2019 Annual INCOSE international workshop
Torrance, CA, USA
January 26 - 29, 2019

Production and Logistics Modeling Challenge Team

Timothy Sprock[a], Leon McGinnis[b], Conrad Bock[a] , George Thiers[c]

[a] National Institute of Standards and Technology, [b] Georgia Tech, [c] MBSE Tools, Inc.

Overview

- Purpose?
- Challenges: why do we exist?
- Collaboration Paradigm
- Making Models and MBSE Ubiquitous in Production and Logistics

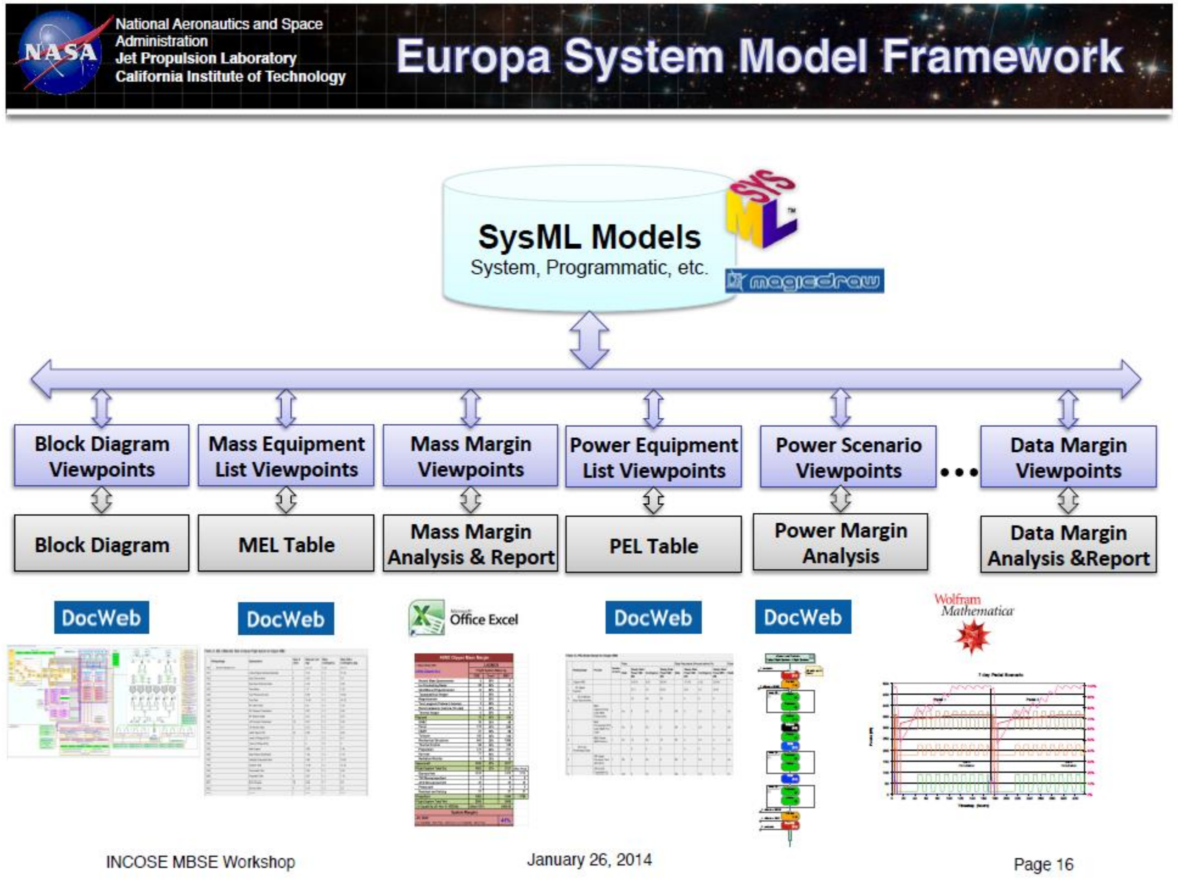
Challenge Team Purpose

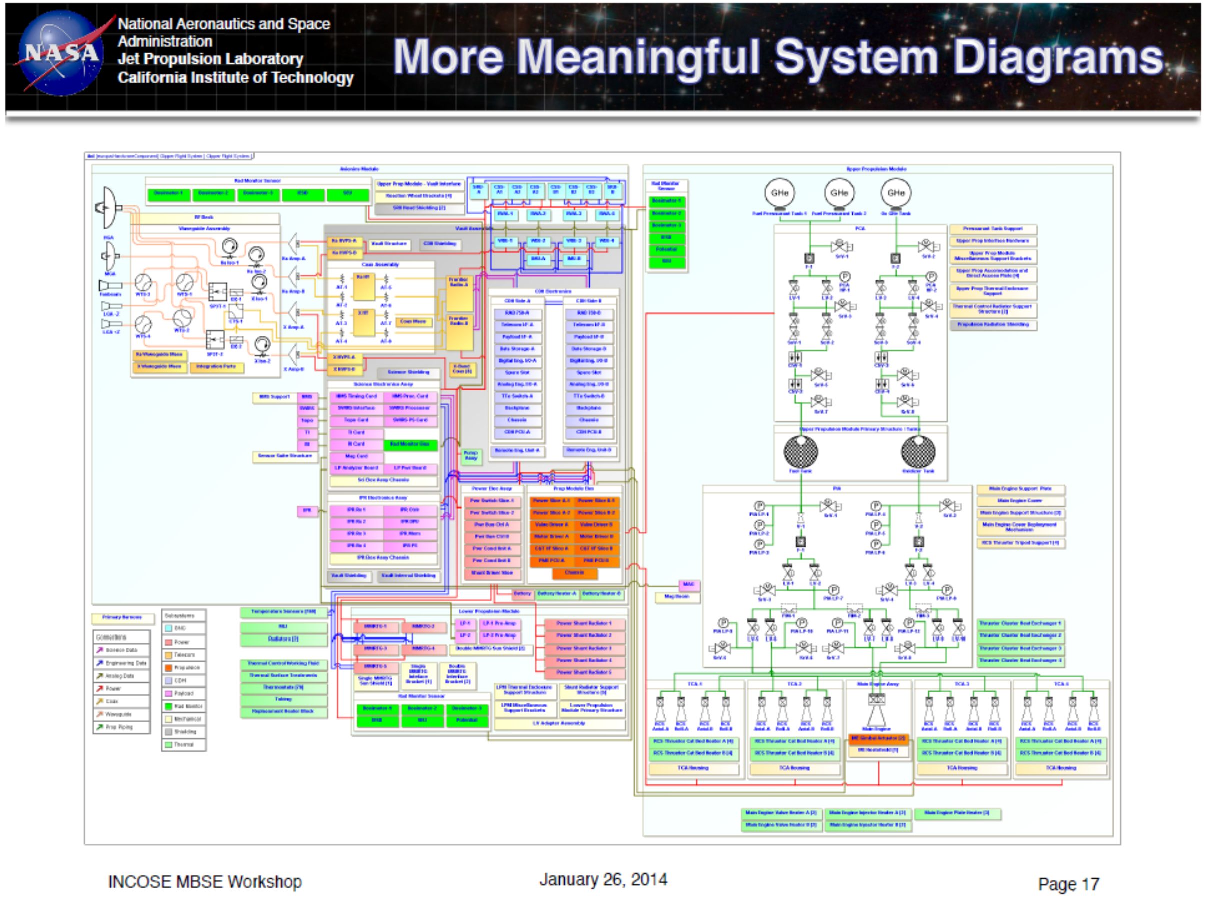
Increase the availability of reference models, awareness of these models and methods, and successful use of MBSE in the production, logistics, and industrial engineering communities.

Specific challenges in providing a foundation to production and logistics [systems] engineering are the lack of:

- Standard reference models
- Well-structured engineering design methodologies
- Integrated analysis models and tools available to support design and operational decision-making.

MBSE in the “Product” domain--JPL





Dave Nichols & Chi Lin, “Integrated Model-Centric Engineering: The Application of MBSE at JPL Through the Life Cycle,” INCOSE IW 2014

What makes this possible?

- Almost 50 years of effort to “standardize” the specification of the product—culminating in the ability to exchange designs between CAD systems
- Similar efforts to integrate product analyses with CAD models
- Emergence of SysML, a systems modeling variant of UML
- Recognition of the potential payoff
- Resulting commitment of resources to accomplish integration

Motivation

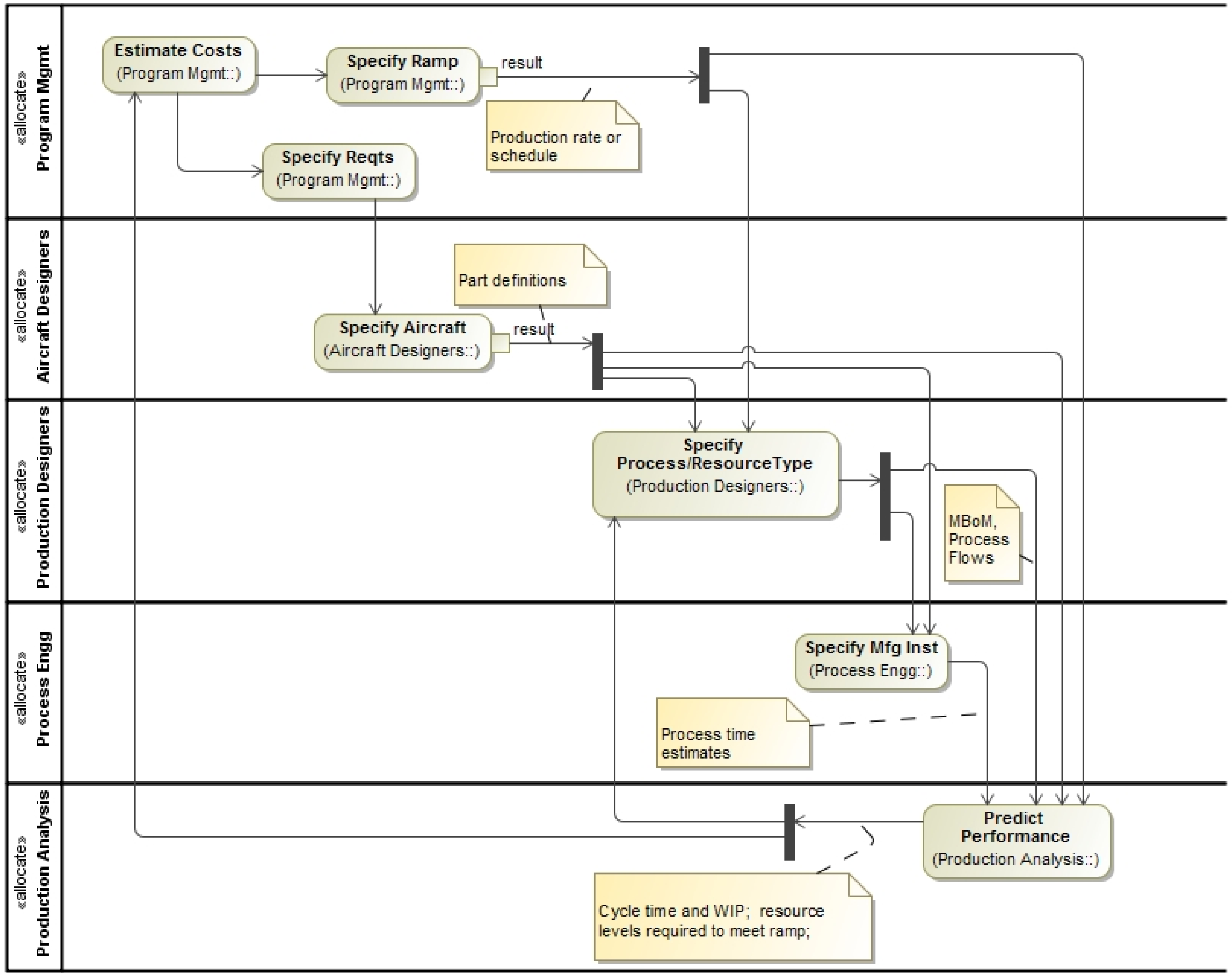
There are multiple stakeholders, with discipline-specific viewpoints

The systems are large, complicated, expensive, and persistent

The contemporary decision support analyses are independent, stand alone efforts

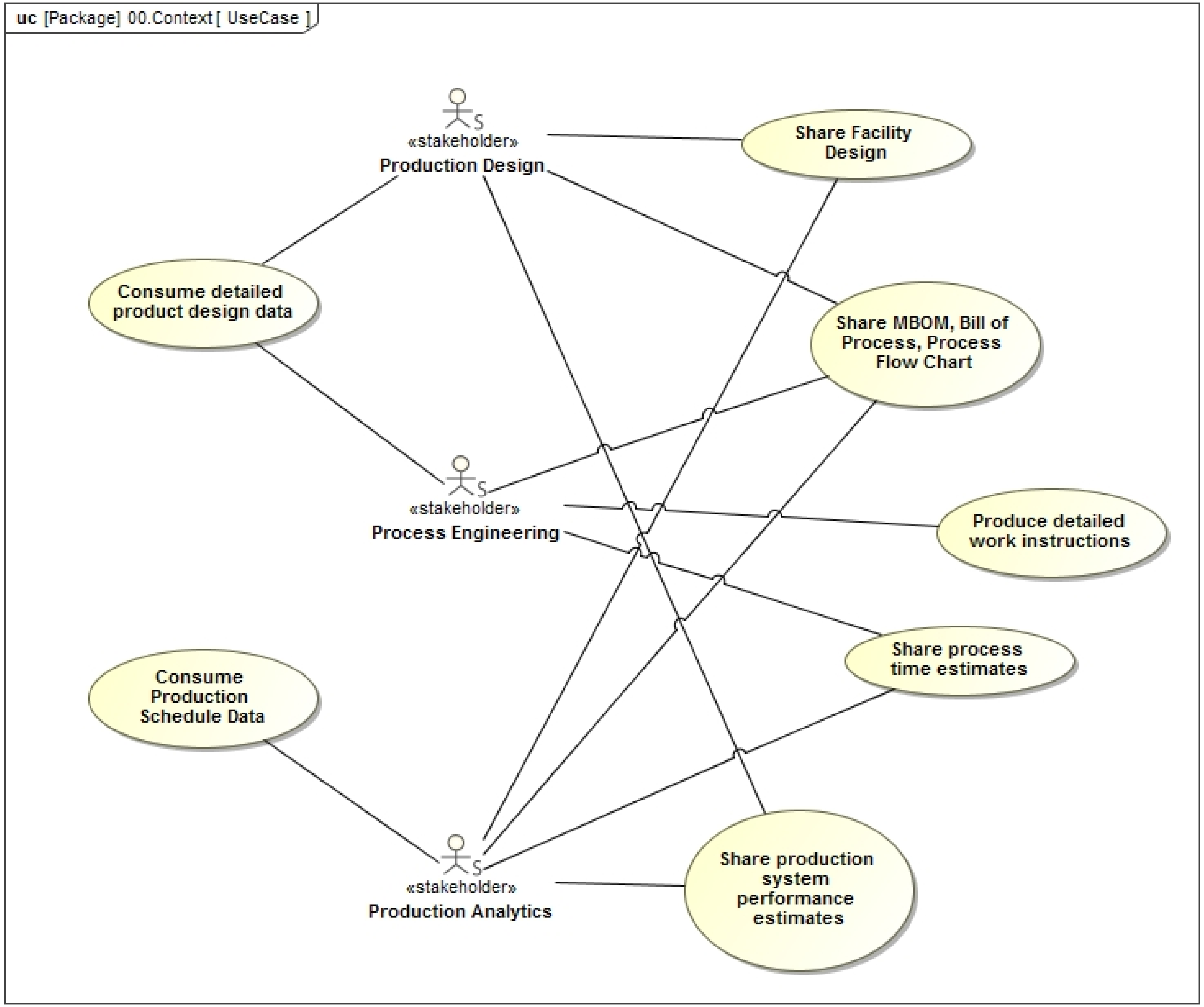
The consequences of poorly integrated decisions can be late to market and/or cost to produce

Stakeholders and interactions in Production



Points of view and responsibilities

- Product requirements
- Product design
- Production system resources
- Process instructions to create
- Process time estimates
- Performance prediction



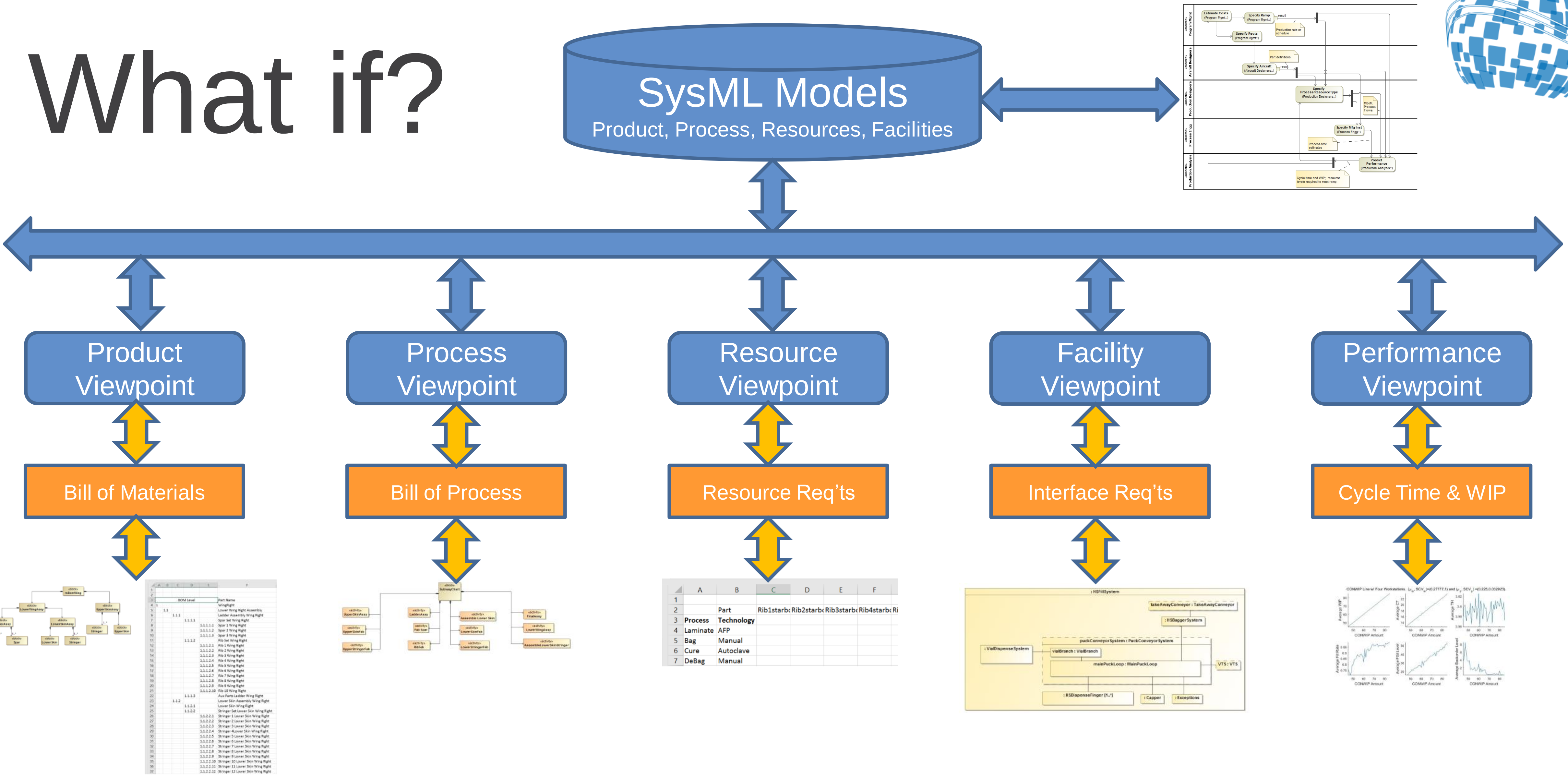
Developing the production system requires sharing a lot of technical information about the product, the intended production processes, the resources that will execute those processes, the instructions for executing those processes, the intended production schedule (or rate or ramp…), and the resulting cycle time and WIP levels.

