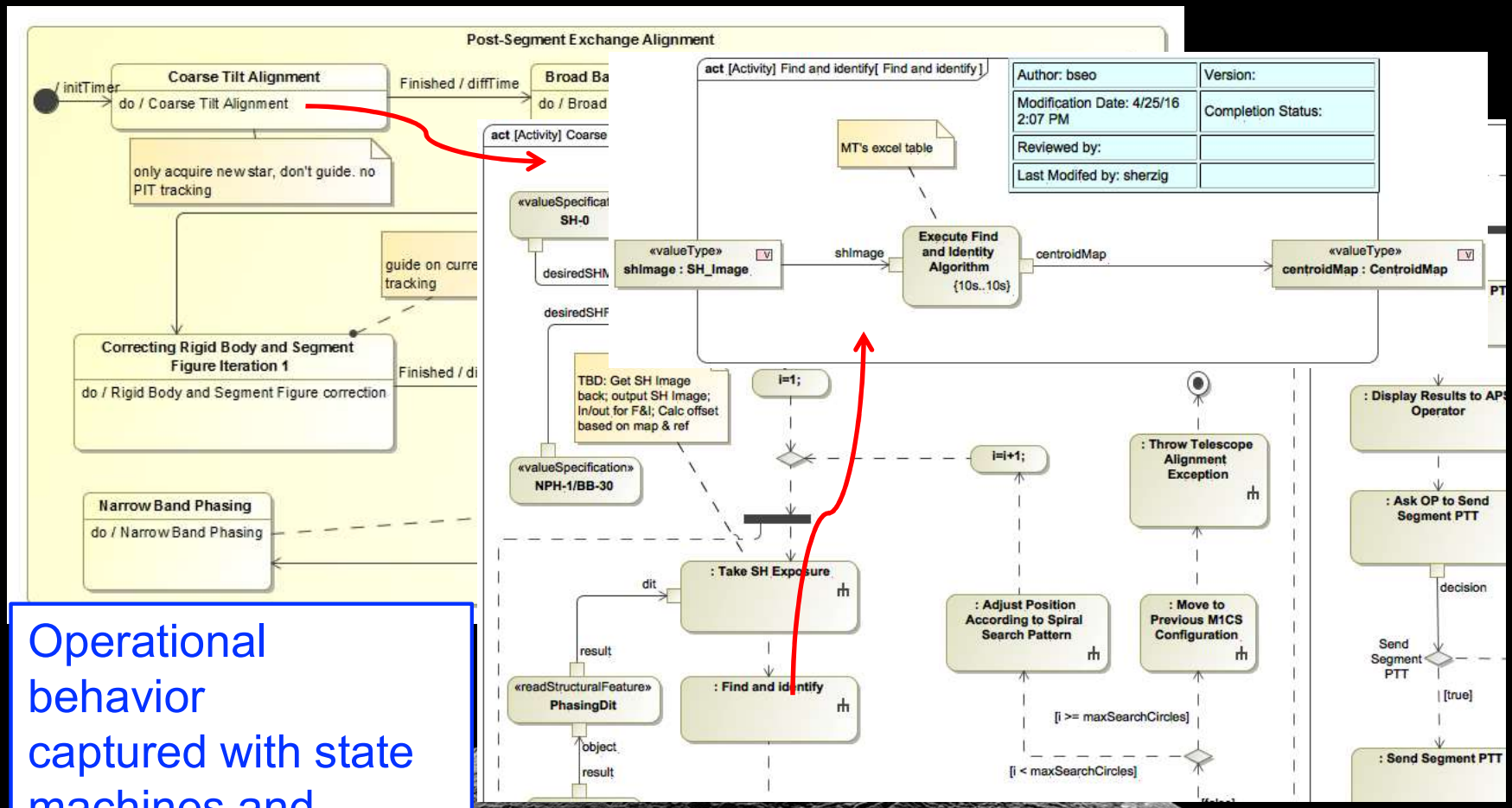
Conceptual Architecture

APS conceptual is broken down into several components



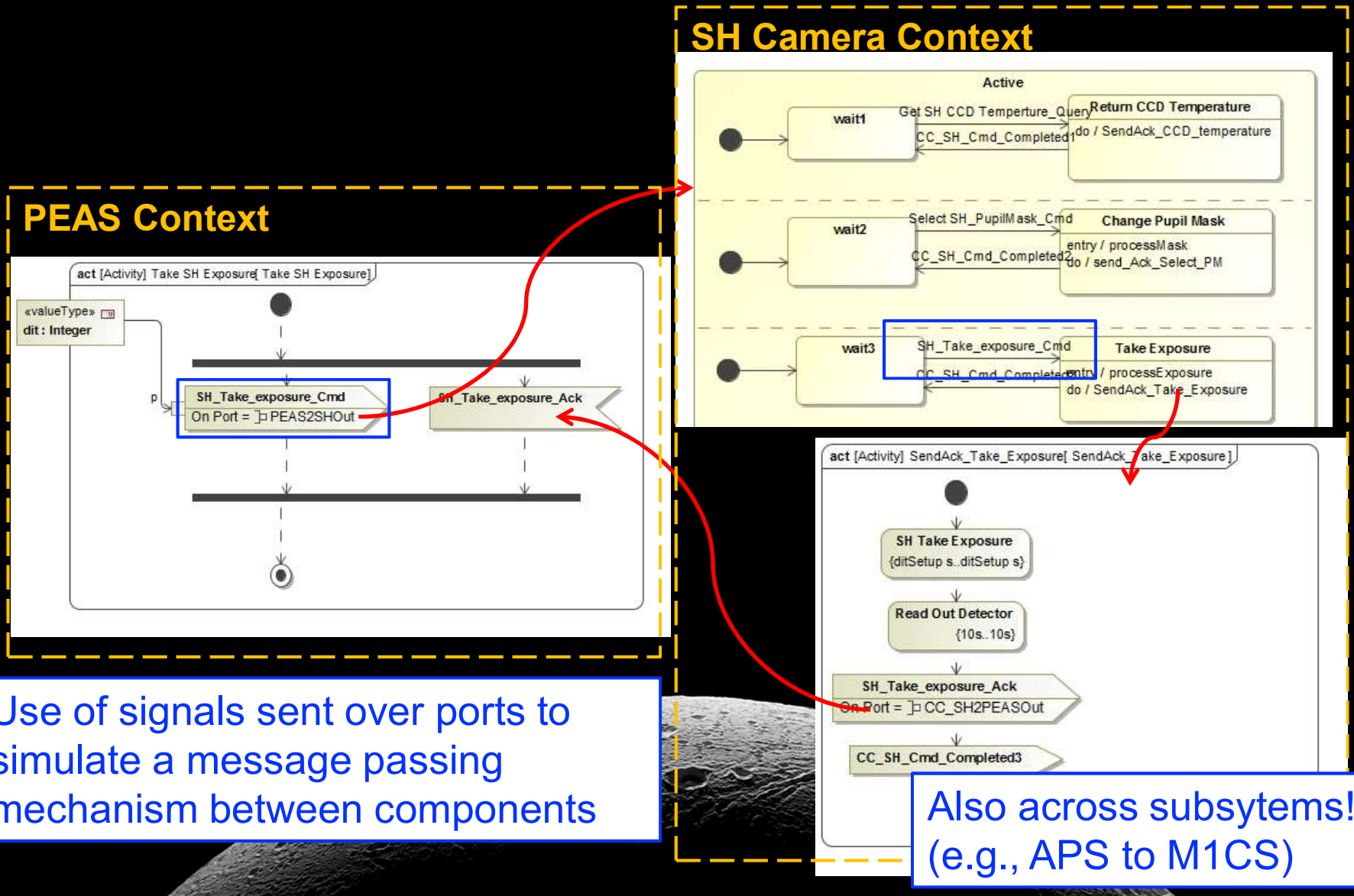
