



Jet Propulsion Laboratory
California Institute of Technology

CAESAR Concept

Presented By

Maged Elaasar, Ph.D.

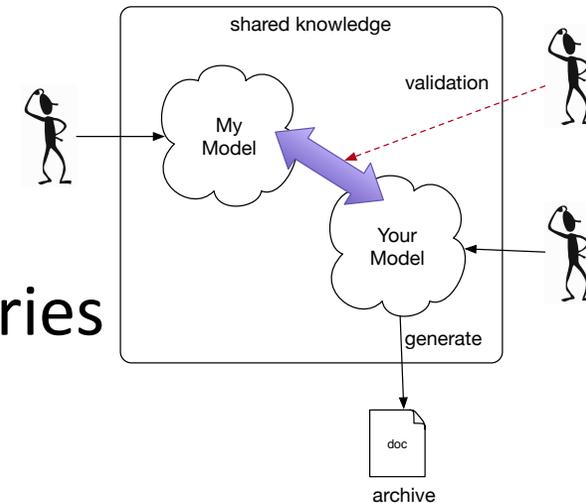
Chief Architect

Integrated Model Centric Engineering

© California Institute of Technology 2018, all rights reserved

MBSE vision

- Engineering work being done in precise models
 - Shared knowledge is authored and managed and communicated through models
 - Documents can be generated from models as needed for archival, or to support suppliers who can't access the model
- Automated analysis in connected tools
 - Reduces uncertainty and latency in decision-making
 - Improves the product, and reduces the risk
- Institutional knowledge captured in curated models and libraries



State of MBSE at JPL

- Significant success in limited scope system modeling (e.g., in ARRM and M2020)
 - Small, but valuable, “baby steps” demonstrated precise, concise communication value of a model
 - Models did not attempt to cover all of the project phase details
- Ambitious modeling effort (e.g., in Europa) has also demonstrated value
 - But hit performance and scalability limits of trying to do everything in one tool (or one model)
- Many modeling tools are in use in SE and subsystem disciplines
 - Tools are weakly connected to each other
 - Performing cross-tool analysis of information is complex
 - Identifying the information baseline across tools is hard
- Focus on tools rather than a coherent information production and management methodology
 - Most often if a methodology is used, it is ill-defined
 - Different methodologies are being used by different projects hampering reuse
 - Some methodologies are not easily supported using the existing tools

Key challenges

Useful system models can be created in SysML, but they are cumbersome to create and manage



We need more **discipline-specific authoring interfaces**

For models to be effective, it has to be easy to present the information in meaningful graphical, and textual views



We need easy to use **reporting tools** to generate graphical and textual views of model information

Many tools and models are used in SE because many discipline-specific analyses are needed



We need effective model **integration tools** and methods to keep models aligned and enable workflow

Model content is rapidly changing as work progresses



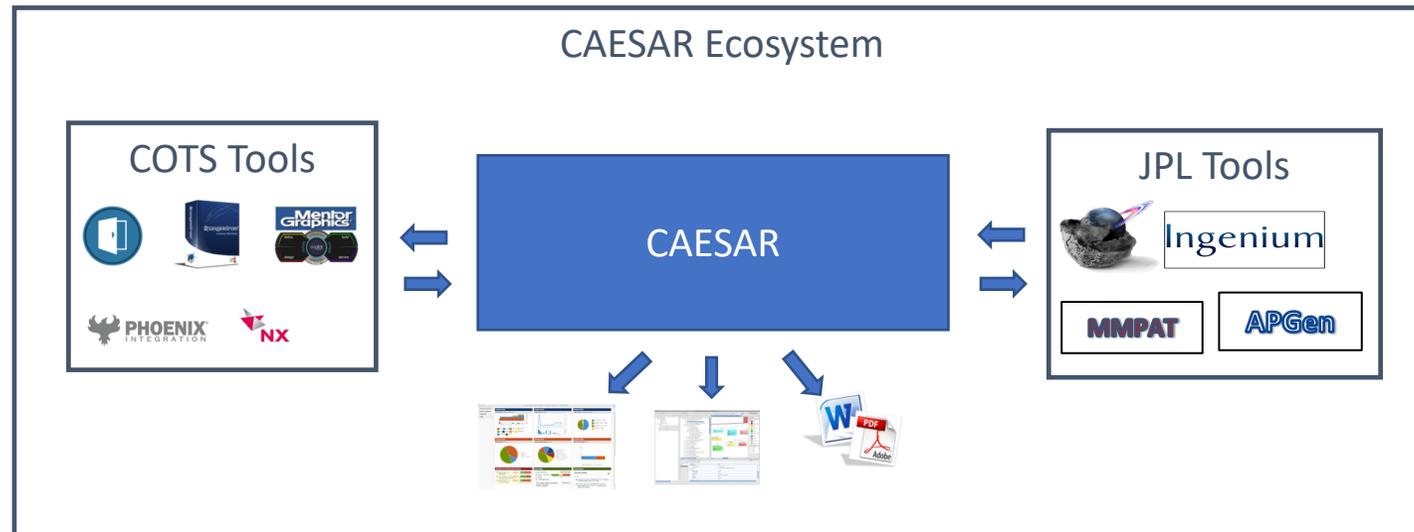
We need effective cross-tool **configuration management**, and validation analysis