Today, this information and the way it is shared is still largely *ad hoc*.

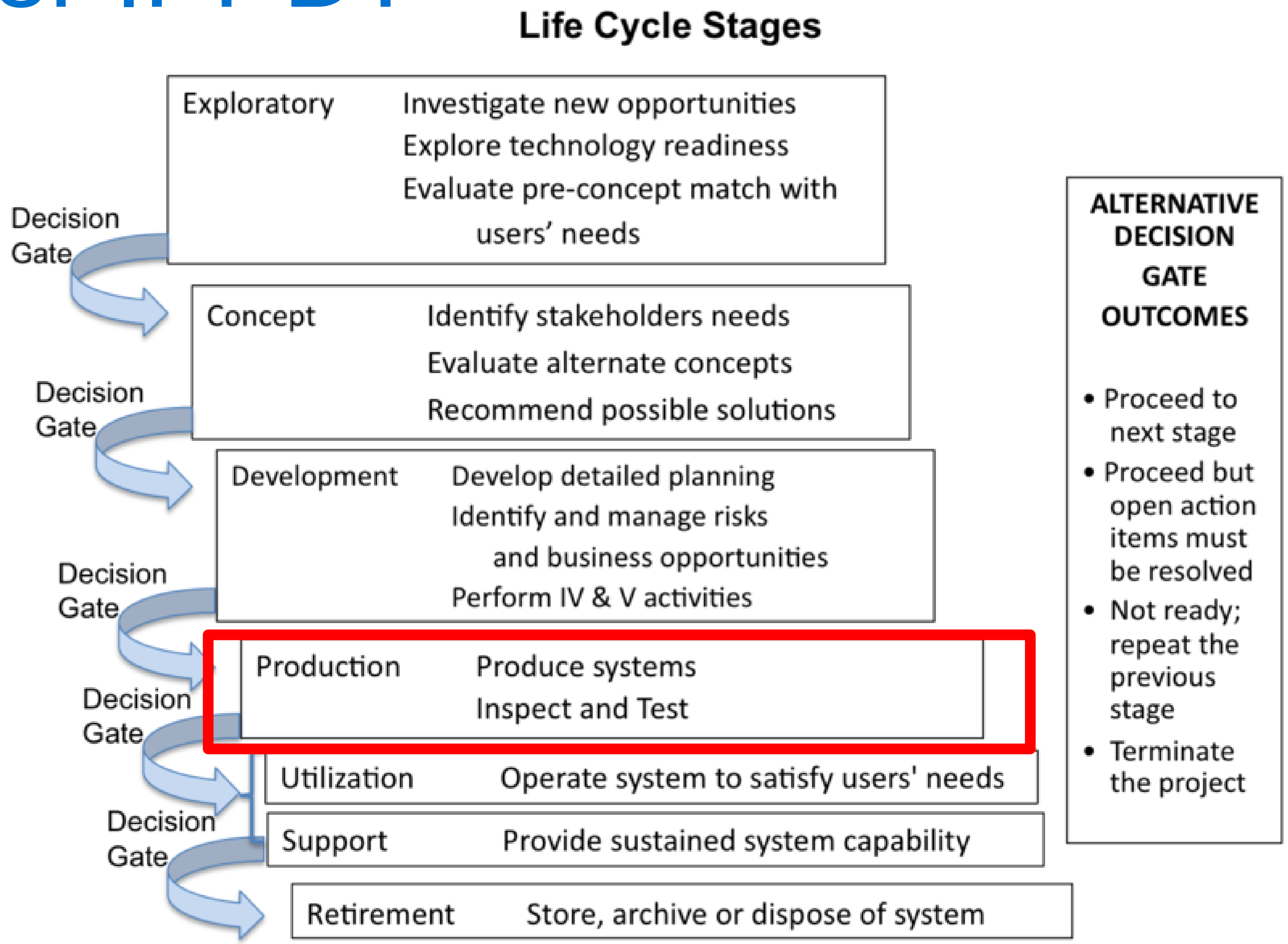
Consequences of current practice

- Time to market (time to full scale production) delays while the production system “bugs” are worked out
- Cost targets missed because
  - Resource capacity additions
  - Cycle time and WIP growth

What if?

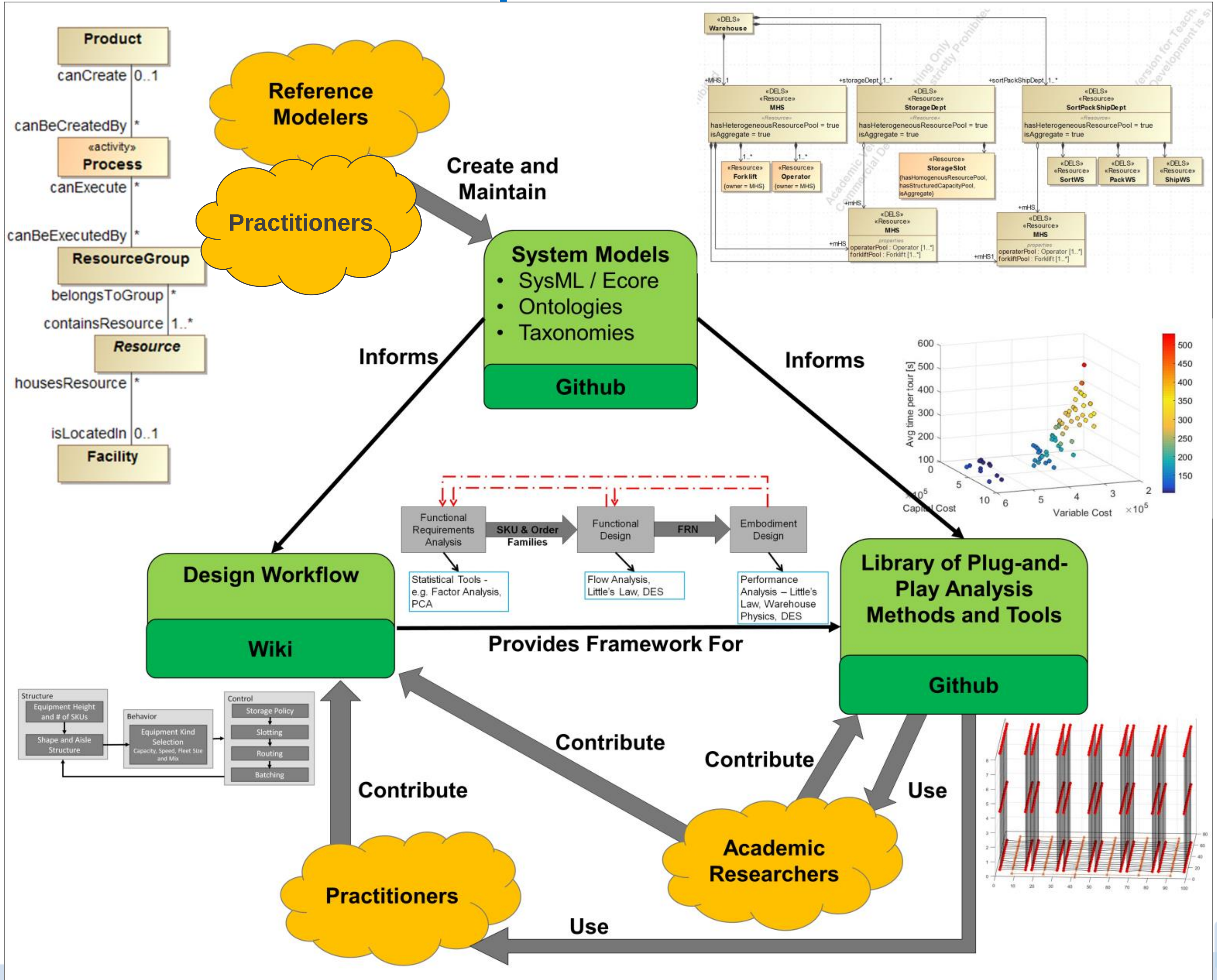

SysML Models
Product, Process, Resources, Facilities
Product Viewpoint
Process Viewpoint
Resource Viewpoint
Facility Viewpoint
Performance Viewpoint
Bill of Materials
Bill of Process
Resource Req’ts
Interface Req’ts
Cycle Time & WIP
Part
Process
Technology
Laminate
AFP
Bag
Manual
Cure
Autoclave
DeBag
Manual

Remember IPPD?



http://sebokwiki.org/wiki/System_Life_Cycle_Process_Models:_Vee

Mechanisms for development collaboration



Ubiquitous System Models: Where to start?

- Product, Process, Resource, & Facility
- How do you control your system?
- What do you want to know about the system?

Progress to date

- “Foundations” document: fundamental concepts and abstractions (-> developers)
- “Playbook” document: how to go about creating discipline- and analysis- agnostic production models (->modelers)
- “Case studies”: central fill pharmacy; composite parts manufacturing; semiconductor manufacturing (->general interest, students)
- All with associated SysML models

Acknowledgements

- NIST
- Collins Aerospace
- McKesson High Value Solutions
- Boeing
- Physical Internet Center, GaTech

It’s (long past) time to bring the power of (model based) systems engineering to production systems and global supply chains!

What does it take to do that?

Where are we in the journey?

Challenge team:
http://www.omgwiki.org/MBSE/doku.php?id=mbse:prodlog

Monday @ 1:00pm in Pier 10

timothy.sprock@nist.gov
leon.mcginnis@isye.gatech.edu
conrad.bock@nist.gov
gthiers3@gmail.com

2019

Annual INCOSE

international workshop

Torrance, CA, USA

January 26 - 29, 2019

www.incose.org/IW2019

2019 Annual INCOSE international workshop
Torrance, CA, USA
January 26 - 29, 2019

Production and Logistics Systems Modeling Challenge Team

Timothy Sprock[a], Leon McGinnis[b], & Conrad Bock[a]

[a] National Institute of Standards and Technology, [b] Georgia Tech

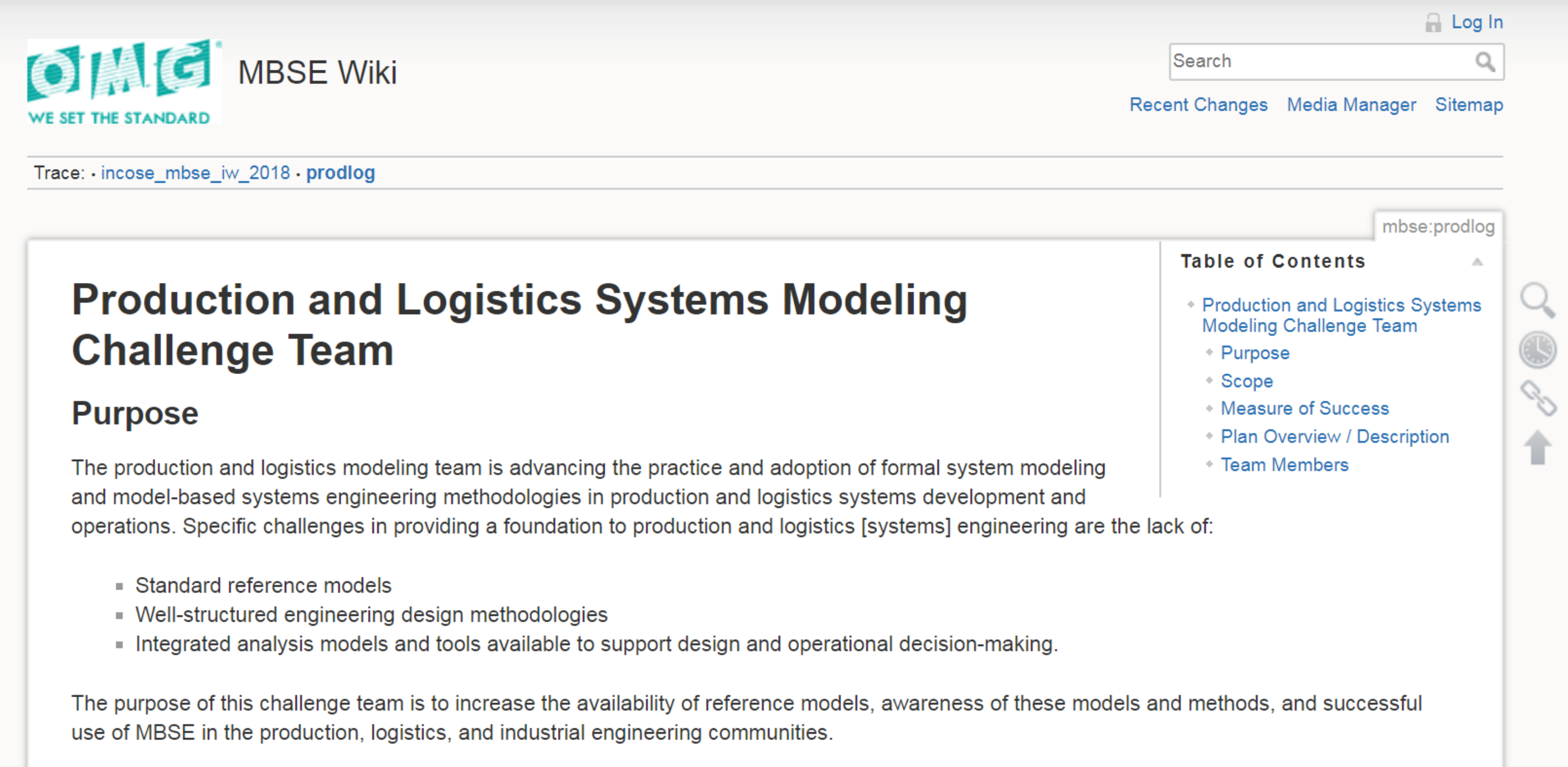
Agenda

- ***Review activities and progress to date;***
- ***feedback an discussion;***
- ***identify opportunities to contribute to existing efforts or important new activities.***

- Overview
- Value Proposition
- 2018 Work Items Status Update
  - Theory of DELS Specification
  - Model-based Industrial and Systems Engineering Playbook
- Case Studies
  - Central Fill Pharmacy Models – Leon McGinnis, Georgia Tech
  - Value Stream Mapping for Production – George Thiers, MBSE Tools
- Roadmap:
  - Document existing models and make them available
  - Identify and Document Use Cases, Refine Value Proposition
  - Identify Additional Case Studies
  - Identify Potential Liaisons

Production and Logistics Systems Modeling Charter

- http://www.omgwiki.org/MBSE/doku.php?id=mbse:prodlog



WE SET THE STANDARD MBSE Wiki

Log In

Search

Recent Changes Media Manager Sitemap

Trace: · incose_mbse_iw_2018 · **prodlog**

mbse:prodlog

Table of Contents

Production and Logistics Systems Modeling Challenge Team

Purpose

The production and logistics modeling team is advancing the practice and adoption of formal system modeling and model-based systems engineering methodologies in production and logistics systems development and operations. Specific challenges in providing a foundation to production and logistics [systems] engineering are the lack of:

- Standard reference models
- Well-structured engineering design methodologies
- Integrated analysis models and tools available to support design and operational decision-making.

The purpose of this challenge team is to increase the availability of reference models, awareness of these models and methods, and successful use of MBSE in the production, logistics, and industrial engineering communities.

Production and Logistics Systems Modeling Challenge Team

Increase the availability of reference models, awareness of these models and methods, and successful use of MBSE in the production, logistics, and industrial engineering communities.

Specific challenges in providing a foundation to production and logistics [systems] engineering are the lack of:

- Standard reference models
- Well-structured engineering design methodologies
- Integrated analysis models and tools available to support design and operational decision-making.

http://www.omgwiki.org/MBSE/doku.php?id=mbse:prodlog

Currently Active Contributors

- Tim Sprock, NIST: lead on “theory”; contributing everywhere
- Conrad Bock, NIST: technical guru
- George Thiers, MBSE Tools, Inc: lead on “playbook”
- Leon McGinnis, Georgia Tech: lead on “cases”
- Greg Pollari, Eugenio Rios, Collins Aerospace: contributing case study for playbook, industry perspective

Agenda

- Overview
- **Value Proposition**
- 2018 Work Items Status Update
  - Theory of DELS Specification
  - Model-based Industrial and Systems Engineering Playbook
- Case Studies
  - Central Fill Pharmacy Models – Leon McGinnis, Georgia Tech
  - Value Stream Mapping for Production – George Thiers, MBSE Tools
- Roadmap:
  - Document existing models and make them available
  - Identify and Document Use Cases, Refine Value Proposition
  - Identify Additional Case Studies
  - Identify Potential Liaisons

A Value Proposition for MBSE for Manufacturing Systems

George Thiers
MBSE Tools, Inc.
Alpharetta, GA, USA

Leon McGinnis
Georgia Tech ISyE
Atlanta, GA, USA

Timothy Sprock
Conrad Bock
National Institute of Standards and Technology
Gaithersburg, MD, USA

Greg Pollari
Eugenio Rios
Rockwell Collins
Cedar Rapids, IA, USA

Adam Graunke
Michael Christian
Boeing Research & Technology
Seattle, WA, USA

Model-Based Systems Engineering (MBSE) is defined as "the formalized application of modeling to support system requirements, design, analysis, verification, and validation activities" throughout all life-cycle phases [1]. When applied to product development, MBSE has demonstrated benefits including shorter time-to-market, increased product quality, and reduced program cost. [2, 3, 4, 5]. A manufacturing system can be regarded as just another product and modeled using conventional MBSE processes, methods, and tools, but this is far from contemporary practice, and is challenging due to the inherent complexity of a manufacturing system. This paper explores contemporary practices for design, diagnosis, and improvement of a discrete manufacturing system throughout its lifecycle, what MBSE's application might look like, and a value proposition for its inclusion.

1 A Manufacturing System's Lifecycle

To discuss contemporary practices for design, diagnosis, and improvement of a manufacturing system, it is first important to acknowledge that a system, models of it, and associated information and data are dynamic, not static. They evolve over time in predictable ways as a manufacturing system advances through its *lifecycle*. One definition of a manufacturing system lifecycle is shown in figure 1 [6].

