International Council on Systems Engineering
A better world through a systems approach

Darth Vader's Personal Library: Models, Models, and More Models

Matt Gagliardi
System Strategy Inc.

Matthew Hause
System Strategy inc.

INCOSE International Symposium 2025 | Ottawa, Canada

Agenda

- Background & Problem
- Approach
- Defining Model Libraries
- Identifying Producers/Consumers
- Model Controls
- Examples
- Managing Model Libraries

Background and Problem

The Galactic Empire is constantly trying to develop systems under compressed schedules. Additionally, the struggle for power within the Empire has driven project managers to refrain from sharing product designs and specifications, resulting in design silos that duplicate product lines, common systems, sub-systems, acquisition processes, and mission planning. However, this has resulted in extended development cycles, as well as key system failures, beyond what is acceptable to the Empire's Sith leaders, and has forced the early "retirement" of project managers

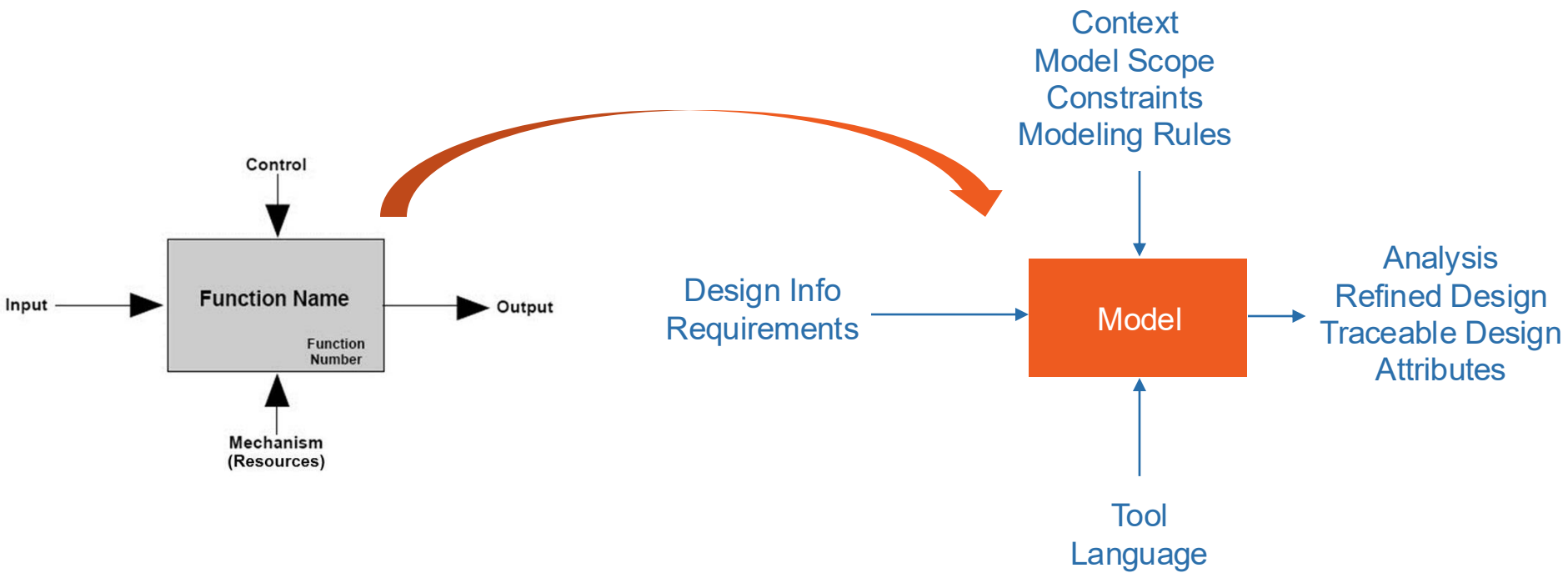
- Not sharing info across the enterprise (no reuse)
- Extended development due to building all models for an effort from scratch
- Models of same/similar things not aligned (different sources and/or different definition)
- Increased supply chain costs/effort

Approach

Model Libraries can be a key part of the solution!

- Define Model Libraries
  - Use
  - Scope
  - Constraints
  - Language
  - Inputs
  - Outputs
- Identify Potential Producers of Inputs and Consumers of Outputs
- Address Gaps
- Modeling Controls
  - Model Scope
  - Model Context
  - Modeling method/style
  - Other Constraints
- Manage Model Library
  - Define process for adding and updating models to libraries
  - Model Curation

Remember IDEF0?