Behavior of all components modeled

Modeling Behavior



Operational behavior captured with state machines and activity models

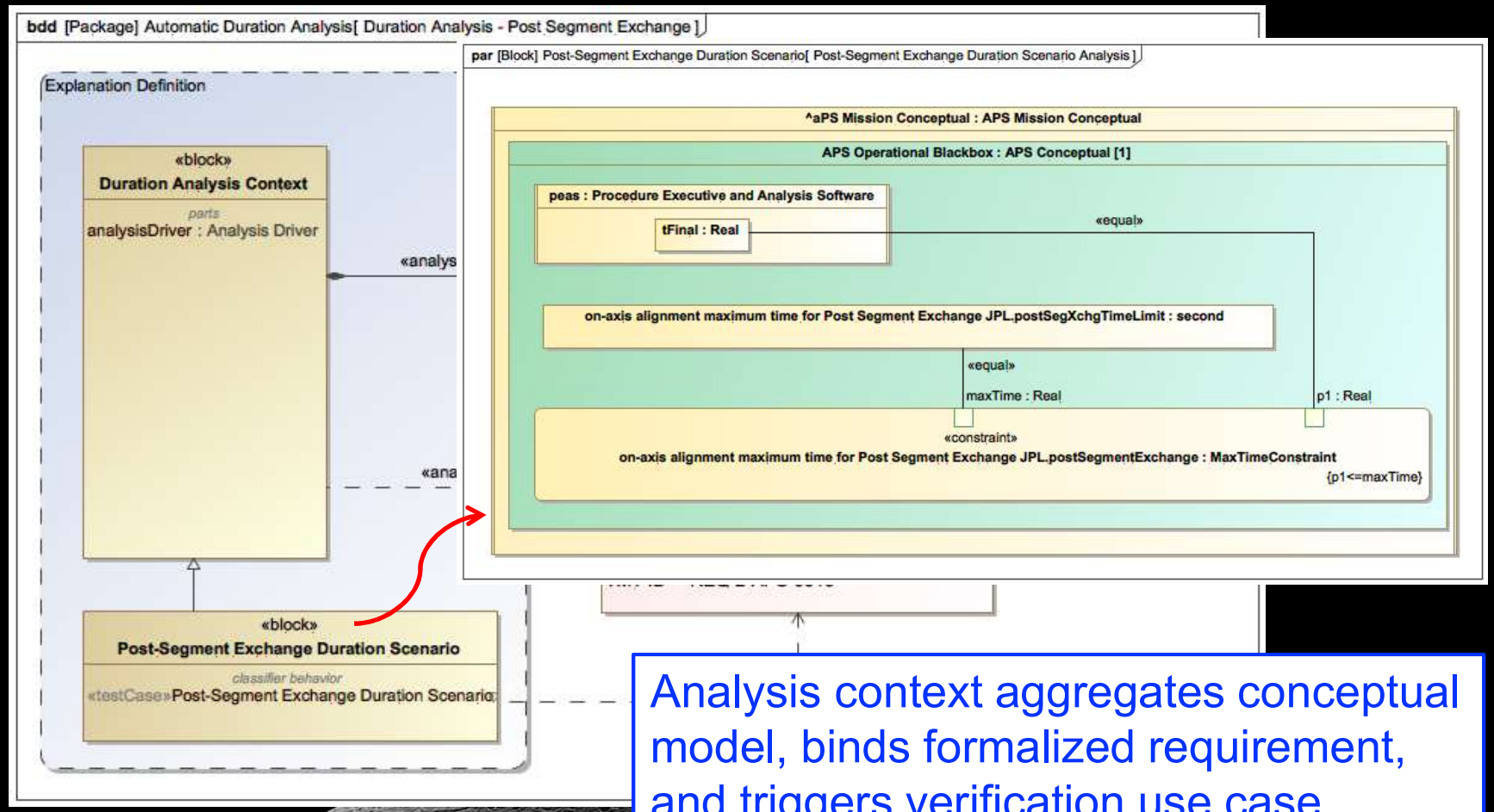
Interactions Between Components



Use of signals sent over ports to simulate a message passing mechanism between components