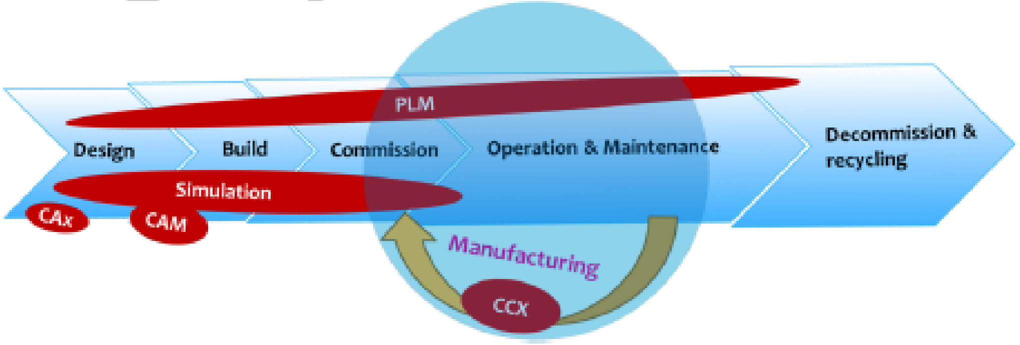


Figure 1: *A Manufacturing System Lifecycle.*

1

Intended audience: potential adopters of MBSE for Production and Logistics, both users and managers

Submitted to MBE Summit

– April 1-4, 2019 at NIST

– Preview:

https://v2.overleaf.com/read/pjjpsvkskgvn

Lifecycle		Concept, Early-Stage Design	Late-Stage Design, Build	Commission	Operation & Maintenance
What You Know	*Product*	Partial EBOM	EBOM, partial MBOM	EBOM, MBOM	EBOM, MBOM, with engineering changes
	Process	Make	Make, Measure, Test, partial Move & Store	Make, Measure, Test, Move, Store	Make, Measure, Test, Move, Store, Control
	Resource	Work Unit: Capability	Work Unit, partial Work Center: Capability, partial Capacity (available resource-hours per hour/shift/day)	Work Unit, Work Center, partial Area: Capability, Capacity, partial Performance	Work Unit, Work Center, Area: Capability, Capacity, Performance
	Facility	n/a	Location, partial Channel	Location, Channel	Location, Channel, Geometry
	Control	n/a	Admission, partial Sequencing (Prioritization of orders? Is expediting allowed? Are changeovers allowed?), partial Resource Assignment (Job shop or dedicated lines?)	Admission, Sequencing, Resource Assignment, partial *Scheduling* (Make to engineer, order, or stock? Push or pull?), partial Resource State Changes, partial Dynamic Process Planning (Is material handling scheduled or requested? Prioritization of requests? Is storage allowed?)	Admission, Sequencing, Resource Assignment, *Scheduling*, Resource State Changes, Dynamic Process Planning
What You Can Do	*Describe*	**(Product)** Does every part have a part number? A make/buy decision? A process plan if make? DFMA analyses? **(Process)** Does every make process have a make-to specification? A resource capable of its execution? **(Resource)** Are all requirements concerning capability, capacity, and performance allocated to resources?	**(Product)** Same, with a richer set of parts. **(Process)** Same, with a richer set of processes, plus: Gross execution capacity per process? With standard hours estimates, max execution rate per process? **(Resource)** Downtime causes per resource? Changeover time estimates? Material movement requirements per part? Channel requirements between resources? **(Facility)** Sizing requirements for Work Units & Work Centers? Storage constraints?	**(Product)** Same, with a richer set of parts. **(Process)** Same, with a richer set of processes, plus: Max operational cost per process? Gross execution capacity & max rate per logistical process? Contingency-triggered alternatives? **(Resource)** Downtime costs per resource? Changeover costs? Max material handling rate per channel? **(Facility)** Sizing requirements for per channel? Per storage buffer? Per Area? **(Control)** TH, CT, WIP, critical path, emerging bottlenecks?	*NEED HELP HERE; biggest change is that operational data is available.* **(Product)** Quality? **(Process)** Process alternatives upon contingencies? Waste? **(Resource)** Utilization, downtime, and changeover data. Material handling data. **(Facility)** Geometry-related. Channel congestion? Storage overflows? **(Control)** TH, CT, WIP, On-time deliveries, (see SCOR for more metrics). Per-job statistics.
	Predict	Lower & upper bounds on expected TH, CT, WIP, with fixed resources?	Refined lower & upper bounds on expected TH, CT, WIP, with fixed resources? Expected critical path? Potential bottlenecks?	Expected TH, CT, WIP? Expected critical path? Potential bottlenecks? Expected schedule delays or fractions of travelled work, per process?	Worst-case, expected, and best-case TH, CT, WIP, bottlenecks, on-time deliveries, schedule delays or fractions of travelled work for alternatives and scenarios?
	Prescribe	Lower & upper bounds on required resources, with fixed TH, CT, WIP requirements?	Refined lower & upper bounds on required resources, with fixed TH, CT, WIP requirements? Lower & upper bounds on material handling capacity? Projected storage buffers? Preliminary facility layout?	Expected resource requirements for make, measure, test processes? Expected resource requirements for move, store processes? Storage buffer capacities? Facility layout?	Adaptive redesigns: If a shortage of part type P, what should we do? If an outage of machine instance M, what should we do? Strategic redesigns, in response to changing external demand or internal technologies.

Model-based and systems engineering for discrete manufacturing systems enable:

- **Consistent Description** by fixing semantic gaps and inconsistencies among all manufacturing stakeholders. PLM and PDM have demonstrated the benefits of all stakeholders sharing consistent product and make-process data \cite{hill2003trendsetter}. It seems a small leap to argue that similar benefits could be realized by all stakeholders sharing consistent resource, facility, and control data.

- **Predictable and Prescribable Performance**: Manufacturing performance projections throughout the lifecycle for metrics including rate and cost, with confidence on par with product performance projections, plus prescribable ways to improve that performance.

- **Data-Driven Decision Making**: Evolving from a messy garage or black hole of one-off analytical models to a single-source-of-truth descriptive model that can be analyzed, interrogated, and the basis of automation. One application of automation is generation of analytical models to answer roughly 80% of ``routine'' questions, and while automatically-generated analytical models may never be as performance-optimized as humans' hand-crafted ones, the cost is almost trivial compared to the benefits gained in validation, verification, and trust.

- **Lifecycle Awareness**: A manufacturing system, its models, and its use cases are dynamic, not static, and evolve over time in predictable ways. Lifecycle awareness sets expectations for model content and utility over time.

- **Digital Integration** of initiatives including "smart manufacturing" and "digital thread" for a discrete manufacturing system. A data schema is a structural model, not a behavioral nor a control one, so without strong semantic-adding contributions from a human interpreter you'll never induce how a system actually works. Data doesn't give you the schema; you can infer one, but the span of that schema will only cover what's in the data - and nothing that's not. Statistical analysis performs description, and limited prediction under strong assumptions, effectively that the future will look a lot like the present and past.

Discussion: Value Proposition

- How would you apply MBISE?
- What would you want to do with it?

Agenda

- Overview
- Value Proposition
- 2018 Work Items Status Update
  - **Theory of DELS Specification**
  - Model-based Industrial and Systems Engineering Playbook
- Case Studies
  - Central Fill Pharmacy Models – Leon McGinnis, Georgia Tech
  - Value Stream Mapping for Production – George Thiers, MBSE Tools
- Roadmap:
  - Document existing models and make them available
  - Identify and Document Use Cases, Refine Value Proposition
  - Identify Additional Case Studies
  - Identify Potential Liaisons

Theory of Discrete Event Logistics Systems (DELS) Specification

Timothy Sprock[a], George Thiers[b], Leon McGinnis[c], Conrad Bock[a]

[a]National Institute of Standards and Technology, Gaithersburg, MD 20899
[b]MBSE Tools, Inc. Alpharetta, GA 30009
[c]H. Milton Stewart School of Industrial and Systems Engineering Georgia Institute of Technology, Atlanta, GA 30332

Abstract

abstract

Keywords: Discrete Event Logistics Systems (DELS); System Modeling; SysML

1. INTRODUCTION

A discrete event logistics system (or DELS) can be described as:

- a network of resources, arranged in a facility; each resource has one or more processing capabilities and for each capability, it has a capacity;
- a set of products flow through this network of resources, and are transformed by processes executed by the resources; a process may require the capabilities of more than one resource; the transformation can change location, age, or condition

The adjective "discrete" in this case recognizes the nature of the flows and processes. Flows are in discrete units, e.g., individual product units or components of product units, or batches of product units. Processes have well-defined start and end events, e.g., the start of a machining or heat-treating process, and the completion of same, even though our knowledge of the well-defined event time may be subject to uncertainty.

The concepts of DELS extend far beyond factories. A warehouse also is a DELS, albeit one with much simpler resources and processes. Similarly, a supply chain is a DELS, but

Email address: timothy.sprock@nist.gov (Timothy Sprock)

Preprint submitted to NISTIR - AMS *January 24, 2019*

Intended audience: developers of methods and tools who need to understand the deep technical foundations

Document (Preview):

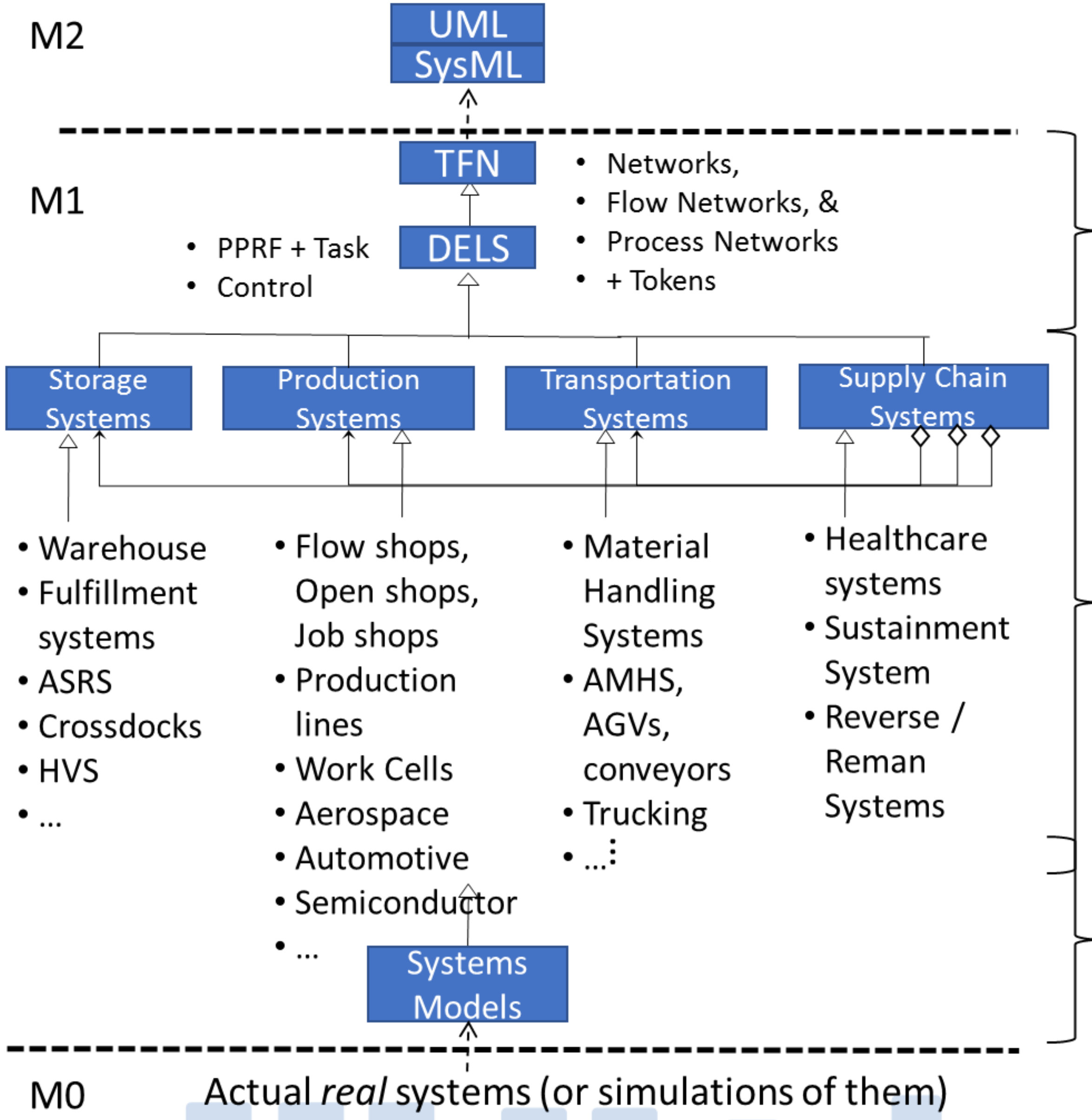
https://v2.overleaf.com/read/hhsmnkssjwcp

SysML Models:

https://github.com/usnistgov/DiscreteEventLogisticsSystems

Email timothy.sprock@nist.gov for access (need github account)

Reusable Model Libraries and Methods for Using Them



<u>Top of M1</u>

- **DELS Reference model**
- Network Abstractions
- PPRF Domain Ontology
- PPRF Taxonomies & Model Libraries
- Control Patterns

<u>Middle of M1</u>

- **(sub-) Domain-specific reference models and architectures**
- Generalization Set aligns with STORE, MAKE, & MOVE processes

<u>Bottom of M1</u>

- **System Models**
- “as-built” or “specification” models

Theory of Discrete Event Logistics Systems (DELS) Specification

1. Introduction
2. Modeling Framework
3. Network Abstractions
   - 3.1 Basic Networks
   - 3.2 Flow Networks
   - 3.3 Process Networks
4. Discrete Event Logistics Systems
   - 4.1 Resource
   - 4.2 Process
   - 4.3 Product
   - 4.4 Facility
   - 4.5 Task
   - 4.6 Interfaces
5. DELS Operational Control
   - 5.1 Patterns for Modeling Operational Control
   - 5.2 DELS Controller
6. Extended DELS Definition
7. Specializing DELS
8. Composing Specialized DELS

Agenda

- Overview
- Value Proposition
- 2018 Work Items Status Update
  - Theory of DELS Specification
  - **Model-based Industrial and Systems Engineering Playbook**
- Case Studies
  - Central Fill Pharmacy Models – Leon McGinnis, Georgia Tech
  - Value Stream Mapping for Production – George Thiers, MBSE Tools
- Roadmap:
  - Document existing models and make them available
  - Identify and Document Use Cases, Refine Value Proposition
  - Identify Additional Case Studies
  - Identify Potential Liaisons

Model-Based Industrial and Systems Engineering Playbook

Manufacturing Edition, Electronics Assembly example

George Thiers[1,2], Leon McGinnis[1], Timothy Sprock[3], Conrad Bock[3], Greg Pollari[4], Eugenio Rios[4], and Adam Graunke[5]

[1]Georgia Tech ISyE, Atlanta, GA 30332
[2]MBSE Tools, Inc., Alpharetta, GA 30009
[3]NIST, Gaithersburg, MD 20899
[4]Rockwell Collins, Cedar Rapids, IA 52402
[5]Boeing Research & Technology, Seattle, WA 98108

January 24, 2019
Document Version: 0.0.2
Tool Version: MagicDraw 18.5 sp3
Modeling Language Version: SysML 1.4, UML 2.5

Intended audience: production and logistics systems modelers; a “how to do it” guide

Document (Preview):

https://v2.overleaf.com/read/rsjqhqzmxtxq

SysML Models (Coming Soon):

https://github.com/usnistgov/DiscreteEventLogisticsSystems

Email timothy.sprock@nist.gov for access (need github account)

PRODUCT

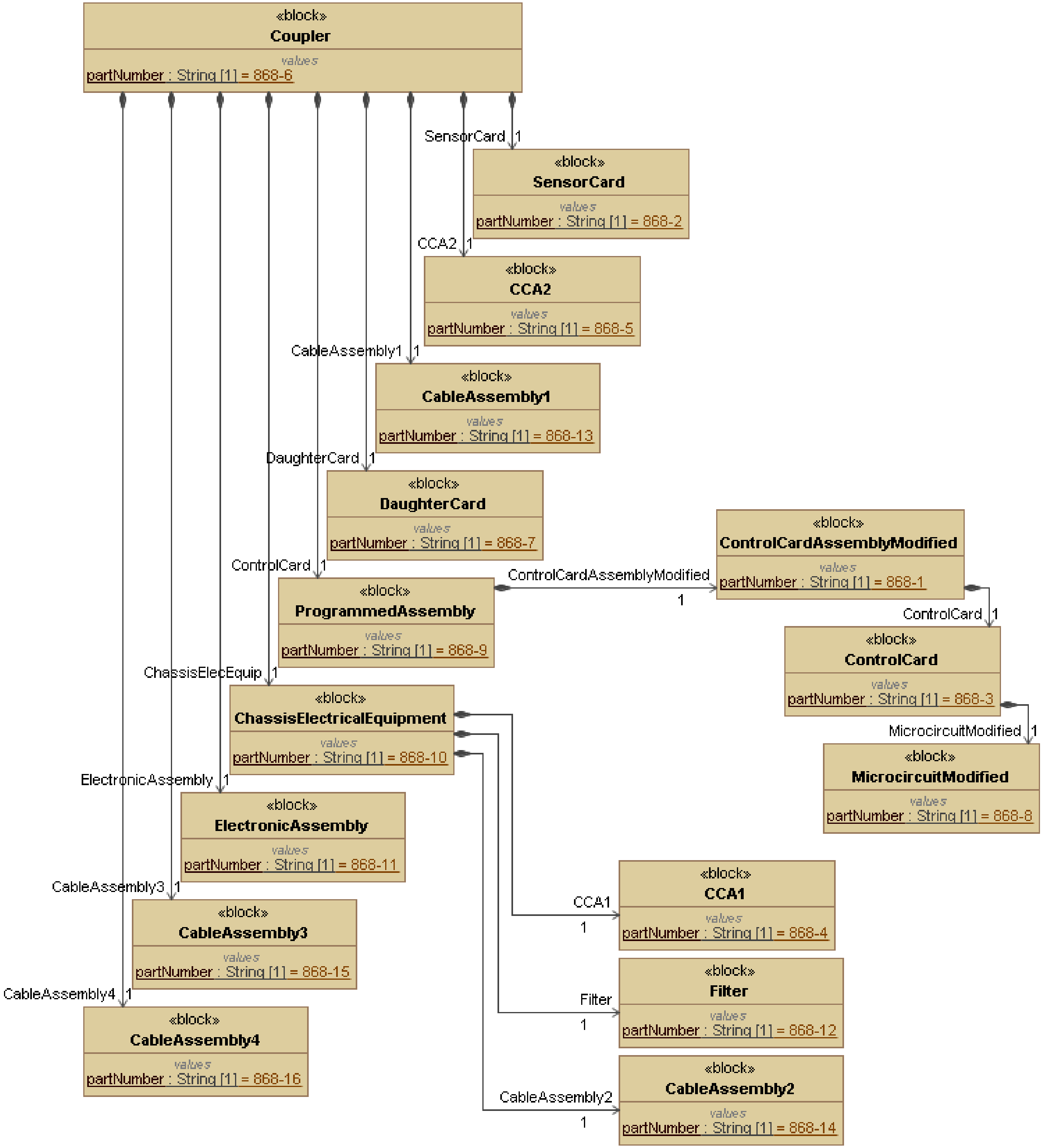
- *Identity and Composition*: Common starting point is an EBOM
- *Classification*: Identifying abstract part families enables reusable process definitions
- *Refinement*: Triggers include EBOM refinement, EBOM-> MBOM transition, EBOM & MBOM refinement
- *Complement Type with State*: Certain dimensions of a part’s state model may be relevant to manufacturing
- *Attach Data*: What part data is relevant to manufacturing, and how to model it?
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Product models can be big

PRODUCT

- *Identity and Composition*: Common starting point is an EBOM
- *Classification*: Identifying abstract part families enables reusable process definitions
- *Refinement*: Triggers include EBOM refinement, EBOM-> MBOM transition, EBOM & MBOM refinement
- *Complement Type with State*: Certain dimensions of a part’s state model may be relevant to manufacturing
- *Attach Data*: What part data is relevant to manufacturing, and how to model it?
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Product models can be big

Level	Material	Description
0.1	868-6	Coupler
..2	868-2	Sensor Card
..2	868-5	CCA 2
...3	868-13	Cable Assembly 1
..2	868-7	Daughter Card
..2	868-9	Programmed Assembly
...3	868-1	Control Card Assembly - Modified
....4	868-3	Control Card
.....5	868-8	Microcircuit, Modified
..2	868-10	Chassis Electrical Equipment
...3	868-4	CCA 1
...3	868-12	Filter
...3	868-14	Cable Assembly 2
..2	868-11	Electronic Assembly
..2	868-15	Cable Assembly 3
..2	868-16	Cable Assembly 4

