DIGITAL ENGINEERING
/ SYSTEMS ENGINEERING

MBSE Strategies: Avoiding Cyclic Project Usage Dependencies by using Black Box and White Box Models

Dr. Jason Kolligs, STC
Stu Masterson, STC
Eric Thome, GD OTS
Christian Knutson, GD OTS

Introduction

Large, complex systems that involve multiple partners and external actors require flexible digital engineering approaches that allow for:

- Modeling to include input from those partners
- Supporting data quality and configuration management

Project usages are a viable approach; however, they can pose challenges

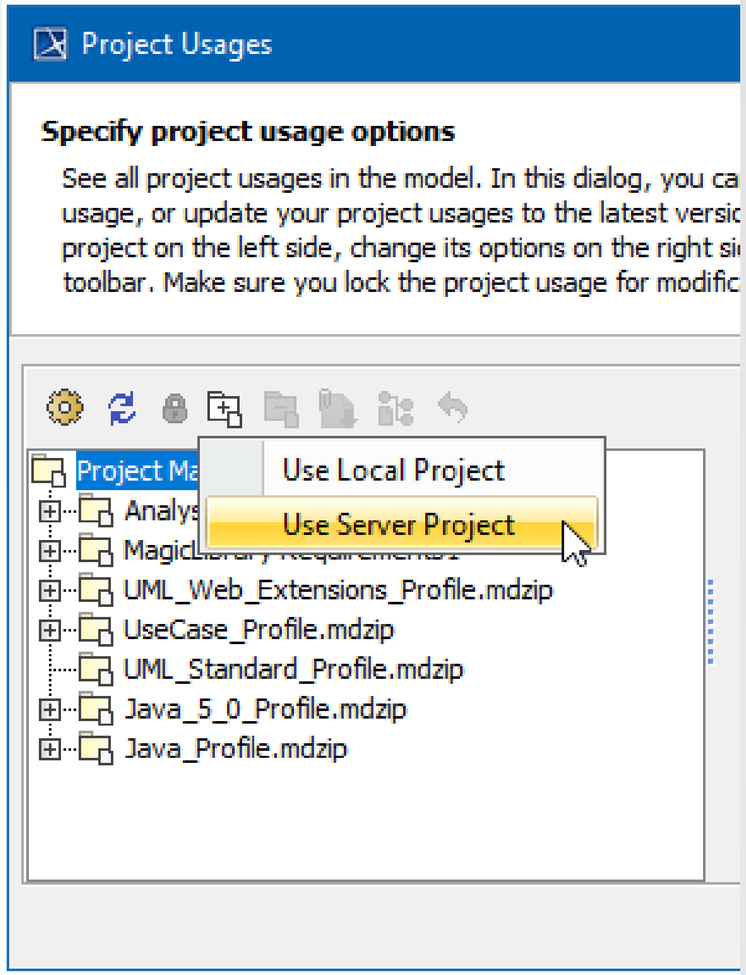
Scenario:

- The planned content of the models are well understood
- Project usage architecture posed potential for cyclic usage

Background – Project Usages

Project usages are used within Cameo and other modeling tools to import a model into another project

- Cannot be modified in the current project but elements are available for usage in the project, i.e. project usages are read-only
- Common practice often used to include custom organizational profiles, reference materials such as regulations and standards, and other template or style guide type projects

