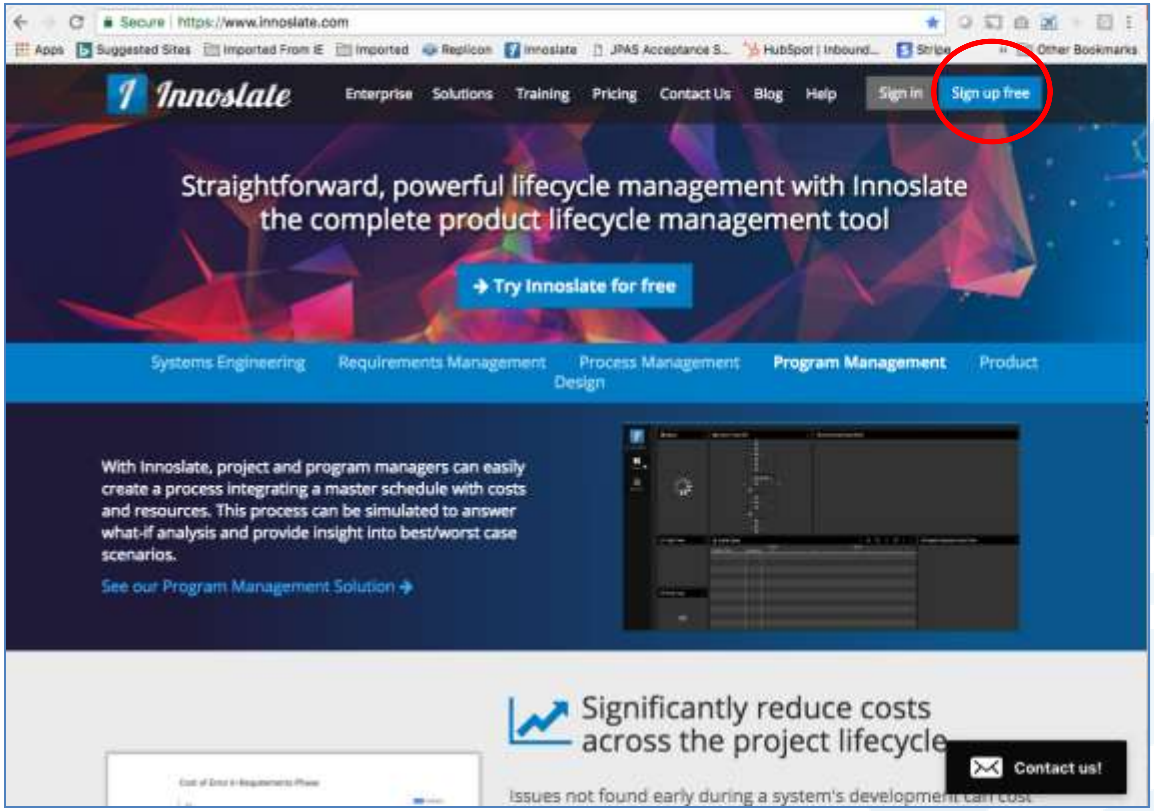
Introduction to Model-Based Systems Engineering (MBSE) and Innoslate®

Steven H. Dam, Ph.D., ESEP
President, SPEC Innovations
571-485-7805
steven.dam@specinnovations.com

SPEC INNOVATIONS

Sign Up at Innoslate.com

- www.innoslate.com
- Activate with your email



Overview

- Introductions

1. Overview of Model-Based Systems Engineering
2. Introduction to the Lifecycle Modeling Language
3. Overview of Innoslate's Capabilities

INTRODUCTIONS

Purpose

- To introduce model-based systems engineering using Innoslate®
- Provide a starting point for learning how to use Innoslate®

Class Guidelines

- Discussion…plus group participation
- Avoid interrupting others
- Ask questions as they arise

SPEC INNOVATIONS

Introductions

- Steve Dam, Ph.D.
  - 40+ years of software development and systems engineering experience
  - Certified INCOSE Expert Systems Engineering Professional
  - Extensive experience in developing broad reaching and detailed architectures and systems engineering design across DoD, DOE, and NASA
  - Author of the books: *DoD Architecture Framework: A Guide to Applying System Engineering to Develop Integrated, Executable Architectures* and *Proposal Engineering: A Guide to Developing Winning, Cost-Effective Proposals*
  - Co-author (with Dr. Dinesh Verma) of Chapter 3, Concept of Operations and System Operational Architecture, *Applied Space Systems Engineering*

Who's Here?

- Name?
- Organization?
- Familiar with Systems Engineering?
- Familiar with MBSE?

1. OVERVIEW OF MODEL-BASED SYSTEMS ENGINEERING

SPEC INNOVATIONS

Views of Systems Engineering

Technical Orientation

A comprehensive, iterative problem-solving process that is used to:

- Transform customer requirements into a solution set
- Generate information for decision-makers
- Provide information for the next phase

– ***MIL-STD-499B (DRAFT)***

Management Orientation

"… the management function which controls the total system development effort for the purpose of achieving an optimum balance of all system elements…"

– ***DSMC Handbook***

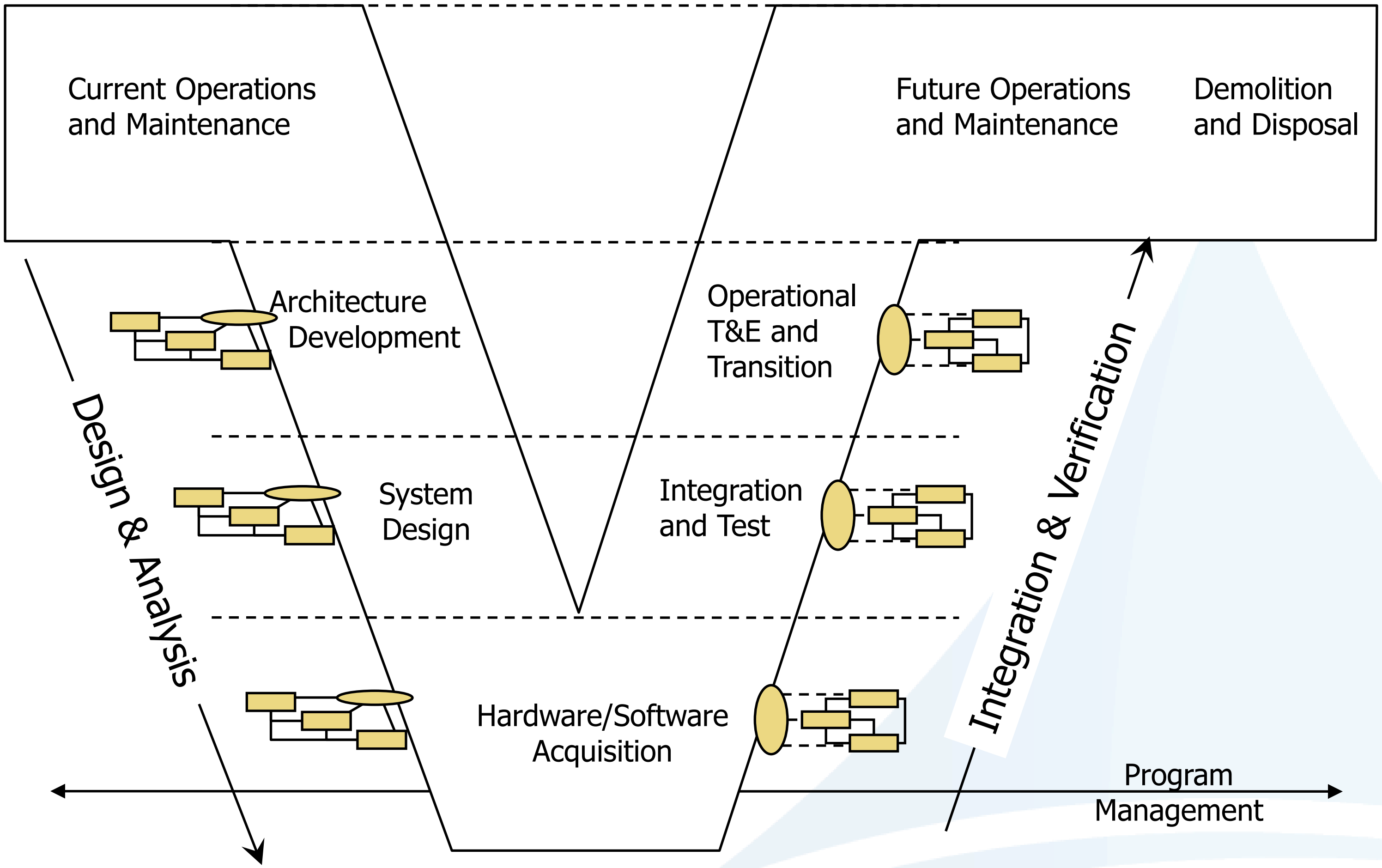
SPEC INNOVATIONS

Systems Engineering [INCOSE]