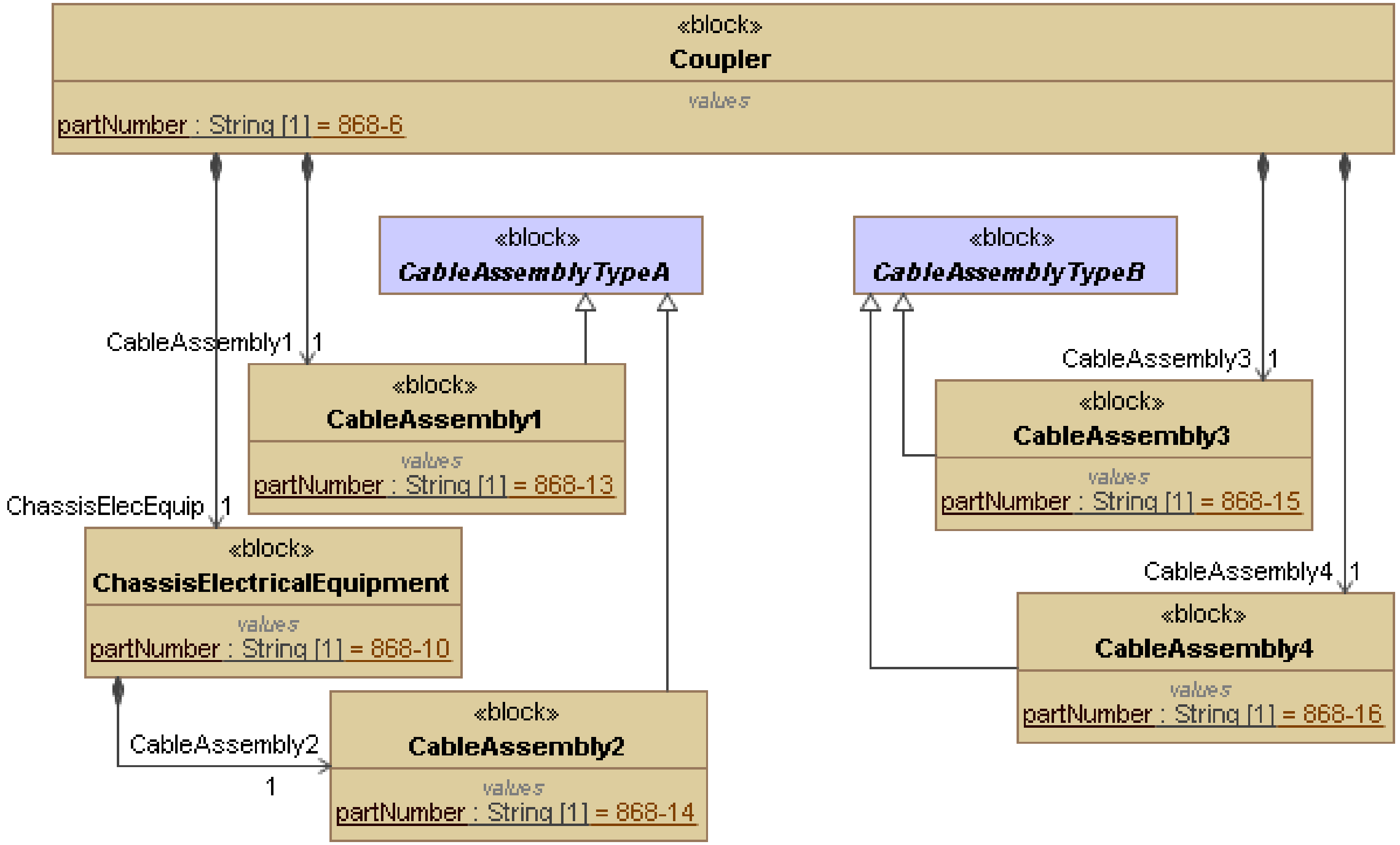
Table 2.1: EBOM information for a part type named a **Coupler**



«block»
Coupler
values
partNumber : String [1] = 868-6
SensorCard 1
«block»
SensorCard
values
partNumber : String [1] = 868-2
CCA2 1
«block»
CCA2
values
partNumber : String [1] = 868-5
CableAssembly1 1
«block»
CableAssembly1
values
partNumber : String [1] = 868-13
DaughterCard 1
«block»
DaughterCard
values
partNumber : String [1] = 868-7
ControlCard 1
«block»
ProgrammedAssembly
values
partNumber : String [1] = 868-9
ControlCardAssemblyModified
1
«block»
ControlCardAssemblyModified
values
partNumber : String [1] = 868-1
ControlCard 1
«block»
ControlCard
values
partNumber : String [1] = 868-3
MicrocircuitModified 1
«block»
MicrocircuitModified
values
partNumber : String [1] = 868-8
ChassisElecEquip 1
«block»
ChassisElectricalEquipment
values
partNumber : String [1] = 868-10
ElectronicAssembly 1
«block»
ElectronicAssembly
values
partNumber : String [1] = 868-11
CableAssembly3 1
«block»
CableAssembly3
values
partNumber : String [1] = 868-15
CableAssembly4 1
«block»
CableAssembly4
values
partNumber : String [1] = 868-16
CCA1
1
«block»
CCA1
values
partNumber : String [1] = 868-4
Filter
1
«block»
Filter
values
partNumber : String [1] = 868-12
CableAssembly2
1
«block»
CableAssembly2
values
partNumber : String [1] = 868-14

PRODUCT

- *Identity and Composition*: Common starting point is an EBOM
- *Classification*: Identifying abstract part families enables reusable process definitions
- *Refinement*: Triggers include EBOM refinement, EBOM-> MBOM transition, EBOM & MBOM refinement
- *Complement Type with State*: Certain dimensions of a part’s state model may be relevant to manufacturing
- *Attach Data*: What part data is relevant to manufacturing, and how to model it?
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Product models can be big



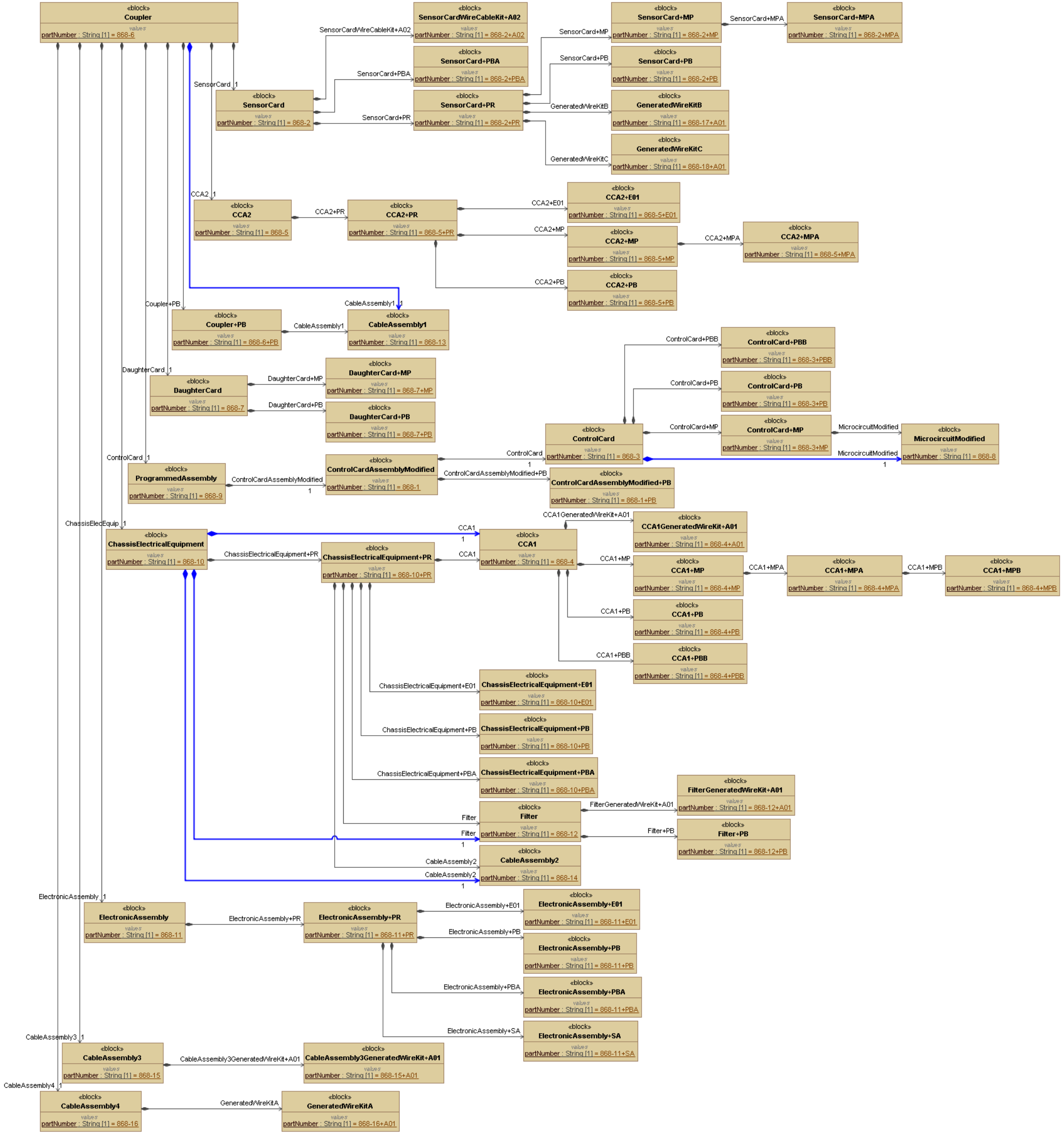
«block»
Coupler
values
partNumber : String [1] = 868-6
«block»
CableAssemblyTypeA
«block»
CableAssemblyTypeB
CableAssembly1
1
«block»
CableAssembly1
values
partNumber : String [1] = 868-13
CableAssembly3
1
«block»
CableAssembly3
values
partNumber : String [1] = 868-15
ChassisElecEquip
1
«block»
ChassisElectricalEquipment
values
partNumber : String [1] = 868-10
CableAssembly4
1
«block»
CableAssembly4
values
partNumber : String [1] = 868-16
CableAssembly2
1
«block»
CableAssembly2
values
partNumber : String [1] = 868-14

PRODUCT

- *Identity and Composition*: Common starting point is an EBOM
- *Classification*: Identifying abstract part families enables reusable process definitions
- *Refinement*: Triggers include EBOM refinement, EBOM-> MBOM transition, EBOM & MBOM refinement
- *Complement Type with State*: Certain dimensions of a part's state model may be relevant to manufacturing
- *Attach Data*: What part data is relevant to manufacturing, and how to model it?
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Product models can be big

Level	Material	Description
0.1	868-6	Coupler
..2	868-2	Sensor Card
...3	868-2+A02	Sensor Card Wire/Cable Kit
...3	868-2+PBA	Sensor Card
...3	868-2+PR	Sensor Card
....4	868-2+MP	Sensor Card
.....5	868-2+MPA	Sensor Card
....4	868-2+PB	Sensor Card
....4	868-17+A01	Generated Wire Kit B
....4	868-18+A01	Generated Wire Kit C
..2	868-5	CCA 2
...3	868-5+PR	CCA 2
....4	868-5+E01	CCA 2
....4	868-5+MP	CCA 2
.....5	868-5+MPA	CCA 2
....4	868-5+PB	CCA 2
..2	868-6+PB	Coupler
...3	868-13	Cable Assembly 1
..2	868-7	Daughter Card
...3	868-7+MP	Daughter Card
...3	868-7+PB	Daughter Card
..2	868-9	Programmed Assembly
...3	868-1	Control Card Assembly - Modified
....4	868-1+PB	Control Card Assembly - Modified
....4	868-3	Control Card
.....5	868-3+MP	Control Card
......6	868-8	Microcircuit, Modified
.....5	868-3+PB	Control Card
.....5	868-3+PBB	Control Card
..2	868-10	Chassis Electrical Equipment
...3	868-10+PR	Chassis Electrical Equipment
....4	868-4	CCA 1
.....5	868-4+A01	CCA 1 Generated Wire Kit
.....5	868-4+MP	CCA 1
......6	868-4+MPA	CCA 1
.......7	868-4+MPB	CCA 1
.....5	868-4+PB	CCA 1
.....5	868-4+PBB	CCA 1
....4	868-10+E01	Chassis Electrical Equipment
....4	868-10+PB	Chassis Electrical Equipment
....4	868-10+PBA	Chassis Electrical Equipment
....4	868-12	Filter
.....5	868-12+A01	Filter Generated Wire Kit
.....5	868-12+PB	Filter
....4	868-14	Cable Assembly 2
..2	868-11	Electronic Assembly
...3	868-11+PR	Electronic Assembly
....4	868-11+E01	Electronic Assembly
....4	868-11+PB	Electronic Assembly
....4	868-11+PBA	Electronic Assembly
....4	868-11+SA	Electronic Assembly
..2	868-15	Cable Assembly 3
...3	868-15+A01	Cable Assembly 3 Generated Wire Kit
..2	868-16	Cable Assembly 4
...3	868-16+A01	Generated Wire Kit A

Table 2.2: MBOM information for a part type named a **Coupler**.



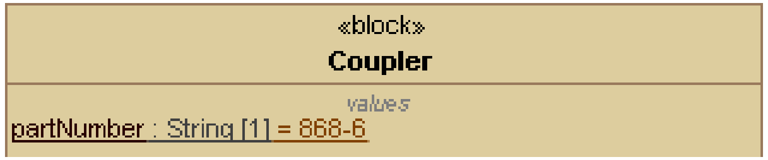
«block» Coupler
values
partNumber : String [1] = 868-6
«block» SensorCardWireCableKit+A02
partNumber : String [1] = 868-2+A02
«block» SensorCard+MP
partNumber : String [1] = 868-2+MP
«block» SensorCard+MPA
partNumber : String [1] = 868-2+MPA
«block» SensorCard+PBA
partNumber : String [1] = 868-2+PBA
«block» SensorCard+PB
partNumber : String [1] = 868-2+PB
«block» SensorCard
partNumber : String [1] = 868-2
«block» SensorCard+PR
partNumber : String [1] = 868-2+PR
«block» GeneratedWireKitB
partNumber : String [1] = 868-17+A01
«block» GeneratedWireKitC
partNumber : String [1] = 868-18+A01
SensorCardWireCableKit+A02
SensorCard+MP
SensorCard+MPA
SensorCard+PB
SensorCard+PBA
SensorCard+PR
GeneratedWireKitB
GeneratedWireKitC
SensorCard 1
CCA2 1
«block» CCA2
partNumber : String [1] = 868-5
CCA2+PR
«block» CCA2+PR
partNumber : String [1] = 868-5+PR
«block» CCA2+E01
partNumber : String [1] = 868-5+E01
CCA2+E01
«block» CCA2+MP
partNumber : String [1] = 868-5+MP
CCA2+MP
«block» CCA2+MPA
partNumber : String [1] = 868-5+MPA
CCA2+MPA
«block» CCA2+PB
partNumber : String [1] = 868-5+PB
CCA2+PB
Coupler+PB
CableAssembly1 1
«block» Coupler+PB
partNumber : String [1] = 868-6+PB
CableAssembly1
«block» CableAssembly1
partNumber : String [1] = 868-13
DaughterCard 1
«block» DaughterCard
partNumber : String [1] = 868-7
DaughterCard+MP
«block» DaughterCard+MP
partNumber : String [1] = 868-7+MP
DaughterCard+PB
«block» DaughterCard+PB
partNumber : String [1] = 868-7+PB
ControlCard+PBB
«block» ControlCard+PBB
partNumber : String [1] = 868-3+PBB
ControlCard+PB
«block» ControlCard+PB
partNumber : String [1] = 868-3+PB
ControlCard+MP
«block» ControlCard+MP
partNumber : String [1] = 868-3+MP
«block» ControlCard
partNumber : String [1] = 868-3
MicrocircuitModified
«block» MicrocircuitModified
partNumber : String [1] = 868-8
ControlCard 1
«block» ProgrammedAssembly
partNumber : String [1] = 868-9
ControlCardAssemblyModified
«block» ControlCardAssemblyModified
partNumber : String [1] = 868-1
ControlCard
ControlCardAssemblyModified+PB
«block» ControlCardAssemblyModified+PB
partNumber : String [1] = 868-1+PB
ChassisElecEquip 1
«block» ChassisElectricalEquipment
partNumber : String [1] = 868-10
CCA1
ChassisElectricalEquipment+PR
«block» ChassisElectricalEquipment+PR
partNumber : String [1] = 868-10+PR
«block» CCA1
partNumber : String [1] = 868-4
CCA1GeneratedWireKit+A01
«block» CCA1GeneratedWireKit+A01
partNumber : String [1] = 868-4+A01
CCA1+MP
«block» CCA1+MP
partNumber : String [1] = 868-4+MP
CCA1+MPA
«block» CCA1+MPA
partNumber : String [1] = 868-4+MPA
CCA1+MPB
«block» CCA1+MPB
partNumber : String [1] = 868-4+MPB
CCA1+PB
«block» CCA1+PB
partNumber : String [1] = 868-4+PB
CCA1+PBB
«block» CCA1+PBB
partNumber : String [1] = 868-4+PBB
ChassisElectricalEquipment+E01
«block» ChassisElectricalEquipment+E01
partNumber : String [1] = 868-10+E01
ChassisElectricalEquipment+PB
«block» ChassisElectricalEquipment+PB
partNumber : String [1] = 868-10+PB
ChassisElectricalEquipment+PBA
«block» ChassisElectricalEquipment+PBA
partNumber : String [1] = 868-10+PBA
FilterGeneratedWireKit+A01
«block» FilterGeneratedWireKit+A01
partNumber : String [1] = 868-12+A01
Filter
«block» Filter
partNumber : String [1] = 868-12
Filter+PB
«block» Filter+PB
partNumber : String [1] = 868-12+PB
CableAssembly2
«block» CableAssembly2
partNumber : String [1] = 868-14
ElectronicAssembly 1
«block» ElectronicAssembly
partNumber : String [1] = 868-11
ElectronicAssembly+PR
«block» ElectronicAssembly+PR
partNumber : String [1] = 868-11+PR
ElectronicAssembly+E01
«block» ElectronicAssembly+E01
partNumber : String [1] = 868-11+E01
ElectronicAssembly+PB
«block» ElectronicAssembly+PB
partNumber : String [1] = 868-11+PB
ElectronicAssembly+PBA
«block» ElectronicAssembly+PBA
partNumber : String [1] = 868-11+PBA
ElectronicAssembly+SA
«block» ElectronicAssembly+SA
partNumber : String [1] = 868-11+SA
CableAssembly3 1
«block» CableAssembly3
partNumber : String [1] = 868-15
CableAssembly3GeneratedWireKit+A01
«block» CableAssembly3GeneratedWireKit+A01
partNumber : String [1] = 868-15+A01
CableAssembly4 1
«block» CableAssembly4
partNumber : String [1] = 868-16
GeneratedWireKitA
«block» GeneratedWireKitA
partNumber : String [1] = 868-16+A01

PRODUCT

- *Identity and Composition*: Common starting point is an EBOM
- *Classification*: Identifying abstract part families enables reusable process definitions
- *Refinement*: Triggers include EBOM refinement, EBOM-> MBOM transition, EBOM & MBOM refinement
- *Complement Type with State*: Certain dimensions of a part's state model may be relevant to manufacturing
- *Attach Data*: What part data is relevant to manufacturing, and how to model it?
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Product models can be big

