

INTERFACE MANAGEMENT WITH MBSE – FROM THEORY TO MODELING TO REALITY

Matthew Hause

Engineering Fellow, MBSE Specialist

October, 2018



AGENDA

1. Introduction
2. Interfaces
3. System of System Interfaces
4. System Interfaces
5. Through the development lifecycle
6. Conclusion

INTRODUCTION



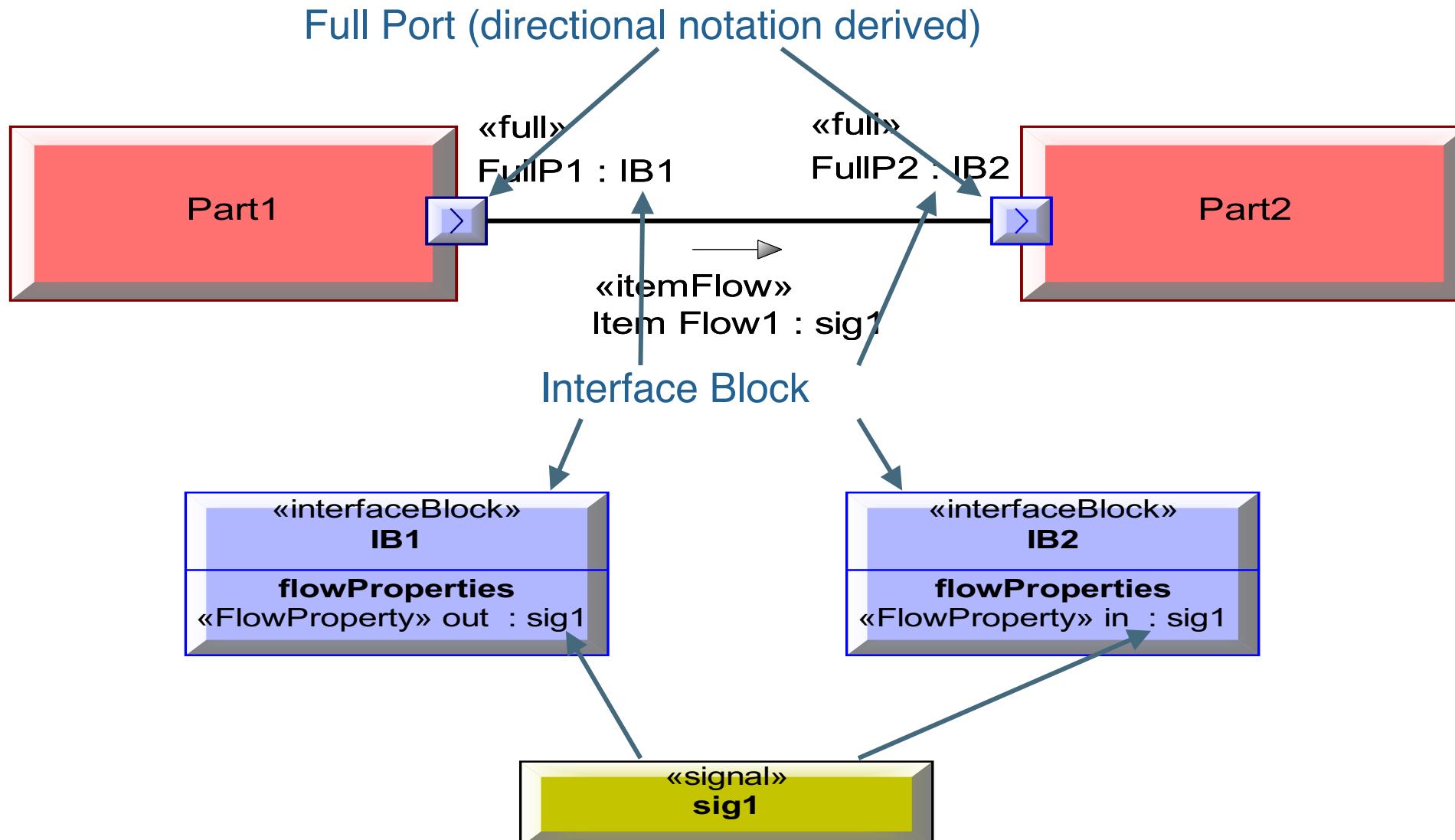
- Interoperability is a key facet of a successful system, and essential to a system of systems.
- Interoperability is a property of a system, whose interfaces are completely understood, to work with other products or systems without any restricted access or implementation.
- Software interoperability is the capability of different programs to exchange data via a common set of exchange formats, (read/write) file formats using same protocols.
- DOD: The condition achieved among communications-electronics systems when information or services can be exchanged directly and satisfactorily.
- So, interoperability begins with interfaces: mechanical, electronic, hardware, software, people-ware, etc.

DESIGNING INTERFACES



- Starts with requirements and stakeholder needs
- System-to-System interfaces
 - Define the required behavior/functionality
 - Identify the Dependencies - interaction with other systems and within the subsystems
 - Identify the necessary interactions
 - Data, physical, logical, electrical, etc.
 - Define logical interface requirements
 - Define interaction performance characteristics
 - Allocate to physical interfaces
- Human Interfaces
 - Identify the characteristics of the (Human) users that will interact with the system.
 - Define the required tasks to be performed
 - Identify the Primary User Interface Elements
 - Define the Navigation Map

FULL PORT NOTATION

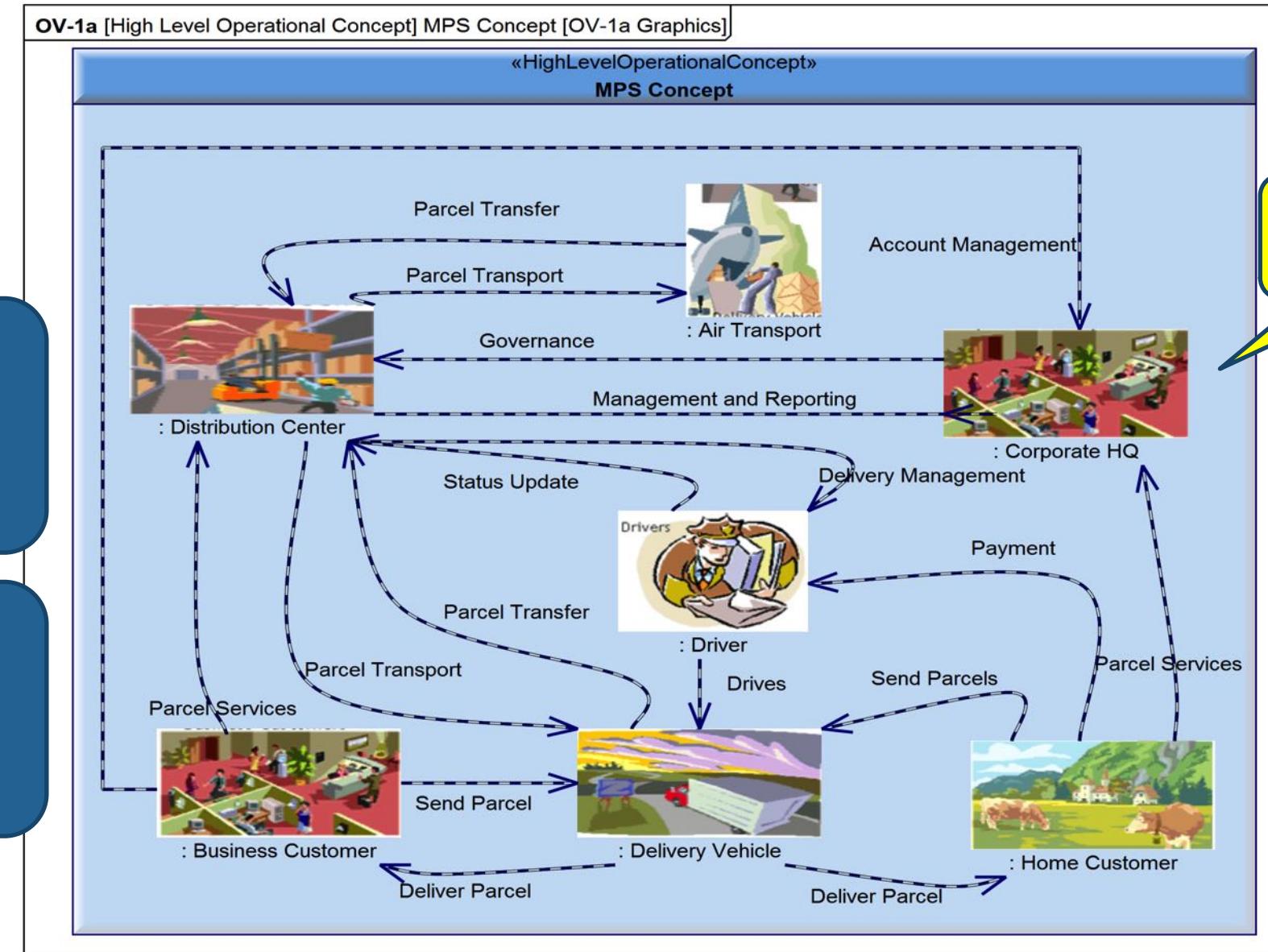


SYSTEMS OF SYSTEMS INTERFACES

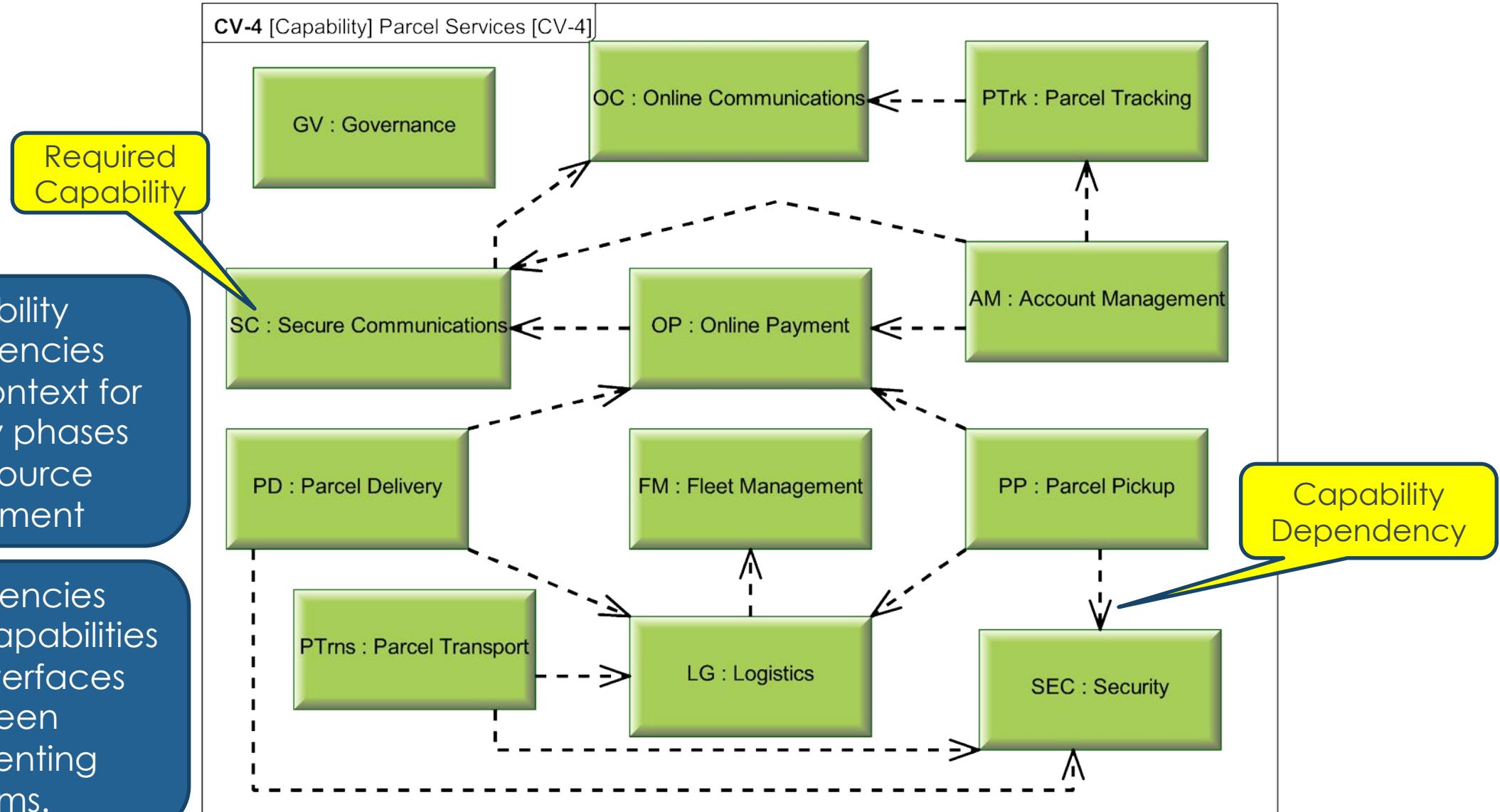
OPERATIONAL CONCEPT GRAPHIC

Provides a means to communicate with non-technical stakeholders while maintaining model consistency