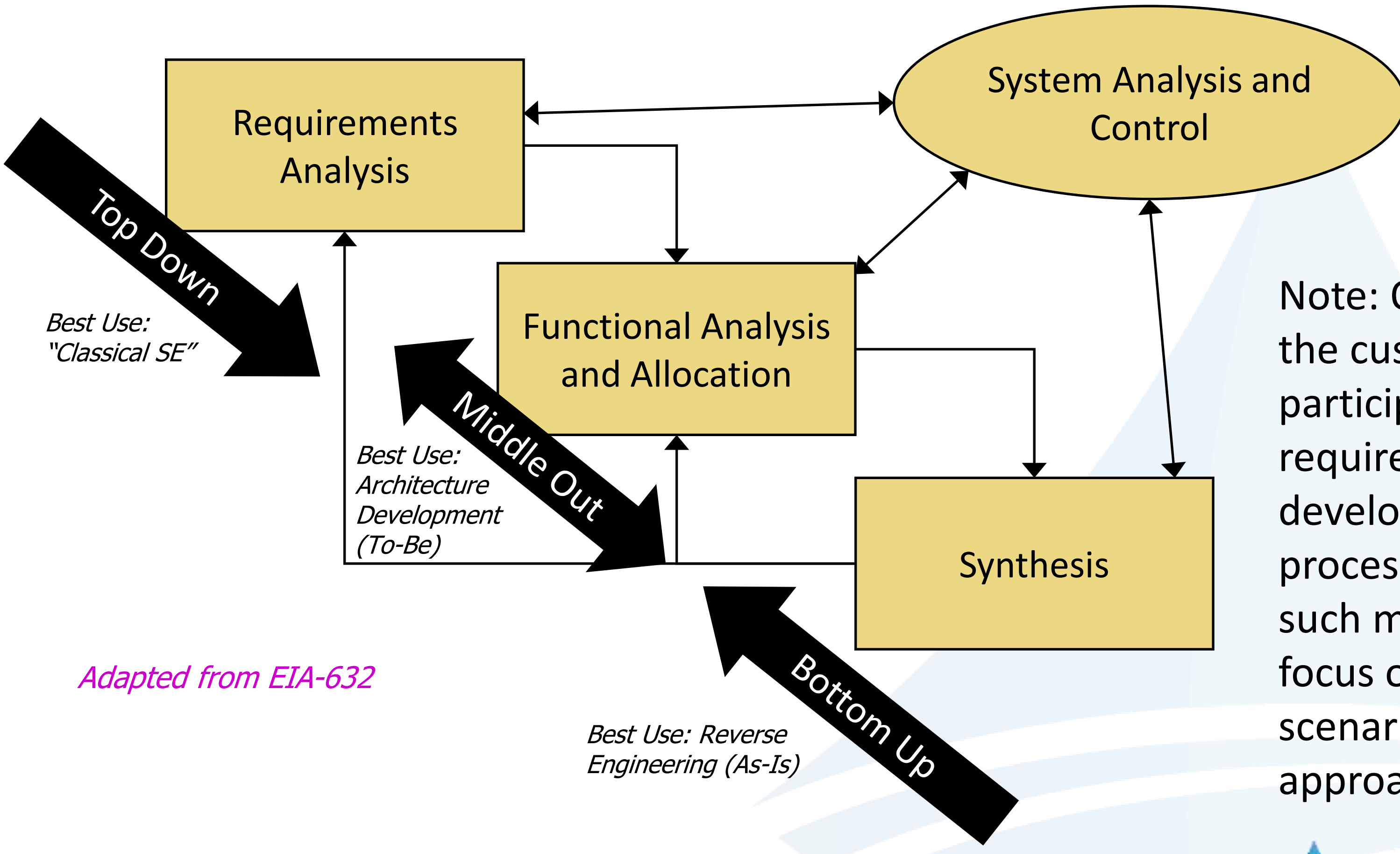
- Systems Engineering is an interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem:
  - Operations
  - Performance
  - Test
  - Manufacturing
  - Cost & Schedule
  - Training & Support
  - Disposal
- Systems Engineering integrates all the disciplines and specialty groups into a team effort forming a structured development process that proceeds from concept to production to operation. Systems Engineering considers both the business and the technical needs of all customers with the goal of providing a quality product that meets the user needs.

SPEC INNOVATIONS

The Lifecycle



Systems Engineering During Design Phase



Note: On the cloud the customer participate in the requirements development process and as such may want to focus on a scenario-based approach

What Is a Model?

- A model:
  - represents reality
    - By only "representing reality" it means that we are simplifying reality as it would take the Universe to model the Universe completely
    - "Essentially, all models are wrong, some models are useful"

https://en.wikipedia.org/wiki/George_E._P._Box accessed 9/23/2015

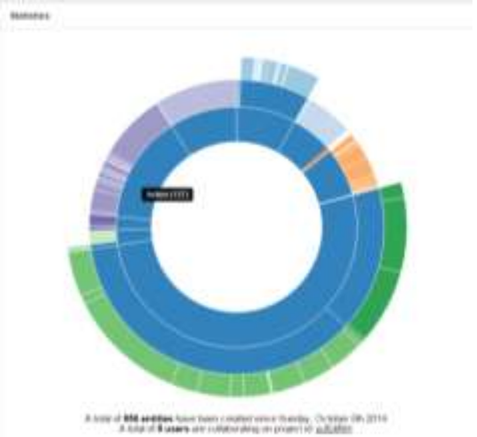
- By useful, a model must meet the needs of both the developers of the model and their audience

George Edward Pelham Box (1919-2013)

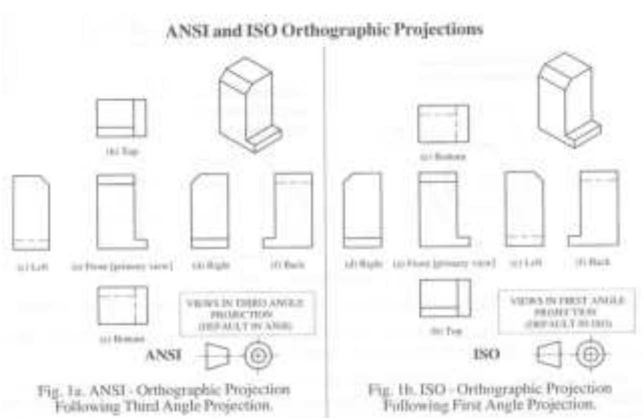
What a Model Means to Me

- As a SE I Need:
  - A way to describe the system and its environment as simply as possible to make it understandable
    - In words AND
    - In pictures
  - AND verifiable through computable representations
- This model becomes a tool for me to describe to other stakeholders the system: what, why, where, when, how, how much, etc.
  - Ultimately, I need to be able to create a specification that can be used to buy or build the system and demonstrate to the owners and users of the system that is meets their needs

Drawings and Computable Models



- A drawing is a type of modeling approach
- A drawing consists of:
  - lines and boxes
  - text
  - pictures
- Drawings are static



- A computable model includes:
  - data that defines the system
  - it may include drawings to visualize the information
  - most “drawings” from a model result as visualizations of the data

Drawings Are Models, But …

- Drawings are useful visualizations of information
- But drawings are:
  - difficult to test
  - require extensive sets of rules
  - provide only 2-4 dimension of information including relationships effectively
- It would require a very large number of drawings to represent all the dimensions in a system model and extensive coordination to keep them consistent

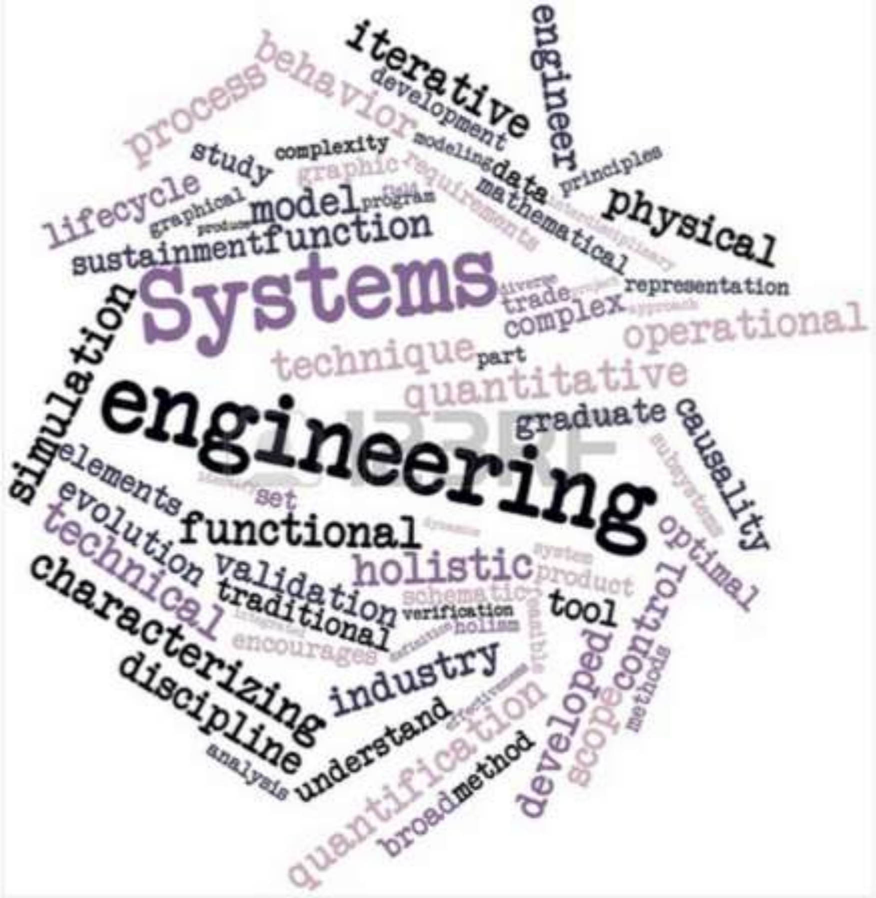
Computable Model Features

- Models provide traceability in many dimensions
- Models can be automatically validated using simulation and rule checkers
- Models can be interrogated
- Models can be reused
- Models can be easily changed

SPEC INNOVATIONS

Model-Based Systems Engineering Definition

- “Model-based systems engineering (MBSE) is the ***formalized application of modeling*** to support system requirements, design, analysis, verification and validation, beginning in the conceptual design phase and continuing throughout development and later life cycle phases.” - INCOSE



What the Definition Tells Us

- By this definition, MBSE has been around for a long time – every since someone drew the first flowchart
- More recently it has come to mean a way to capture the essential elements of information in a database and then visualize that information in many different ways, including the production of the many SE documents needed (plans, specifications, risk reports, etc.)
- Many systems engineers equate SysML with MBSE – we disagree

SPEC INNOVATIONS

Perhaps a Better Definition

- Model-Based Systems Engineering is the formalized application of modeling (static and dynamic) to support system design and analysis, throughout all phases of the system lifecycle, through the collection of modeling languages, structures, model-based processes, and presentation frameworks used to support the discipline of systems engineering in a model-based or model-driven context. The four tenets of this definition are:
  - Modeling Languages
  - Structure
  - Model-Based Processes
  - Presentation Frameworks

From "Proceedings of the IEEE Systems Conference, 2016," W. Vaneman, Ph.D.