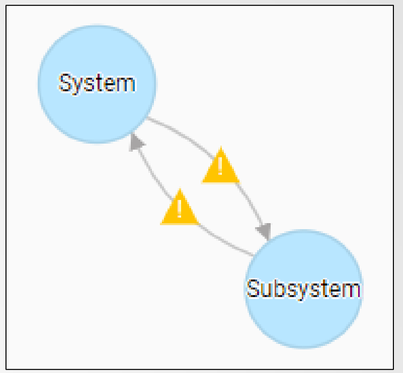


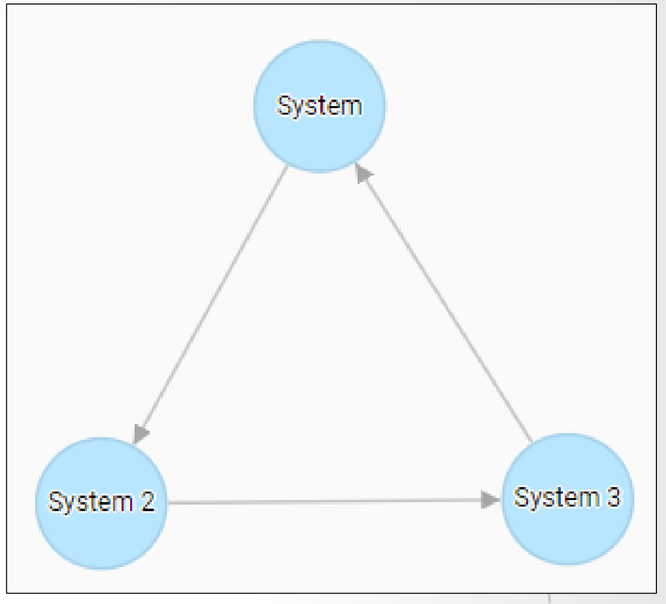
Background – Cyclic Usages

Cyclic usages, or cyclic dependencies, are project usages within model-based systems engineering (MBSE) that reference each other in a circular manner

- Creates an infinite loop or references that will eventually lead to resource issues that can be realized as model crashes or even model corruption (data loss) at the extreme cases
- Cyclic usages can be accidently created and difficult to identify in more complex model federations
- Cameo Resource Map

NOTE: Warning for simple, but not complex



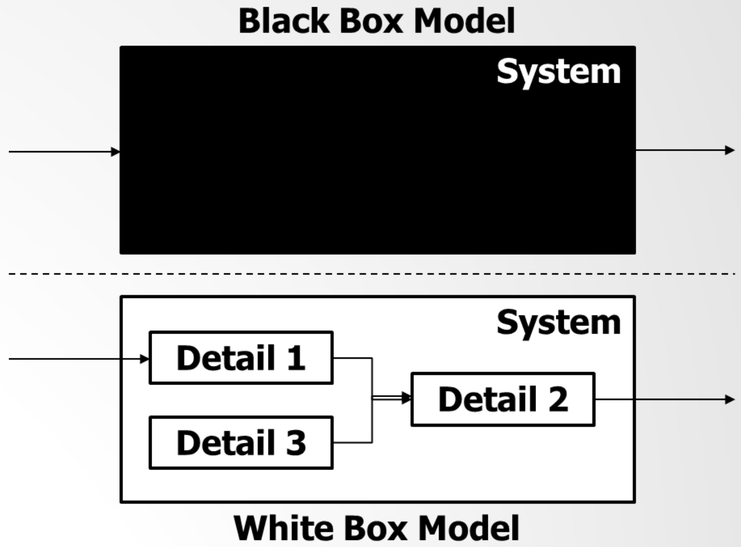


Background – Black Box (BB) and White Box (WB) Modeling

Black box modeling (black box view)

- Focuses on the system's external behaviors, such as the interfaces and data flows
- Does not reveal the internal workings (and therefore can be used to obfuscate complexity and/or Intellectual Property)
- Establishes system boundaries

White box models or views show the internal workings of a system while acknowledging the same external behaviors as the black box.



The Situation

The stakeholders of a new engineering effort wanted to develop a model utilizing the feedback from external partners.

The concept was to drive lower-level design and development based on high-level needs and expectations.

The lower-level designs would influence the higher-level design and therefore will have to be consolidated into the higher-level model to facilitate the final design.

A preliminary model that established the system boundary, desired inputs, and expected outputs was needed by the external partners so that they could develop their subsystems which would then be integrated into the system model.

One Big Model vs. Distributed Projects (Generalized)

	One Big Model	Distributed Projects
Pros	• All Data is Centrally Located • Full data accessibility	• Increased Performance at Scale Scale
Cons	• Needs to be accessible to different different companies • More iterations and revisions which which leads to challenging merges merges and cloud publishing	• Data is only available to the original original system developer when when iterations are sent • Requires integration plan to properly coordinate final model model effort e.g. avoid cyclic usages

Engineering Effort did not want or need centrally located data or full data accessibility

Energy should be focused on mitigating this challenge

Risks: Project Usages vs. Merges

Project Usages: If the subsystem models are using the original system model as a project usage (which is preferred), then the original system model cannot import the subsystem models as project usages (which is preferred) as it would create a cyclic usage.

Merges: multiple model branches are merged back into the trunk. Requires consideration of two (2) general concepts that are conflicting:

- Keep trunk revisions low, i.e. use branches as long as possible. (good for long term model health and future operations such as migrations, exports, and managing history)
- Keep branches small enough to allow for merging. Merging (currently) is a memory intensive process based on the differences from the trunk

The merge capabilities of the modeling tool being used by the team were difficult and inconsistent, which led to that option being removed from consideration

Potential Solutions

There are two (2) potential solutions to the cyclic usage issue.