Solution objectives

- Re-implementing all the tools (and associated methodology) is a prohibitive task, but...
- We can afford to develop a **platform** that enables a larger community of developers (including vendors) to produce cooperative, reusable, and maintainable tools
- We can also afford to develop some **discipline-focused applications** and associated methodology to validate the platform and provide value to customers

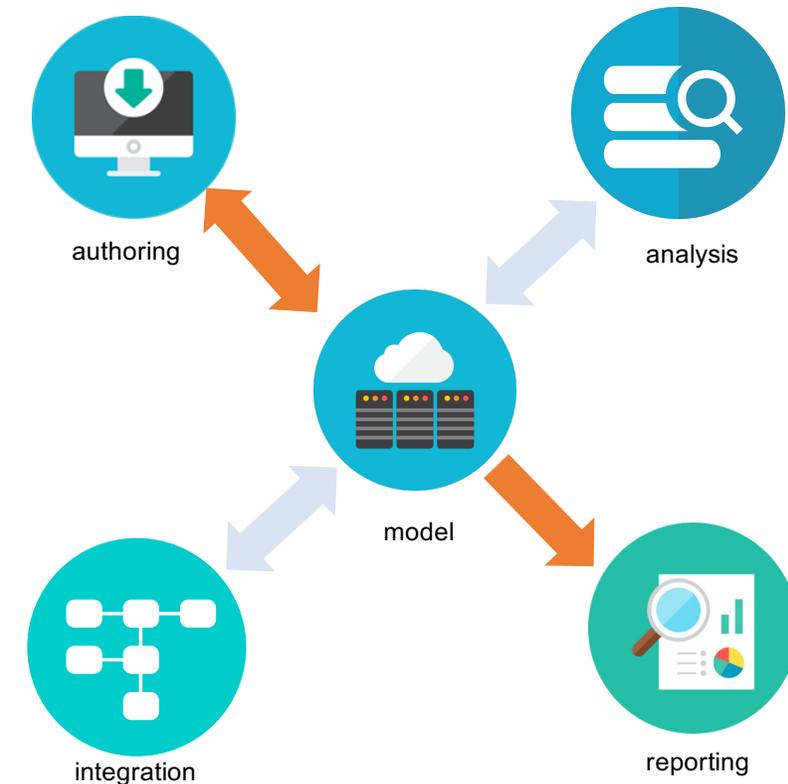
CAESAR concept

- CAESAR stands for **C**omputer **A**ided **E**ngineering for **S**ystems **A**rchitecture
- CAESAR is an platform for transforming current SE practices into rigorous and integrated model-centric engineering processes