Hence, now MBSE isn't just automated SE business as usual – its something new, improved, and valued by the community

SPEC INNOVATIONS

2. INTRODUCTION TO THE LIFECYCLE MODELING LANGUAGE

Lifecycle Modeling Language (LML)

- LML combines the logical constructs with an ontology to capture information
  - SysML – mainly constructs – limited ontology
  - DoDAF Metamodel 2.0 (DM2) ontology only
- LML simplifies both the “constructs” and ontology to make them more complete, yet easier to use
- Goal: A language that works across the full lifecycle

LML Ontology* Overview

*Ontology = Taxonomy + relationships among terms and concepts
** Taxonomy = Collection of standardized, defined terms or concepts

- Taxonomy**:
  - 12 primary element classes
  - Many types of each element class
    - Action (types = Function, Activity, Task, etc.)
- Relationships: almost all classes related to each other and themselves with consistent words
  - Asset performs Action/Action performed by Asset
  - Hierarchies: decomposed by/decomposes
  - Peer-to-Peer: related to/relates

SPEC INNOVATIONS

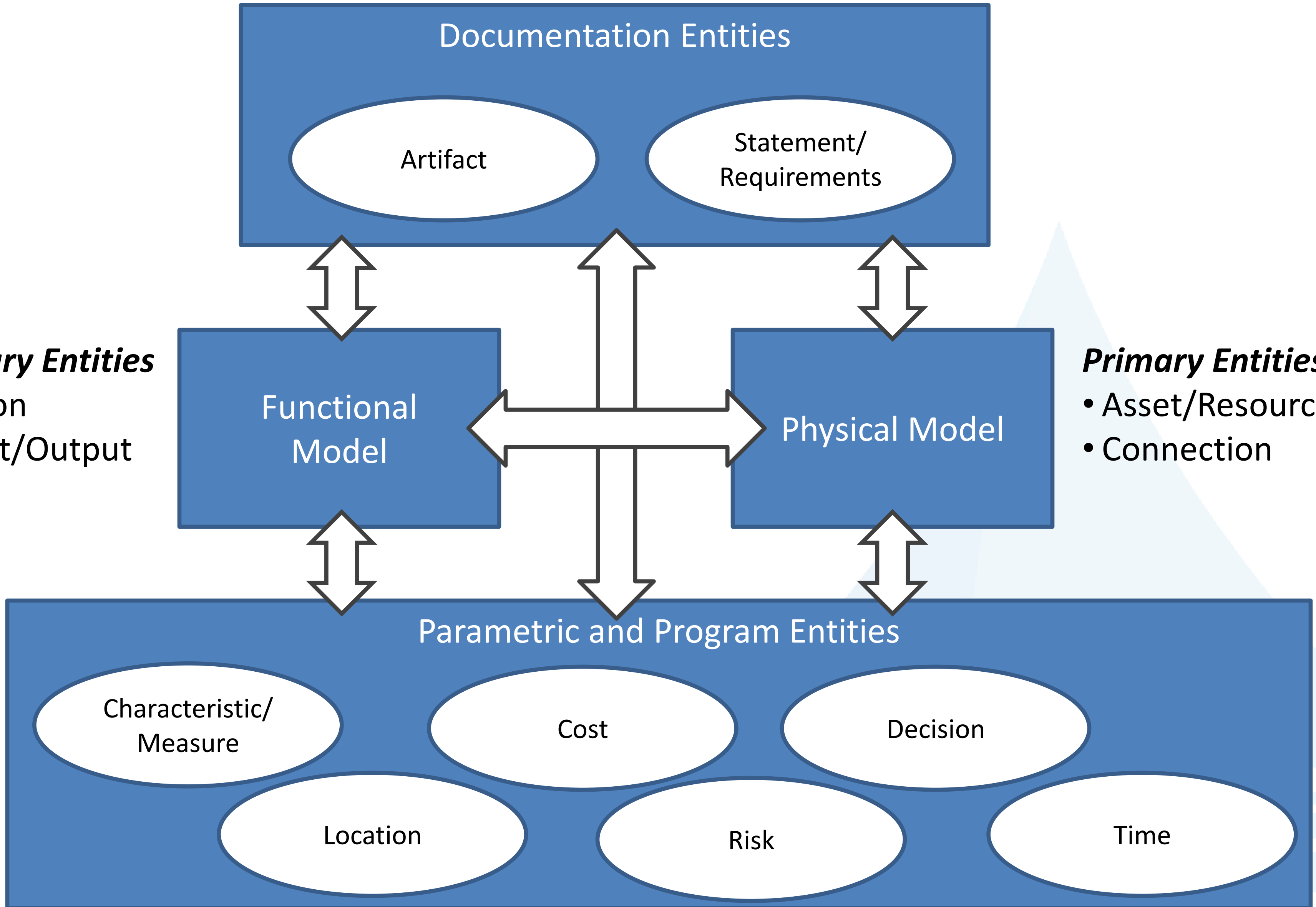
LML's Simplified Schema

- Action
- Artifact
- Asset
  - Resource
- Characteristic
  - Measure
- Connection
  - Conduit
  - Logical
- Cost
- Decision
- Input/Output
- Location
  - Physical, Orbital, Virtual
- Risk
- Statement
  - Requirement
- Time

Supports capturing information throughout the lifecycle

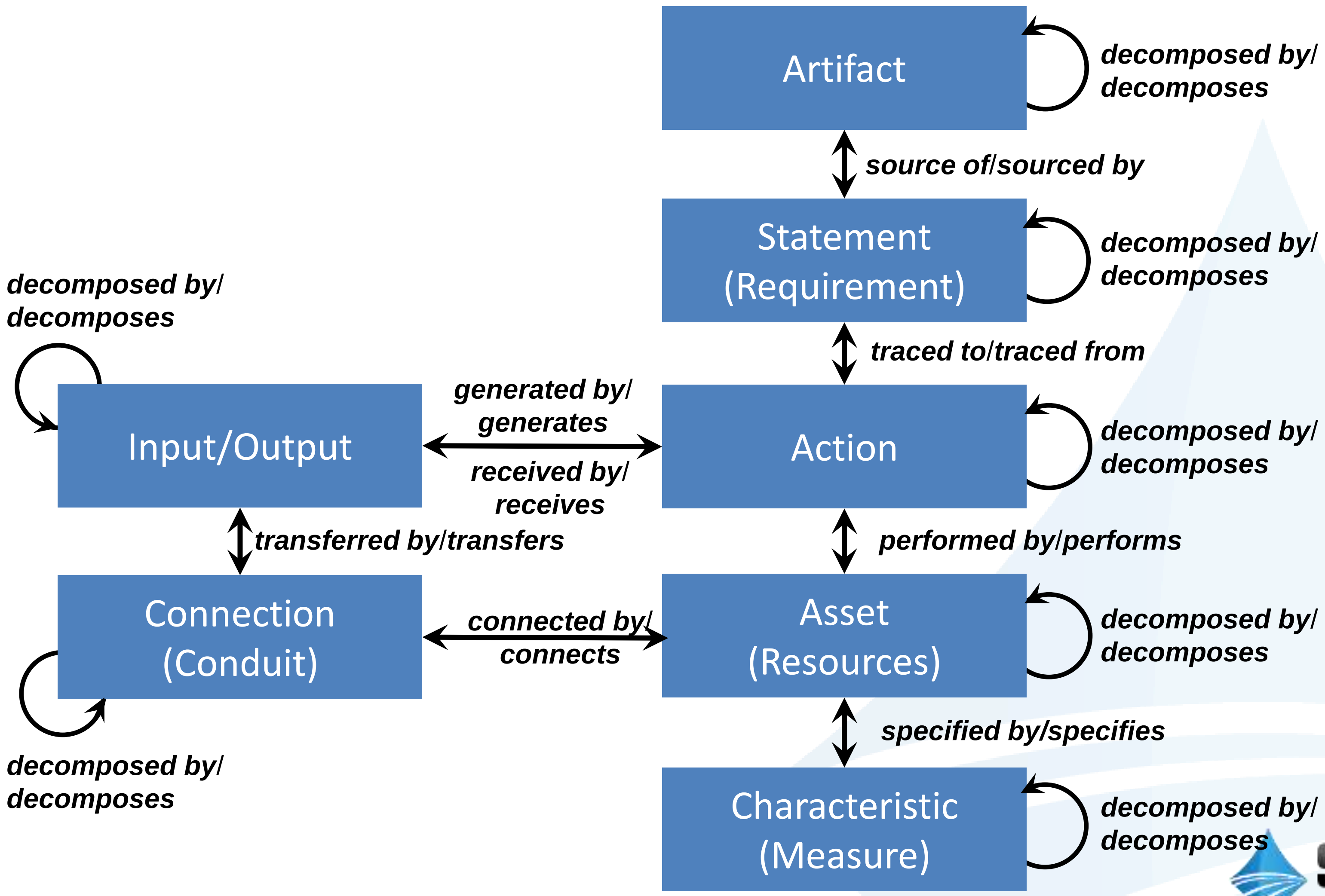
LML Models

Primary Entities
- Action
- Input/Output



Primary Entities
- Asset/Resource
- Connection

LML Primary Entities and Relationships for SE Support



LML Relationships Provide Linkage Needed Between the Classes

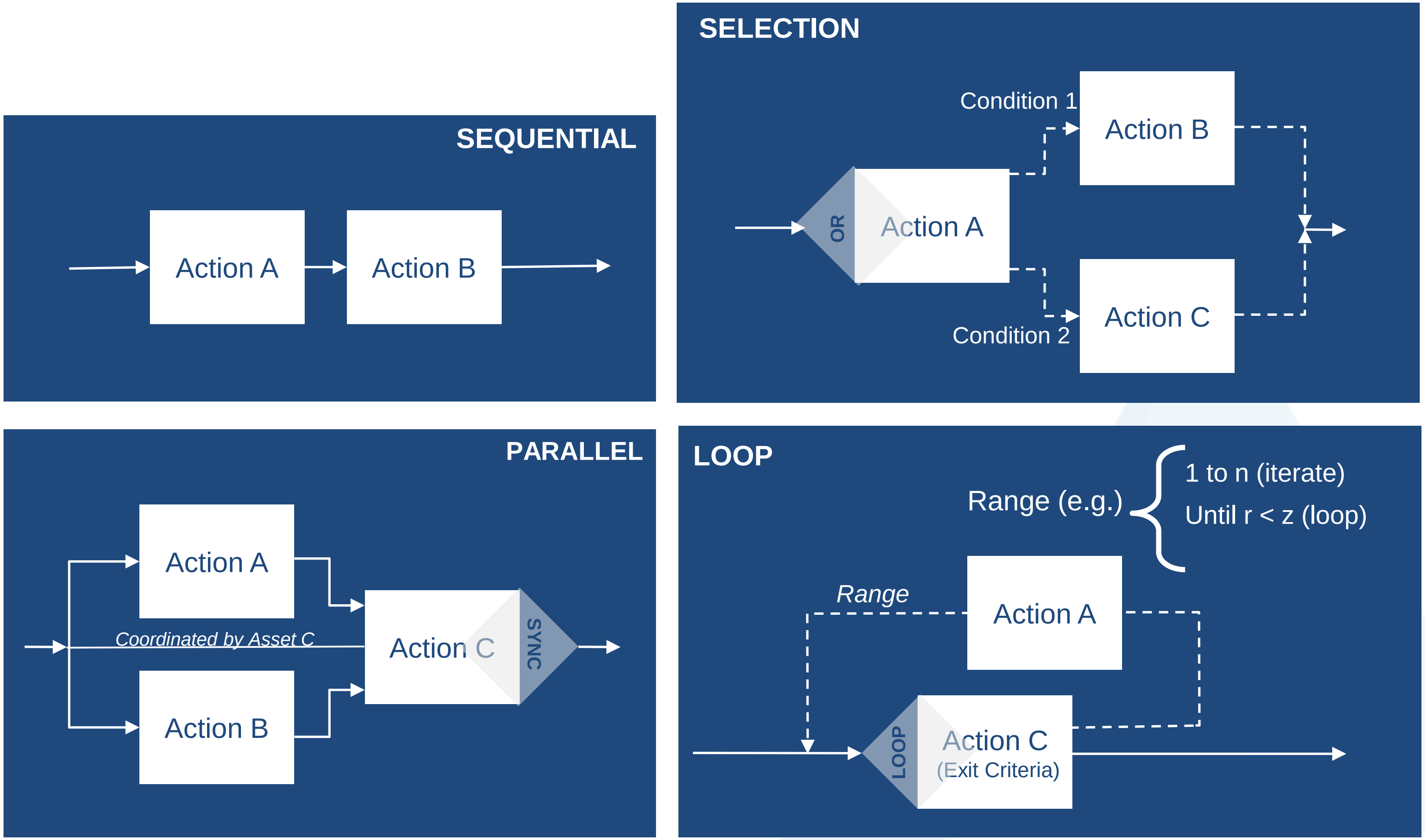
- decomposed by/decomposes
- orbited by/orbits
- related to/relates