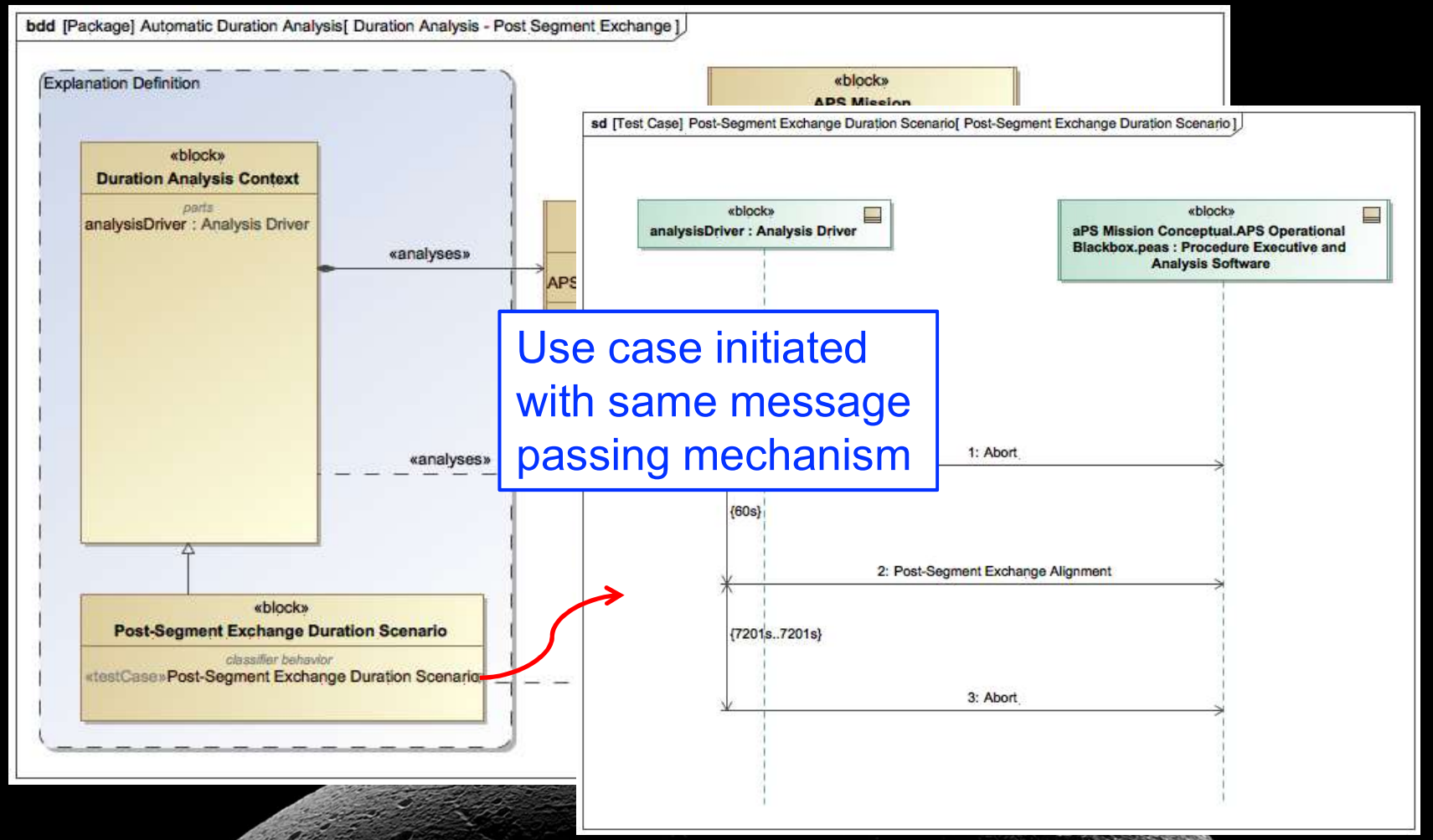
Also across subsytems! (e.g., APS to M1CS)