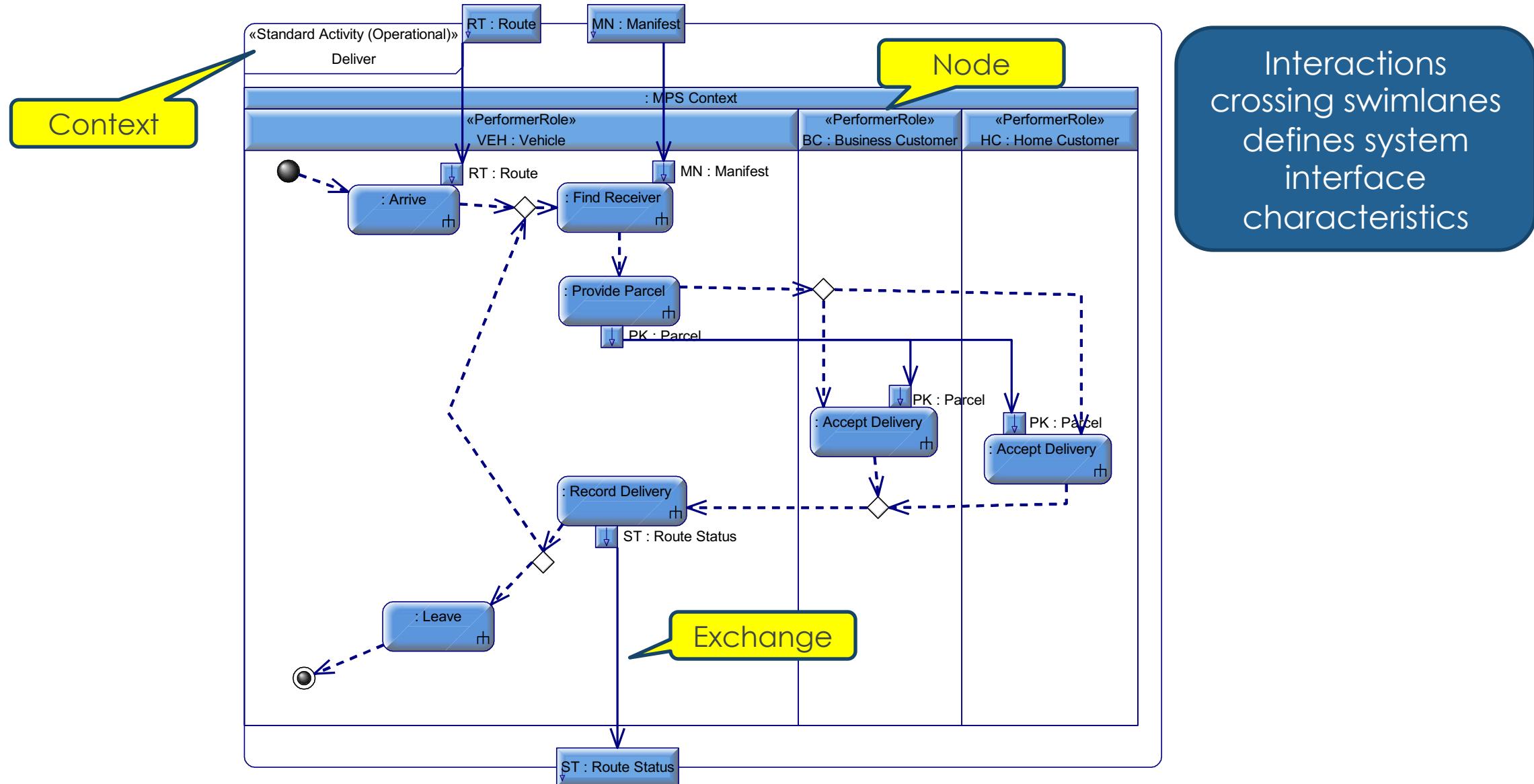
Defines nominal interfaces between conceptual entities in the context.



CAPABILITY DEPENDENCIES



LOGICAL ARCHITECTURE INTERACTIONS



Interactions crossing swimlanes defines system interface characteristics

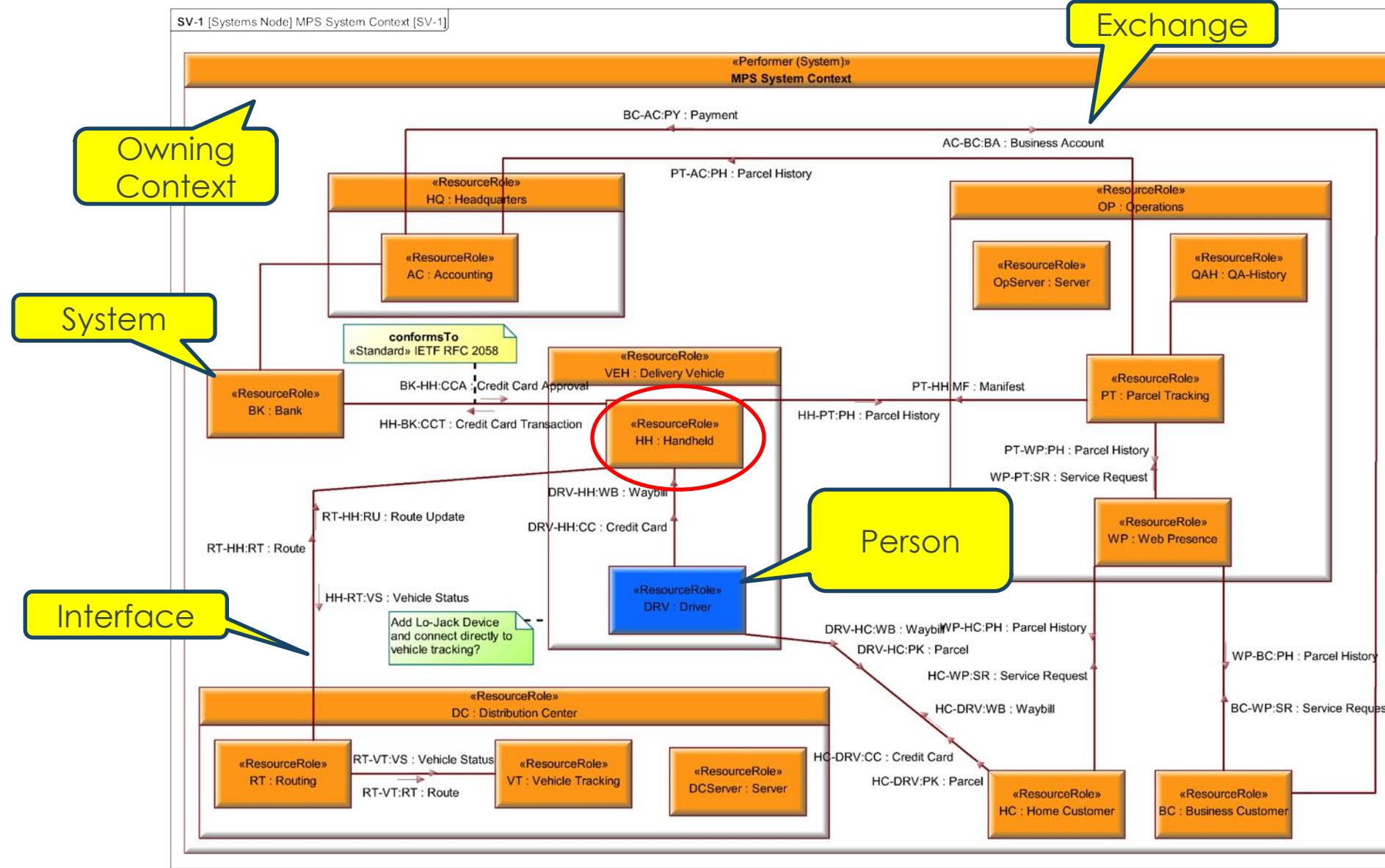
LOGICAL ARCHITECTURE ICD (FRAGMENT)

[Architectural Description] Structure [OV-3 Info Exchange]

Operational		Producer		Needline	Consumer	
Name	Conveyed	Performer (Operational)	Activity (Operational)	Name	Performer (Operational)	Activity (Operational)
CHQ-BC:BL	«Exchange Elements» Bill	«Performer (Operational)» Corporate HQ		BC - CHQ	«Performer (Operational)» Business Customer	
BC-VEH:PK	«System» Parcel	«Performer (Operational)» Business Customer	«Activity (Operational)» Provide Waybill	BC - VEH	«Performer (Operational)» Vehicle	«Activity (Operational)» Verify Waybill and Payment
BC-VEH:PW	«Exchange Elements» Parcel Waybill	«Performer (Operational)» Business Customer	«Activity (Operational)» Provide Waybill	BC - VEH	«Performer (Operational)» Vehicle	«Activity (Operational)» Verify Waybill and Payment
VEH-BC:PK	«System» Parcel	«Performer (Operational)» Vehicle	«Activity (Operational)» Provide Parcel	BC - VEH	«Performer (Operational)» Business Customer	«Activity (Operational)» Accept Delivery
SF-DC:PK	«System» Parcel	«Performer (Operational)» Storefront		SF - DC	«Performer (Operational)» Distribution Center	
DC-VEH:MN	«Exchange Elements» Manifest	«Performer (Operational)» Distribution Center	«Activity (Operational)» Find and Record Outgoing Parcels	VEH - DC	«Performer (Operational)» Vehicle	«Activity (Operational)» Load Vehicle «Activity (Operational)» Find Receiver «Activity (Operational)» Find Sender
DC-VEH:PK	«System» Parcel	«Performer (Operational)» Distribution Center	«Activity (Operational)» Find and Record Outgoing Parcels	VEH - DC	«Performer (Operational)» Vehicle	«Activity (Operational)» Load Vehicle
DC-VEH:PW	«Exchange Elements» Parcel Waybill	«Performer (Operational)» Distribution Center	«Activity (Operational)» Find and Record Outgoing Parcels	VEH - DC	«Performer (Operational)» Vehicle	«Activity (Operational)» Load Vehicle
DC-VEH:RT	«Exchange Elements» Route	«Performer (Operational)» Distribution Center		VEH - DC	«Performer (Operational)» Vehicle	«Activity (Operational)» Arrive
VEH-DC:MN	«Exchange Elements» Manifest	«Performer (Operational)» Vehicle	«Activity (Operational)» Unload Vehicle	VEH - DC	«Performer (Operational)» Distribution Center	«Activity (Operational)» Record and Store Incoming Parcels
VEH-DC:PK	«System» Parcel	«Performer (Operational)» Vehicle	«Activity (Operational)» Unload Vehicle	VEH - DC	«Performer (Operational)» Distribution Center	«Activity (Operational)» Record and Store Incoming Parcels
VEH-DC:PW	«Exchange Elements» Parcel Waybill	«Performer (Operational)» Vehicle	«Activity (Operational)» Unload Vehicle	VEH - DC	«Performer (Operational)» Distribution Center	«Activity (Operational)» Record and Store Incoming Parcels
VEH-DC:ST	«Exchange Elements» Route Status	«Performer (Operational)» Vehicle	«Activity (Operational)» Record Delivery «Activity (Operational)» Record Pickup	VEH - DC	«Performer (Operational)» Distribution Center	
HC-VEH:PK	«System» Parcel	«Performer (Operational)» Home Customer	«Activity (Operational)» Provide Waybill	VEH - HC	«Performer (Operational)» Vehicle	«Activity (Operational)» Verify Waybill and Payment
HC-VEH:PW	«Exchange Elements» Parcel Waybill	«Performer (Operational)» Home Customer	«Activity (Operational)» Provide Waybill	VEH - HC	«Performer (Operational)» Vehicle	«Activity (Operational)» Verify Waybill and Payment
HC-VEH:PY	«Exchange Elements» Payment	«Performer (Operational)» Home Customer	«Standard Activity (Operational)» Provide Payment	VEH - HC	«Performer (Operational)» Vehicle	«Activity (Operational)» Verify Waybill and Payment
VEH-HC:PK	«System» Parcel	«Performer (Operational)» Vehicle	«Activity (Operational)» Provide Parcel	VEH - HC	«Performer (Operational)» Home Customer	«Activity (Operational)» Accept Delivery