CAESAR supported functions

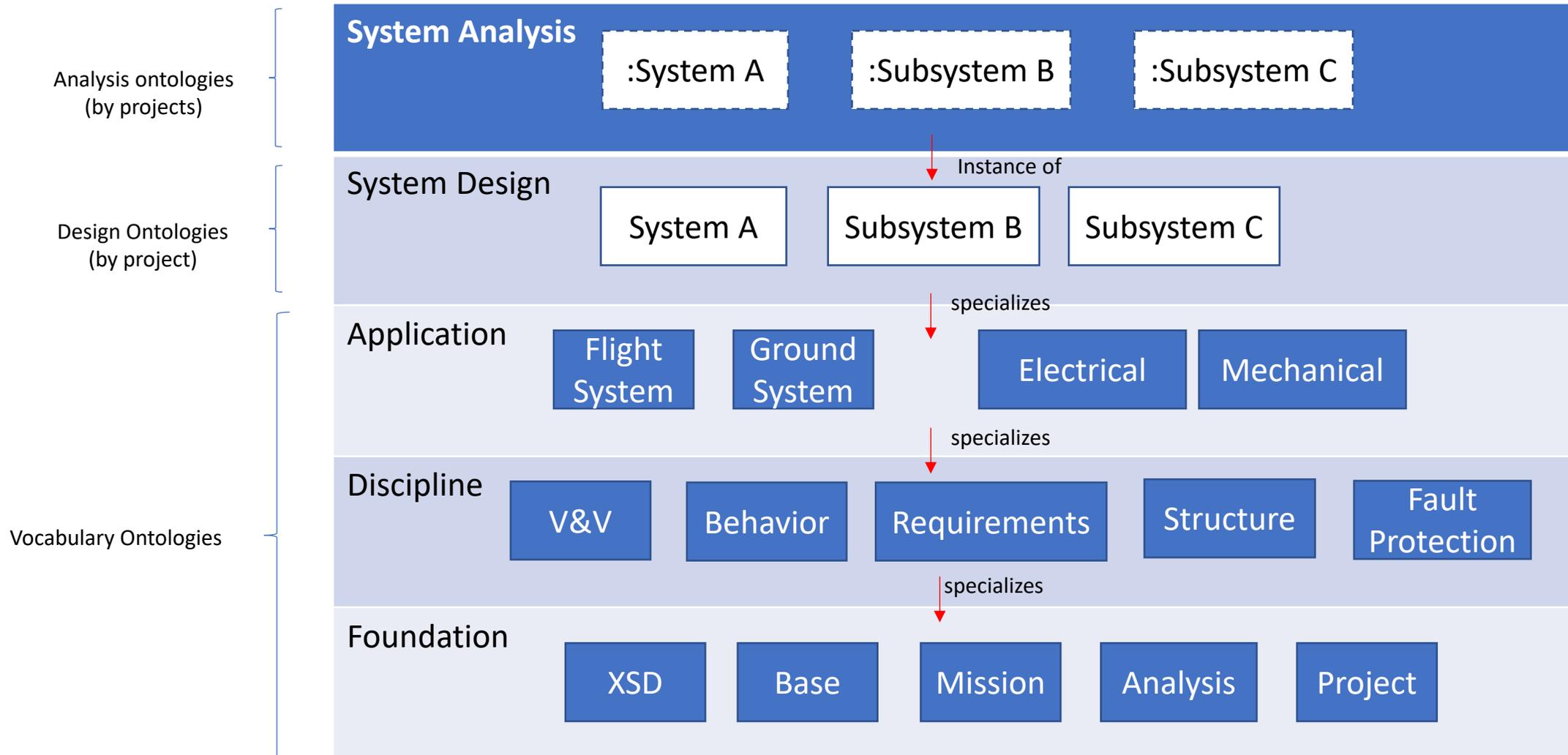
- Information Management
 - Common representation using rigorous semantic vocabularies
 - Configuration management, version control, and governance
 - Curated starting point templates and library models
- Information Authoring
 - Support for building new domain-specific model authoring viewpoints
 - Support for adapting viewpoints in existing authoring tools
- Information Integration
 - Supports the design of integration flows where model fragments can be imported from different data sources in a common format and incrementally compared, analyzed, merged, and managed together
- Information Analysis
 - Support for querying model data and performing various analyses
- Information Reporting
 - Support for publishing and organizing reporting viewpoints, and generating other model representations, and documents