	Action	Artifact	Asset (Resource)	Characteristic (Measure)	Connection (Conduit, Logical)	Cost	Decision	Input/Output	Location (Orbital, Physical, Virtual)	Risk	Statement (Requirement)	Time
Action	decomposed by* related to*	references	(consumes) performed by (produces) (seizes)	specified by	-	incurs	enables results in	generates receives	located at	causes mitigates resolves	(satisfies) traced from (verifies)	occurs
Artifact	referenced by	decomposed by* related to*	referenced by	referenced by specified by	defines protocol for referenced by	incurs referenced by	enables referenced by results in	referenced by	located at	causes mitigates referenced by resolves	referenced by (satisfies) source of traced from (verifies)	occurs
Asset (Resource)	(consumed by) performs (produced by) (seized by)	references	decomposed by* orbited by* related to*	specified by	connected by	incurs	enables made responds to results in	-	located at	causes mitigates resolves	(satisfies) traced from (verifies)	occurs
Characteristic (Measure)	specifies	references specifies	specifies	decomposed by* related to* specified by*	specifies	incurs specifies	enables results in specifies	specifies	located at specifies	causes mitigates resolves specifies	(satisfies) spacifies traced from (verifies)	occurs specifies
Connection (Conduit, Logical)	-	defined protocol by references	connects to	specified by	decomposed by* joined by* related to*	incurs	enables results in	transfers	located at	causes mitigates resolves	(satisfies) traced from (verifies)	occurs
Cost	incurred by	incurred by references	incurred by	incurred by specified by	incurred by	decomposed by* related to*	enables incurred by results in	incurred by	located at	causes incurred by mitigates resolves	incurred by (satisfies) traced from (verifies)	occurs
Decision	enabled by result of	enabled by references result of	enabled by made by responded by result of	enabled by result of specified by	enabled by result of	enabled by incurs result of	decomposed by* related to*	enabled by result of	located at	causes enabled by mitigated by result of resolves	alternative enabled by traced from result of	date resolved by decision due occurs
Input/Output	generated by received by	references	-	specified by	transferred by	incurs	enables results in	decomposed by* related to*	located at	causes mitigates resolves	(satisfies) traced from (verifies)	occurs
Location (Orbital, Physical, Logical)	locates	locates	locates	locates specified by	locates	locates	locates	locates	decomposed by* related to*	locates mitigates	locates (satisfies) traced from (verifies)	occurs
Risk	caused by mitigated by resolved by	caused by mitigated by references resolved by	caused by mitigated by resolved by	caused by mitigated by resolved by specified by	caused by mitigated by resolved by	caused by incurs mitigated by resolved by	caused by enables mitigated by results in resolved by	caused by mitigated by resolved by	located at mitigated by	caused by* decomposed by* related to* resolved by*	caused by mitigated by resolved by	occurs mitigated by
Statement (Requirement)	(satisfied by) traced to (verified by)	references (satisified by) sourced by traced to (verified by)	(satisfied by) traced to (verified by)	(satisified by) specified by traced to (verified by)	(satisfied by) traced to (verified by)	incurs (satisified by) traced to (verified by)	alternative of enables traced to results in	(satisfied by) traced to (verified by)	located at (satisfied by) traced to (verified by)	causes mitigates resolves	decomposed by* traced to* related to*	occurs (satisified by) (verified by)
Time	occurred by	occurred by	occurred by	occurred by specified by	occurred by	occurred by	date resolves decided by occurred by	occurred by	occurred by	occurred by mitigates	occurred by (satisfies) (verifies)	decomposed by* related to*

Diagrams Are Needed for Every Class

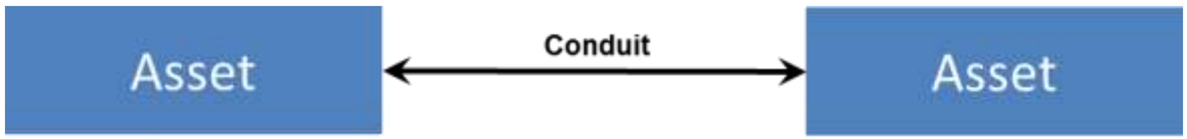
- Action Diagram (Mandatory)
- Asset Diagram (Mandatory)
- Spider Diagram (Mandatory)
- Interface Diagrams
  - N2 (Assets or Actions)
- Hierarchy Diagrams
  - Automatically color coded by class
- Time Diagrams
  - Gantt Charts
  - Timeline Diagram
- Location Diagrams
  - Maps for Earth
  - Orbital charts
- Class/Block Definition Diagram
  - Data modeling
- Risk Chart
  - Standard risk/opportunity chart
- Organization Charts
  - Showing lines of communication, as well as lines of authority
- Pie/Bar/Line Charts
  - For cost and performance
- Combined Physical and Functional Diagram

Action Diagram (Mandatory)

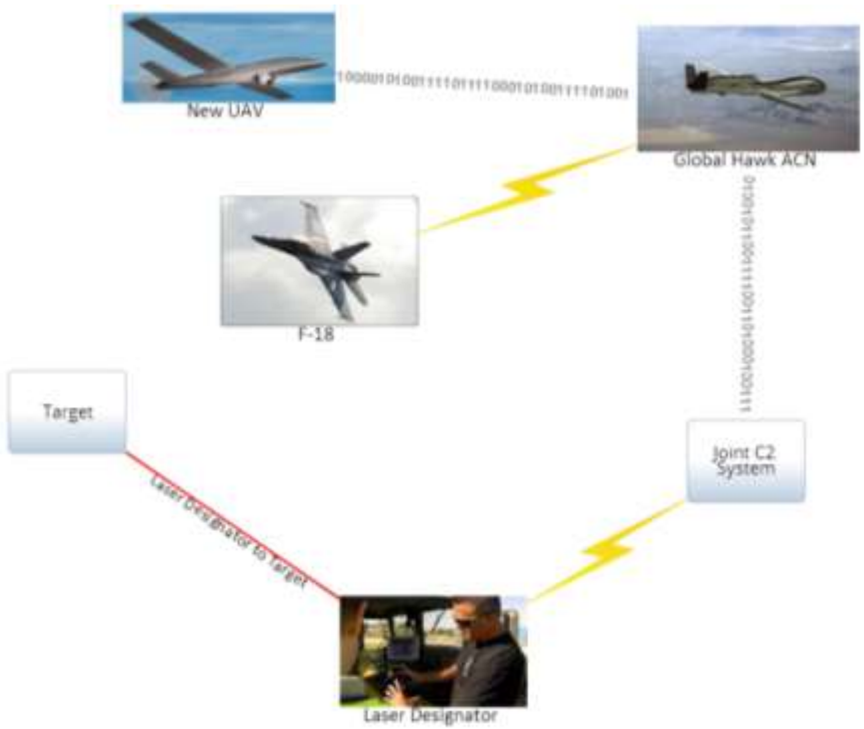


No constructs – only special types of Actions – ones that enable the modeling of command and control/ information assurance to capture the critical decisions in your model

Asset Diagram (mandatory)

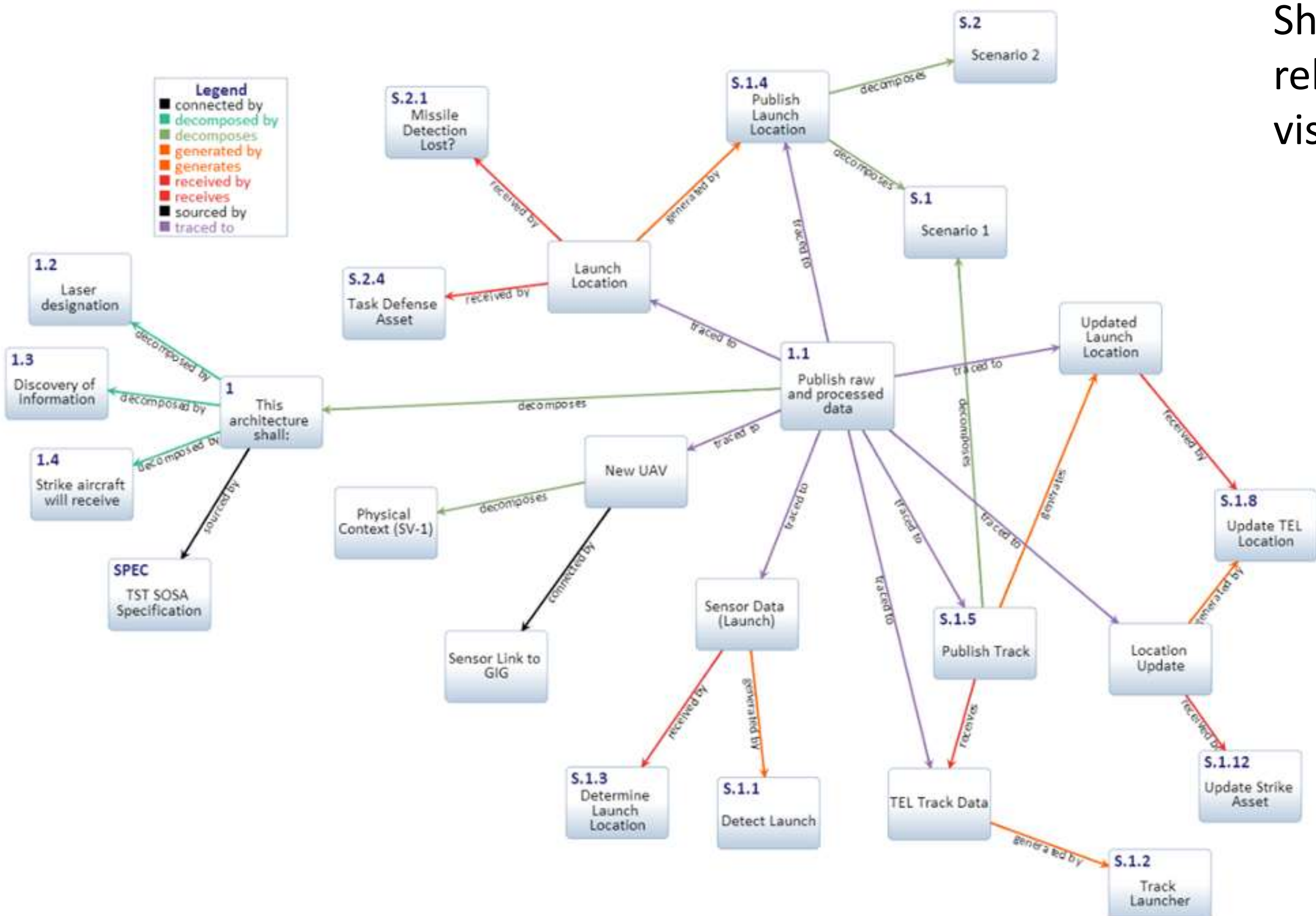


Block diagram general form



Block diagram using pictures

Spider Diagram (Mandatory for Traceability)