Defining Model Libraries

- Select Categories of Models based on Purpose and/or type

Some Examples from the Empire

Signal

Interface

Component

System

Family of Systems

System of Systems

Mission Engineering

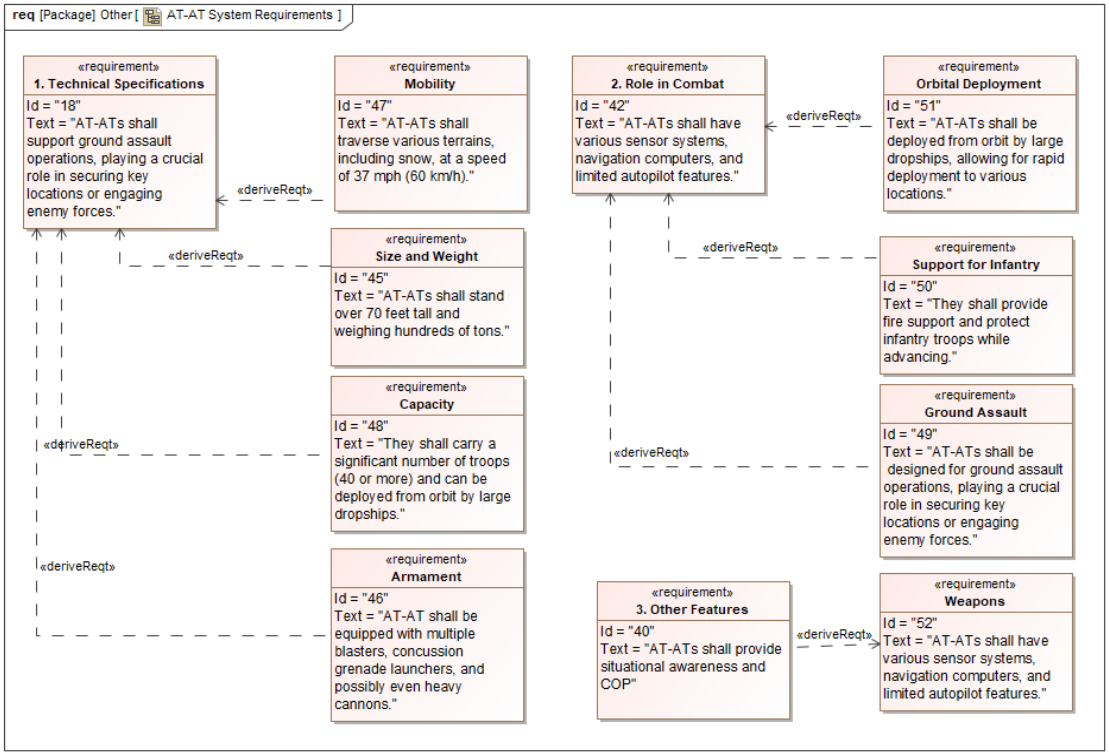
Increasing Level of Model Integration

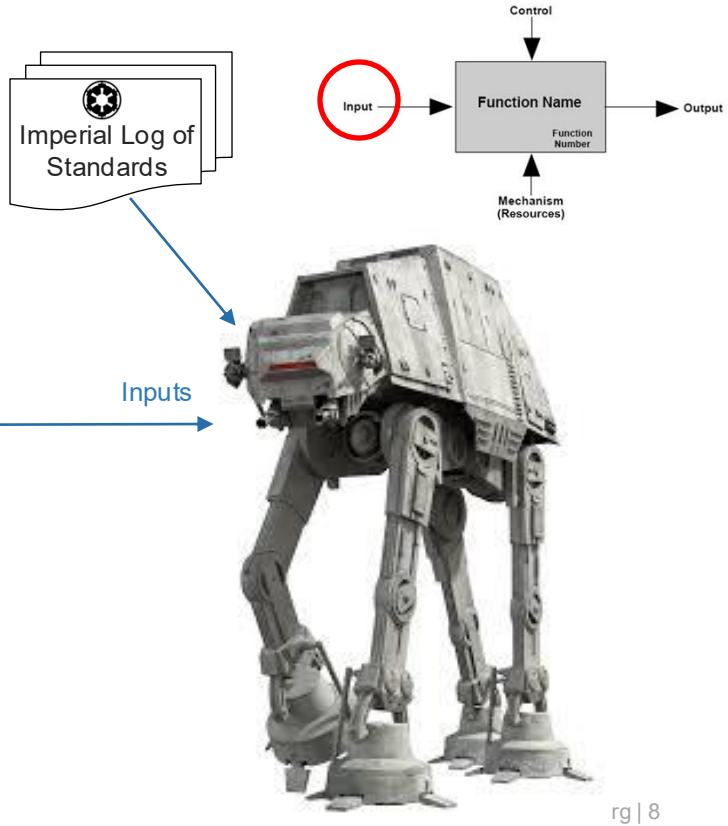
Increasing Level of Reuse

Defining Model Libraries

- Identify the kinds of information needed, and their sources, to develop models in each category/library