Generated automatically from the architecture

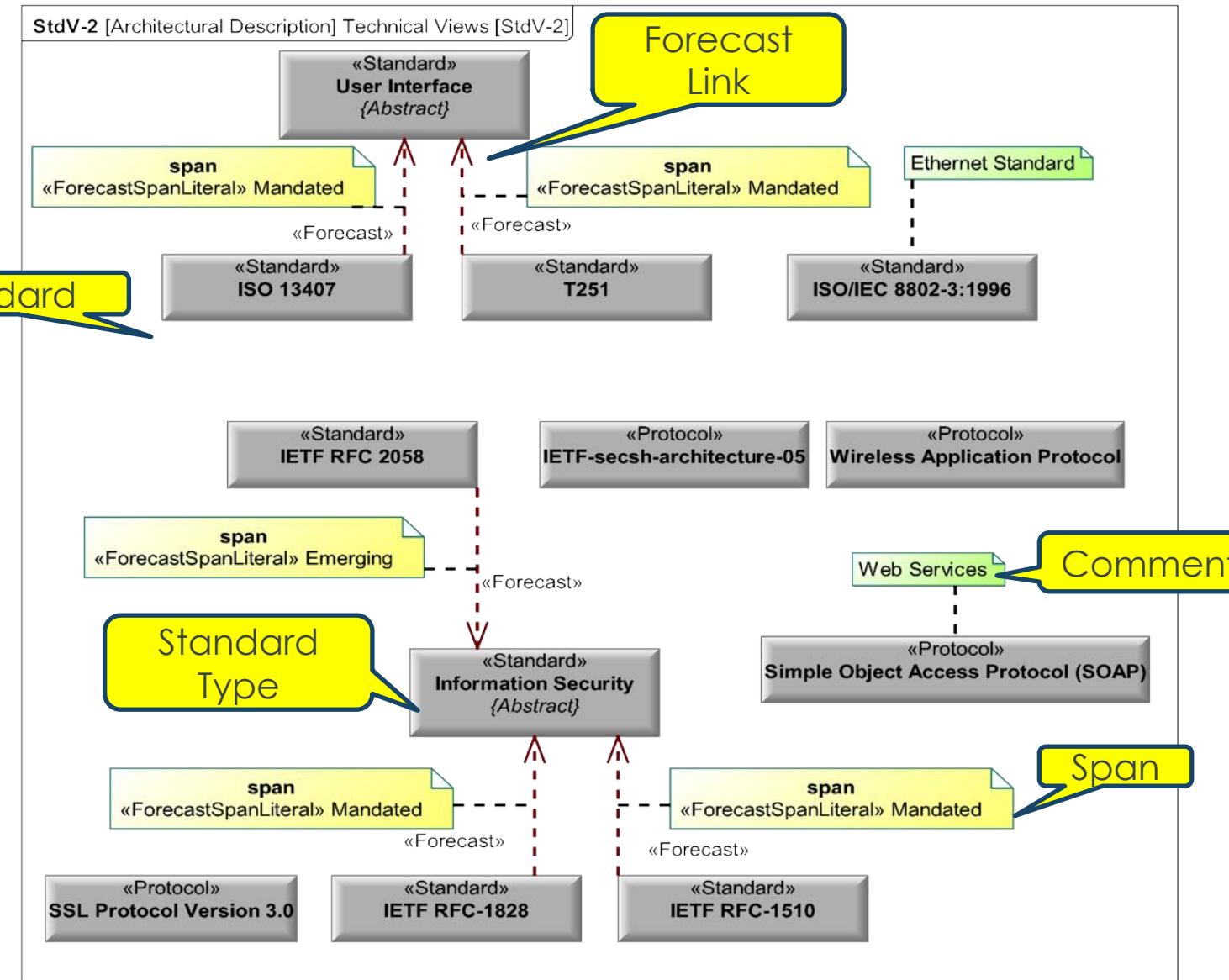
SYSTEM INTERCHANGE SPECIFICATION



Systems can also be specified as services

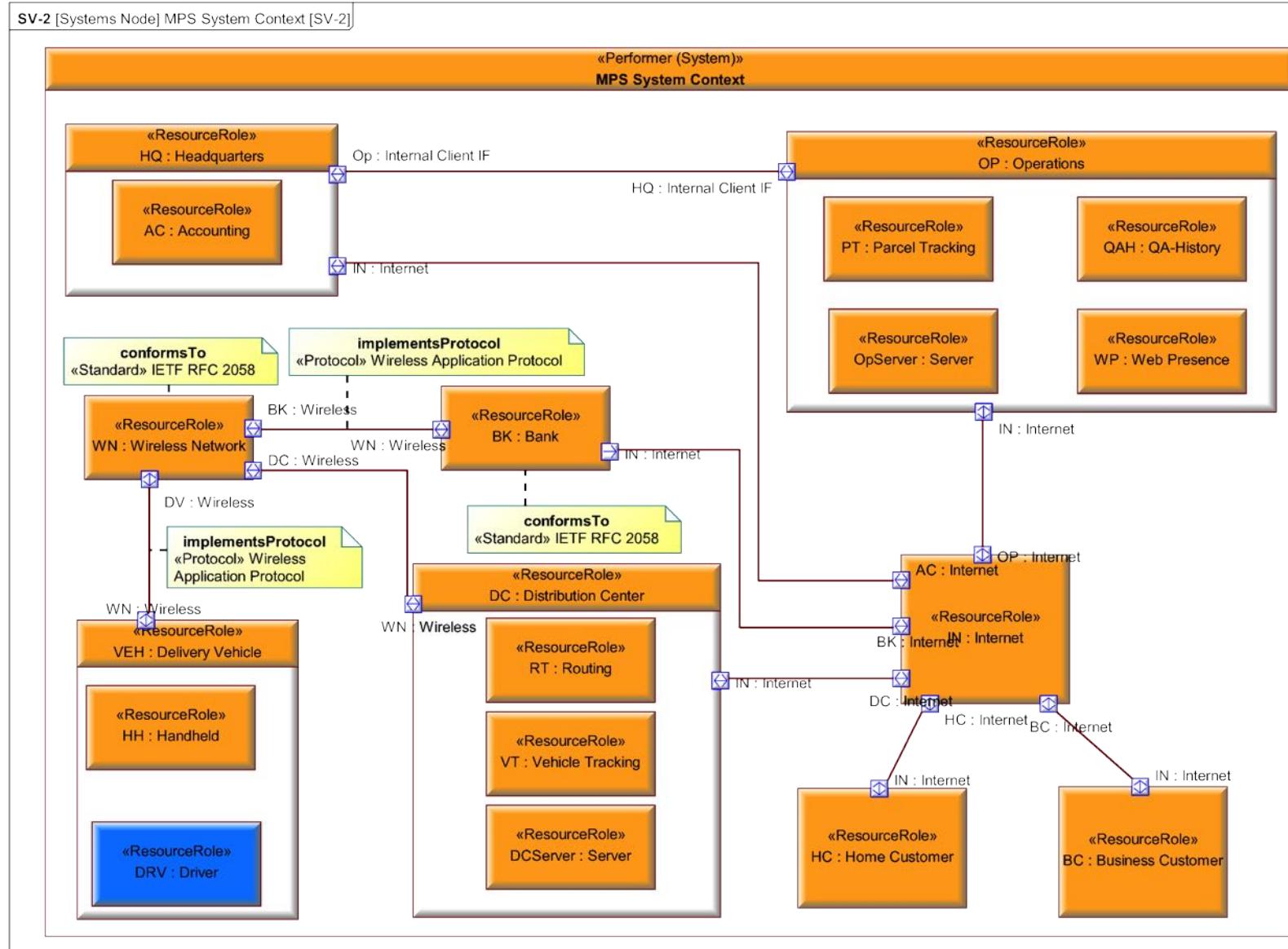
Defines system and human interface requirements and interactions

THE EVOLUTION OF STANDARDS OVER TIME



Defines standards and standards forecasts

SYSTEM INTERFACE SPECIFICATION



Defines how systems will interact to provide capabilities

STANDARDS COMPLIANCE MATRIX

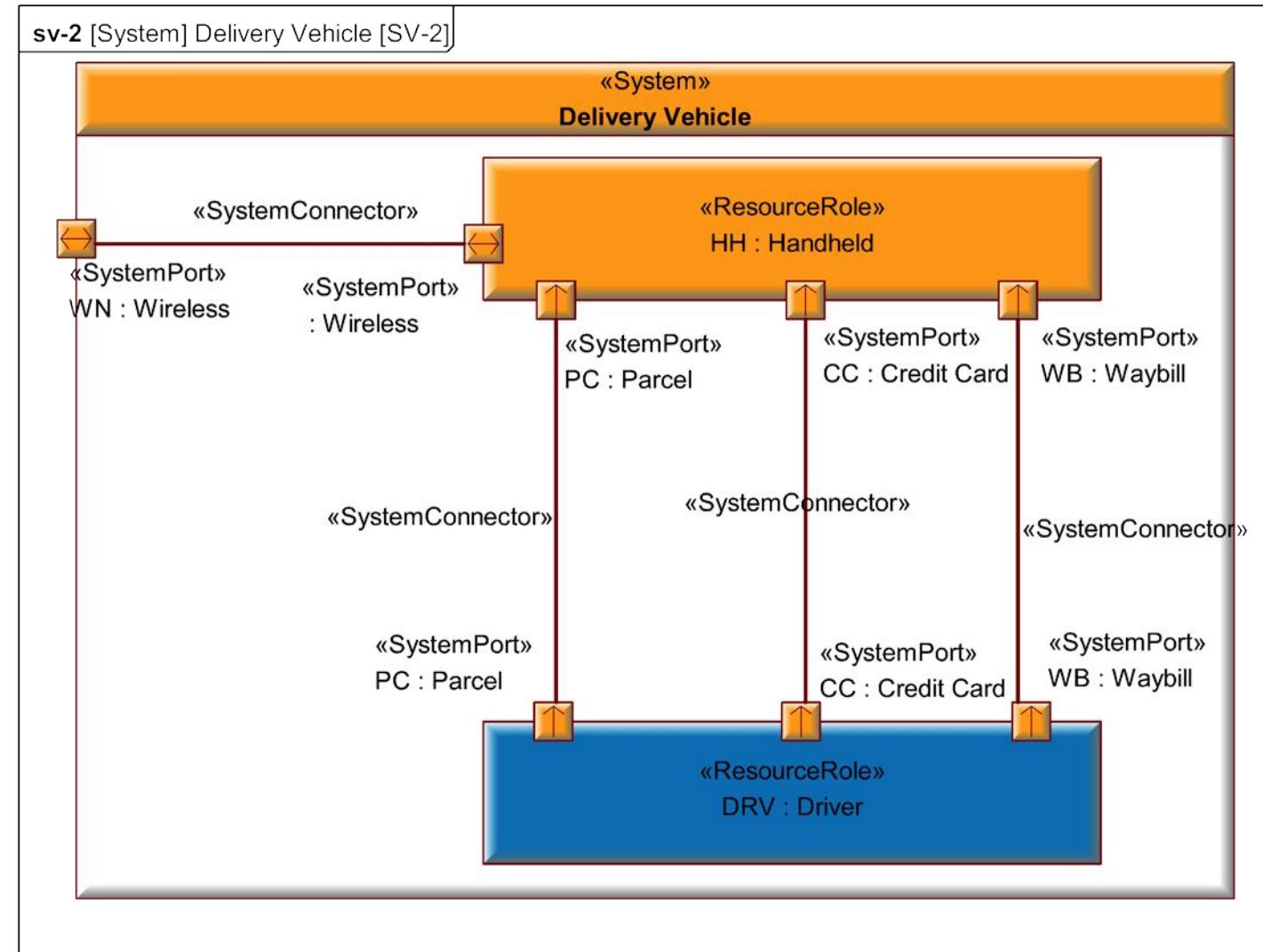
[Architectural Description] Technical Views [StdV-1 Matrix]

		Standards																							
		«Protocol» IETF-secstr-architecture-05	«Protocol» Simple Object Access Protocol (SOAP)	«Protocol» SSL Protocol Version 3.0	«Protocol» Wireless Application Protocol	«Standard» IETF RFC 2058	«Standard» IETF RFC-1510	«Standard» IETF RFC-1828	«Standard» Information Security	«Standard» ISO 13407	«Standard» ISO/IEC 8802.3:1996	«Standard» T251	«Standard» User Interface	«Protocol» IETF-secstr-architecture-05	«Protocol» Simple Object Access Protocol (SOAP)	«Protocol» SSL Protocol Version 3.0	«Protocol» Wireless Application Protocol	«Standard» IETF RFC 2058	«Standard» IETF RFC-1510	«Standard» IETF RFC-1828	«Standard» Information Security	«Standard» ISO 13407	«Standard» ISO/IEC 8802.3:1996	«Standard» T251	«Standard» User Interface
		«ResourceRole» BK												«ResourceRole» HH											
Conforming Elements	«SystemInterface» HH - BK													«ResourceRole» HH											
	«Performer (System)» Wireless Network													«ResourceRole» WN											
	«ResourceRole» WP																								

Generated automatically.
Summarizes standards conformance

Conformance

DRIVER-HANDHELD MODULAR INTERFACES



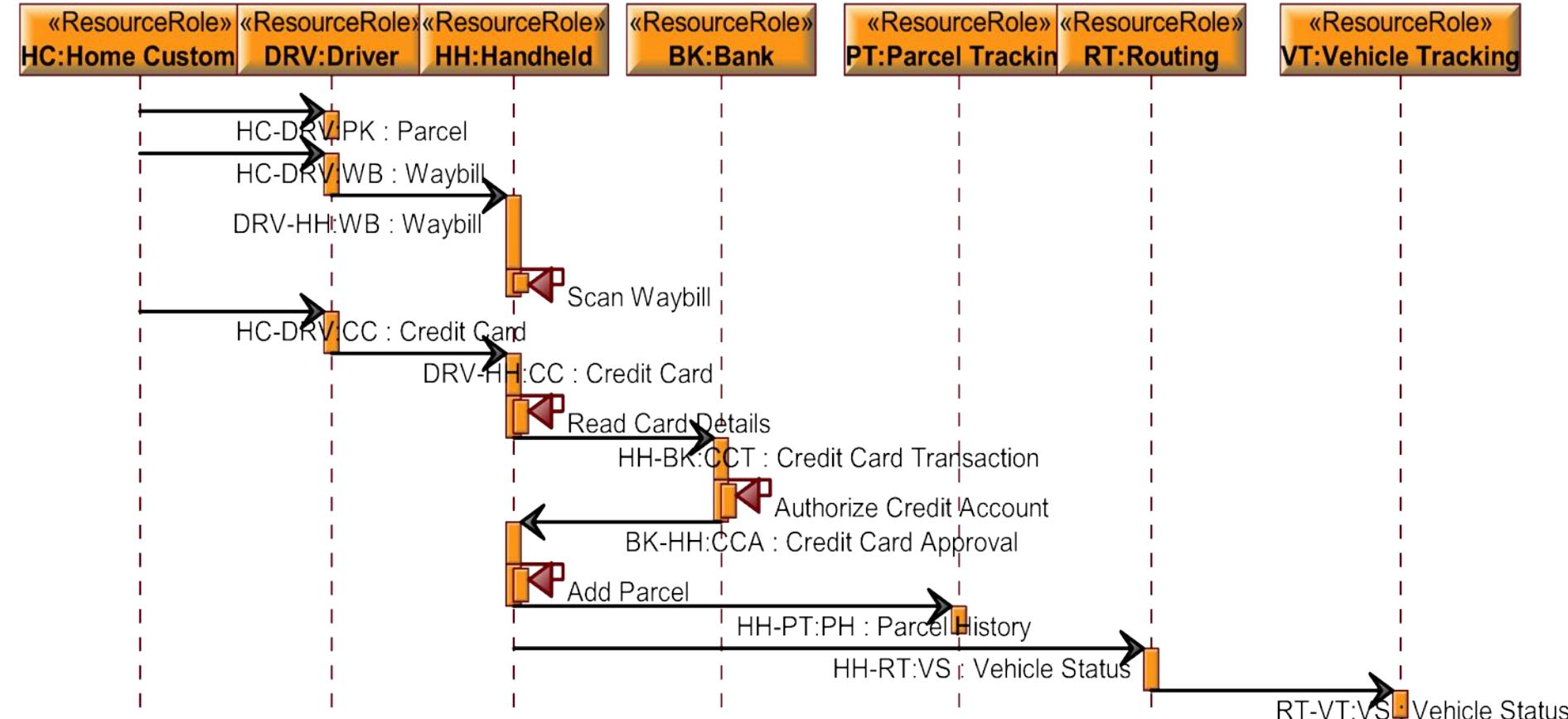
SYSTEM EVENT TRACE DESCRIPTION

- The order and timing of the interactions is just as critical as the interface definition itself: not just what happens, but when and why it happens.