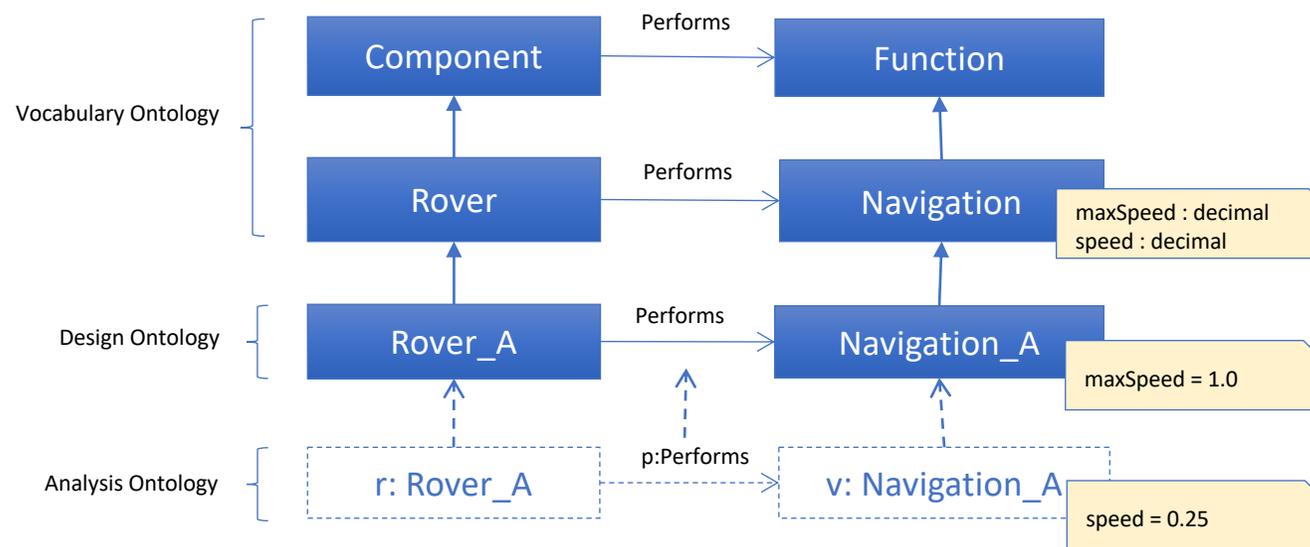
Information management

- CAESAR represents information in OML (Ontology Modeling Language)
 - Allows information to be represented in a tool independent way
 - Is inspired by the semantic web standard OWL2-DL
 - Abstract syntax is defined at a higher level of abstraction than OWL2-DL (easier to express)
 - Semantics is defined by mapping to a subset of OWL2-DL plus SWRL rules (allows reasoning)
 - Supports multiple concrete syntaxes
 - Textual grammar (readability), compressed canonical syntax (SCM friendly), graphical notation
 - Supports multiple API
 - Java, Scala, and JavaScript
 - Has a small core language that is used to define vocabularies (DSLs) as libraries
 - Allows both class-level (design) and instance-level (analysis) modeling
 - Allows both open-world and closed-world assumption modeling
 - Allows information to be componentized in multiple model fragments
 - Allows the same information to be expressed by different authorities

OML Ontology layers



OML Example



```

open terminology <https://imce.jpl.nasa.gov/application/flightsystem> {
  extends <https://imce.jpl.nasa.gov/foundation/mission>
  concept Rover
  Rover extendsConcept mission:Component
  concept Navigation
  Navigation extendsConcept mission:Function
  allEntities Rover . mission:Performs in Navigation
  entityScalarDataProperty maxSpeed {
    domain Navigation
    range XMLSchema:decimal
  }
  entityScalarDataProperty speed {
    domain Navigation
    range XMLSchema:decimal
  }
}

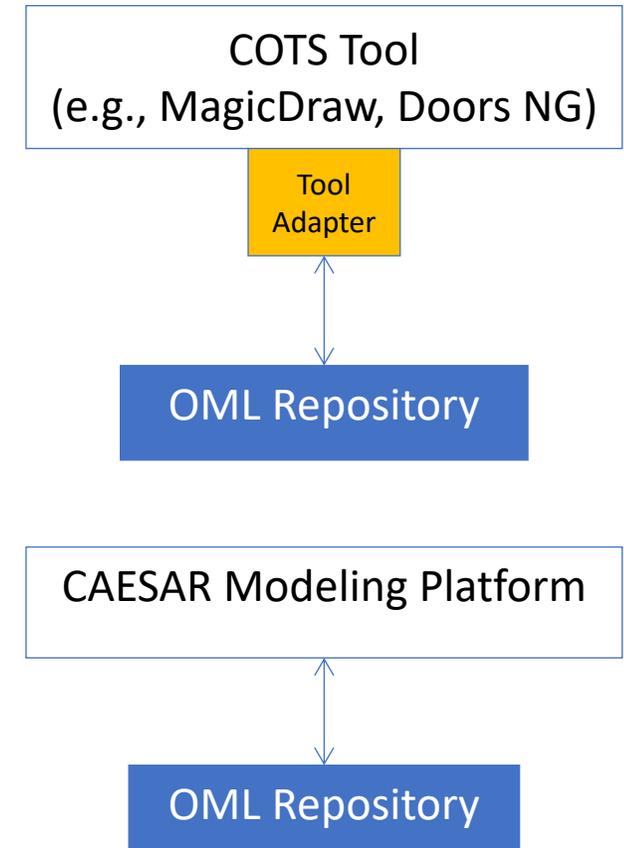
closed terminology <https://imce.jpl.nasa.gov/project/design> {
  extends <https://imce.jpl.nasa.gov/application/flightsystem>
  concept Rover_A
  Rover_A extendsConcept flightsystem:Rover
  concept Navigation_A
  Navigation_A extendsConcept flightsystem:Navigation
  allEntities Rover_A . mission:Performs in Navigation_A
  allData Navigation_A . flightsystem:maxSpeed in 1.0
}

closed description <https://imce.jpl.nasa.gov/project/analysis> {
  extends <https://imce.jpl.nasa.gov/project/design>
  conceptInstance r isA design:Rover_A
  conceptInstance v isA design:Navigation_A
  refidRelationshipInstance p isA mission:Performs
  domain (p) = r
  range (p) = v
  v . flightsystem:speed = 0.25
}