Verifying Timing Requirements by Simulation

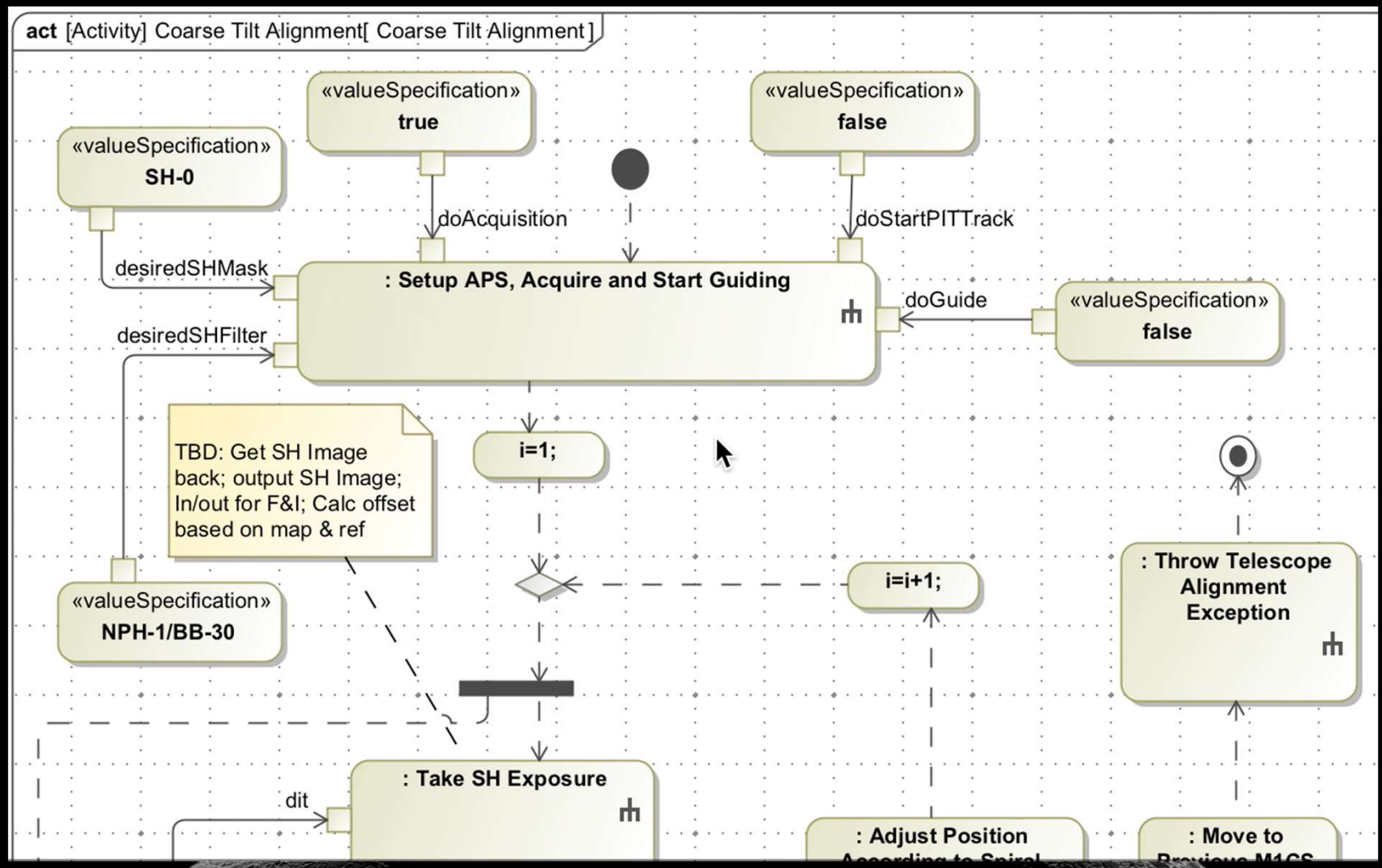


Analysis context aggregates conceptual model, binds formalized requirement, and triggers verification use case