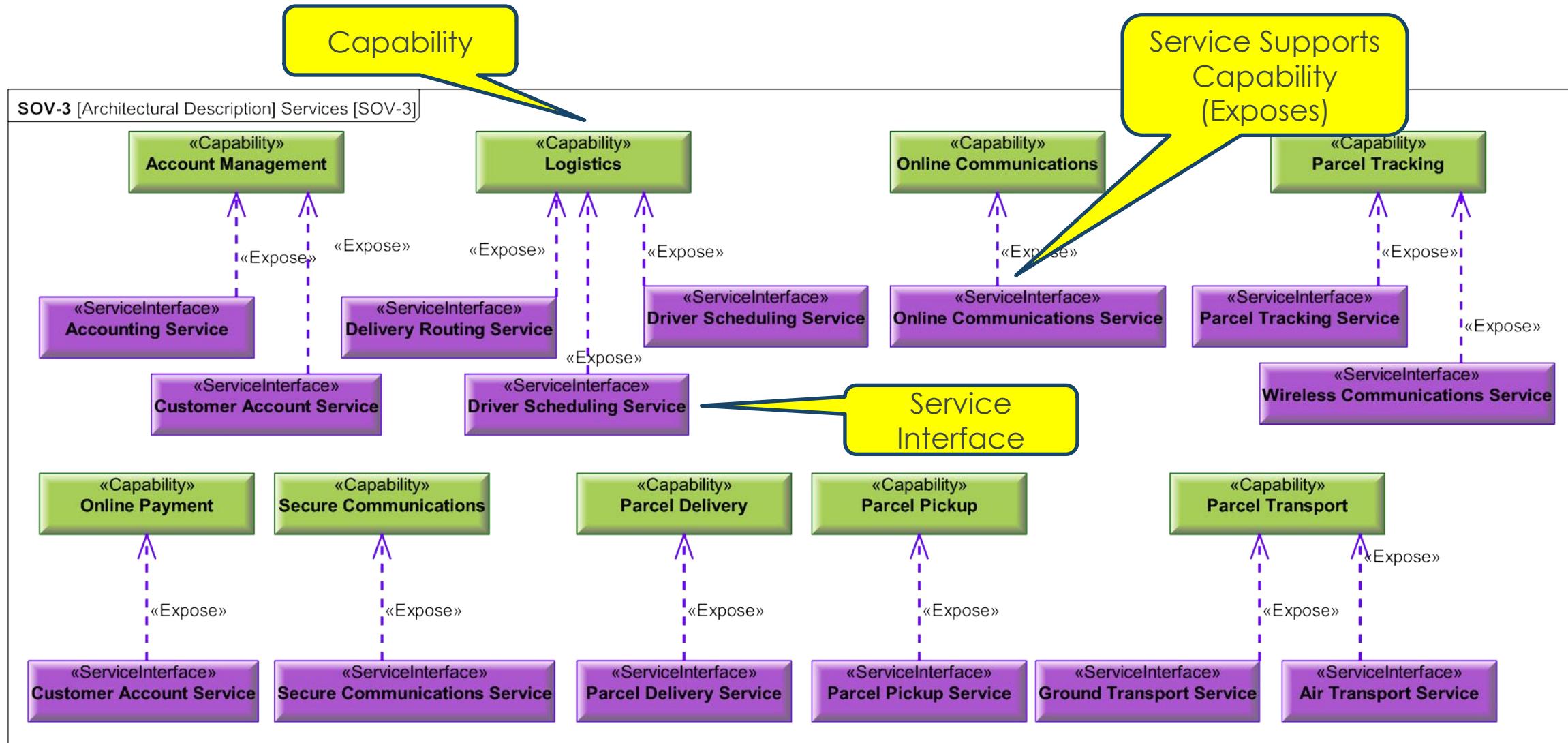
MPS System Context

Description

- Provide Parcel
- Provide Waybill
- Place Waybill in Front of Scanner
- Scan and Store Waybill
- Provide Credit
- Place Card in Scanner
- Scan and Store Card
- Request Card Authorization
- Authorize Card
- Send Approval
- Update Parcel Status
- Update Parcel Status
- Update Vehicle Status
- Update Tracking Status