```

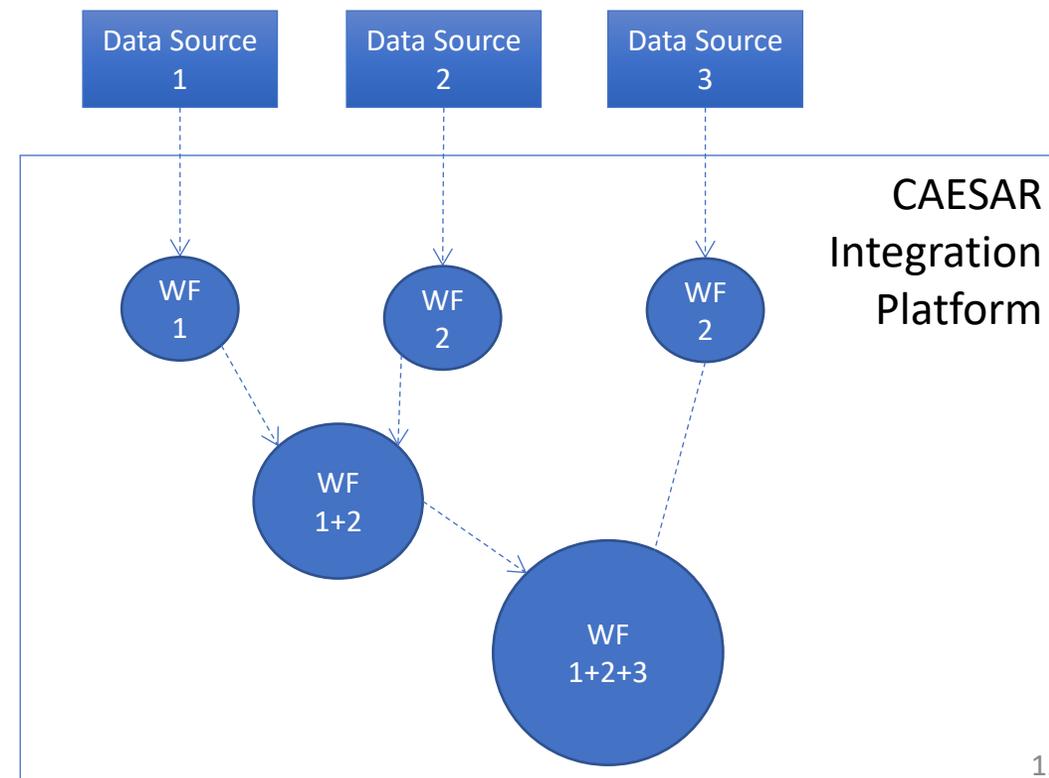
Information authoring

- CAESAR supports authoring information in OML in two ways:
 - Using existing COTS tools (via tool adapters)
 - The adapters map back and forth between the tools' native language and OML
 - The adapters may extend the tool's native language to support the mapping
 - E.g., Map ontology to SysML profiles and use them to model in MagicDraw
 - The subset of information that is mapped is based on the used OML vocabulary
 - The adapters can be run manually or automatically (based on policy)
 - Using the CAESAR Modeling Platform (natively in OML)
 - The platform provides an IDE with custom discipline authoring viewpoints
 - To add missing authoring viewpoints in COTS tools
 - To address non-functional concerns with existing COTS tool viewpoints
- Authored OML information is managed in repositories
 - Supports configuration management capabilities (branches)
 - Supports version control capabilities (version timelines)
 - Supports governance capabilities (version metadata)
 - Ex. which model version from the the COTS tool corresponds to an OML version



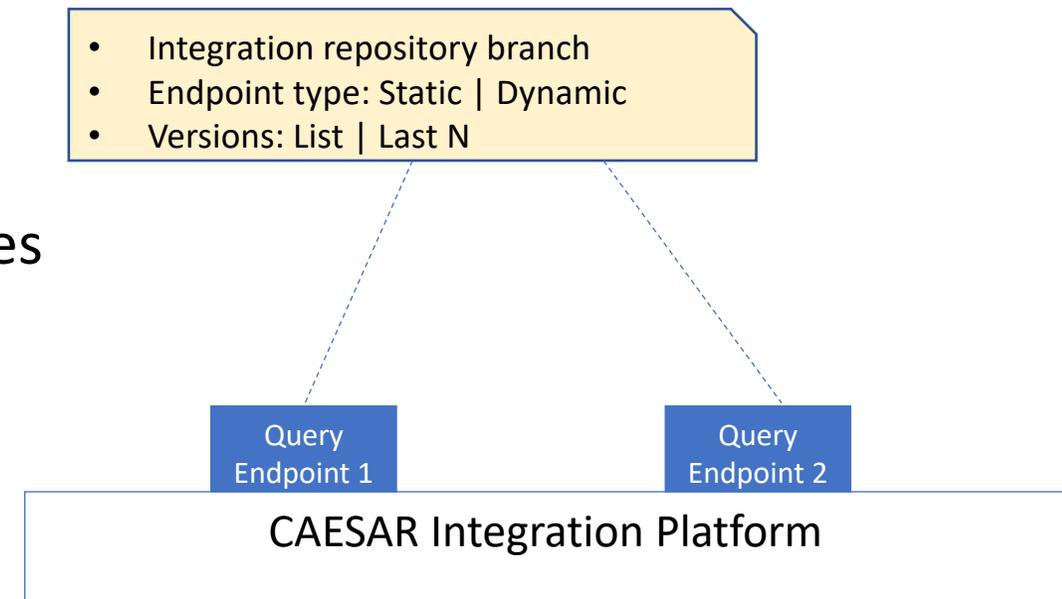
Information integration

- CAESAR supports information integration using the CAESAR Integration Platform
 - Allows integration of multiple independent data sources with OML repositories
 - Allows definition of integration workflows that integrate data sources incrementally
 - Every workflow:
 - merges one or more OML datasets
 - Commits the merged dataset to a unique branch of the integration repository
 - Runs integration analyses on the merged dataset
 - Reports issues resulting from the analyses (if any)
 - If no issues, triggers the next workflow downstream
 - Integration can be triggered when one or more data sources changes (manually or auto)
 - Supports base and variant configurations of data sources (e.g., experimental branches)



Information analysis

- CAESAR supports information analysis using the CAESAR Integration Platform
 - Supports both integration analysis and post-integration analysis
 - Integration analyses may include:
 - Satisfiability analysis (using reasoning)
 - Well-formedness analysis (using queries)
 - Consistency analysis (using queries)
 - Post integration analysis can use arbitrary queries
 - Any integration branch can be analyzed using a query endpoint that is configured with
 - One or more versions from the branch
 - The versions can be static or dynamic
 - Query endpoints accept queries in SPARQL
 - Common queries have dedicated endpoints
 - Query endpoints are public and can be used by external analytics tools