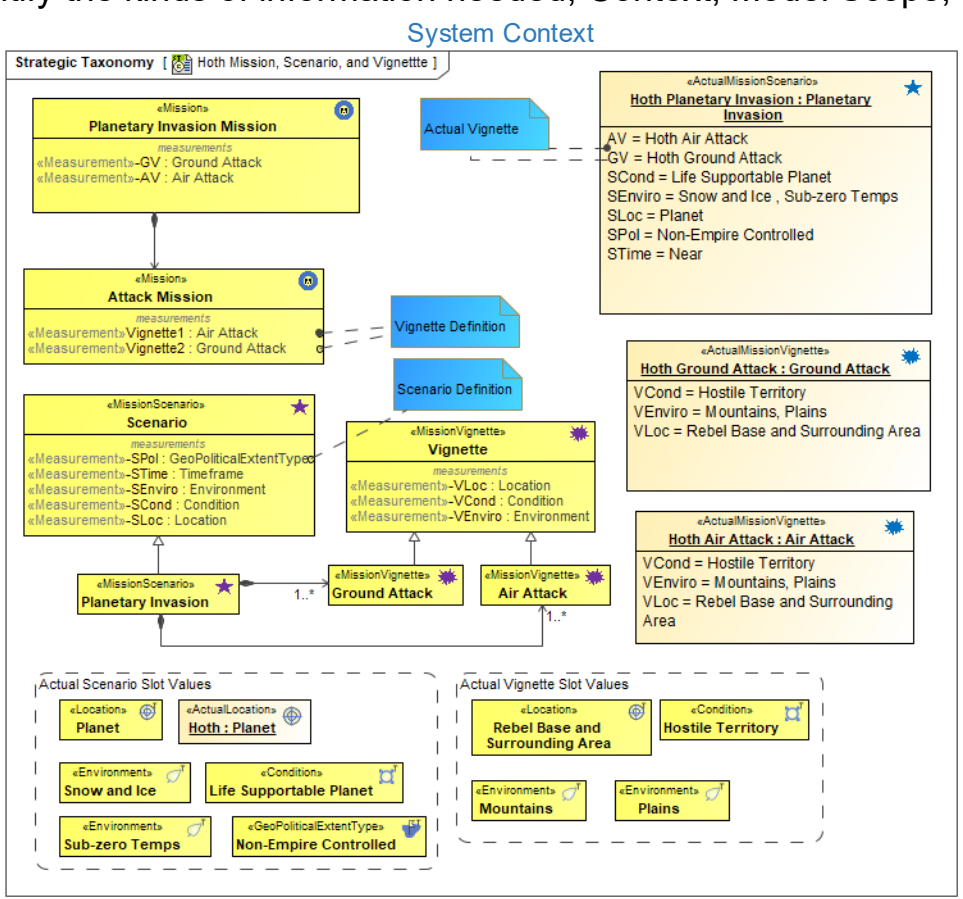
Specifications



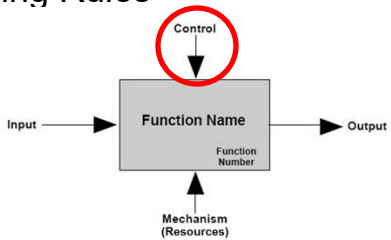


Defining Model Libraries

- Identify the kinds of information needed, Context, Model Scope, Constraints, Modeling Rules



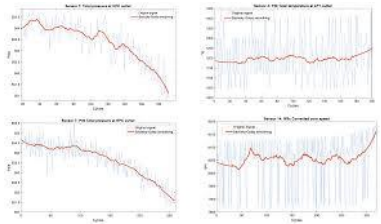
Controls



Defining Model Libraries

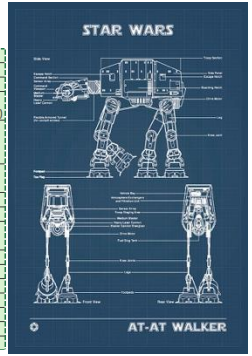
- Identify the kinds of information available to export from models in each category/library