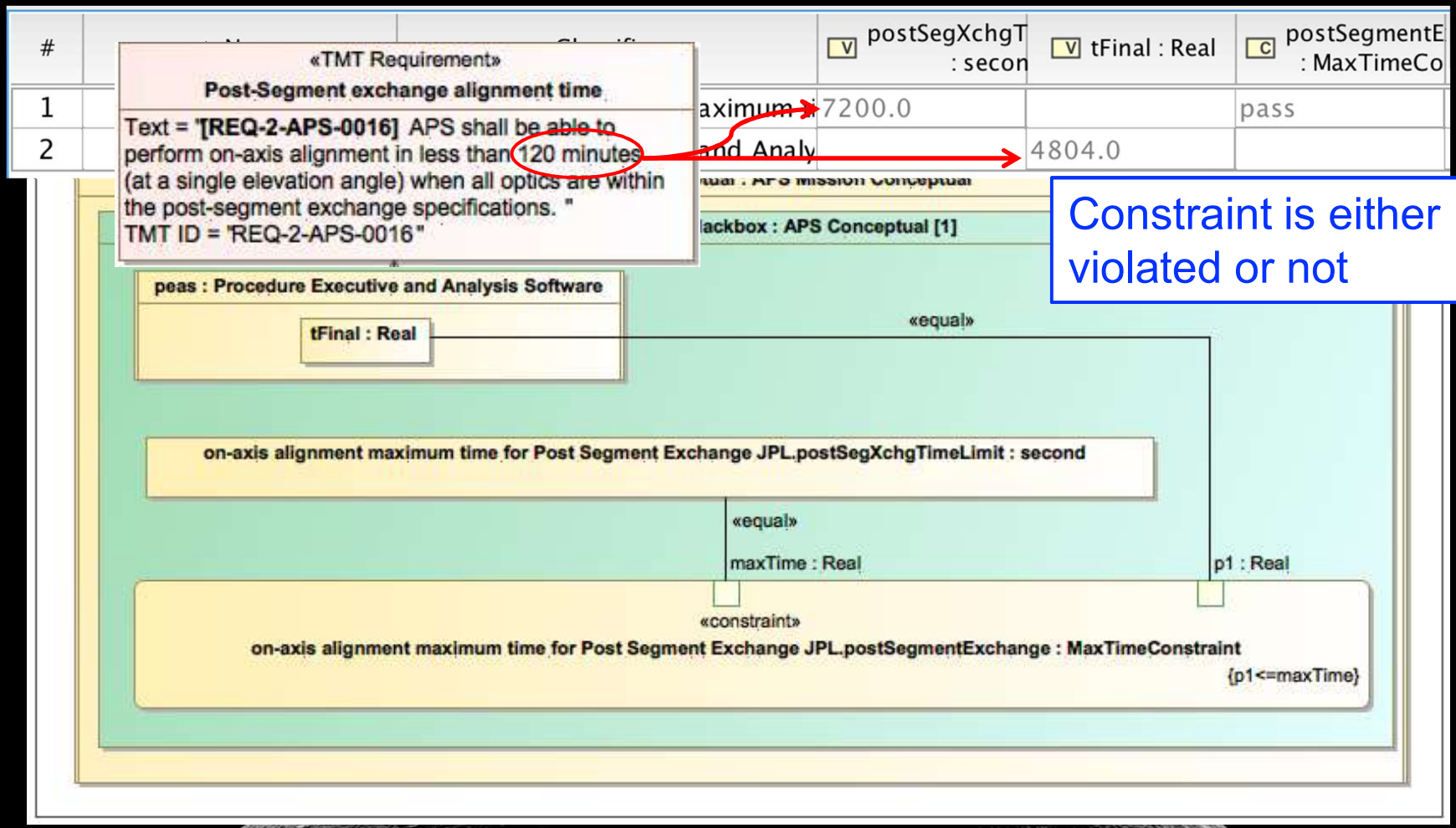
Verifying Timing Requirements by Simulation