PRODUCT

- *Identity and Composition*: Common starting point is an EBOM
- *Classification*: Identifying abstract part families enables reusable process definitions
- *Refinement*: Triggers include EBOM refinement, EBOM-> MBOM transition, EBOM & MBOM refinement
- *Complement Type with State*: Certain dimensions of a part's state model may be relevant to manufacturing
- *Attach Data*: What part data is relevant to manufacturing, and how to model it?
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Product models can be big

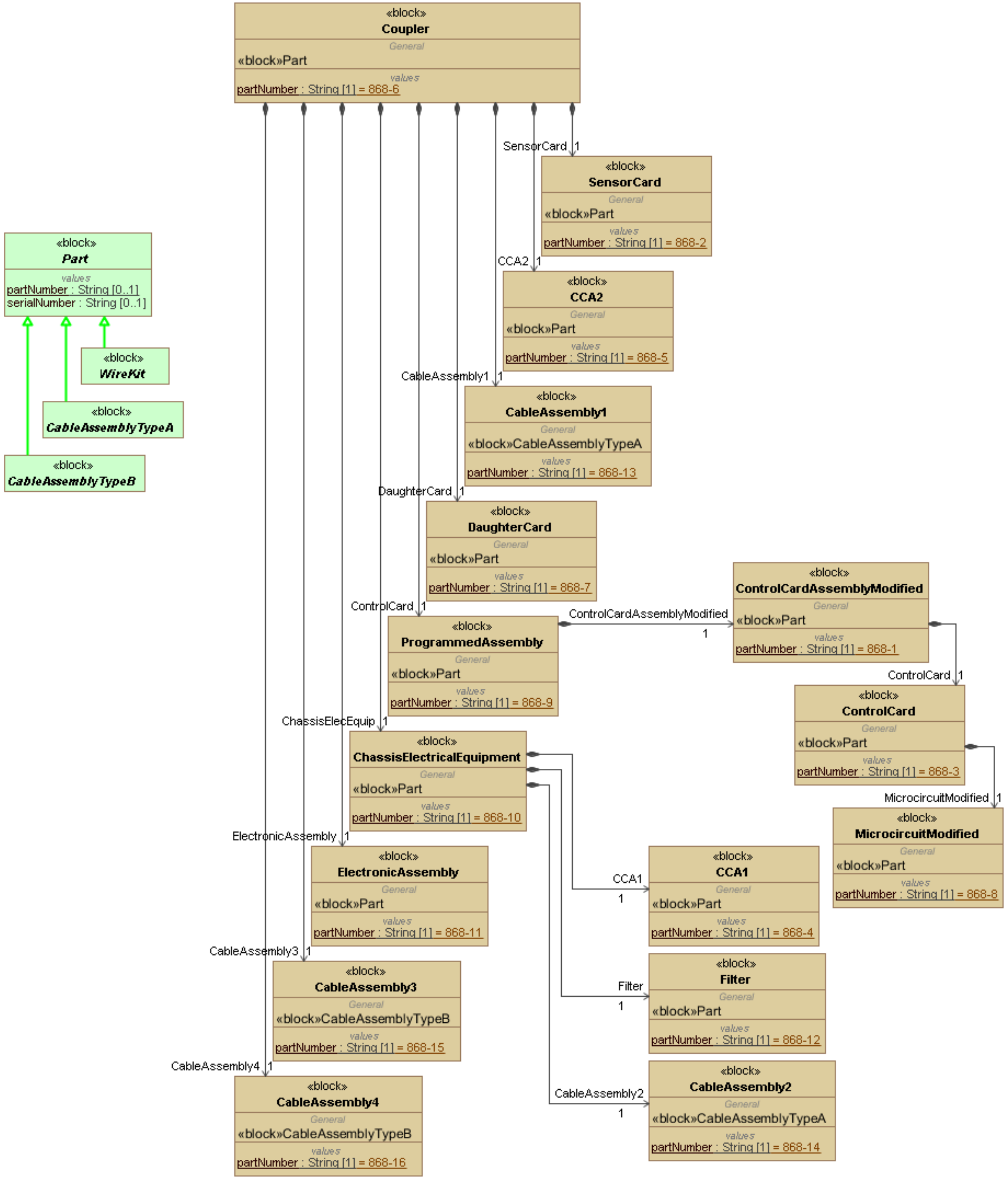


The primary mechanism to attach data values is the SysML value property. Options include:

- Per-instance data (e.g. Serial Number): Model instantiation is required in order to enter unique data values.
- Per-type data (e.g. Part Number): No instantiation required, use property’s “default value”.
- Per-usage data (no examples yet): No instantiation required, use usage’s “context-specific initial value”.

PRODUCT

- *Identity and Composition*: Common starting point is an EBOM
- *Classification*: Identifying abstract part families enables reusable process definitions
- *Refinement*: Triggers include EBOM refinement, EBOM-> MBOM transition, EBOM & MBOM refinement
- *Complement Type with State*: Certain dimensions of a part's state model may be relevant to manufacturing
- *Attach Data*: What part data is relevant to manufacturing, and how to model it?
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Product models can be big


«block»
Coupler
General
«block»Part
values
partNumber : String [1] = 868-6
SensorCard 1
«block»
SensorCard
General
«block»Part
values
partNumber : String [1] = 868-2
CCA2 1
«block»
CCA2
General
«block»Part
values
partNumber : String [1] = 868-5
CableAssembly1 1
«block»
CableAssembly1
General
«block»CableAssemblyTypeA
values
partNumber : String [1] = 868-13
DaughterCard 1
«block»
DaughterCard
General
«block»Part
values
partNumber : String [1] = 868-7
ControlCard 1
«block»
ProgrammedAssembly
General
«block»Part
values
partNumber : String [1] = 868-9
ControlCardAssemblyModified
1
«block»
ControlCardAssemblyModified
General
«block»Part
values
partNumber : String [1] = 868-1
ControlCard 1
«block»
ControlCard
General
«block»Part
values
partNumber : String [1] = 868-3
MicrocircuitModified 1
«block»
MicrocircuitModified
General
«block»Part
values
partNumber : String [1] = 868-8
ChassisElecEquip 1
«block»
ChassisElectricalEquipment
General
«block»Part
values
partNumber : String [1] = 868-10
CCA1
1
«block»
CCA1
General
«block»Part
values
partNumber : String [1] = 868-4
Filter
1
«block»
Filter
General
«block»Part
values
partNumber : String [1] = 868-12
CableAssembly2
1
«block»
CableAssembly2
General
«block»CableAssemblyTypeA
values
partNumber : String [1] = 868-14
ElectronicAssembly 1
«block»
ElectronicAssembly
General
«block»Part
values
partNumber : String [1] = 868-11
CableAssembly3 1
«block»
CableAssembly3
General
«block»CableAssemblyTypeB
values
partNumber : String [1] = 868-15
CableAssembly4 1
«block»
CableAssembly4
General
«block»CableAssemblyTypeB
values
partNumber : String [1] = 868-16
«block»
Part
values
partNumber : String [0..1]
serialNumber : String [0..1]
«block»
WireKit
«block»
CableAssemblyTypeA
«block»
CableAssemblyTypeB

PRODUCT

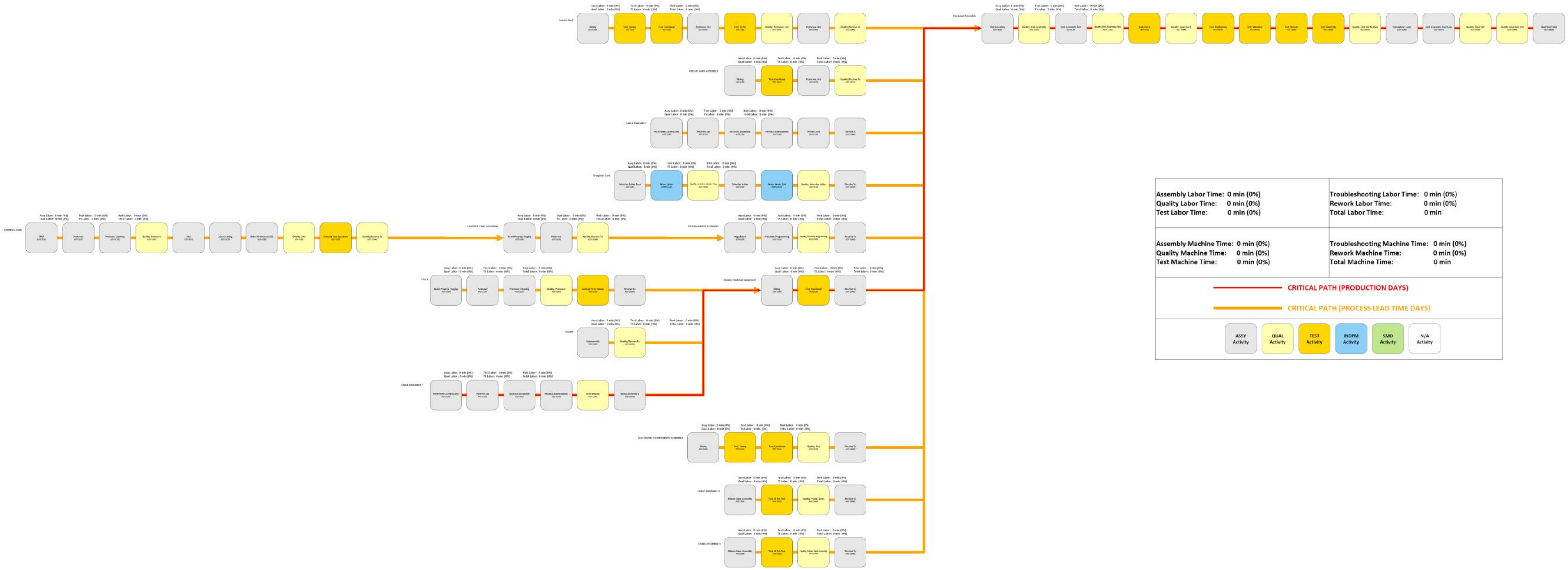
- *Identity and Composition*: Common starting point is an EBOM
- *Classification*: Identifying abstract part families enables reusable process definitions
- *Refinement*: Triggers include EBOM refinement, EBOM-> MBOM transition, EBOM & MBOM refinement
- *Complement Type with State*: Certain dimensions of a part's state model may be relevant to manufacturing
- *Attach Data*: What part data is relevant to manufacturing, and how to model it?
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Product models can be big

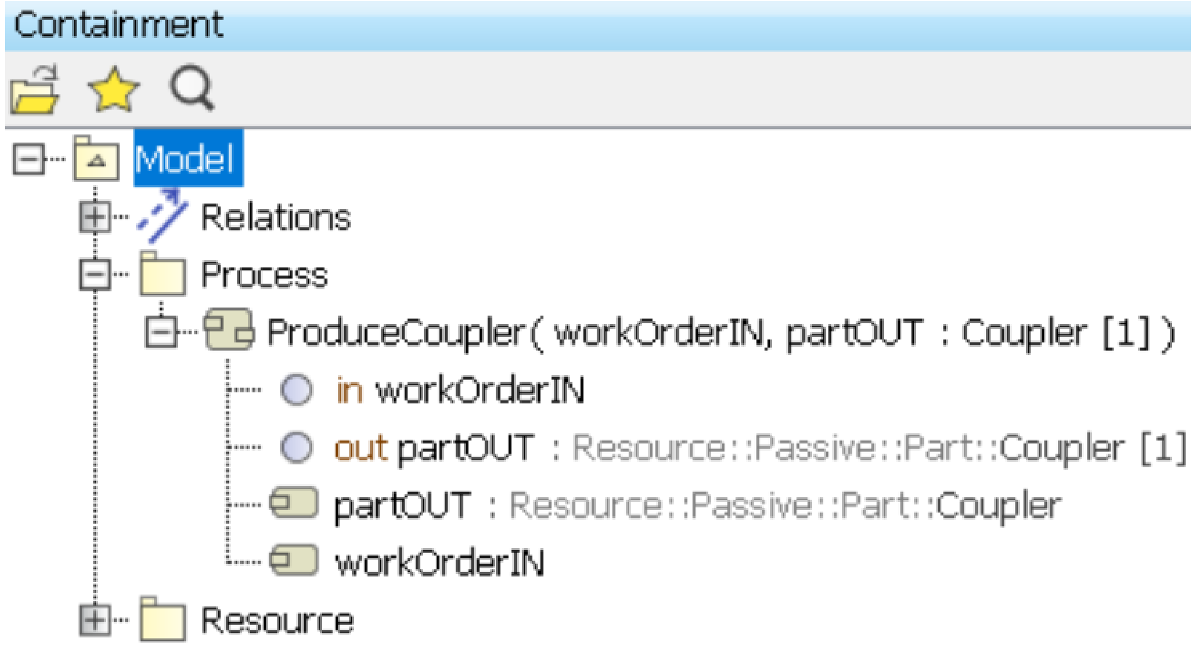
PROCESS

- *Top-Level Process and its I/O*: Black-box definition of the top-level manufacturing transformation
- *Input/Output*: Parts already have type; Work Orders need types too
- *Refinement*: Add a lower-level process
- *Exclusions from the Process Model*
- *Refinement*: To a leaf-level
- *Which Process is being Requested*: Make versus Deliver
- *Complement Type with State*: For processes' part I/O
- *Attach Data*: To both processes and work orders
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Process models can be big, just like Product models

PROCESS

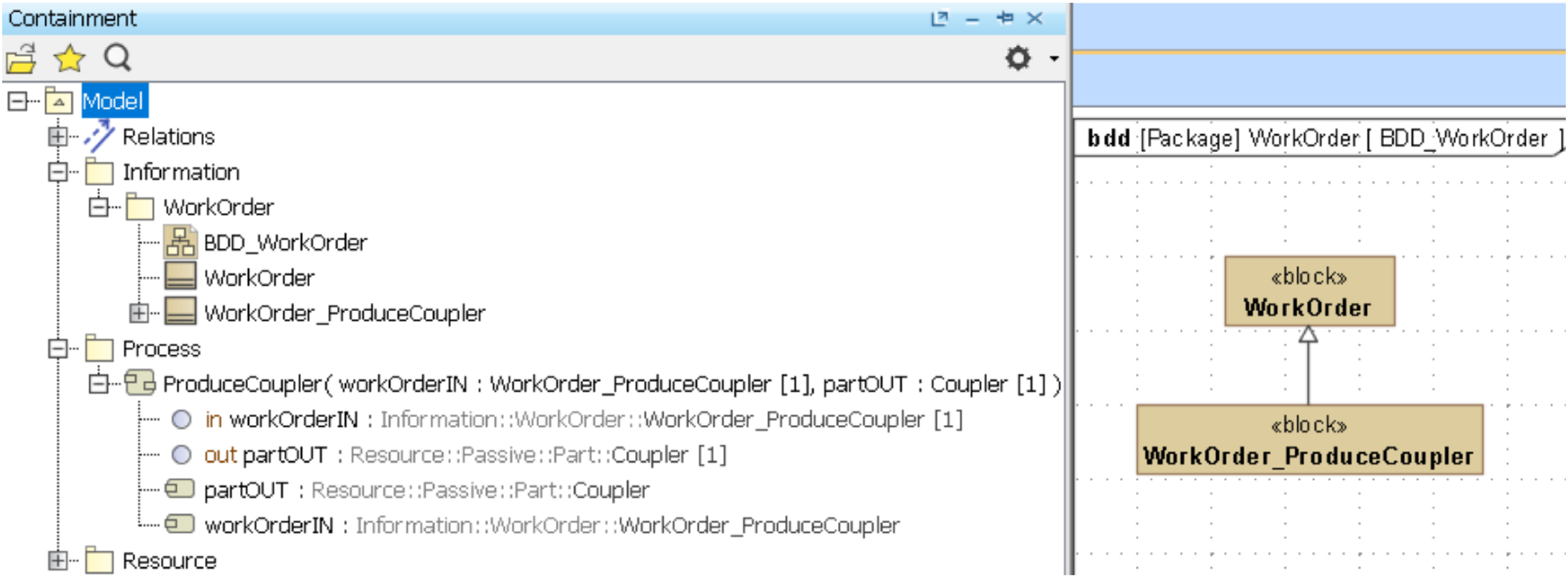
- *Top-Level Process and its I/O*: Black-box definition of the top-level manufacturing transformation
- *Input/Output*: Parts already have type; Work Orders need types too
- *Refinement*: Add a lower-level process
- *Exclusions from the Process Model*
- *Refinement*: To a leaf-level
- *Which Process is being Requested*: Make versus Deliver
- *Complement Type with State*: For processes' part I/O
- *Attach Data*: To both processes and work orders
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Process models can be big, just like Product models

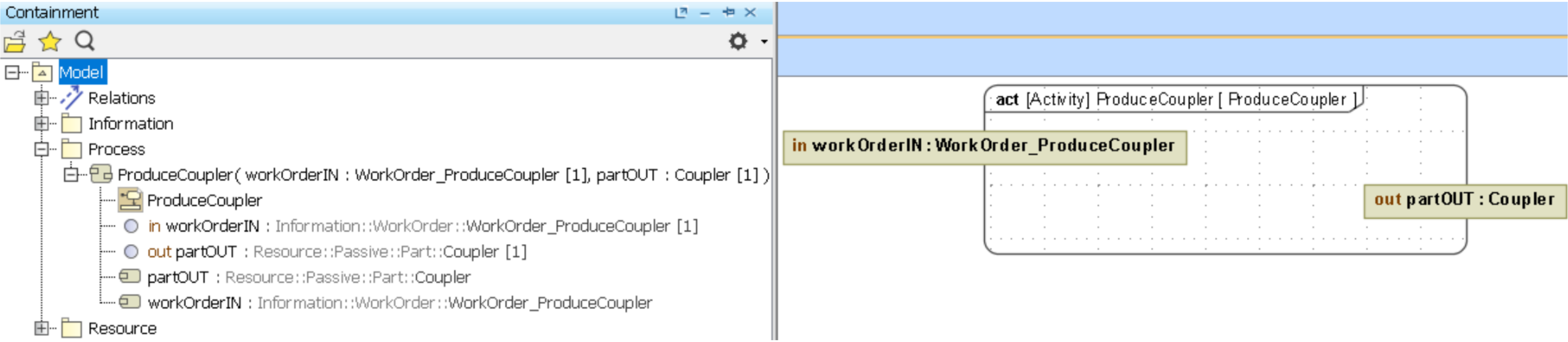

Assembly Labor Time: 0 min (0%)
Quality Labor Time: 0 min (0%)
Test Labor Time: 0 min (0%)
Troubleshooting Labor Time: 0 min (0%)
Rework Labor Time: 0 min (0%)
Total Labor Time: 0 min
Assembly Machine Time: 0 min (0%)
Quality Machine Time: 0 min (0%)
Test Machine Time: 0 min (0%)
Troubleshooting Machine Time: 0 min (0%)
Rework Machine Time: 0 min (0%)
Total Machine Time: 0 min
CRITICAL PATH (PRODUCTION DAYS)
CRITICAL PATH (PROCESS LEAD TIME DAYS)
ASSY Activity
QUAL Activity
TEST Activity
INDPM Activity
SMD Activity
N/A Activity


Containment
Model
Relations
Process
ProduceCoupler(workOrderIN, partOUT : Coupler [1])
in workOrderIN
out partOUT : Resource::Passive::Part::Coupler [1]
partOUT : Resource::Passive::Part::Coupler
workOrderIN
Resource