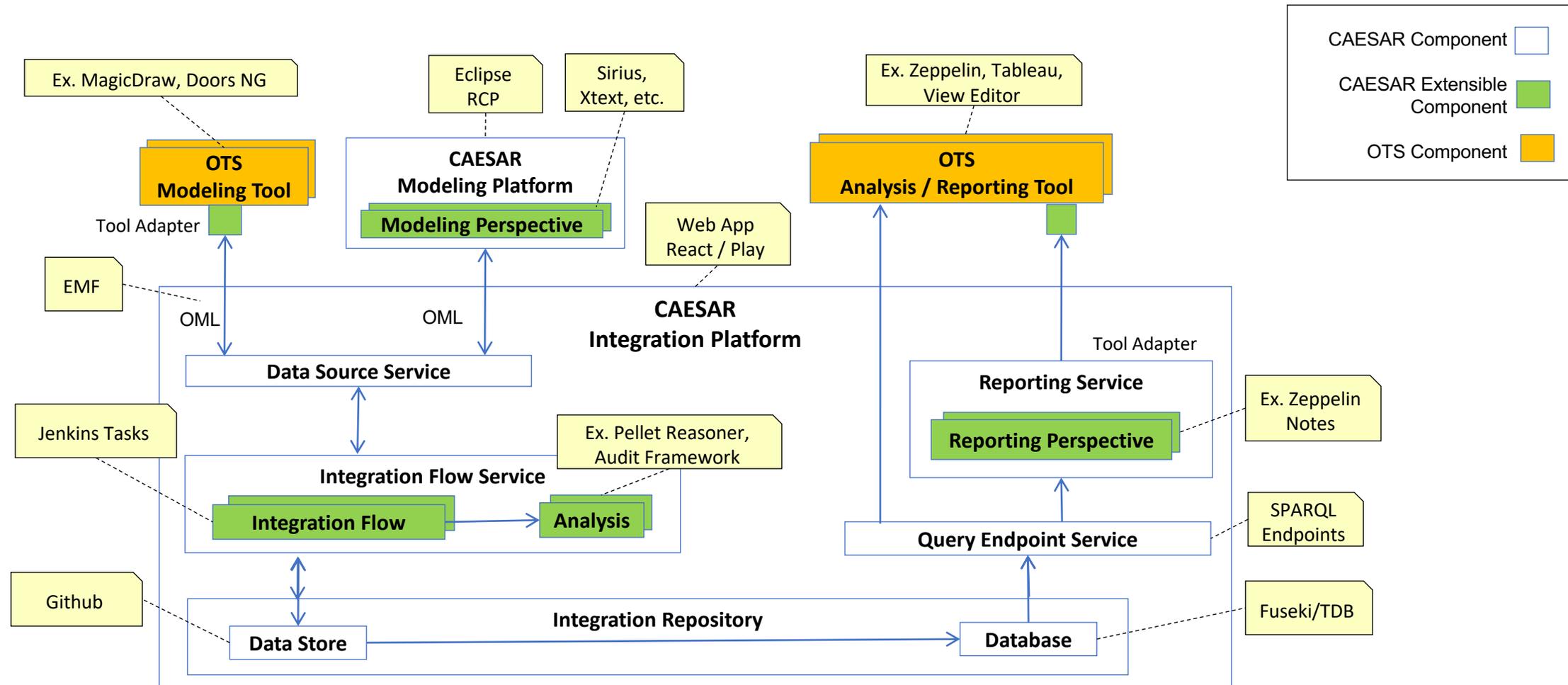


Information reporting

- CAESAR supports information reporting using the CAESAR Integration Platform
 - Supports the ability to generate gate products (reports, documents or models)
 - Gate products can be designed using templates that get bound to a query endpoint dynamically
 - Templates can be designed using a supported analytics tool (through a tool adapter)
 - Templates can be parameterized and those parameters bound in different configurations
 - Gate products can be static or interactive (allowing drilling, navigation, or changing parameter values)
 - Supports the ability to organize gate products in dashboards
 - A dashboard can organize the gate products hierarchically and allow navigation to them
 - Supports the ability to review, comment on, and approve gate products