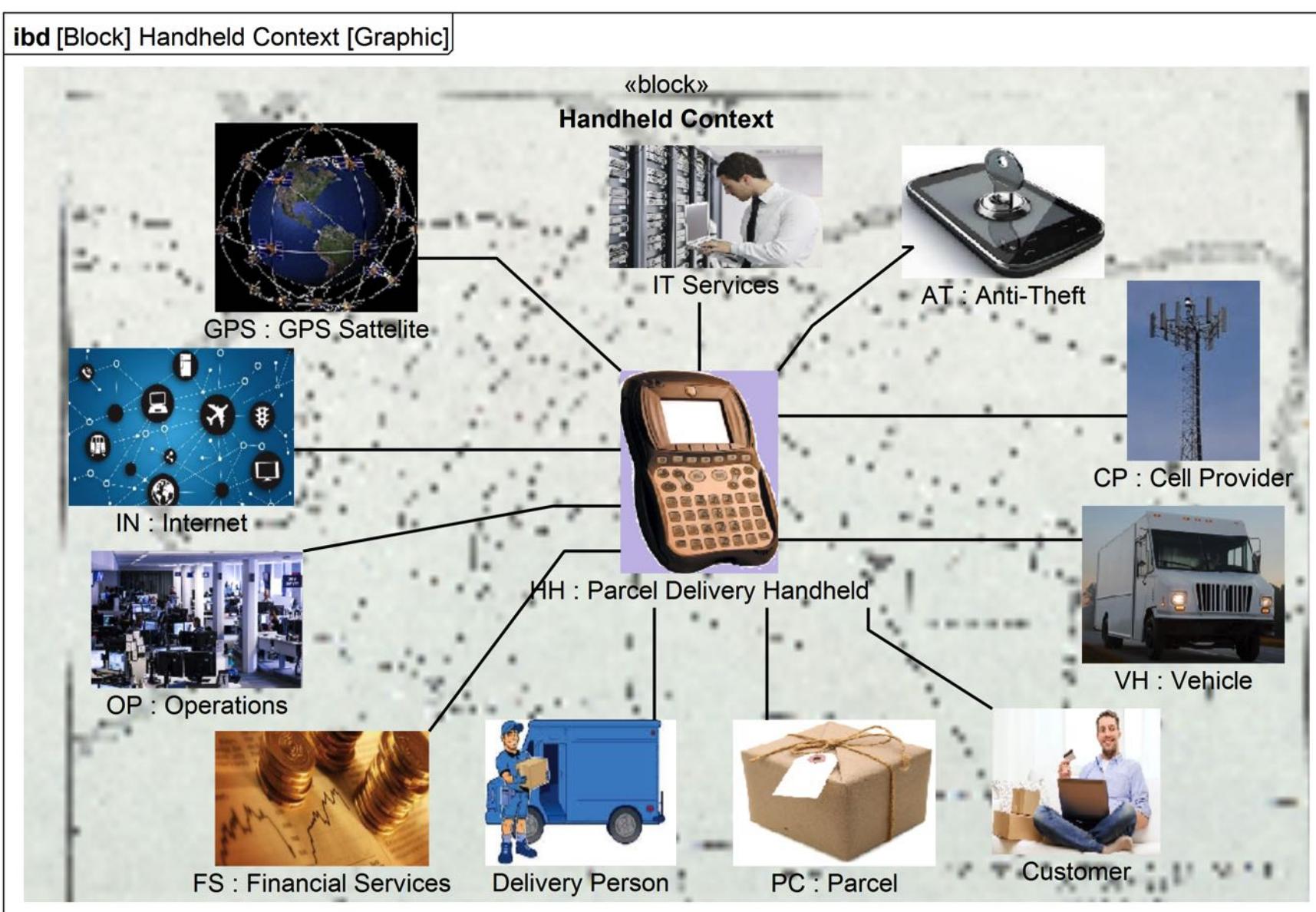


DERIVING SERVICES FROM CAPABILITIES

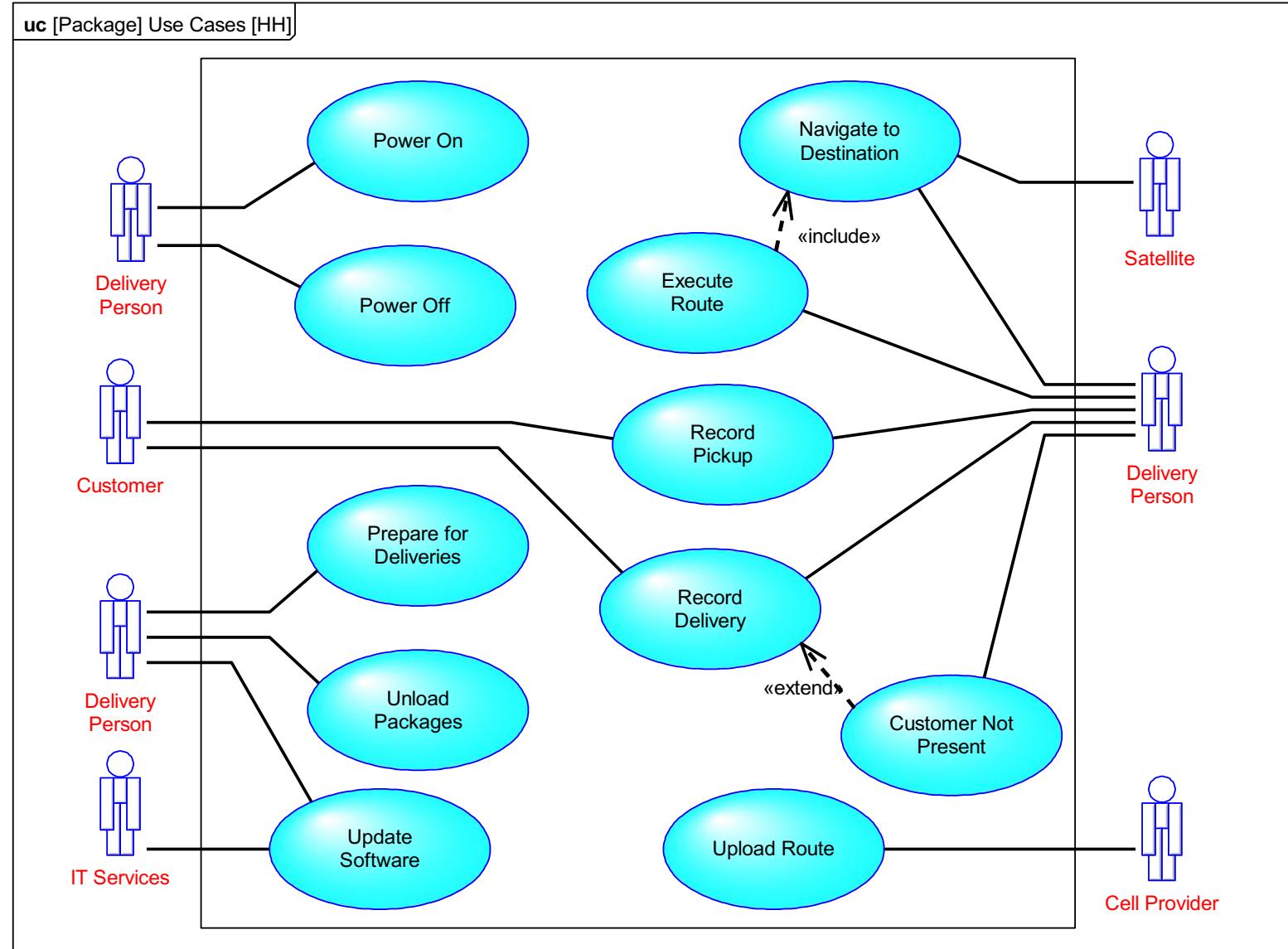


SYSTEMS INTERFACES

CONTEXT OF HANDHELD DEVICE



USE CASES DEFINE INTERACTIONS WITH ACTORS

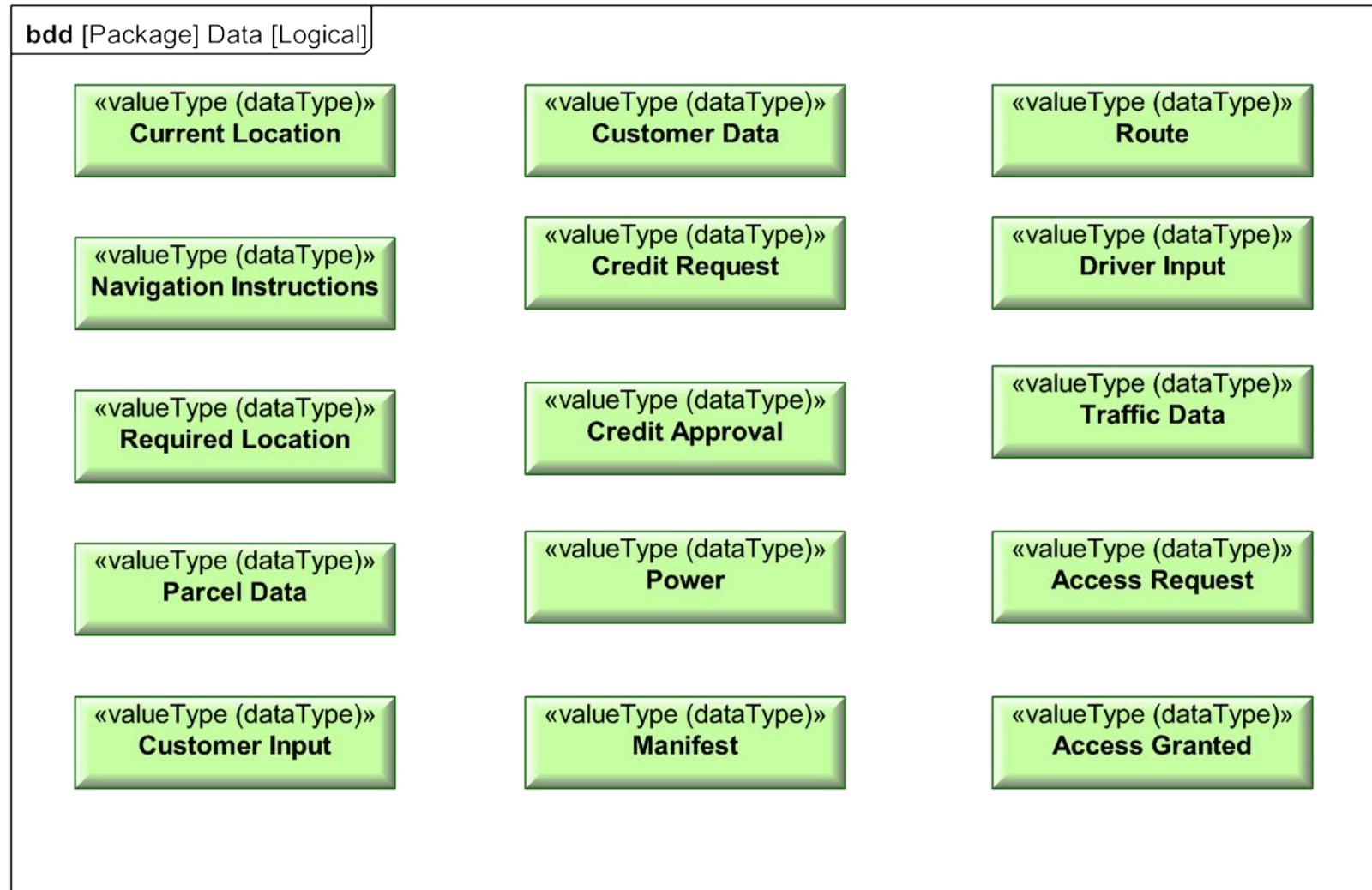


LOGICAL V. PHYSICAL MODELING WITH IBDS

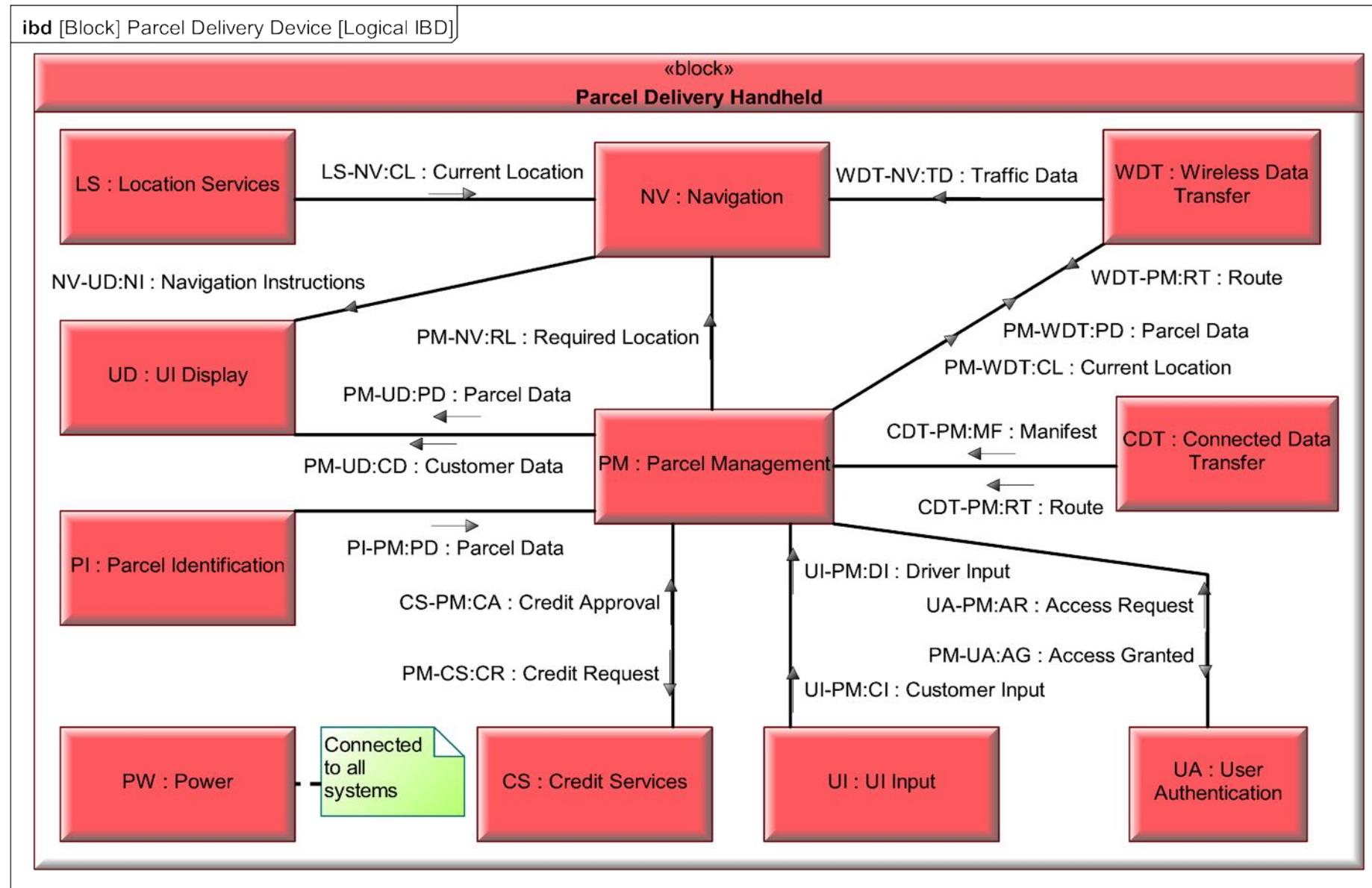


- IBDs can be used to capture both a logical model of parts, connections and flows, and a physical model
- Logical model focuses on logical parts and flows and may not show ports or types (unless logical types defined)
 - Based on specification rather than implementation ('what' not 'how')
 - Abstract types (if any)
- Physical model focuses on physical parts and flows and normally shows ports and physical (implementation) types
 - Normally follows logical modeling
 - May be many physical models for one logical model
 - Real-world types
- May affect package structure
 - Logical package contains logical types
 - Physical package contains physical types
- Can link logical model items to physical model items via Allocation