Analysis

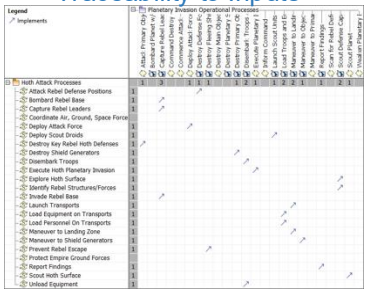


Outputs

Length	20m
Height/Depth	22.5m
Max Speed	60km/hr
Power Plant	KDY FW62 compact fusion drive systems (2)
Armament	2 Taim & Bak MS-1 fire-linked heavy laser cannons
	2 FF-4 medium repeating blasters[4]
	Durasteel footpads (4)
Complement	Speeder bikes (5)
	All Terrain Scout Transports (2)
	Heavy weaponry
Crew	Pilot (1)
	Gunner/co-pilot (1)
	Commander (1)
	Deck officers (2)
Passengers	40+ Troops
Cargo Capacity	1 metric ton



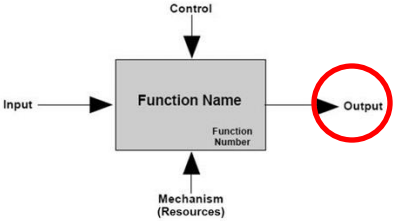
Technical Attributes



Traceability to Inputs

Critical Function List

External Interfaces



Identify Potential Producers and Consumers

- Done by Model Category (Library)
- Model Matchmaking (Curation) – what models can give me what I want, and what can I provide with my model
- Address any overlap between libraries
  - Do I really need more than one source of the same/similar info?
  - If I do, how do I decide which one to use when?
  - Do you impose an ASOT, or allow multiple?
- Address any gaps between libraries (model, library)
  - If there's a missing input, the model is incomplete
  - How would I have gotten this info in the past?

Model Controls

- As noted before, Context and Scope of a model are controls that can be used to define model library categories
- But other controls can be used to constrain the model inputs/outputs enough to enable consistent integration of models, and should cover basic modeling elements (reqs, structure, behavior, interfaces, etc.):
  - Language
  - Modeling Style/Method
  - Units
  - Sources (e.g. Outputs from other Libraries)

Model Controls – Example (Modeling Style)

- Reqs
  - ONLY use base SysML Requirements type
  - Must have Title, Text, Verification Method, Measurement, and Units defined
  - If derived, must have traceability to parent (and use <<derivedreq>> relationship)
- Functions
  - Can be represented only by Operations
  - Must be owned by a block that is a part of the system that the model represents
  - Shall be represented in the Empire's common function library

Example Using PLE Model

Component, Feature Set, and Interface Libraries

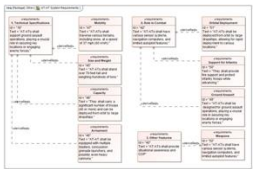
Empire Walker Component Library

Empire Walker Feature Set Library

Empire Ground Domain Interface Library

PLE Model

AT-AT Reqs



System Model Library

AT-ACT Model

AT-AT Model

Elite AT-AT Model

Hoth Mission Model (usage)

Example Using PLE Model

PLE Model

AT-AT Model

Hoth Mission Model (usage)

Defined:

- Mission Type
- Environment

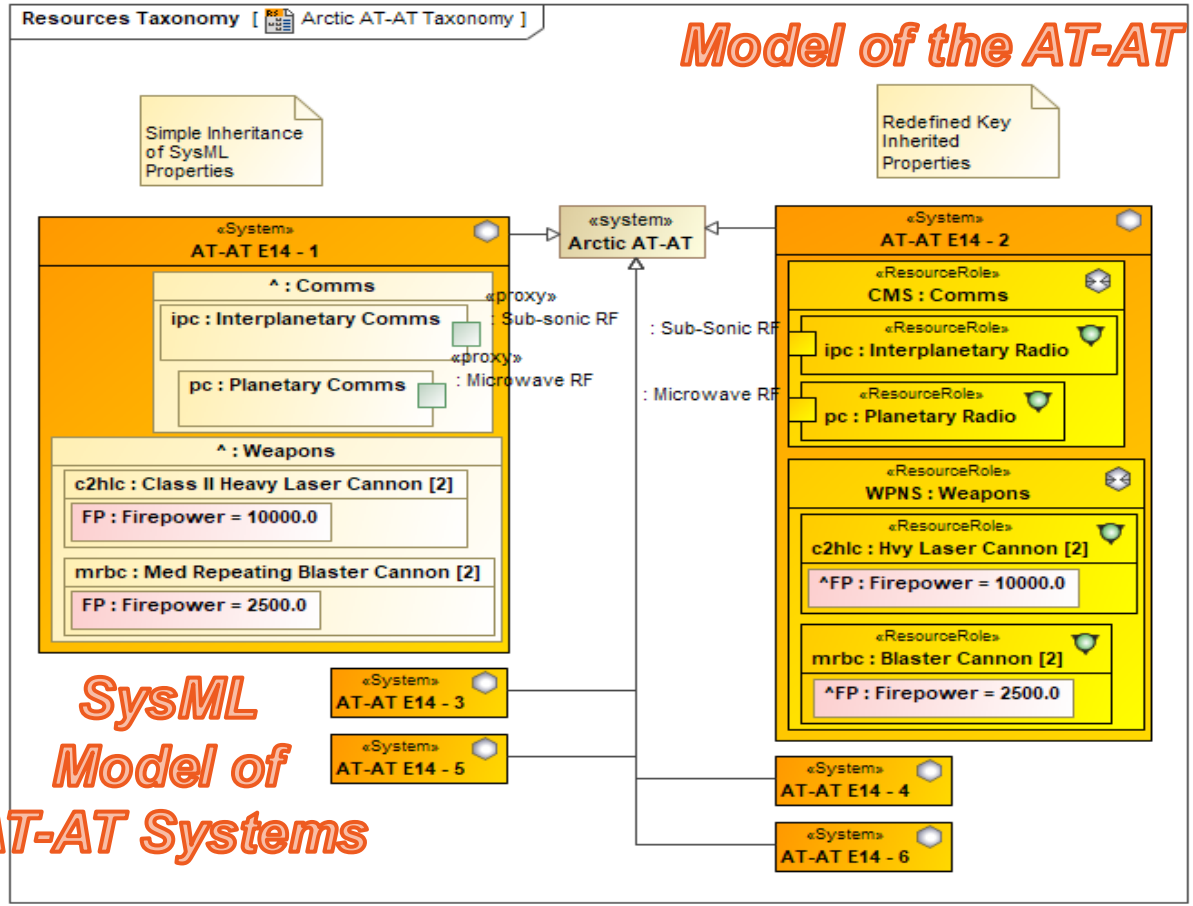
Defined:

- Comms Interfaces
- Weapon Firepower (measurement)

Using the AT-AT System model in the Mission Model

- The figure shows two variations of the reused model: the left where the SysML parts are maintained, and the right where they are redefined to UAF elements
- Because UAF limits interactions and connection to only UAF-UAF elements, ports need to be redefined

May need to constrain how SysML models are integrated into UAF (i.e.require conversion to UAF elements)



Manage Model Libraries

Once Model Categories/Libraries are defined, need to diligently manage them

- Process for identifying models missing from library (model curation)
  - "Suggestion Box" - let users find new library models for you
  - Active Search
  - Adoption of Other libraries
- Process for accepting models into a library
  - Defined Criteria (Input/Output Standards at a Minimum)
  - Allow for Model Modification (e.g. extraction of sub-models from submitted model)

Manage Model Libraries – Cont'd

- Model with modularity in mind
  - You may want/need to split off useful portions of larger models into appropriate libraries to make best use of them
  - Maximizes reuse potential
- Publish/advertise the model elements so that engineers know what is available and how to find what they need.
  - Organize Libraries based on Org/Domain/Product (other?)
  - Provide documentation that explains Library organization
  - Use Key descriptors (meta-data) that enable searches (see next slide)

Manage Model Libraries – Cont'd

Example Meta-Data for Model Search

- Model Name and Classification: A descriptive and unique name that helps in identifying and referring to the Model.
- Intent: A description of the goal behind the Model and the reason for using it.
- Aliases: Other names for the Model.
- Motivation (Forces): A scenario consisting of a problem and a context in which this Model can be used.
- Applicability: Situations in which this Model is usable; the context for the Model.
- Collaboration: A description of how elements used in the Model interact with each other.
- Consequences: A description of the results, side effects, and trade offs caused by using the Model.
- Implementation: A description of an implementation of the Model; the solution part of the Model.
- Known Uses: Examples of real usages of the Model.
- Related Models: Other Models that have some relationship with the Model; discussion of the differences between the Model and similar Models.
- Dependencies: Other models (from other libraries) that are used in this Model

Conclusion

- To help the Empire improve product design time, consistency, and quality (as well as not have an "early retirement"), Project Leaders looked at how they can define/develop Model Libraries, including:
  - Library Inputs/Outputs/Constraints/Controls
  - Identifying existing and potential model consumers
  - Model Curation
  - Model Meta Data (for searchability)

- But, they cannot forget that:
  - After definition, You're Not Done . . . . . Library adaptability will be necessary
  - Library development requires resources/effort (not free), but enables development savings in the future

Don't Reinvent, Re-Use!

Thank You!!