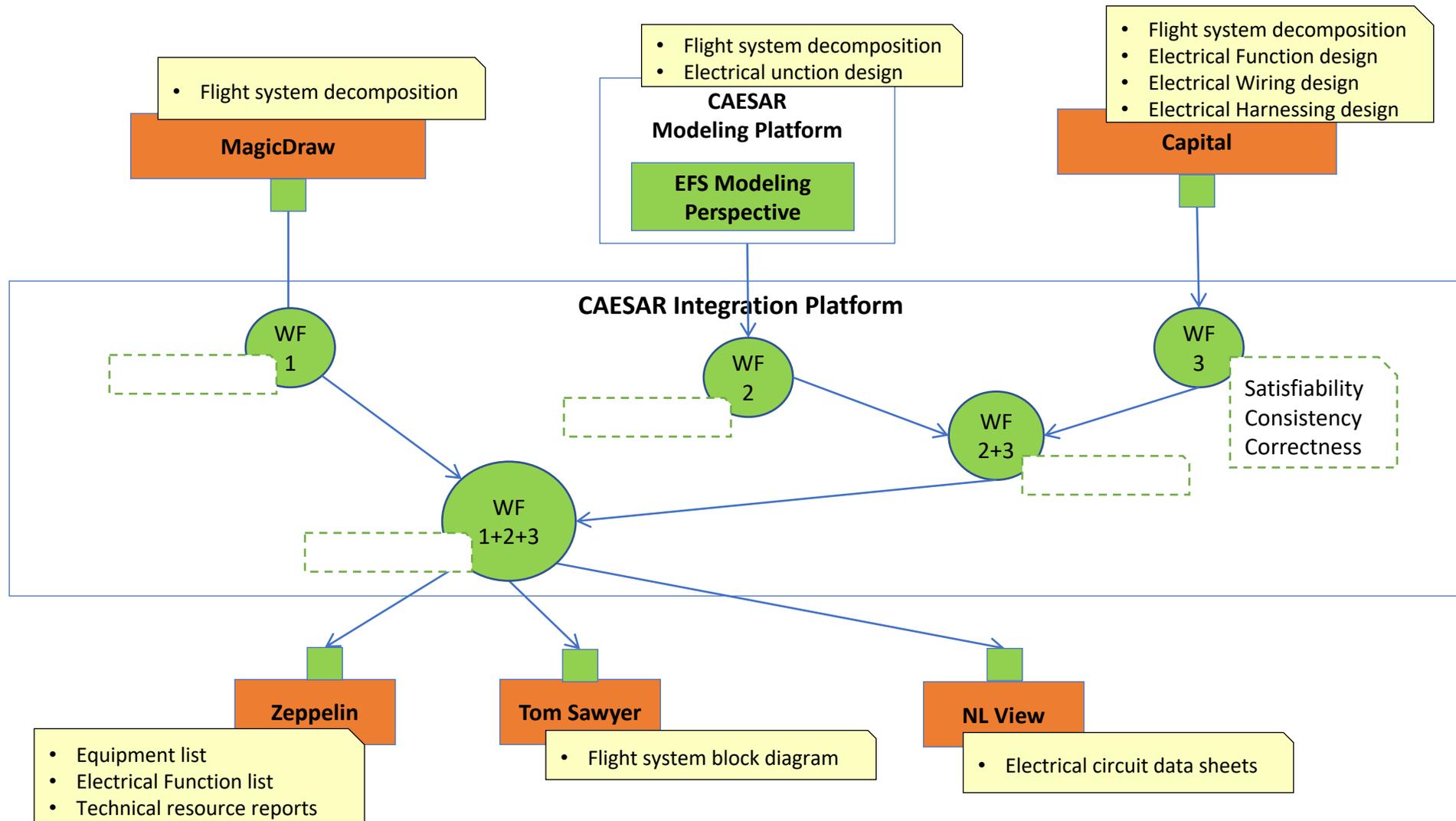
CAESAR system architecture



CAESAR development process

- Work with the domain experts (the line organizations) to implement applications
 - Develop relevant ontologies
 - Identify relevant data sources (and their authoring tools)
 - Develop new authoring viewpoints if needed
 - Develop integration workflows
 - Develop analyses to run
 - Develop gate products to produce
- Work with the projects to configure the applications on the platform
 - Define data sources
 - Configure integration workflows
 - Configure query endpoints
 - Configure report templates

CAESAR Application: flight system electrical engineering



Development strategy

- JPL plans to realize the CAESAR platform incrementally over time
- JPL has open sourced some components of CAESAR
 - OML API and tools (<https://github.com/JPL-IMCE/gov.nasa.jpl.imce.oml>)
- JPL is open to discussing collaboration on CAESAR
- JPL likes to work with vendors to implement tool adapters