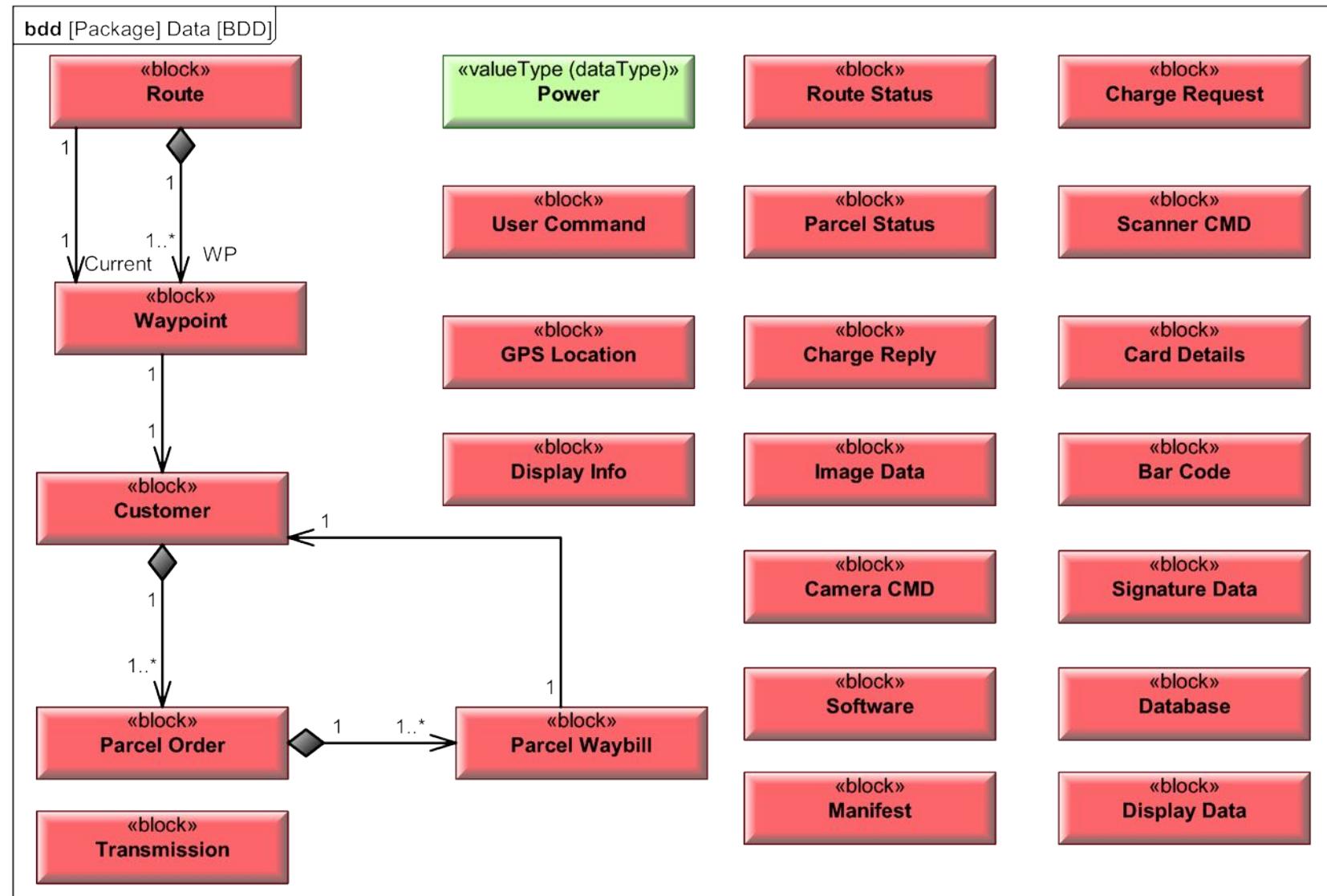
LOGICAL DATA



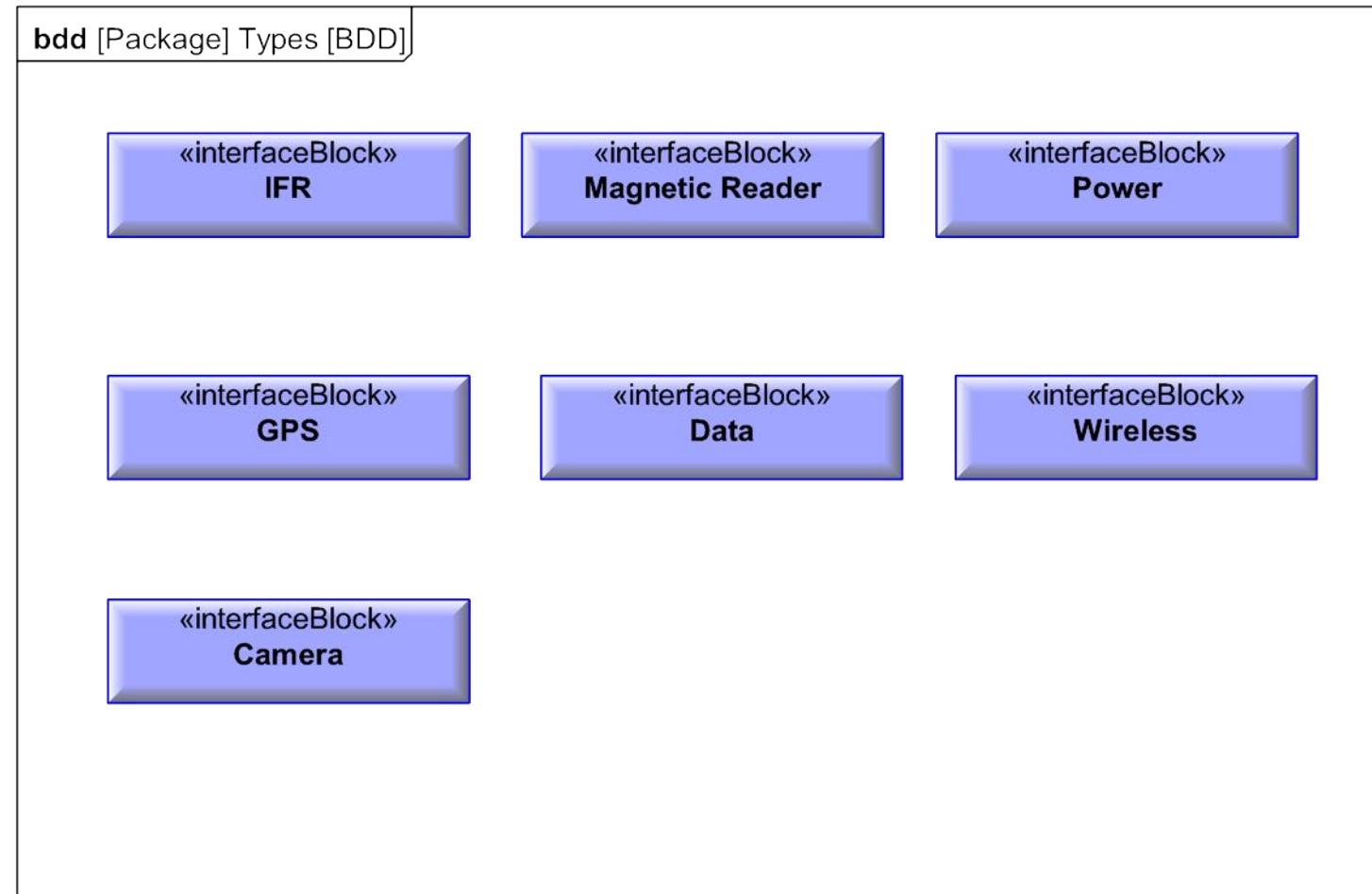
EXAMPLE IBD - LOGICAL MODEL



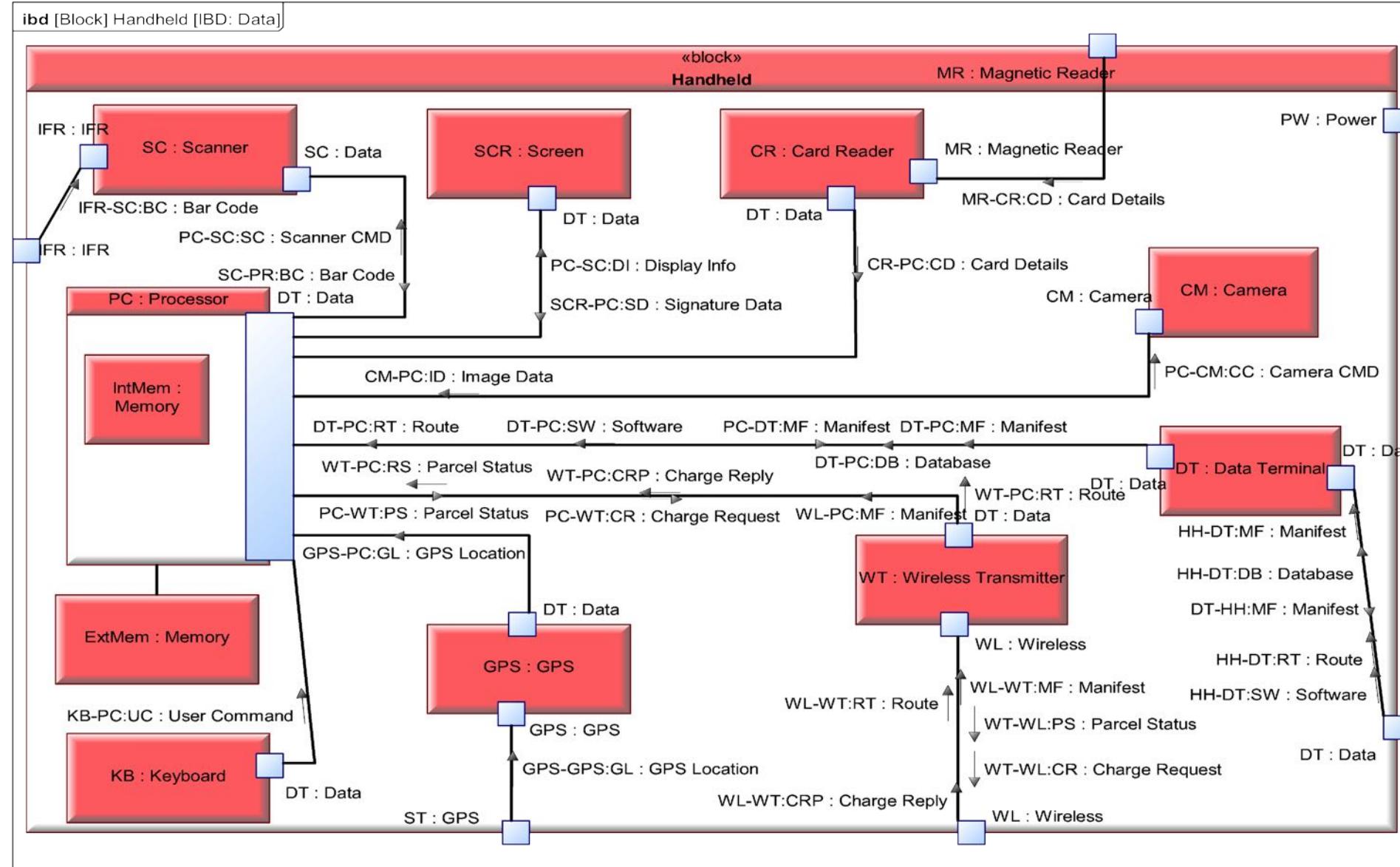
PHYSICAL DATA



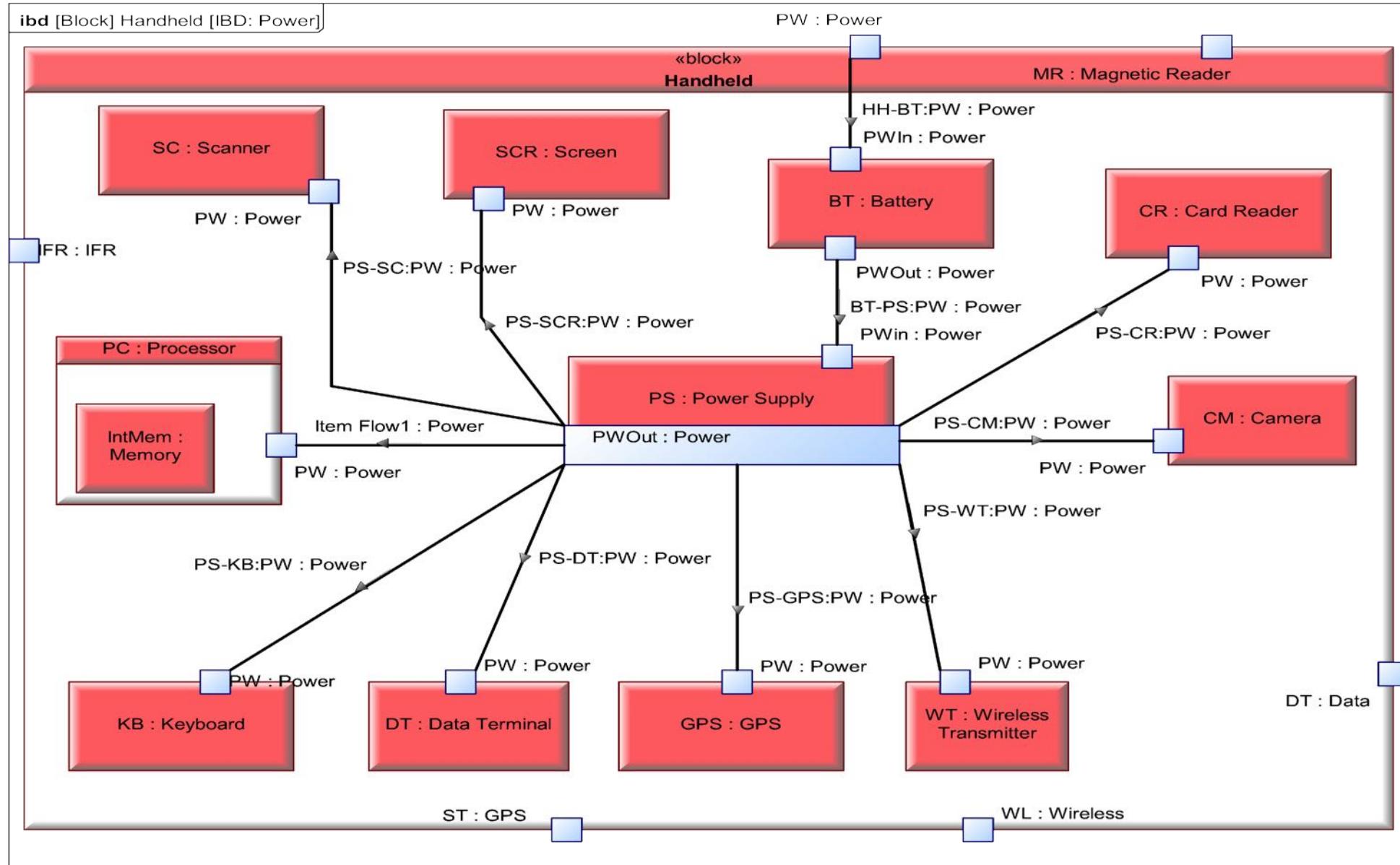
INTERFACES



EXAMPLE IBD – PHYSICAL MODEL



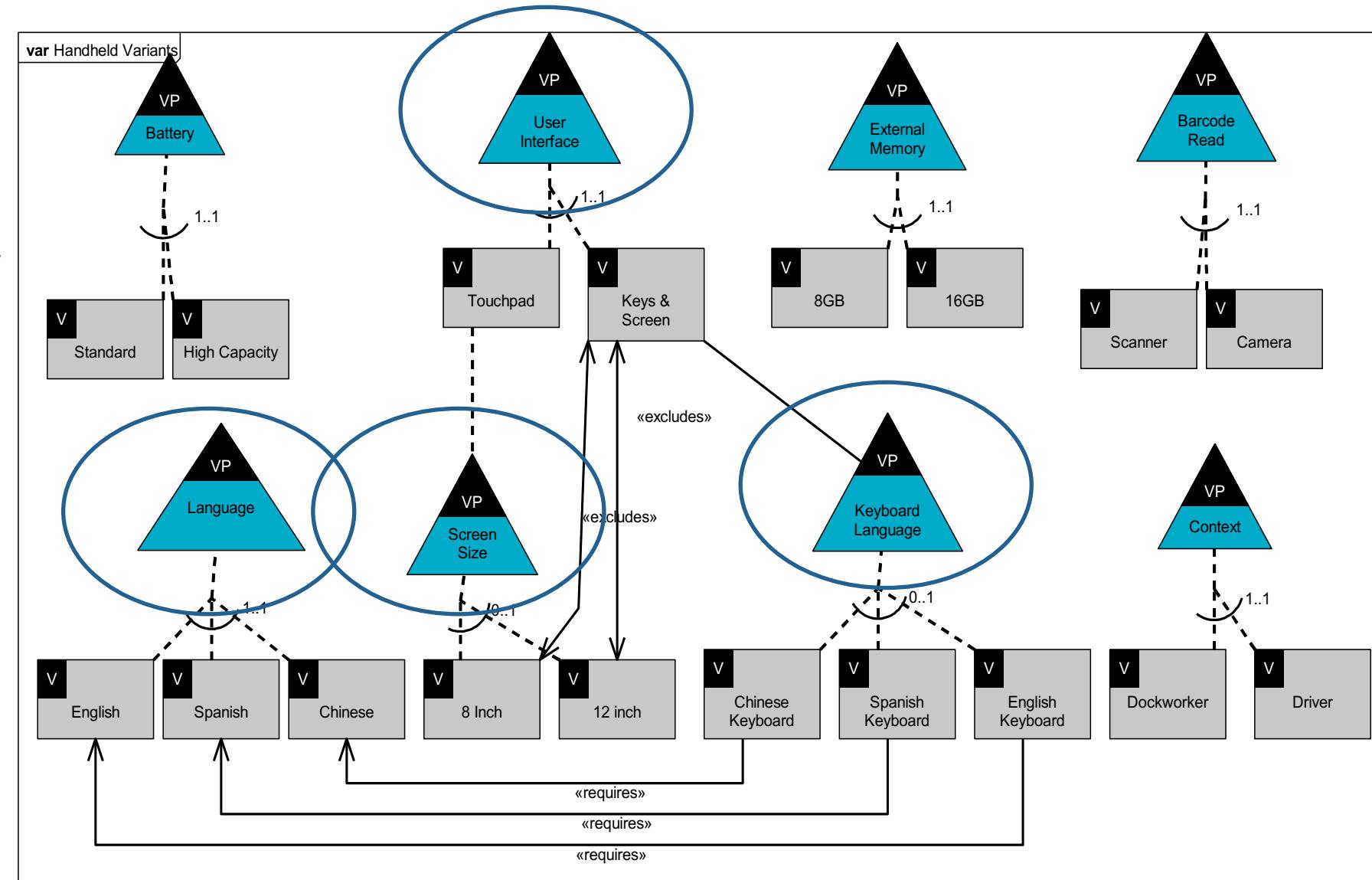