PROCESS

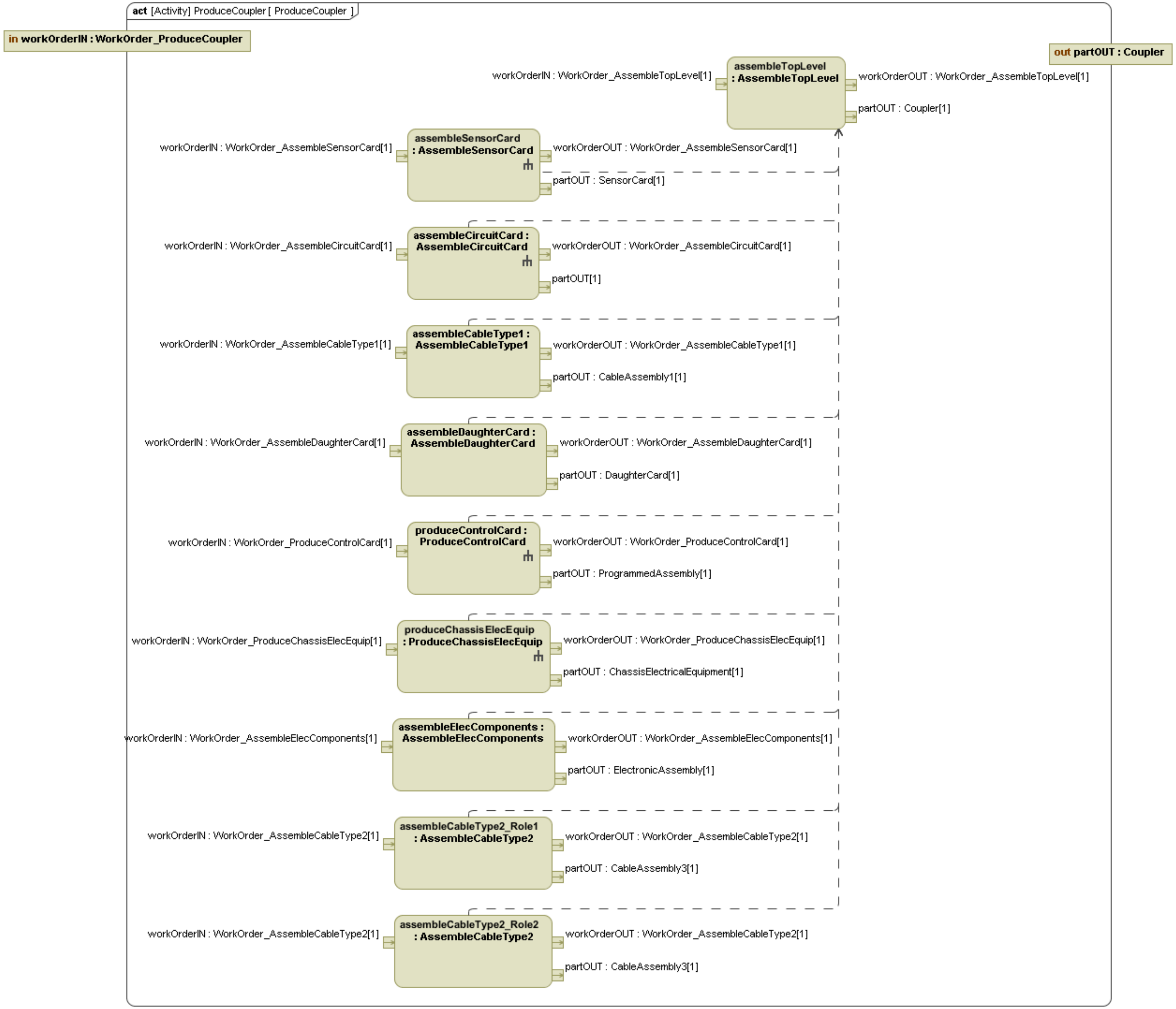
- *Top-Level Process and its I/O*: Black-box definition of the top-level manufacturing transformation
- *Input/Output*: Parts already have type; Work Orders need types too
- *Refinement*: Add a lower-level process
- *Exclusions from the Process Model*
- *Refinement*: To a leaf-level
- *Which Process is being Requested*: Make versus Deliver
- *Complement Type with State*: For processes' part I/O
- *Attach Data*: To both processes and work orders
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Process models can be big, just like Product models


Containment
Model
Relations
Information
WorkOrder
BDD_WorkOrder
WorkOrder
WorkOrder_ProduceCoupler
Process
ProduceCoupler(workOrderIN : WorkOrder_ProduceCoupler [1], partOUT : Coupler [1])
in workOrderIN : Information::WorkOrder::WorkOrder_ProduceCoupler [1]
out partOUT : Resource::Passive::Part::Coupler [1]
partOUT : Resource::Passive::Part::Coupler
workOrderIN : Information::WorkOrder::WorkOrder_ProduceCoupler
Resource
bdd [Package] WorkOrder [BDD_WorkOrder]
«block»
WorkOrder
«block»
WorkOrder_ProduceCoupler


Containment
Model
Relations
Information
Process
ProduceCoupler(workOrderIN : WorkOrder_ProduceCoupler [1], partOUT : Coupler [1])
ProduceCoupler
in workOrderIN : Information::WorkOrder::WorkOrder_ProduceCoupler [1]
out partOUT : Resource::Passive::Part::Coupler [1]
partOUT : Resource::Passive::Part::Coupler
workOrderIN : Information::WorkOrder::WorkOrder_ProduceCoupler
Resource
act [Activity] ProduceCoupler [ProduceCoupler]
in workOrderIN : WorkOrder_ProduceCoupler
out partOUT : Coupler

PROCESS

- *Top-Level Process and its I/O*: Black-box definition of the top-level manufacturing transformation
- *Input/Output*: Parts already have type; Work Orders need types too
- *Refinement*: Add a lower-level process
- *Exclusions from the Process Model*
- *Refinement*: To a leaf-level
- *Which Process is being Requested*: Make versus Deliver
- *Complement Type with State*: For processes' part I/O
- *Attach Data*: To both processes and work orders
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Process models can be big, just like Product models



act [Activity] ProduceCoupler [ProduceCoupler]
in workOrderIN : WorkOrder_ProduceCoupler
out partOUT : Coupler
workOrderIN : WorkOrder_AssembleTopLevel[1]
assembleTopLevel : AssembleTopLevel
workOrderOUT : WorkOrder_AssembleTopLevel[1]
partOUT : Coupler[1]
workOrderIN : WorkOrder_AssembleSensorCard[1]
assembleSensorCard : AssembleSensorCard
workOrderOUT : WorkOrder_AssembleSensorCard[1]
partOUT : SensorCard[1]
workOrderIN : WorkOrder_AssembleCircuitCard[1]
assembleCircuitCard : AssembleCircuitCard
workOrderOUT : WorkOrder_AssembleCircuitCard[1]
partOUT[1]
workOrderIN : WorkOrder_AssembleCableType1[1]
assembleCableType1 : AssembleCableType1
workOrderOUT : WorkOrder_AssembleCableType1[1]
partOUT : CableAssembly1[1]
workOrderIN : WorkOrder_AssembleDaughterCard[1]
assembleDaughterCard : AssembleDaughterCard
workOrderOUT : WorkOrder_AssembleDaughterCard[1]
partOUT : DaughterCard[1]
workOrderIN : WorkOrder_ProduceControlCard[1]
produceControlCard : ProduceControlCard
workOrderOUT : WorkOrder_ProduceControlCard[1]
partOUT : ProgrammedAssembly[1]
workOrderIN : WorkOrder_ProduceChassisElecEquip[1]
produceChassisElecEquip : ProduceChassisElecEquip
workOrderOUT : WorkOrder_ProduceChassisElecEquip[1]
partOUT : ChassisElectricalEquipment[1]
workOrderIN : WorkOrder_AssembleElecComponents[1]
assembleElecComponents : AssembleElecComponents
workOrderOUT : WorkOrder_AssembleElecComponents[1]
partOUT : ElectronicAssembly[1]
workOrderIN : WorkOrder_AssembleCableType2[1]
assembleCableType2_Role1 : AssembleCableType2
workOrderOUT : WorkOrder_AssembleCableType2[1]
partOUT : CableAssembly3[1]
workOrderIN : WorkOrder_AssembleCableType2[1]
assembleCableType2_Role2 : AssembleCableType2
workOrderOUT : WorkOrder_AssembleCableType2[1]
partOUT : CableAssembly3[1]

PROCESS

- *Top-Level Process and its I/O*: Black-box definition of the top-level manufacturing transformation
- *Input/Output*: Parts already have type; Work Orders need types too
- *Refinement*: Add a lower-level process
- *Exclusions from the Process Model*
- *Refinement*: To a leaf-level
- *Which Process is being Requested*: Make versus Deliver
- *Complement Type with State*: For processes' part I/O
- *Attach Data*: To both processes and work orders
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Process models can be big, just like Product models

Exclusions from the process model so far developed, whether intentional or pending, include:

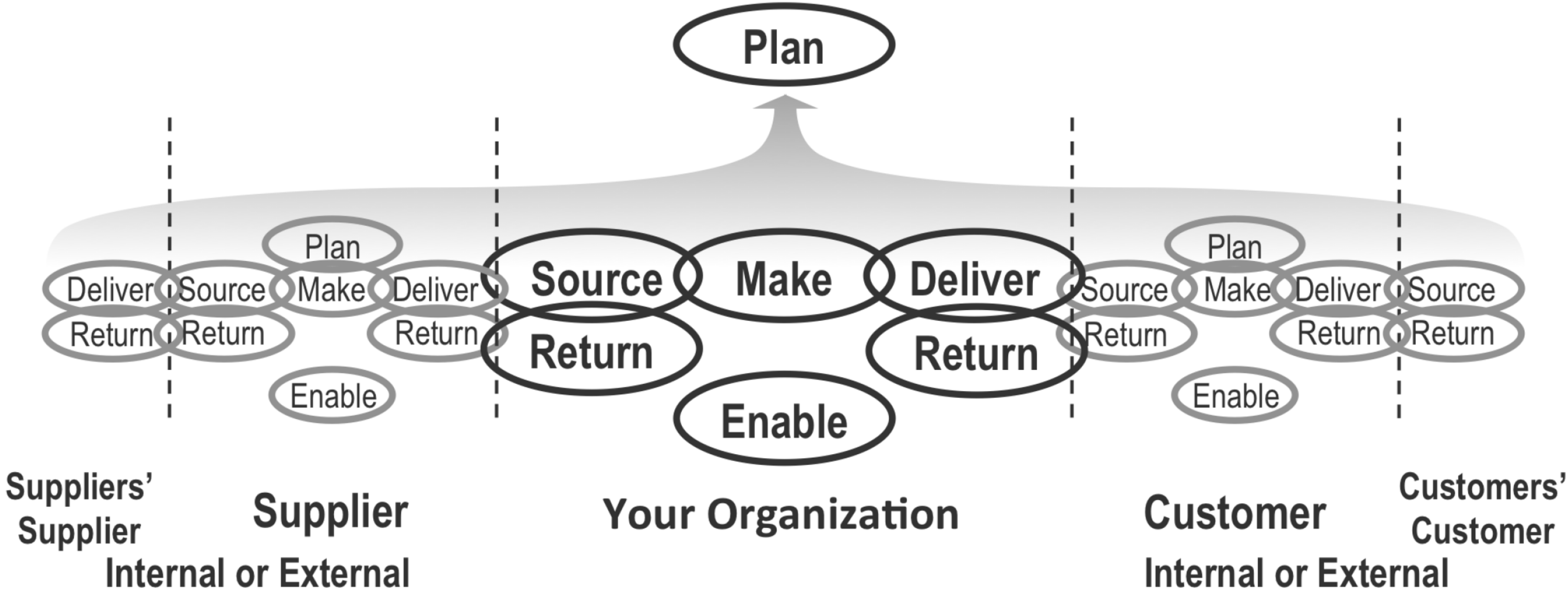
- I/O of passive resources may include more than just parts, for example fixtures too.
- To be precise, I/O of parts may need to specify both type and state.
- Controls for the flow of Work Orders (e.g. Operational Control)
- Controls for the flow of Resources (e.g. Material and Resource Handling)
- Contingencies. Process models so far say nothing about faults, exceptions, failures, or things going wrong.

PROCESS

- *Top-Level Process and its I/O*: Black-box definition of the top-level manufacturing transformation
- *Input/Output*: Parts already have type; Work Orders need types too
- *Refinement*: Add a lower-level process
- *Exclusions from the Process Model*
- *Refinement*: To a leaf-level
- *Which Process is being Requested*: Make versus Deliver
- *Complement Type with State*: For processes' part I/O
- *Attach Data*: To both processes and work orders
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Process models can be big, just like Product models

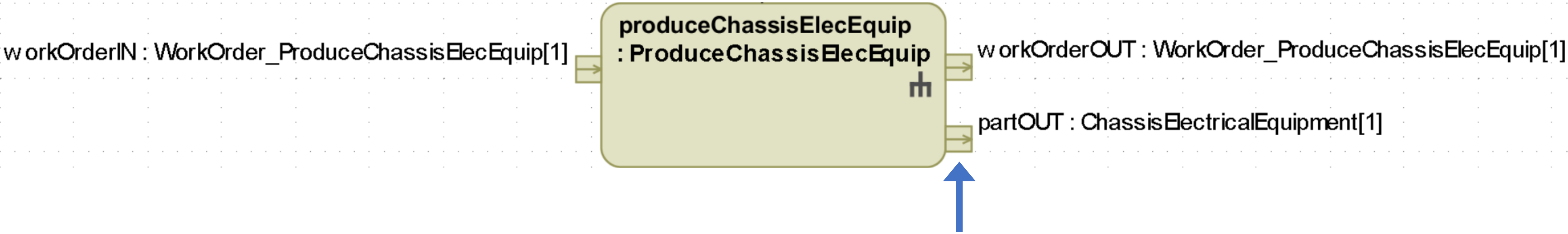
PROCESS

- *Top-Level Process and its I/O*: Black-box definition of the top-level manufacturing transformation
- *Input/Output*: Parts already have type; Work Orders need types too
- *Refinement*: Add a lower-level process
- *Exclusions from the Process Model*
- *Refinement*: To a leaf-level
- *Which Process is being Requested*: Make versus Deliver
- *Complement Type with State*: For processes' part I/O
- *Attach Data*: To both processes and work orders
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Process models can be big, just like Product models


Plan
Deliver
Return
Plan
Source
Make
Deliver
Return
Return
Enable
Source
Make
Deliver
Return
Return
Enable
Plan
Source
Make
Deliver
Return
Return
Enable
Source
Return
Suppliers' Supplier
Supplier
Internal or External
Your Organization
Customer
Internal or External
Customers' Customer

PROCESS

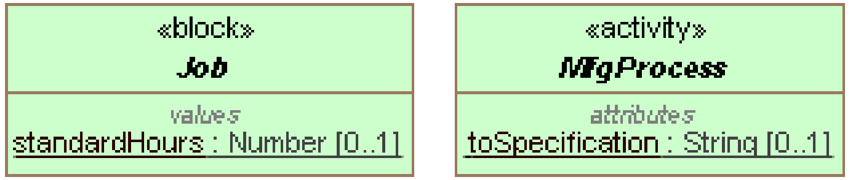
- *Top-Level Process and its I/O*: Black-box definition of the top-level manufacturing transformation
- *Input/Output*: Parts already have type; Work Orders need types too
- *Refinement*: Add a lower-level process
- *Exclusions from the Process Model*
- *Refinement*: To a leaf-level
- *Which Process is being Requested*: Make versus Deliver
- *Complement Type with State*: For processes' part I/O
- *Attach Data*: To both processes and work orders
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Process models can be big, just like Product models



In SysML, pins have an optional “InState” property. This enables specification of not just the output type, but also state – such as a manufacturing specification, a physical property (temperature), an orientation, etc.

PROCESS

- *Top-Level Process and its I/O*: Black-box definition of the top-level manufacturing transformation
- *Input/Output*: Parts already have type; Work Orders need types too
- *Refinement*: Add a lower-level process
- *Exclusions from the Process Model*
- *Refinement*: To a leaf-level
- *Which Process is being Requested*: Make versus Deliver
- *Complement Type with State*: For processes' part I/O
- *Attach Data*: To both processes and work orders
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Process models can be big, just like Product models


«block»
Job
values
standardHours : Number [0..1]
«activity»
MfgProcess
attributes
toSpecification : String [0..1]

PROCESS

- *Top-Level Process and its I/O*: Black-box definition of the top-level manufacturing transformation
- *Input/Output*: Parts already have type; Work Orders need types too
- *Refinement*: Add a lower-level process
- *Exclusions from the Process Model*
- *Refinement*: To a leaf-level
- *Which Process is being Requested*: Make versus Deliver
- *Complement Type with State*: For processes' part I/O
- *Attach Data*: To both processes and work orders
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Process models can be big, just like Product models

PROCESS

- *Top-Level Process and its I/O*: Black-box definition of the top-level manufacturing transformation
- *Input/Output*: Parts already have type; Work Orders need types too
- *Refinement*: Add a lower-level process
- *Exclusions from the Process Model*
- *Refinement*: To a leaf-level
- *Which Process is being Requested*: Make versus Deliver
- *Complement Type with State*: For processes' part I/O
- *Attach Data*: To both processes and work orders
- *Abstraction*: Connect to model libraries using generalization relationships
- *Scalability*: Process models can be big, just like Product models

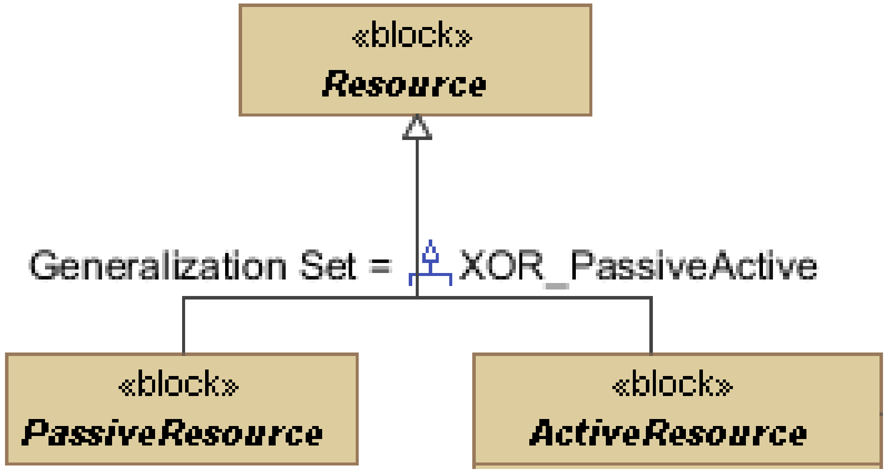
ACTIVE RESOURCE

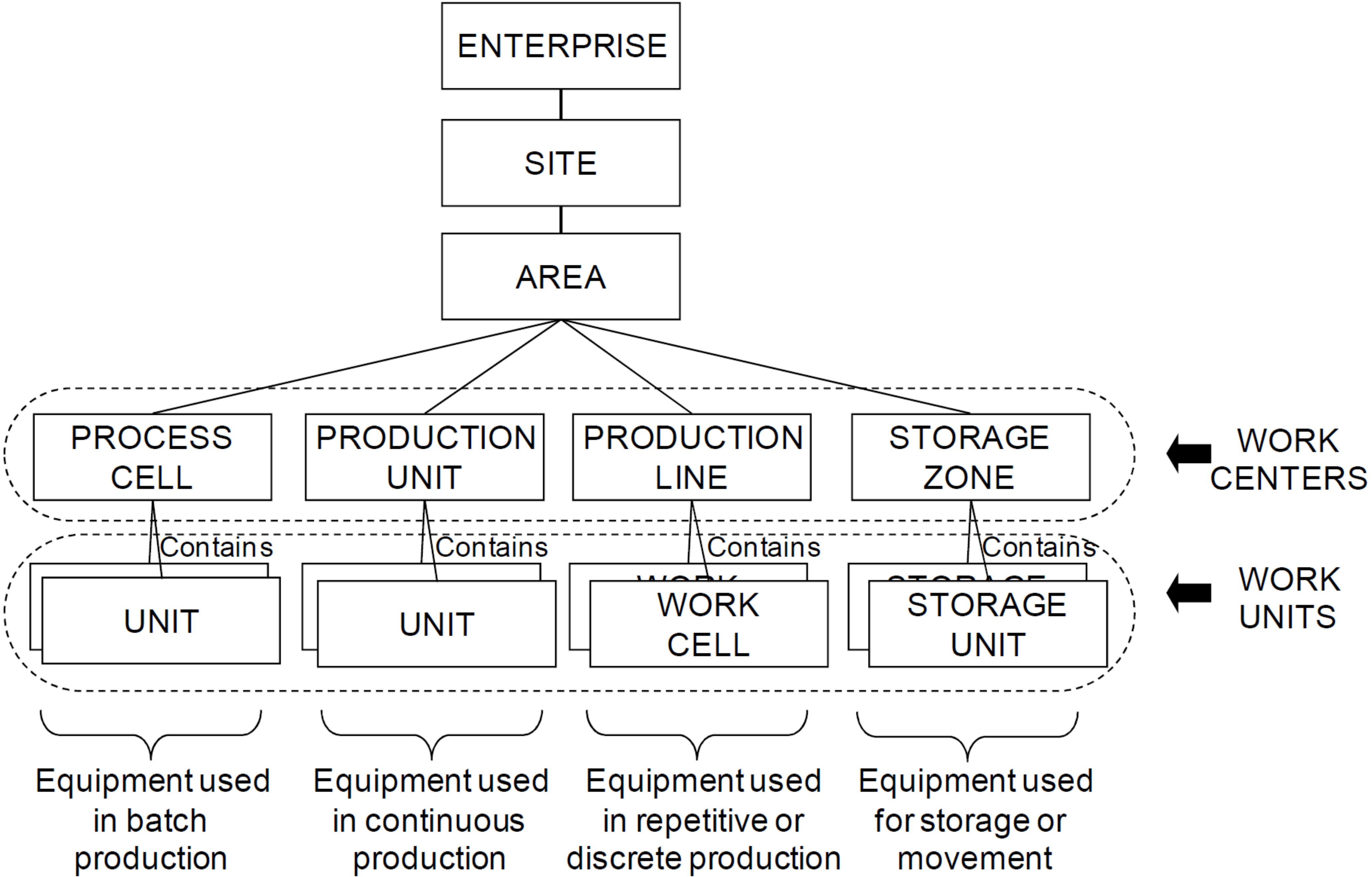
- *Define*: Active Resources.
- *Identify*: Active resources and their composition.
- *Capability*: Identify processes that active resources are capable of executing.
- *Capacity*: Modeling active resources’ capacity for process execution.
- *Performance*: Modeling active resources’ performance in process execution.