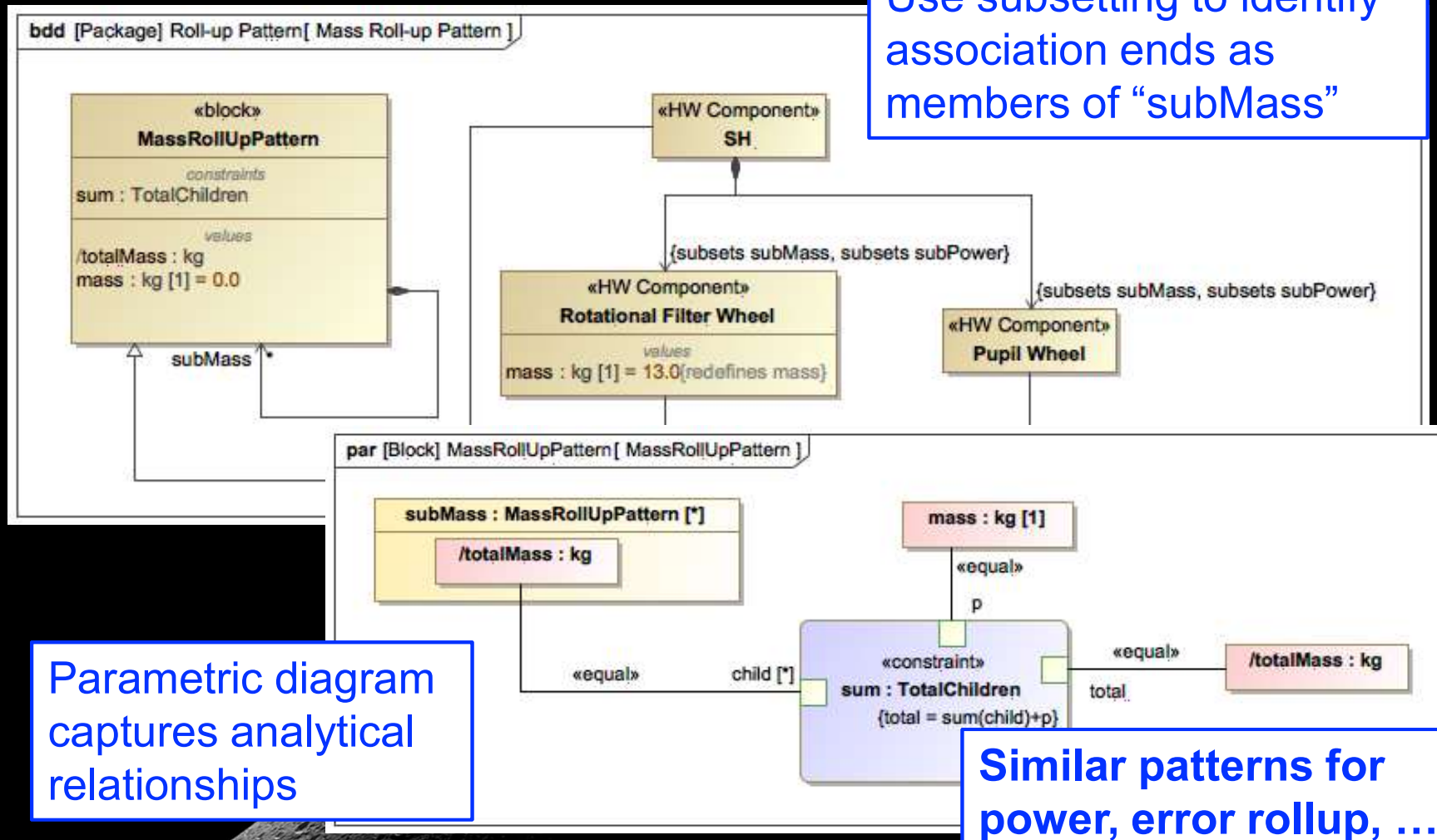
Verifying Timing Requirements by Simulation



Verifying Timing Requirements by Simulation



“Static” Rollup Analyses – Example: Mass



Use subsetting to identify association ends as members of “subMass”

Parametric diagram captures analytical relationships

Similar patterns for power, error rollup, …