EXAMPLE IBD – PHYSICAL MODEL



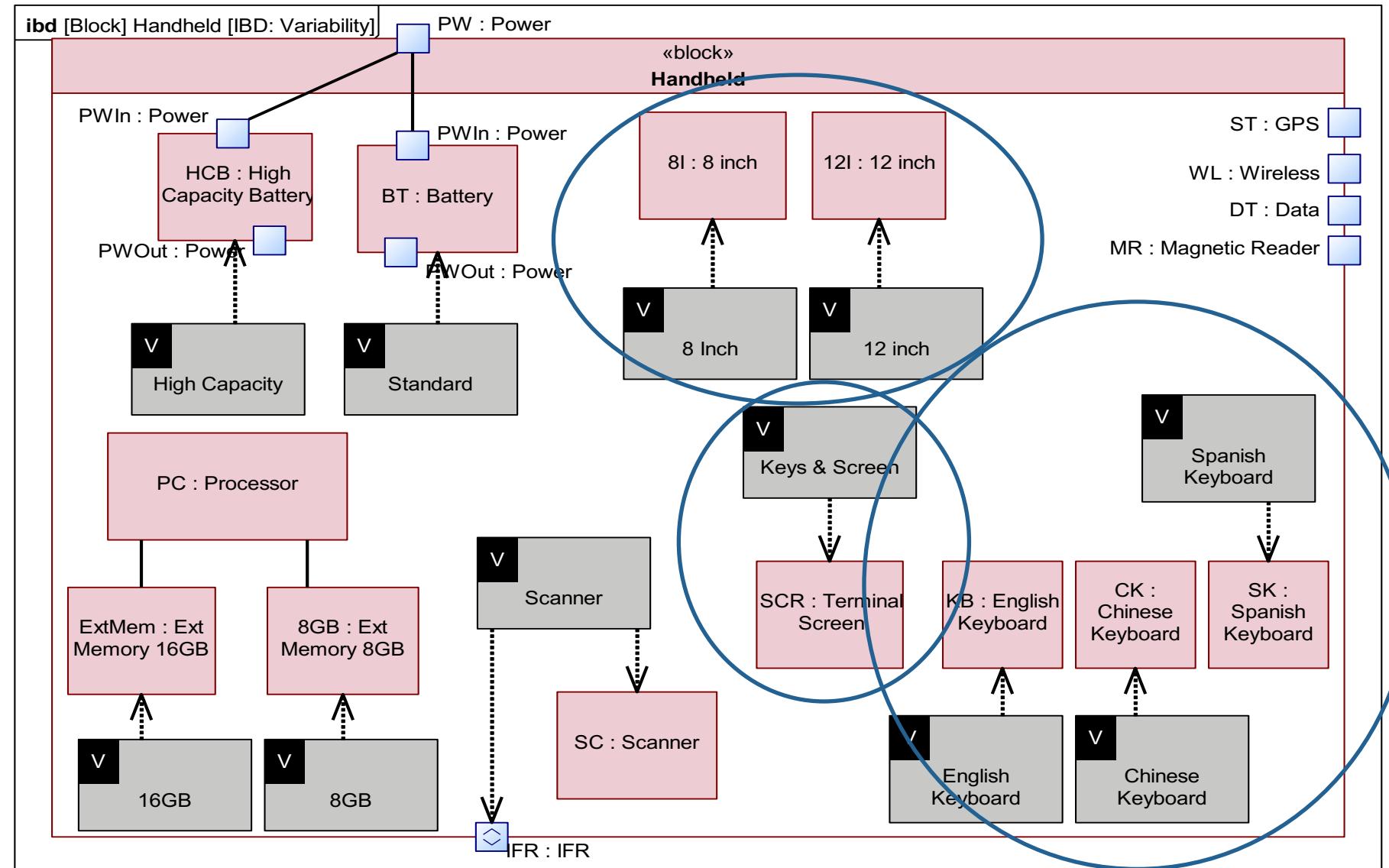
VARIABILITY

VARIABILITY MODELING USING OVM

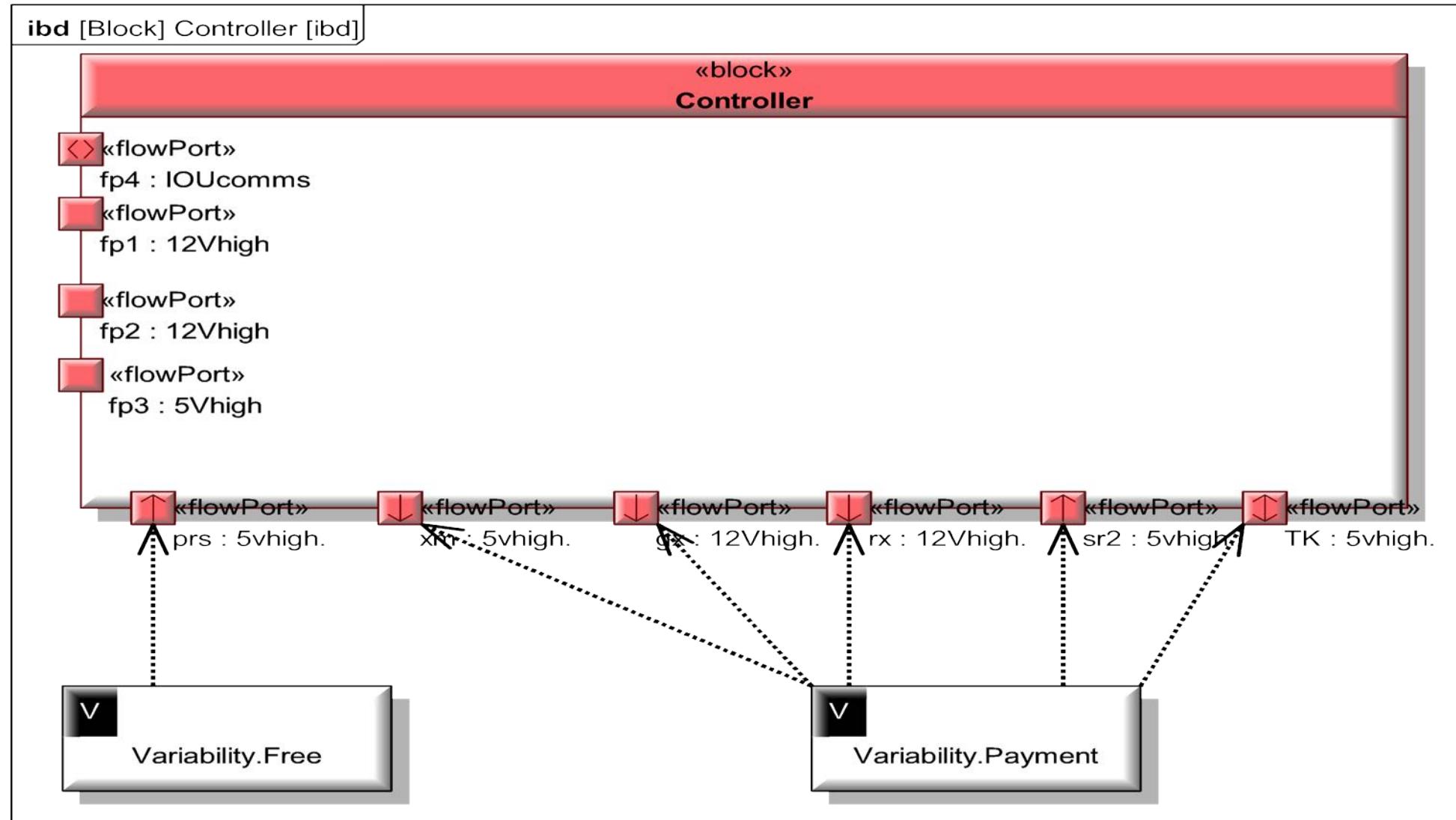
-  Variation Point
-  Variant
-  Variability Dependency
-  Mandatory/Optional
-  Requires Dependency
-  Excludes Dependency
-  Artifact Dependency
-  Alternate Choice