Shows entities and relationships in visual form

LML Translation

- Two types of mapping for tailoring:
  - Map names of classes to enable other “schema” models to be used
  - Map symbols used (e.g., change from LML Logic to Electrical Engineering symbols)
  - Enable diagram translations (e.g., Action Diagram to IDEF 0)

LML Class	DM2	SysML	…
Action	Activity	Activity	
Asset	Performer	Actor	

LML Symbol	Electrical Engineering	BPMN	…
			

LML Summary

- LML provides the fundamental foundation for a tool to support SE
- LML contains the basic technical and programmatic classes needed for the lifecycle
- LML defines the Action Diagram to enable better definition of logic as functional requirements
- LML uses Physical Diagram to provide for abstraction, instances, and clones, thus simplifying physical models
- LML provides the “80% solution”
  - It can be extended to meet specific needs (e.g. adding Question and Answer classes for a survey tool that feeds information into the modeling)

SPEC INNOVATIONS

3. OVERVIEW OF INNOSLATE'S CAPABILITIES

The Innoslate® Solution

Simplicity

- Built-in help and common options
- Primary tool language is easy to learn
- Web browser user interface

Collaboration

- Real-Time Collaboration
- Easy Communication to other engineers

Accuracy

- Full Discrete Event Simulator which simulates cost, schedule, and performance
- Full Monte Carlo Simulator to simulate variance
- Intelligence View and Requirements Quality Checker

Scalability

- Tested to over 10 Million entities and 1,000 simultaneous users

Full Lifecycle Management Support

- Includes full requirements capability with Requirements View
- Includes full modeling capability (SysML/LML/IDEF0)
- Includes Test Center, CONOPS, Project Plan, and Test Plan document views

Interoperability

- CAD Integration
- Automatically generate and/or use other representations (SysML, DoDAF)
- Import from other RM and modeling tools (Word DOCX, DOORS CSV, Excel CSV, XMI)
- Integration with 700+ tools through Zapier (GitHub/Jira etc)

SPEC INNOVATIONS

System Requirements

- Platform Independent
  - Works on Windows XP/7/8/RT, MAC OS X, Linux, iOS, Android
- Software
  - Any modern web browser

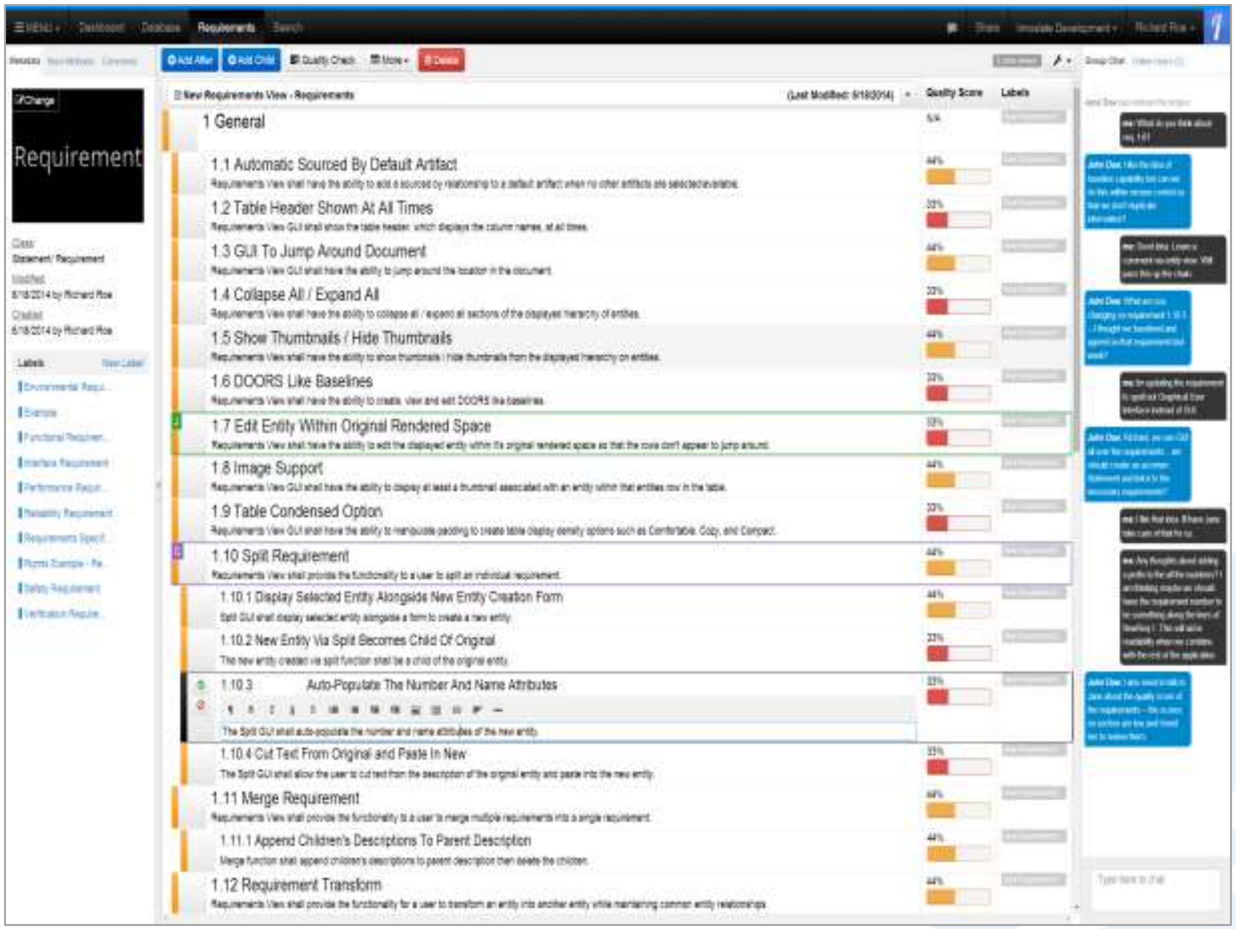
  (Google Chrome, Mozilla Firefox, Safari, IE 10 or 11)
- No downloads required

SPEC INNOVATIONS

Innoslate Security

- All connections are SSL encrypted in transit
- New files uploaded are 256bit AES encrypted at rest
- All developers in Northern Virginia
- Public cloud provider has the following security certifications:
  - ISO 27001:2005
  - SAS70 Type II
  - SSAE 16 Type II
  - ISAE 3402 Type II
  - FedRAMP
- On-site version can be deployed locally behind your firewall

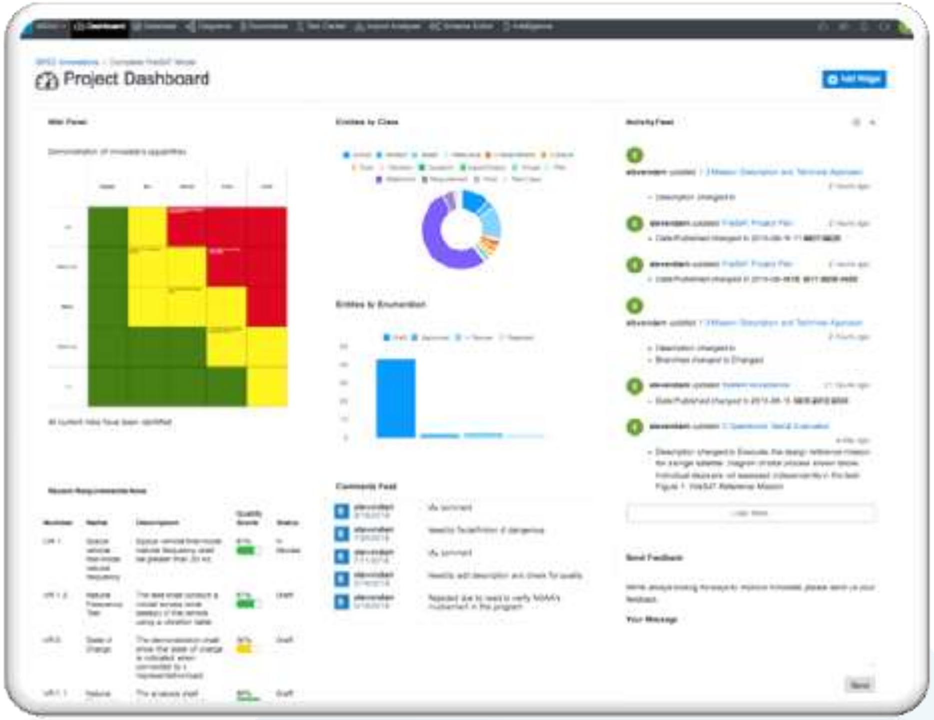
Cross View Real-time Collaboration



- Collaborate with your team members across multiple views
- Simultaneously shows real-time user status
- Group chat
- Project notifications