ACTIVE RESOURCE

- *Define*: Active Resources.
- *Identify*: Active resources and their composition.
- *Capability*: Identify processes that active resources are capable of executing.
- *Capacity*: Modeling active resources’ capacity for process execution.
- *Performance*: Modeling active resources’ performance in process execution.

An **Active Resource**’s defining characteristic is an ability to execute processes.

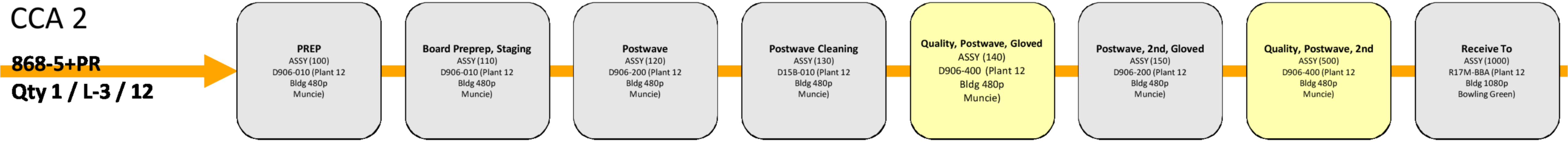



ENTERPRISE
SITE
AREA
PROCESS CELL
PRODUCTION UNIT
PRODUCTION LINE
STORAGE ZONE
WORK CENTERS
Contains
Contains
Contains
Contains
UNIT
UNIT
WORK CELL
STORAGE UNIT
WORK UNITS
Equipment used in batch production
Equipment used in continuous production
Equipment used in repetitive or discrete production
Equipment used for storage or movement

ACTIVE RESOURCE

- *Define*: Active Resources.
- *Identify*: Active resources and their composition.
- *Capability*: Identify processes that active resources are capable of executing.
- *Capacity*: Modeling active resources’ capacity for process execution.
- *Performance*: Modeling active resources’ performance in process execution.

CCA 2


868-5+PR
Qty 1 / L-3 / 12
PREP
ASSY (100)
D906-010 (Plant 12
Bldg 480p
Muncie)
Board Preprep, Staging
ASSY (110)
D906-010 (Plant 12
Bldg 480p
Muncie)
Postwave
ASSY (120)
D906-200 (Plant 12
Bldg 480p
Muncie)
Postwave Cleaning
ASSY (130)
D15B-010 (Plant 12
Bldg 480p
Muncie)
Quality, Postwave, Gloved
ASSY (140)
D906-400 (Plant 12
Bldg 480p
Muncie)
Postwave, 2nd, Gloved
ASSY (150)
D906-200 (Plant 12
Bldg 480p
Muncie)
Quality, Postwave, 2nd
ASSY (500)
D906-400 (Plant 12
Bldg 480p
Muncie)
Receive To
ASSY (1000)
R17M-BBA (Plant 12
Bldg 1080p
Bowling Green)

ACTIVE RESOURCE

- *Define*: Active Resources.
- *Identify*: Active resources and their composition.
- *Capability*: Identify processes that active resources are capable of executing.
- *Capacity*: Modeling active resources’ capacity for process execution.
- *Performance*: Modeling active resources’ performance in process execution.

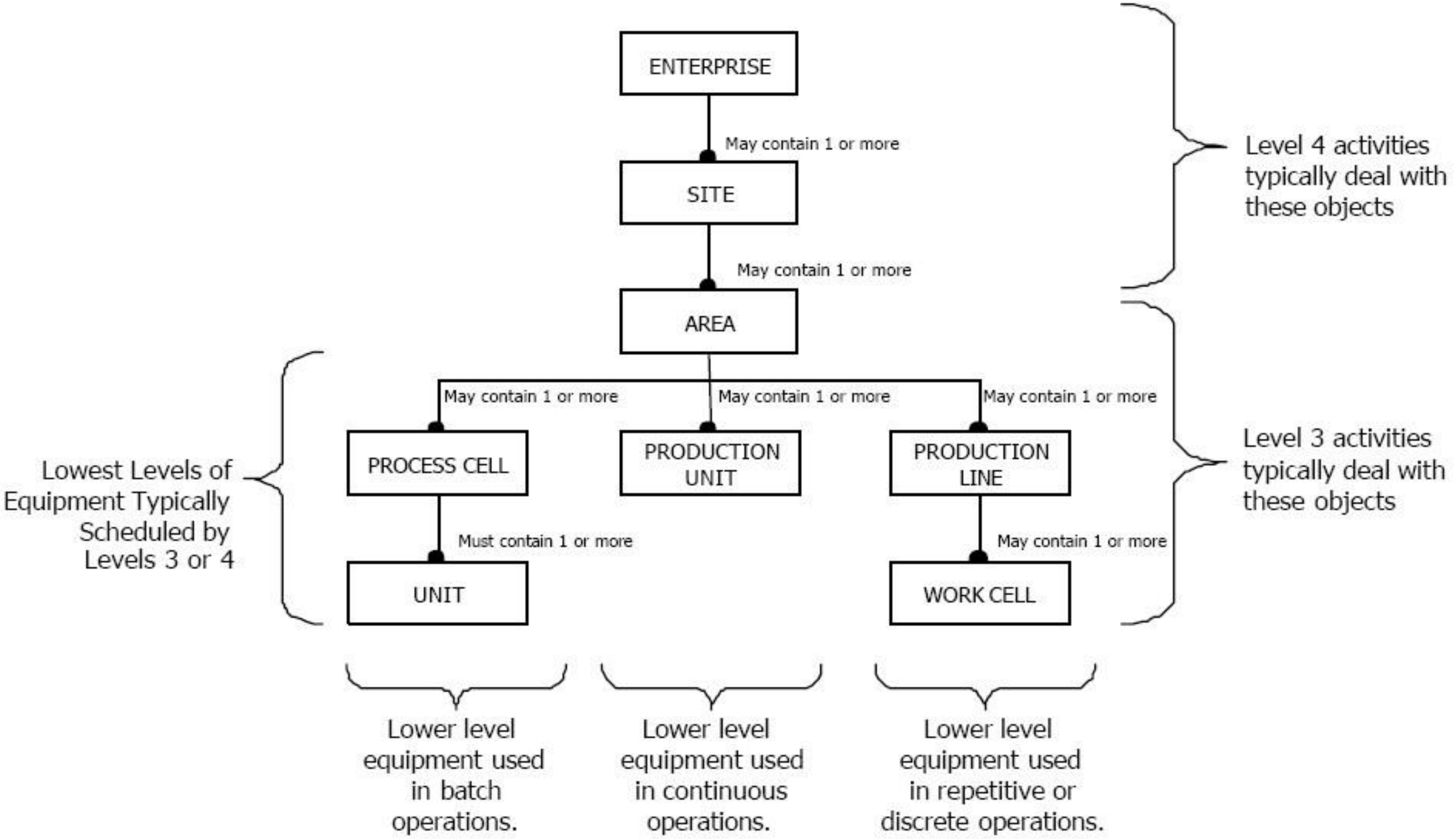
OPERATIONAL CONTROL

- *Define*: PERA / ISA-95 / B2MML “Levels” of Enterprise Control
- *Getting Started*: Define Controllers for Active Resources.
- *Define*: Level 3 Functions
- *Refinement*: Model each controller’s level 3 functionality.

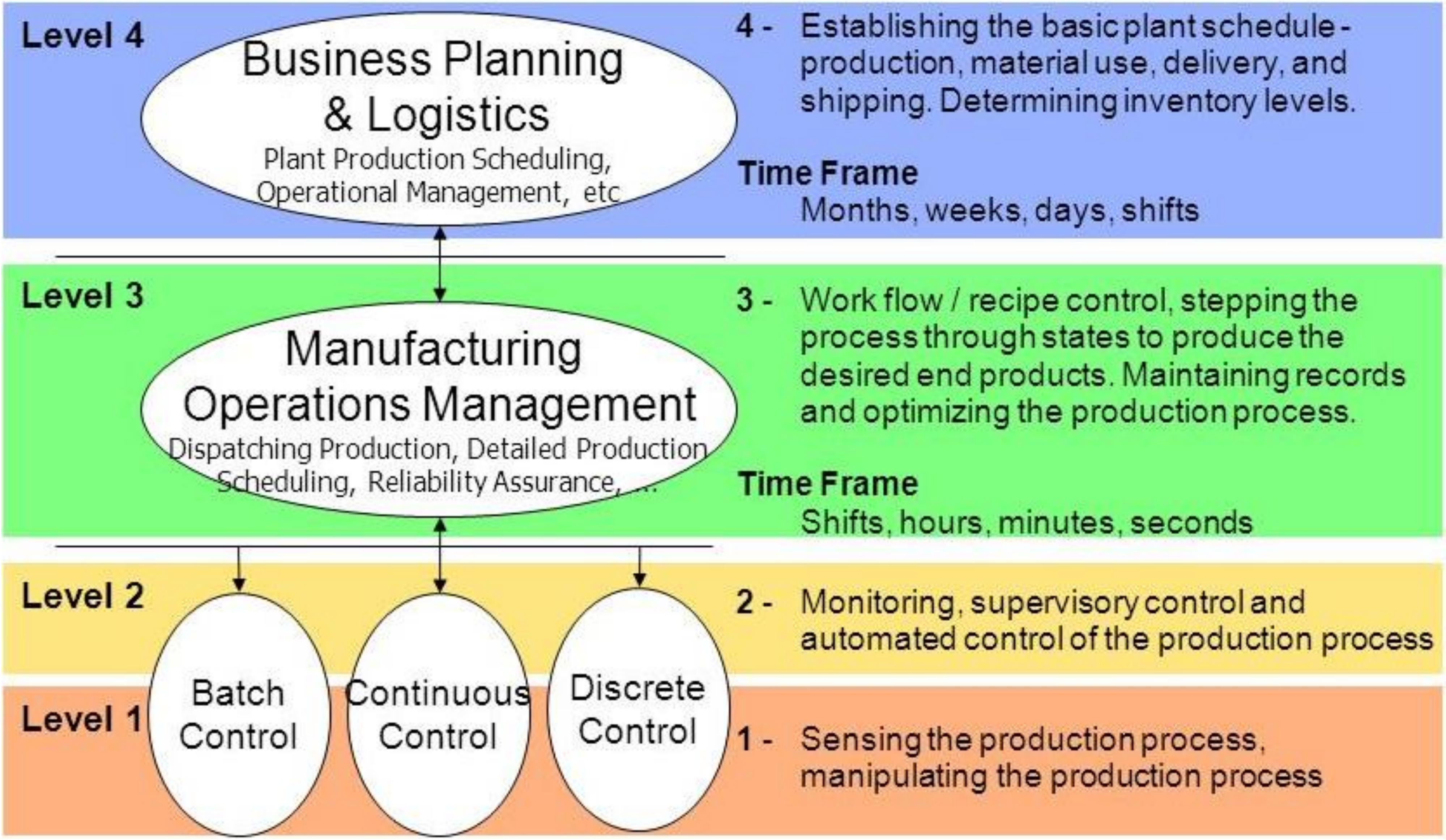
OPERATIONAL CONTROL

- *Define*: PERA / ISA-95 / B2MML “Levels” of Enterprise Control
- *Getting Started*: Define Controllers for Active Resources.
- *Define*: Level 3 Functions
- *Refinement*: Model each controller’s level 3 functionality.

Define: PERA / ISA-95 / B2MML "Levels" of Enterprise Control

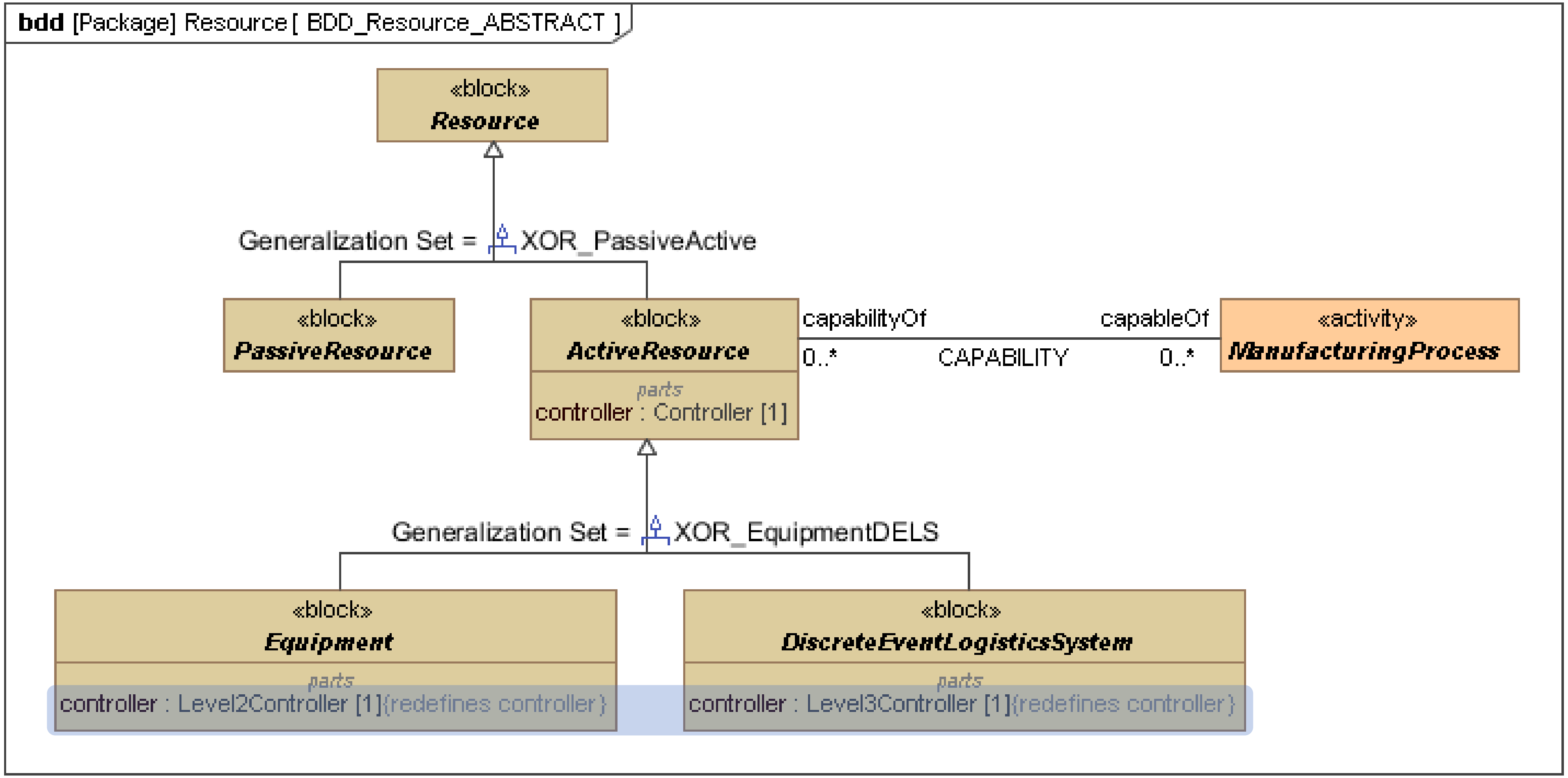


Define: PERA / ISA-95 / B2MML “Levels” of Enterprise Control



OPERATIONAL CONTROL

- *Define*: PERA / ISA-95 / B2MML “Levels” of Enterprise Control
- *Getting Started*: Define Controllers for Active Resources.
- *Define*: Level 3 Functions
- *Refinement*: Model each controller’s level 3 functionality.



bdd [Package] Resource [BDD_Resource_ABSTRACT]
«block»
Resource
Generalization Set = XOR_PassiveActive
«block»
PassiveResource
«block»
ActiveResource
parts
controller : Controller [1]
capabilityOf
0..*
CAPABILITY
capableOf
0..*
«activity»
ManufacturingProcess
Generalization Set = XOR_EquipmentDELS
«block»
Equipment
parts
controller : Level2Controller [1]{redefines controller}
«block»
DiscreteEventLogisticsSystem
parts
controller : Level3Controller [1]{redefines controller}

OPERATIONAL CONTROL

- *Define*: PERA / ISA-95 / B2MML “Levels” of Enterprise Control
- *Getting Started*: Define Controllers for Active Resources.
- *Define*: Level 3 Functions
- *Refinement*: Model each controller’s level 3 functionality.