- Ignore it – translates to serial process in which the original system model that is distributed to the external actors is never updated as the system design process iterates.
  - This solution is not viable in a multi-partner systems engineering effort.
- The second solution is to avoid or prevent cyclical usages
  - What does that architecture look like?
  - Is the architecture rational?

Another System Model

Project usages might be perceived as potentially cyclic is based on belief that there can only be a single system model (M1).

- This effort considered the potential of a second system model (M2) that would integrate the details from external partners.

The M1 and M2 models describe the same system, but they serve different purposes:

- M1 expands customer needs into requirements and logical design concepts.
- This represents divergent systems thinking as the problem space is being defined and the solution space is being opened.
- M2 is the convergence of subsystem design details that home in on a system solution.

BB and WB Principles

The M1 and M2 models align with black box and white box modeling

	M1 (BB)	M2 (WB)
Concern of the the System Model	How the system will interact with its its environment and represents a black black box perspective of the system system	Subsystem models provide further further detail to the system to create create the white box perspective. perspective.
Facilitates Systems Engineering	• System architecture design • Requirements development • Analyses based on the inputs and and outputs • External interfaces maturation.	• Analysis on subsystem interactions • Lower-level interfaces • Requirements traceability across across the subsystems • Verification method traceability traceability • Subsystem level test data

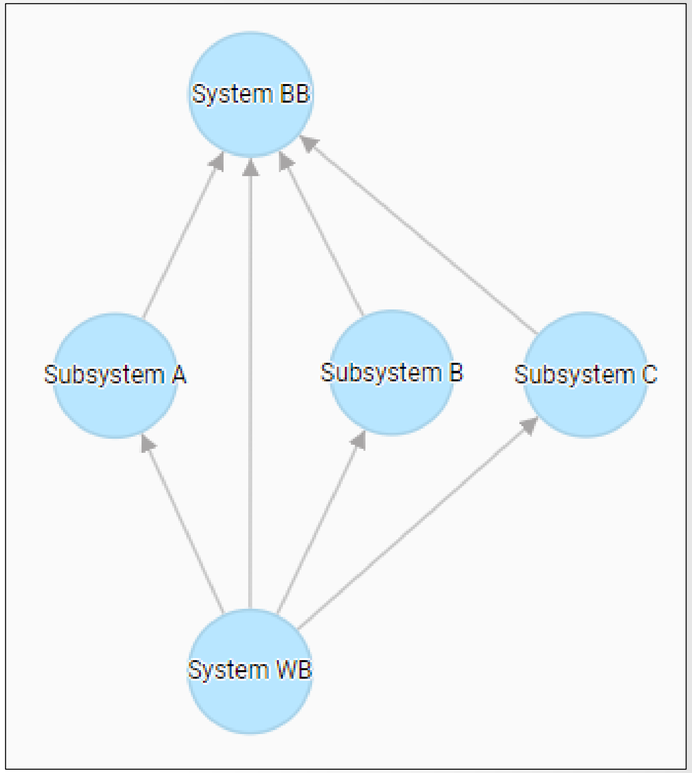
Path Forward

The two (2) model revelation led to an extensible project usage architecture. (See Image)

Recall that an arrow means that the source project is using the contents of the destination project.

The architecture includes a project usage from System WB to System BB which mitigates two (2) minor issues:

- Designating which subsystem project usage supports the System WB project need for data in the System BB model.
- Nested project usages, which involves updating each echelon of project usages.



Benefits vs. Potential Issues

Benefits:

- Distributed – mitigates the performance issues of a potentially massive model.
- Configuration Management – inherently protects the System BB model from external actors and Subsystem models from System WB usages.
- Extensible – supports many subsystems and external actors.
- Reusable – can be templated and used throughout the organization with teams that have similar situations.
- Intellectual Property (IP) Obfuscation – the system integrator can maintain control of IP while disseminating a meaningful model to support partner needs.
- Product Line Engineering (PLE) – the customer has plans for PLE that are well supported by the WB model details tracing to the requirements in the BB model.

Potential Issues:

- Non-Real-Time Data
- Multiple System Models
- (Recall these are directly attributable to the distributed model approach)

Summary / Conclusion

It is important to customize the response, within reason and based on the available resources, to the needs of the project and contract.

For this case, the consideration of a large, singular model was overturned for a distributed model.

The potential for a cyclic usage, which can be catastrophic in both performance issues and data loss, was mitigated by an innovative approach that allowed for two (2) systems models – one (1) Black Box and one (1) White Box.

The concerns of having two (2) system models were mitigated via acceptance and understanding that each system model would play its part appropriately based on model function, system development, and contract deliverables.