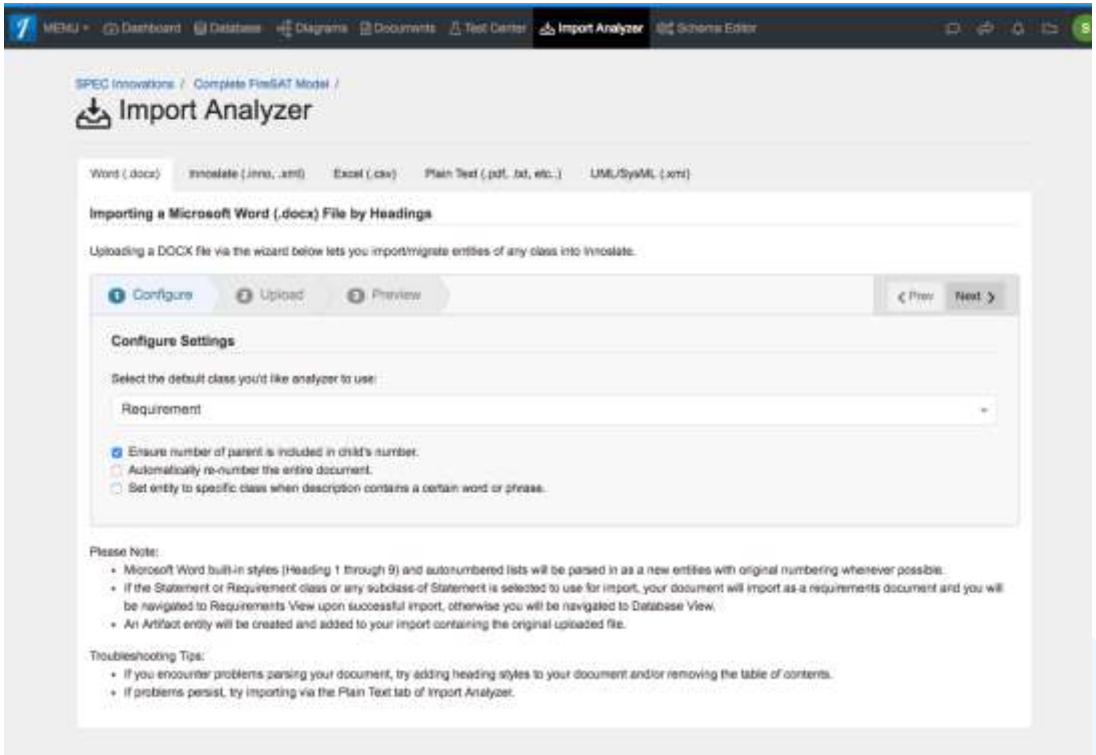
Innoslate Dashboard

- Setup “widgets” to see the information you want
  - Wiki
  - Tables
  - Bar and pie chart
  - Comment feed
- Activity Feed shows project changes
- Send Feedback



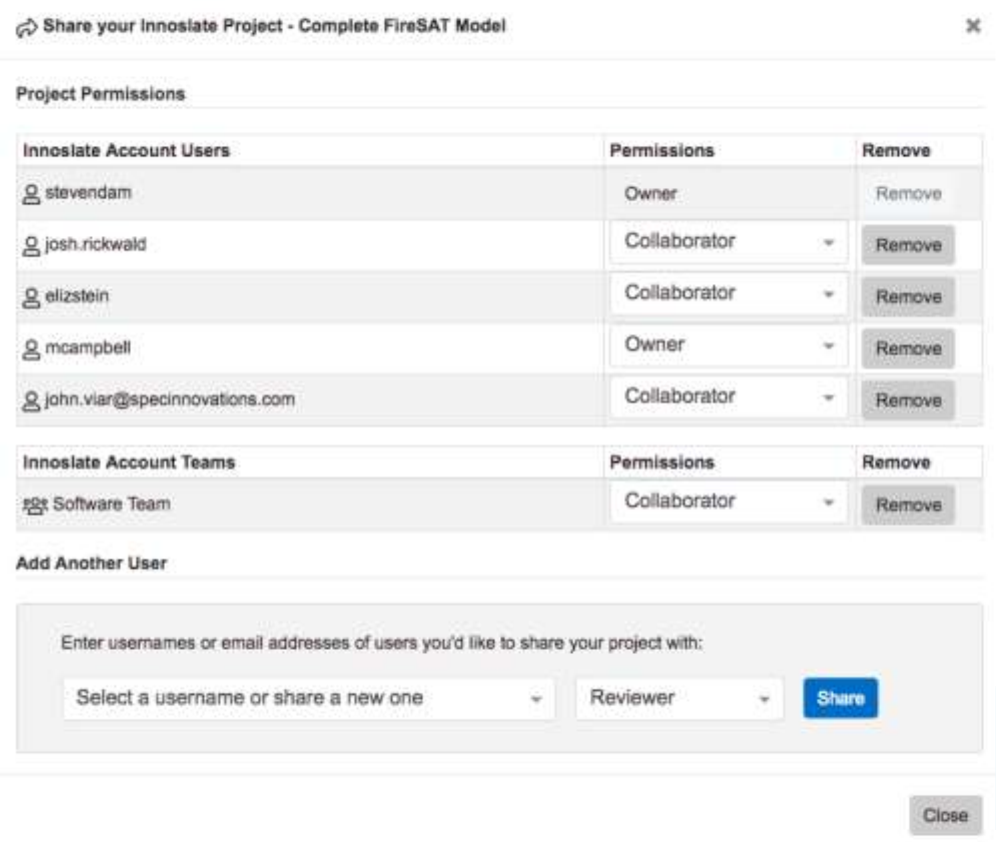
Innoslate Import Analyzer

- Import documents from spreadsheets, MS Word, and text/PDF document
- Import XMI and XML from other Innoslate projects and other tools



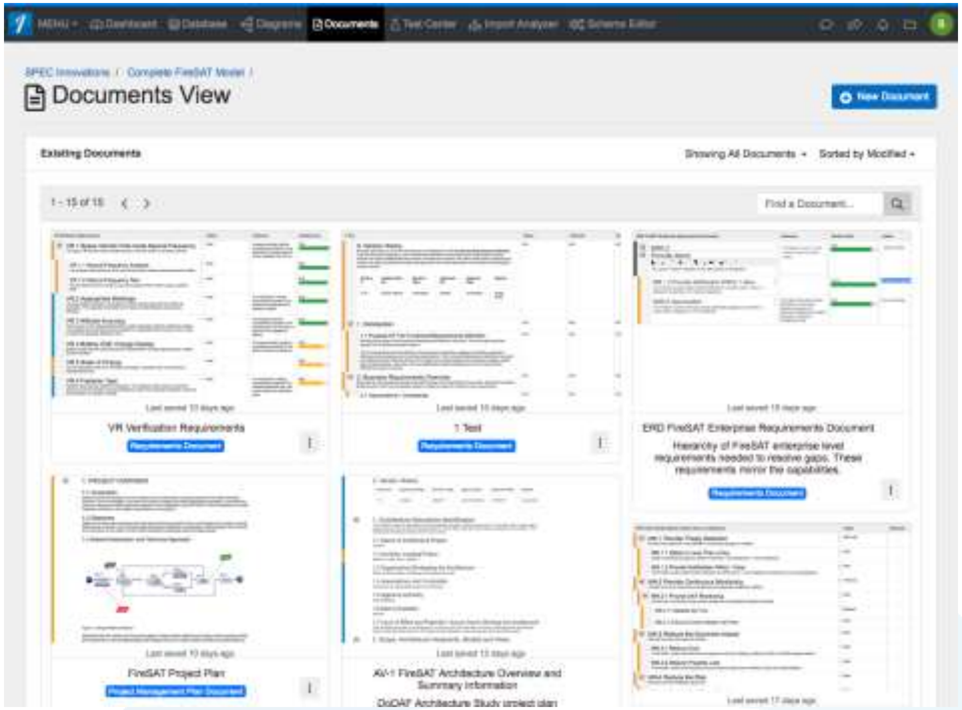
Project Sharing

- You can share your project with your colleagues
  - Owner
  - Reviewer
  - Viewer
  - Collaborator
  - Or any other roll defined by the Organization
- Team, or user name/e-mail address



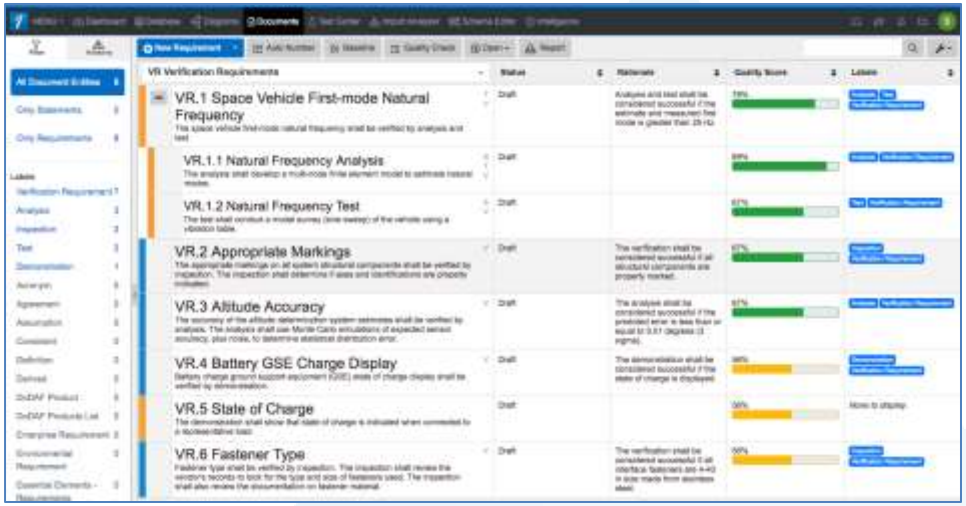
Documents View

- Author all you documents in Innoslate
- Create any document type
- Use pre-defined templates or develop your own
- Use search to find specific documents
- Add your own document types by creating a new label for the Artifact class with the word "Document" in the name



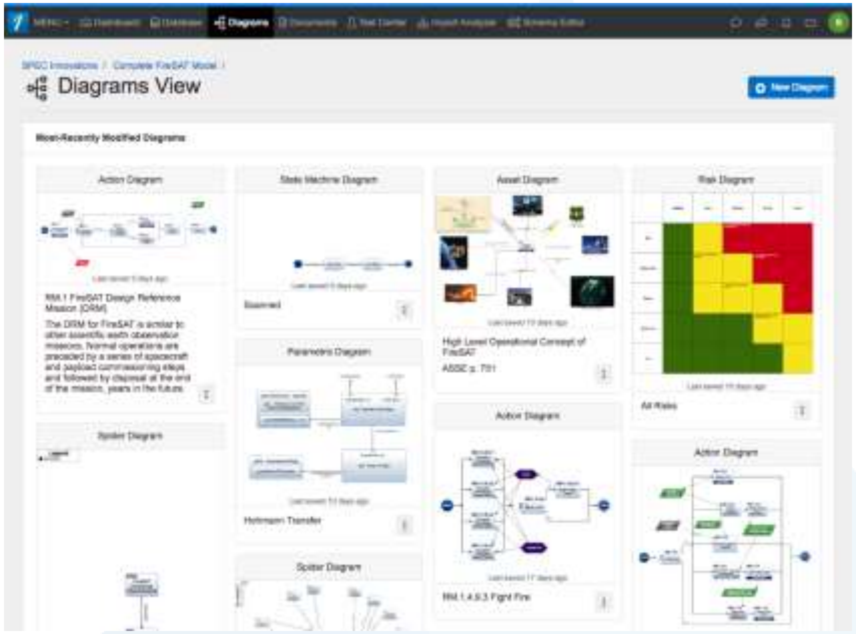
Requirements Documents

- Create Requirements hierarchies
- Track status
- Check quality
- Organize using labels
- Baseline
- Reports