Define: Level 3 Functions

ATOMIC functions: (to "fulfill" a job is to execute its requested process)

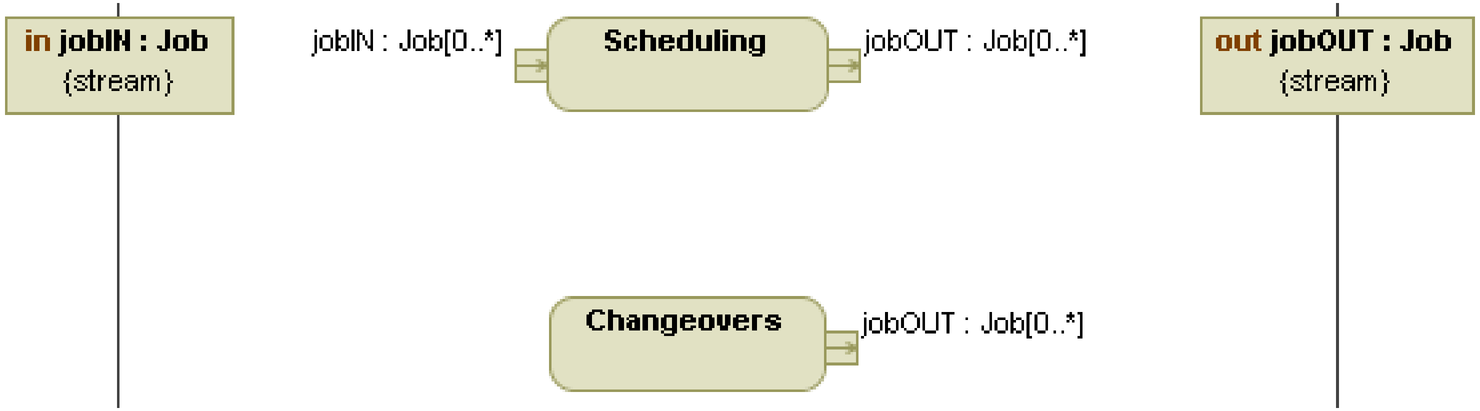
- *Admission* - Which jobs to fulfill?
- *Sequencing* - When, or in what order, is an admitted job fulfilled?
- *Assignment* - Which resource is assigned to fulfill a job?
- *Dynamic Process Planning* - Which process step does job fulfillment require next?
- *Changing State* - Which state should a resource be in?

COMPOUND functions:

- *Scheduling* – A combination of *sequencing* and *assignment*
- *Routing* – A combination of *assignment* and *dynamic process planning*

OPERATIONAL CONTROL

- *Define*: PERA / ISA-95 / B2MML “Levels” of Enterprise Control
- *Getting Started*: Define Controllers for Active Resources.
- *Define*: Level 3 Functions
- *Refinement*: Model each controller’s level 3 functionality.



Each of the call actions is a behavior, not just an algorithm. However, if decision-making logic is all that's of initial interest, start there. The called behavior could be opaque, for example to specify a well-known rule such as "FIFO" for sequencing. The called behavior could be a state machine. The called behavior could be another activity, modeling both an algorithm and how decisions are actuated.

Agenda

- Overview
- Value Proposition
- 2018 Work Items Status Update
  - Theory of DELS Specification
  - Model-based Industrial and Systems Engineering Playbook
- Case Studies
  - **Central Fill Pharmacy Models – Leon McGinnis, Georgia Tech**
  - Value Stream Mapping for Production – George Thiers, MBSE Tools
- Roadmap:
  - Document existing models and make them available
  - Identify and Document Use Cases, Refine Value Proposition
  - Identify Additional Case Studies
  - Identify Potential Liaisons

Model-Based Systems Engineering
for High Volume Central Fill Pharmacies

Leon F. McGinnis

Professor Emeritus

School of Industrial and Systems Engineering

The Georgia Institute of Technology

January 24, 2019

Rev 04

Acknowledgements: This case is based on research conducted in the W. M. Keck Virtual Factory Lab at Georgia Tech and supported by the National Institutes of Science and Technology and by McKesson High Value Solutions. It has benefited from the participation of many individual researchers, particularly Dr. Tim Sprock, Dr. George Thiers, Dr. Doug Bodner, Camillo Bernes, Francisco xxx, and Di Liu.

Intended audience: general; non-technical description of CFP; SysML-based analysis-agnostic system model; decision-support analyses referencing the system model.

Download most recent version from http://leonmcginnis.com/dels-case-studies/

high speed dispensing technologies require considerable integration of all the individual resources and the puck conveyor, but can be very effective for dispensing drugs for which there is a high demand rate.

A high flexibility resource operates quite differently. It is essentially a robotic workstation, which may have as many as 200 or more canisters, or pill types. Labeled and tare weighted vials may be delivered to the workstation via pucks and the vials removed from the pucks by the robot. Alternatively, the workstation may have its own capability to dispense, label and tare weigh vials. Figure 3 shows a robot holding a vial under a dispensing canister. For high flexibility workstations with vial dispensing capability, the filled vials are dropped into totes moved on a tote conveyor. There can be multiple high-flexibility workstations, as well as manual fill stations integrated via the tote conveyor. This technology can be effective for drugs that are ordered often enough to keep the robot

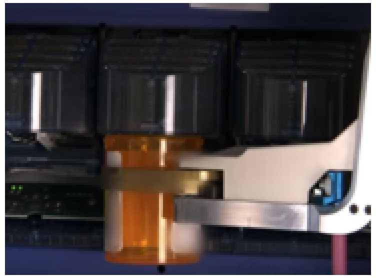


Figure 3 Robotic Workstation

https://www.youtube.com/watch?v=cBUig

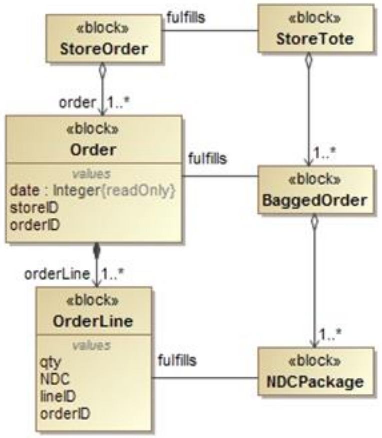


Figure 1 CFP "Product"

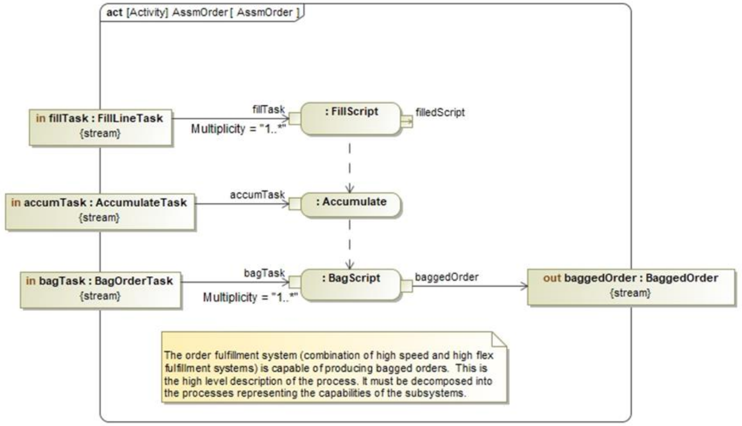


Figure 1 Customer Order Assembly Process

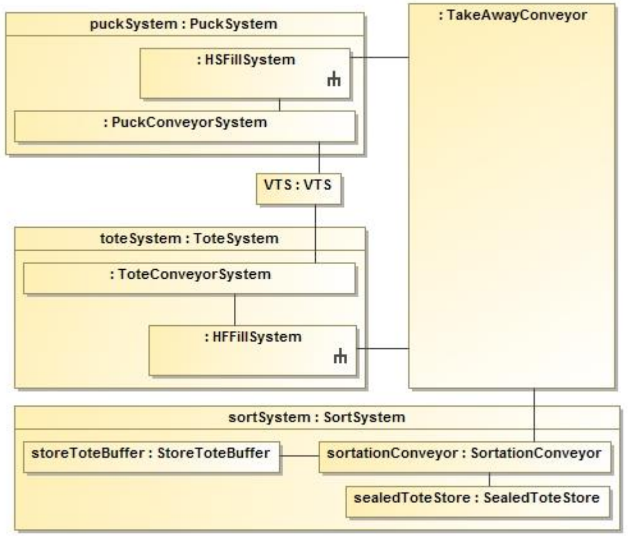


Figure 1 Hypothetical HVCFP

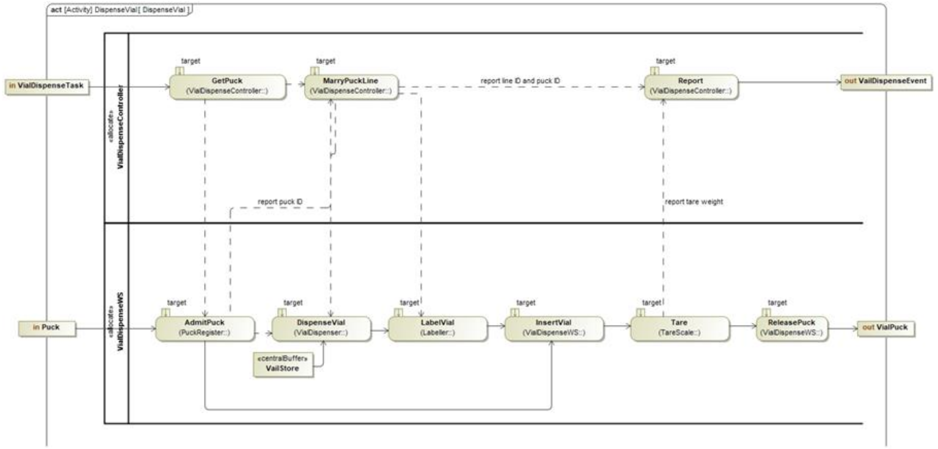


Figure 2 Behavior of the Vial Dispense System Controller and Vial Dispense Workstation

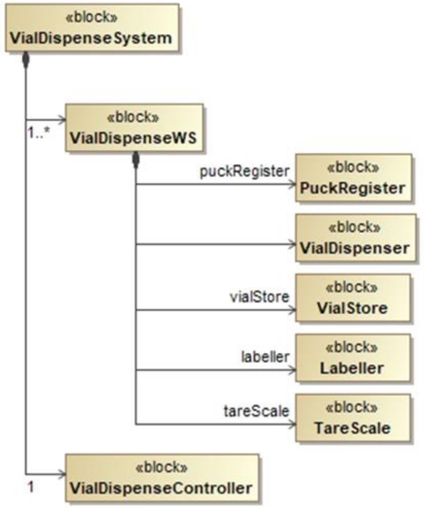


Figure 1 Vial Dispense System Structure

Additional Case Studies

- Semiconductor manufacturing (Intel Mini-Fab case)
- Composite wing production (open source)

Agenda

- Overview
- Value Proposition
- 2018 Work Items Status Update
  - Theory of DELS Specification
  - Model-based Industrial and Systems Engineering Playbook
- Case Studies
  - Central Fill Pharmacy Models – Leon McGinnis, Georgia Tech
  - **Value Stream Mapping for Production – George Thiers, MBSE Tools**
- Roadmap:
  - Document existing models and make them available
  - Identify and Document Use Cases, Refine Value Proposition
  - Identify Additional Case Studies
  - Identify Potential Liaisons

- Insert Slides from MBSE Tools
  - Possibly related to SBIR Phase I report?

Agenda

- Overview
- Value Proposition
- 2018 Work Items Status Update
  - Theory of DELS Specification
  - Model-based Industrial and Systems Engineering Playbook
- Case Studies
  - Central Fill Pharmacy Models – Leon McGinnis, Georgia Tech
  - Value Stream Mapping for Production – George Thiers, MBSE Tools
- **Roadmap:**
  - Document existing models and make them available
  - Identify and Document Use Cases, Refine Value Proposition
  - Identify Additional Case Studies
  - Identify Potential Liaisons

Agenda

- Overview
- Value Proposition
- 2018 Work Items Status Update
  - Theory of DELS Specification
  - Model-based Industrial and Systems Engineering Playbook
- Case Studies
  - Central Fill Pharmacy Models – Leon McGinnis, Georgia Tech
  - Value Stream Mapping for Production – George Thiers, MBSE Tools
- **Roadmap:**
  - Document existing models and make them available
  - **Identify and Document Use Cases**, Refine Value Proposition
  - Identify Additional Case Studies
  - Identify Potential Liaisons

MBISE – Shop Floor Operations Use Case

- Describe
- Predict
- Control
- Design

- **System:** Small flexible job shop or flow shop; 2-5ish kinds of machines; robots, AGVs, or conveyors for MH; storage solution
- **Describe:** Conceptual Model (PPRF) vs Engineering Model (interfaces & protocols)
  - How do we build models? use the model libraries? When is the model done/complete?
- **Describe**: As-is control – MES, flow rules, assignment rules, SCADA/PLC (if necessary)
  - **(re-)Design**: If I want to make the system flow better, where/how do I make changes?
- **Describe**: Sensors & Data Acquisition – what data do/can we collect from the shop floor?
  - **Design**: Where to add sensors? (IOT)
- **Predict**: Shop floor simulation generation – progress on closing “fidelity gap”
  - **(re-)Design**: If I want to make the system flow better, what will the impact be of any changes I make?
- **Control**: Scheduling – what information is available
  - Information: heterogeneous sources, inconsistent formats, fidelity, aggregation

Roadmap - Identify a Case Study

- Include all SysML diagrams and syntax
- Domain-specific concepts:
  – Product, Process, Resource, & Facility
  – How do you control your system?
  – What do you want to know about the system?
  – System Architecture

Agenda

- Overview
- Value Proposition
- 2018 Work Items Status Update
  - Theory of DELS Specification
  - Model-based Industrial and Systems Engineering Playbook
- Case Studies
  - Central Fill Pharmacy Models – Leon McGinnis, Georgia Tech
  - Value Stream Mapping for Production – George Thiers, MBSE Tools
- **Roadmap:**
  - Document existing models and make them available
  - Identify and Document Use Cases, **Refine Value Proposition**
  - Identify Additional Case Studies
  - Identify Potential Liaisons

Discussion: Value Proposition

- How would you apply MBISE?
- What would you want to do with it?

Lifecycle		Concept, Early-Stage Design	Late-Stage Design, Build	Commission	Operation & Maintenance
What You Know	*Product*	Partial EBOM	EBOM, partial MBOM	EBOM, MBOM	EBOM, MBOM, with engineering changes
	Process	Make	Make, Measure, Test, partial Move & Store	Make, Measure, Test, Move, Store	Make, Measure, Test, Move, Store, Control
	Resource	Work Unit: Capability	Work Unit, partial Work Center: Capability, partial Capacity (available resource-hours per hour/shift/day)	Work Unit, Work Center, partial Area: Capability, Capacity, partial Performance	Work Unit, Work Center, Area: Capability, Capacity, Performance
	Facility	n/a	Location, partial Channel	Location, Channel	Location, Channel, Geometry
	Control	n/a	Admission, partial Sequencing (Prioritization of orders? Is expediting allowed? Are changeovers allowed?), partial Resource Assignment (Job shop or dedicated lines?)	Admission, Sequencing, Resource Assignment, partial *Scheduling* (Make to engineer, order, or stock? Push or pull?), partial Resource State Changes, partial Dynamic Process Planning (Is material handling scheduled or requested? Prioritization of requests? Is storage allowed?)	Admission, Sequencing, Resource Assignment, *Scheduling*, Resource State Changes, Dynamic Process Planning
What You Can Do	*Describe*	**(Product)** Does every part have a part number? A make/buy decision? A process plan if make? DFMA analyses? **(Process)** Does every make process have a make-to specification? A resource capable of its execution? **(Resource)** Are all requirements concerning capability, capacity, and performance allocated to resources?	**(Product)** Same, with a richer set of parts. **(Process)** Same, with a richer set of processes, plus: Gross execution capacity per process? With standard hours estimates, max execution rate per process? **(Resource)** Downtime causes per resource? Changeover time estimates? Material movement requirements per part? Channel requirements between resources? **(Facility)** Sizing requirements for Work Units & Work Centers? Storage constraints?	**(Product)** Same, with a richer set of parts. **(Process)** Same, with a richer set of processes, plus: Max operational cost per process? Gross execution capacity & max rate per logistical process? Contingency-triggered alternatives? **(Resource)** Downtime costs per resource? Changeover costs? Max material handling rate per channel? **(Facility)** Sizing requirements for per channel? Per storage buffer? Per Area? **(Control)** TH, CT, WIP, critical path, emerging bottlenecks?	*NEED HELP HERE; biggest change is that operational data is available.* **(Product)** Quality? **(Process)** Process alternatives upon contingencies? Waste? **(Resource)** Utilization, downtime, and changeover data. Material handling data. **(Facility)** Geometry-related. Channel congestion? Storage overflows? **(Control)** TH, CT, WIP, On-time deliveries, (see SCOR for more metrics). Per-job statistics.
	Predict	Lower & upper bounds on expected TH, CT, WIP, with fixed resources?	Refined lower & upper bounds on expected TH, CT, WIP, with fixed resources? Expected critical path? Potential bottlenecks?	Expected TH, CT, WIP? Expected critical path? Potential bottlenecks? Expected schedule delays or fractions of travelled work, per process?	Worst-case, expected, and best-case TH, CT, WIP, bottlenecks, on-time deliveries, schedule delays or fractions of travelled work for alternatives and scenarios?
	Prescribe	Lower & upper bounds on required resources, with fixed TH, CT, WIP requirements?	Refined lower & upper bounds on required resources, with fixed TH, CT, WIP requirements? Lower & upper bounds on material handling capacity? Projected storage buffers? Preliminary facility layout?	Expected resource requirements for make, measure, test processes? Expected resource requirements for move, store processes? Storage buffer capacities? Facility layout?	Adaptive redesigns: If a shortage of part type P, what should we do? If an outage of machine instance M, what should we do? Strategic redesigns, in response to changing external demand or internal technologies.

Agenda

- Overview
- Value Proposition
- 2018 Work Items Status Update
  - Theory of DELS Specification
  - Model-based Industrial and Systems Engineering Playbook
- Case Studies
  - Central Fill Pharmacy Models – Leon McGinnis, Georgia Tech
  - Value Stream Mapping for Production – George Thiers, MBSE Tools
- **Roadmap:**
  - Document existing models and make them available
  - Identify and Document Use Cases, Refine Value Proposition
  - **Identify Additional Case Studies**
  - Identify Potential Liaisons

Roadmap - Identify a Case Study

- "… advancing the practice and adoption of formal system modeling and model-based systems engineering methodologies in production and logistics systems development and operations."

- "Do you have any examples to get me started?"

- Sandy Friedenthal & Chris Oster – "Architecting Spacecraft with SysML: A Model-based Systems Engineering Approach"
  - http://sysml-models.com/spacecraft/index.html

Roadmap - Identify a Case Study

- Include all SysML diagrams and syntax
- Domain-specific concepts:
  - Product, Process, Resource, & Facility
  - How do you control your system?
  - What do you want to know about the system?
  - System Architecture

Roadmap - Liaisons

- ManTIS
- IISE
- Winter Simulation Conference
- SDOs (OMG, others?)
- Others?

Challenge team weekly meeting at 11 am (EST) Fridays.

For February, 2018, the meeting information is:

To join the Meeting:
https://bluejeans.com/406291803

To join via Room System:
Video Conferencing System: bjn.vc -or- 199.48.152.152
Meeting ID : 406291803

To join via phone :

1) Dial:
   +1.408.740.7256 (US (San Jose))
   +1.888.240.2560 (US Toll Free)
   +1.408.317.9253 (US (Primary, San Jose))
   (see all numbers - http://bluejeans.com/numbers)
2) Enter Conference ID : 406291803

Contact Us:

timothy.sprock@nist.gov
leon.mcginnis@isye.gatech.edu
conrad.bock@nist.gov

Links:

http://www.omgwiki.org/MBSE/doku.php?id=mbse:prodlog

https://github.com/usnistgov/DiscreteEventLogisticsSystems

2019

Annual INCOSE

international workshop

Torrance, CA, USA

January 26 - 29, 2019

www.incose.org/IW2019