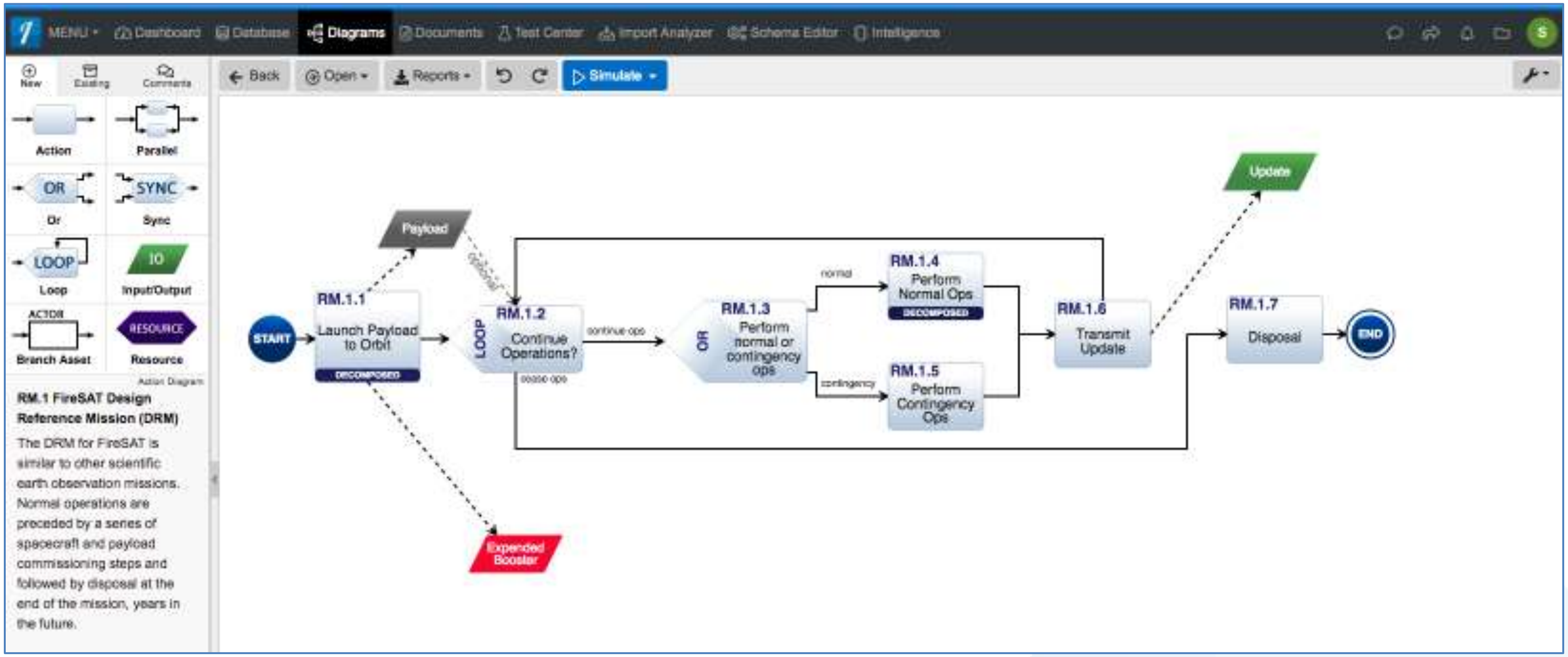
Diagrams View

- Create or modify diagrams directly from this view
  - Select from 22 diagram types including all SysML
- Search for diagrams using complex searches
- Create reports in MS Word or MS PowerPoint



SPEC INNOVATIONS

Key Diagram: Action Diagram

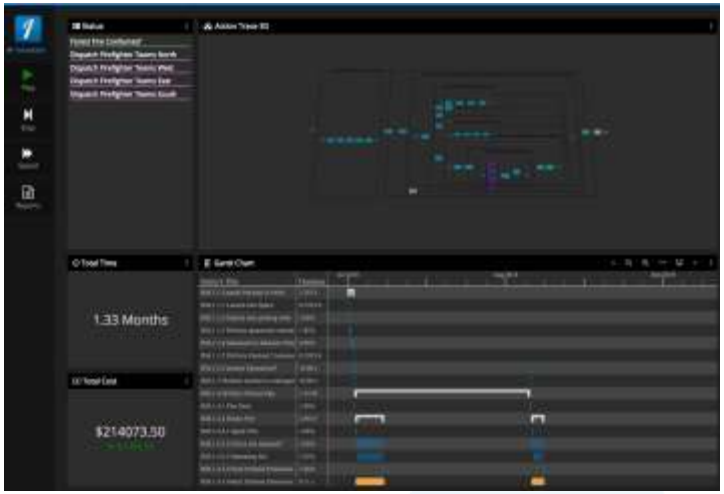


- Capture functional behavior with simple logic
- Include JavaScripts for decisions and complex calculations
- Allocate to Assets
- Use Inputs/Outputs to trigger events
- Resource modeling

Simulate Models

- Use Action Diagram or Activity Diagram
- Model functional behavior constrained by physical architecture
- Include resource modeling
- Develop cost and schedules from execution

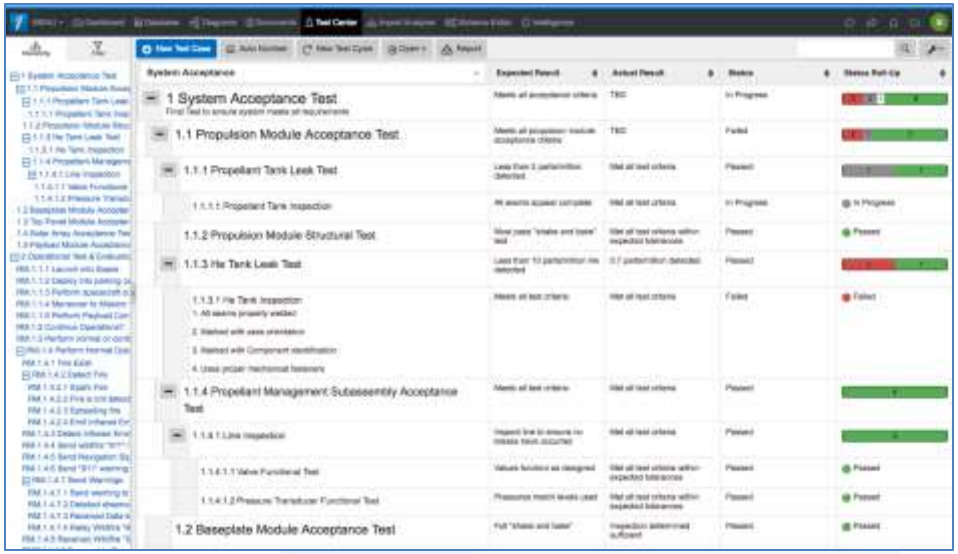
Discrete Event Simulator



Monte Carlo Simulator

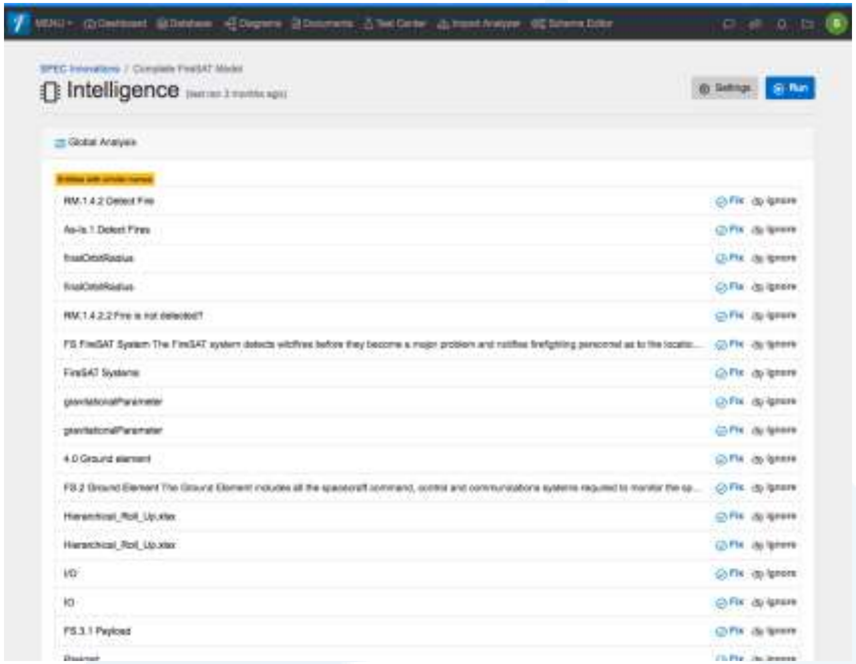
Test Center

- Create test cases and suites
- Capture expected and actual results
- Track status
- Create Action Diagrams from Test Cases to model and cost test processes and procedures

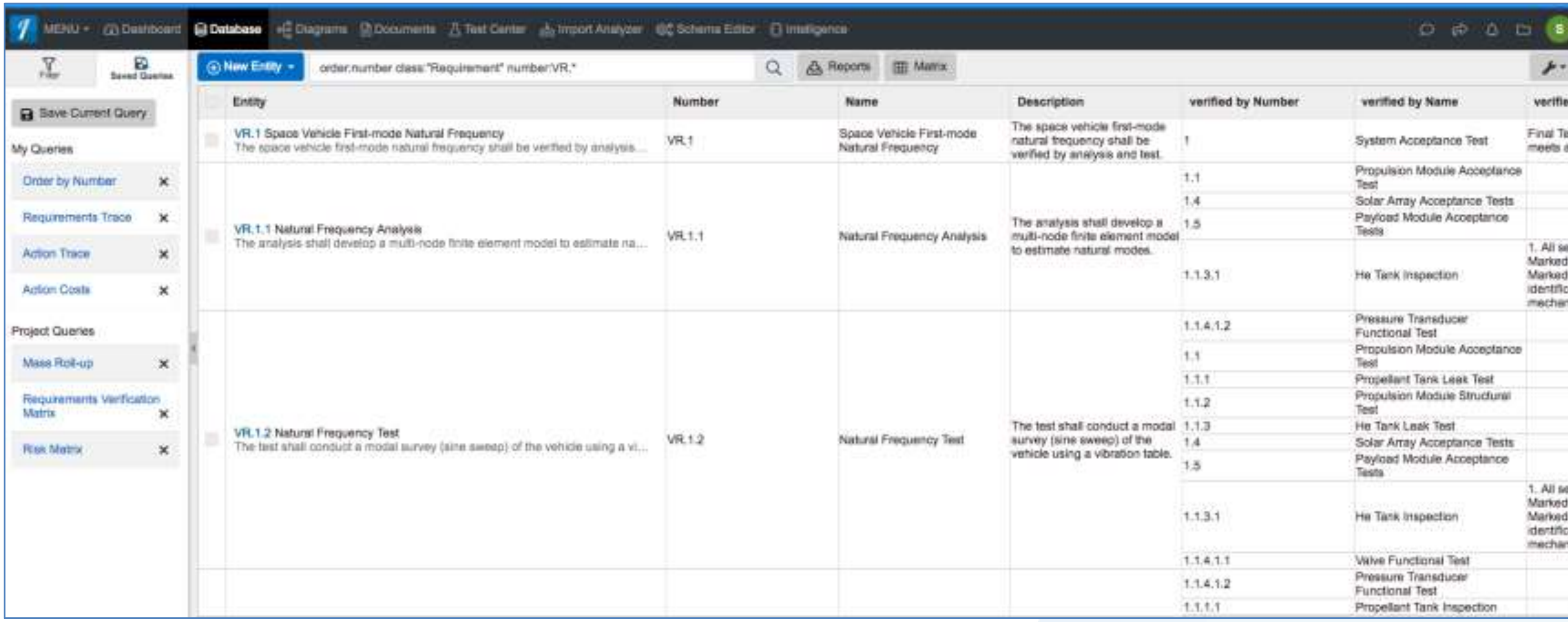


Use Intelligence to Enhance Model Quality

- Provides 68 heuristics to improve modeling by applying NLP technology
- Select between warnings, errors, and ignoring heuristics
- Fix problems or ignore using buttons



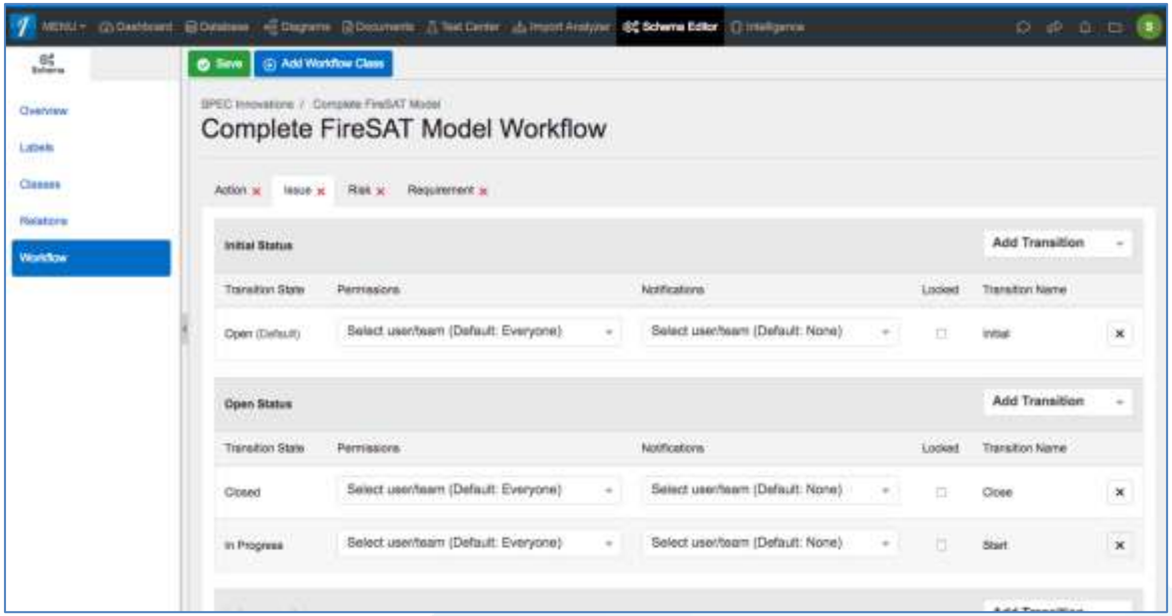
Database View



- Filter and organize information
- Show and edit related entities
- Use complex search queries
- Save those queries
- Show information in hierarchical views

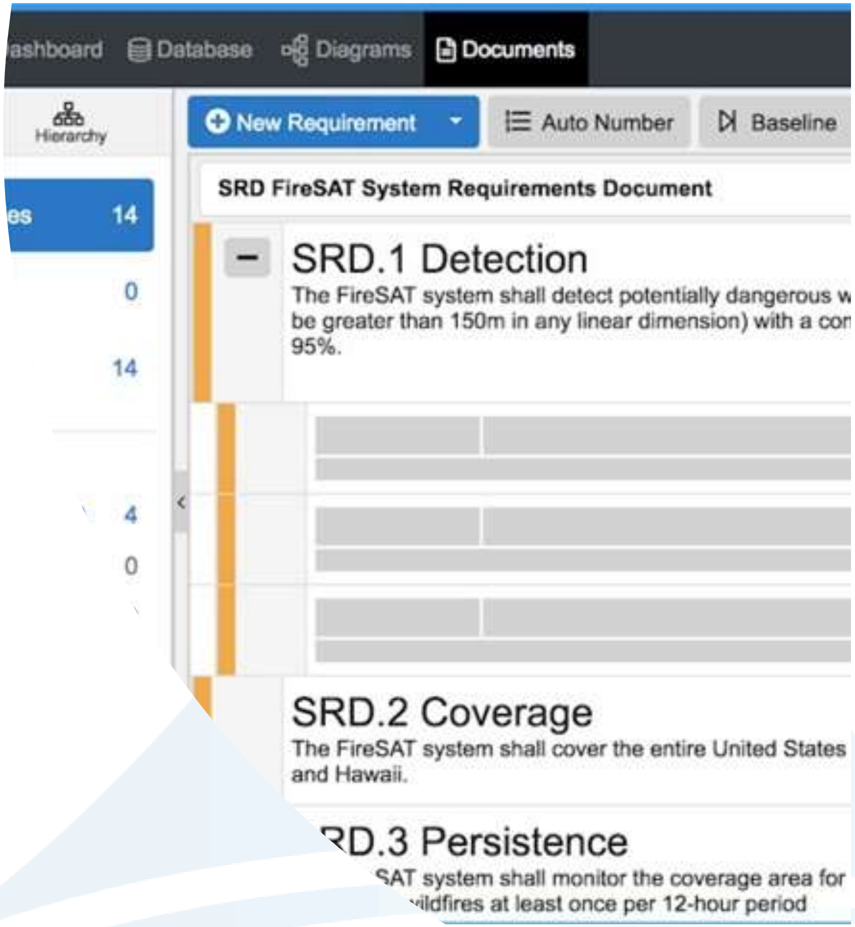
Edit Schema and Create Work Flows

- Add or modify classes, relationships, and attributes
- Hide schema elements
- Create work flows for any class using enumerated attributes



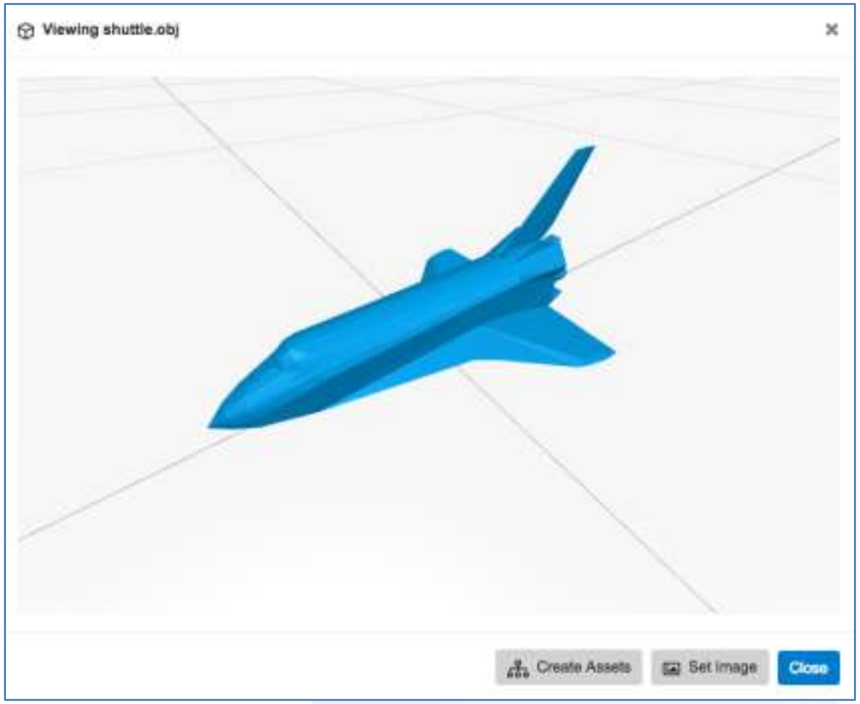
Use Cross-Project Relationships

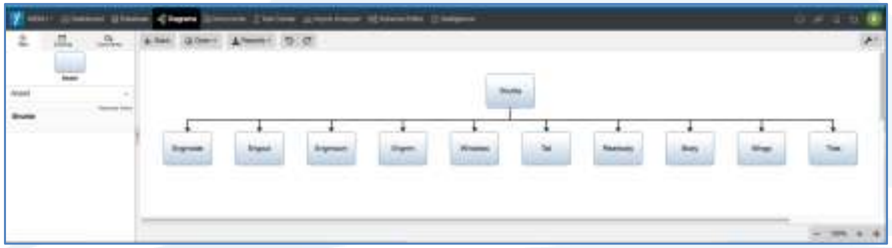
- Cross-Project relationships enable the inclusion of information from one project into another
- If the person viewing the information does not have permission to see the other project then that information will be redacted in any view



Capture Product Design Information

- Capture and visualize CAD files
- Decompose into database objects
- Capture artifacts from other, more detailed analyses





SPEC INNOVATIONS

Use Traceability Matrices to Connect Information

- Trace anything to anything else using the relationships
- Use NLP technology to help identify how information should be traces
- Identify suspect links too

Capture and Relate Program Management Information

- Model management processes using Action Diagram
- Create work breakdown structures
- Capture decisions
- Identify and manage risks and issues
- Get Gantt Charts and costs from simulations
- Get MS Project files from simulations

SPEC INNOVATIONS

SUMMARY

Summary

The Full Lifecycle in One Tool

Requirements Management	MBSE	Test Center
Intelligence View	Modeling	SysML/LML and more
DoDAF	Discrete Event Simulator	Configuration Management
CAD Viewer	Monte Carlo Simulator	Customizable reports

SPEC INNOVATIONS