REFERENCING THE UI INTERFACE ELEMENTS



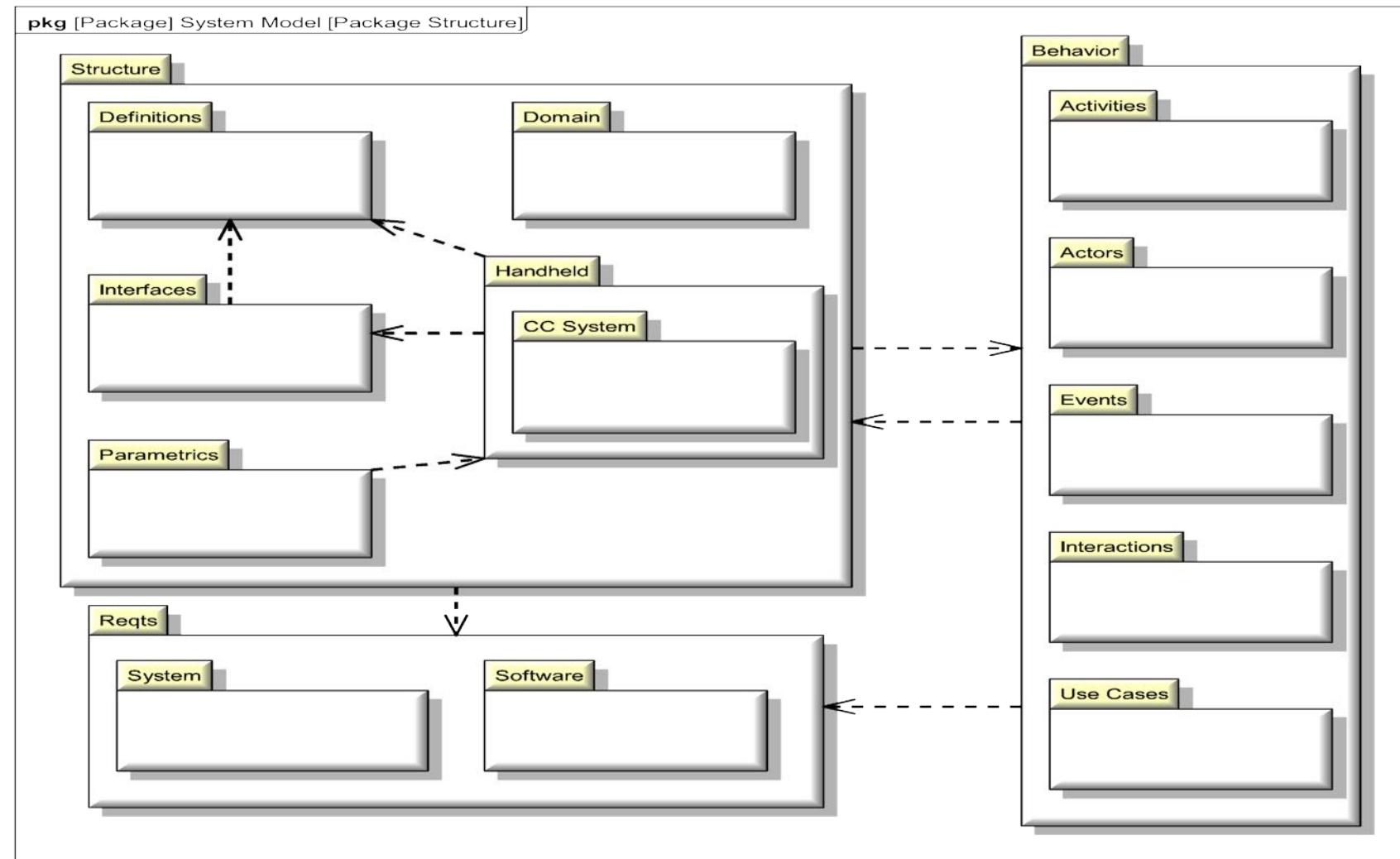
INTERFACE VARIANTS



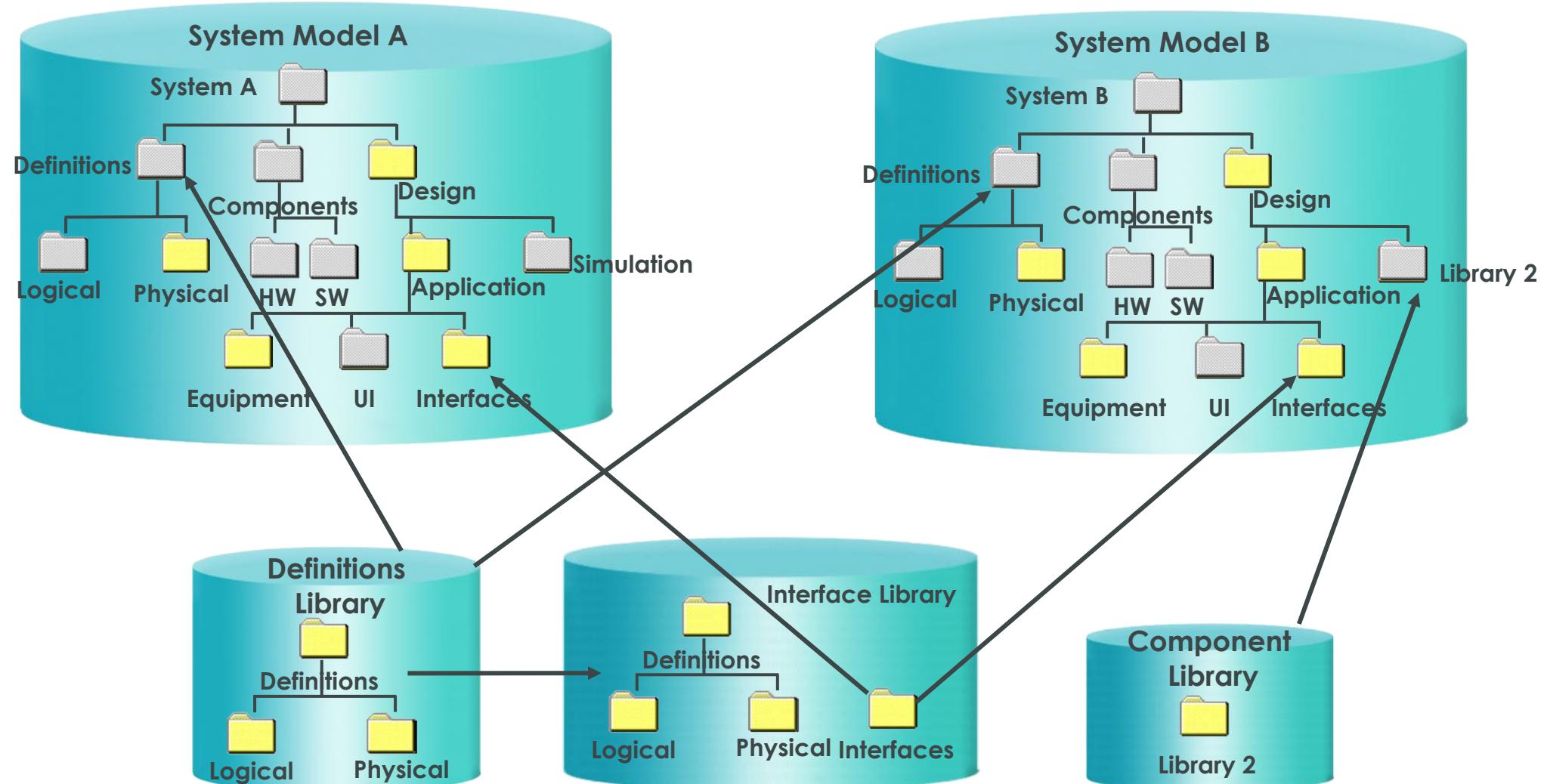
INTERFACE LIBRARIES

MODEL PACKAGE STRUCTURE

- Shows Dependencies within model to interfaces



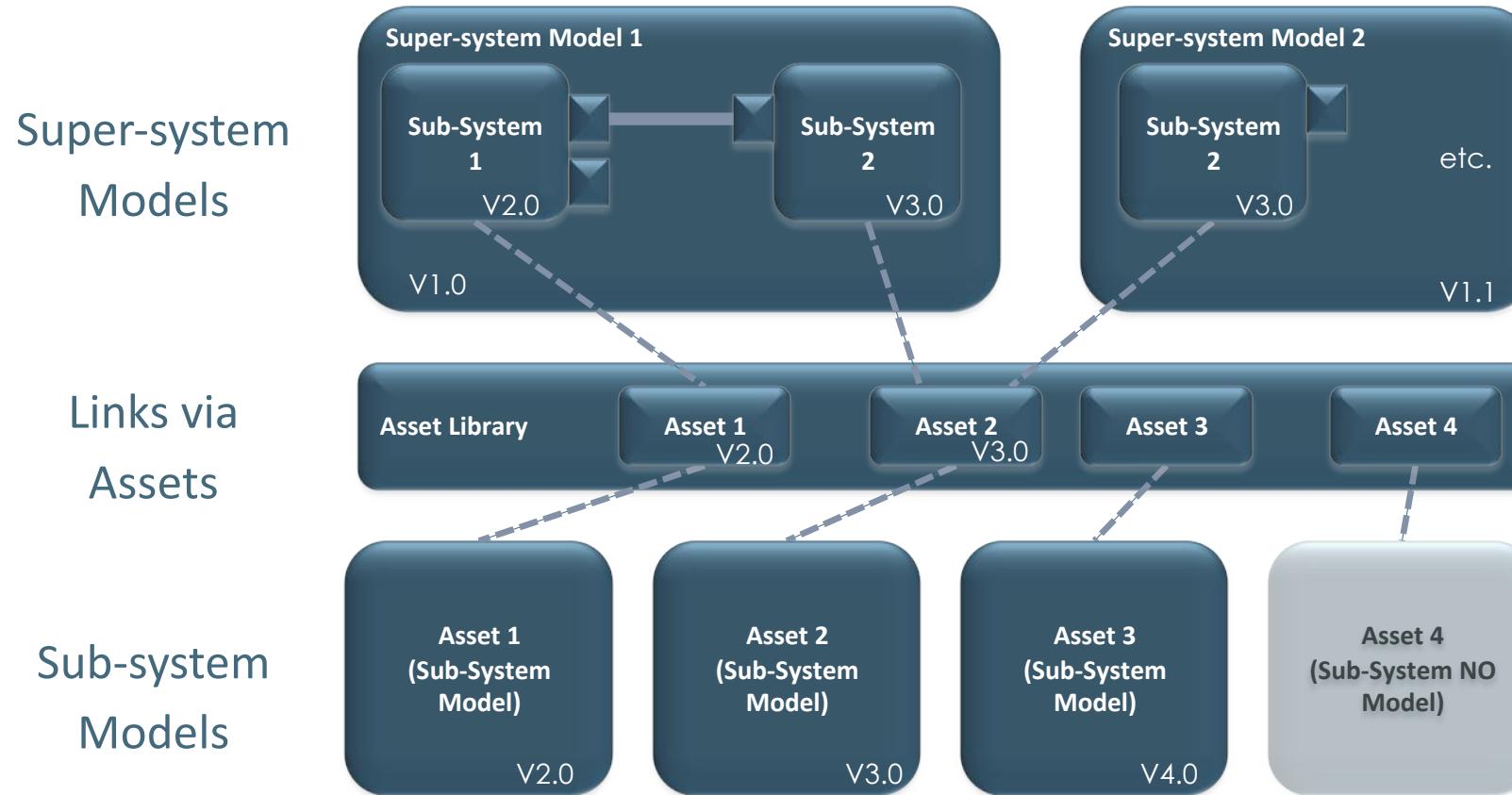
REUSING AND SHARING MODEL LIBRARIES



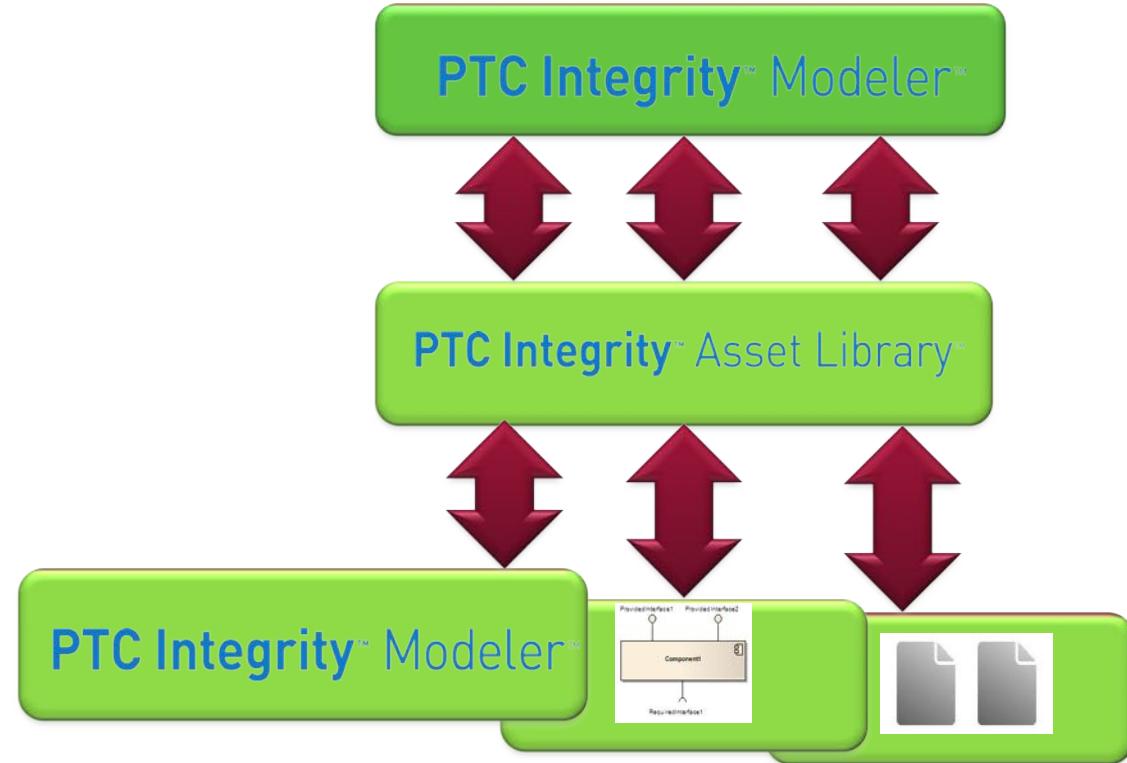
ASSET-BASED DESIGN ENABLES COLLABORATION AND VIRTUAL TEAMS

ASSET-BASED MODULAR DESIGN

- Super-system Model = Configuration of Versioned Sub-systems



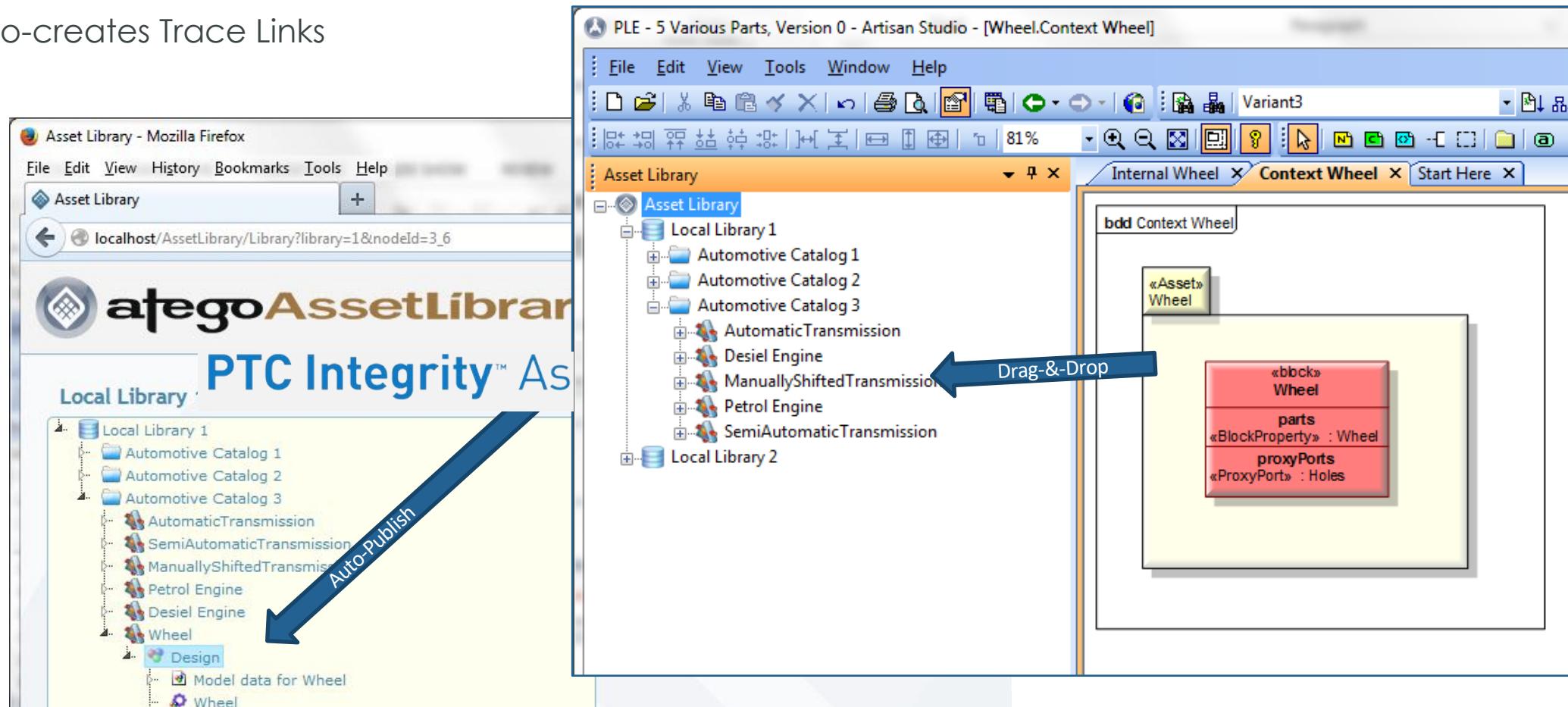
- Design the same way you Build
 - Construct Systems of Sub-Systems (SoS)
 - Use Services to build your Application (SOA)
 - Plug Components together (CBD)
- Modular Design
 - Top-Down, Architected
 - Specification (& Requirements) Driven
 - Parallel Working
 - Separation of Concerns
 - Bottom-Up, Asset Mining
 - Un-modeled Assets
 - Other Modeling Tools
 - Legacy Integration
 - Published Interfaces (e.g. IDL, SysML)
 - Uses the Reusable Asset Specification (RAS) and OSLC



ASSET-BASED MODULAR DESIGN

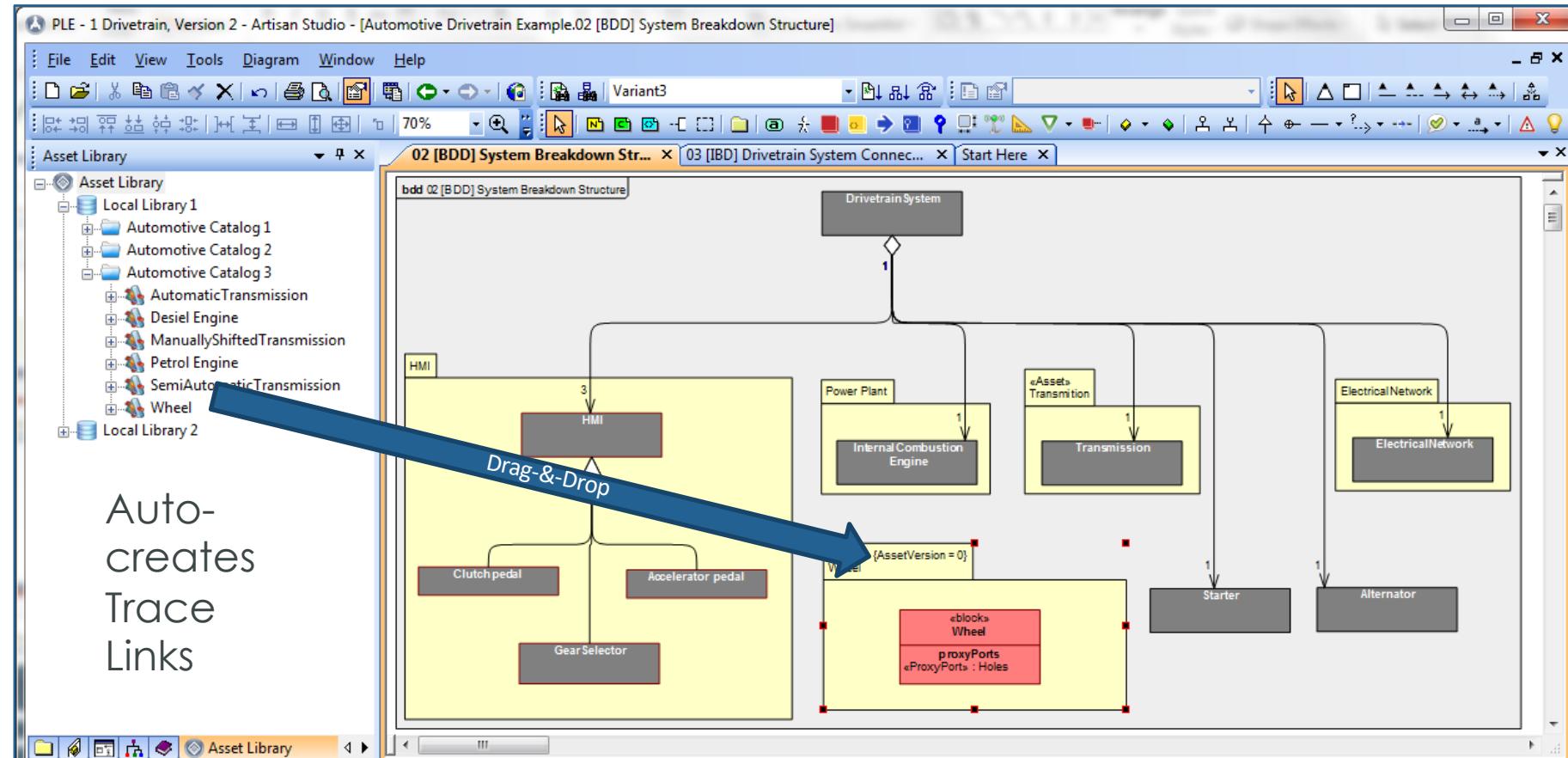


- Publish from Sub-system model into PTC Integrity Asset Library
 - Publishes the asset as a black box
 - Enables reuse as opposed to clone and own
 - Auto-creates Trace Links



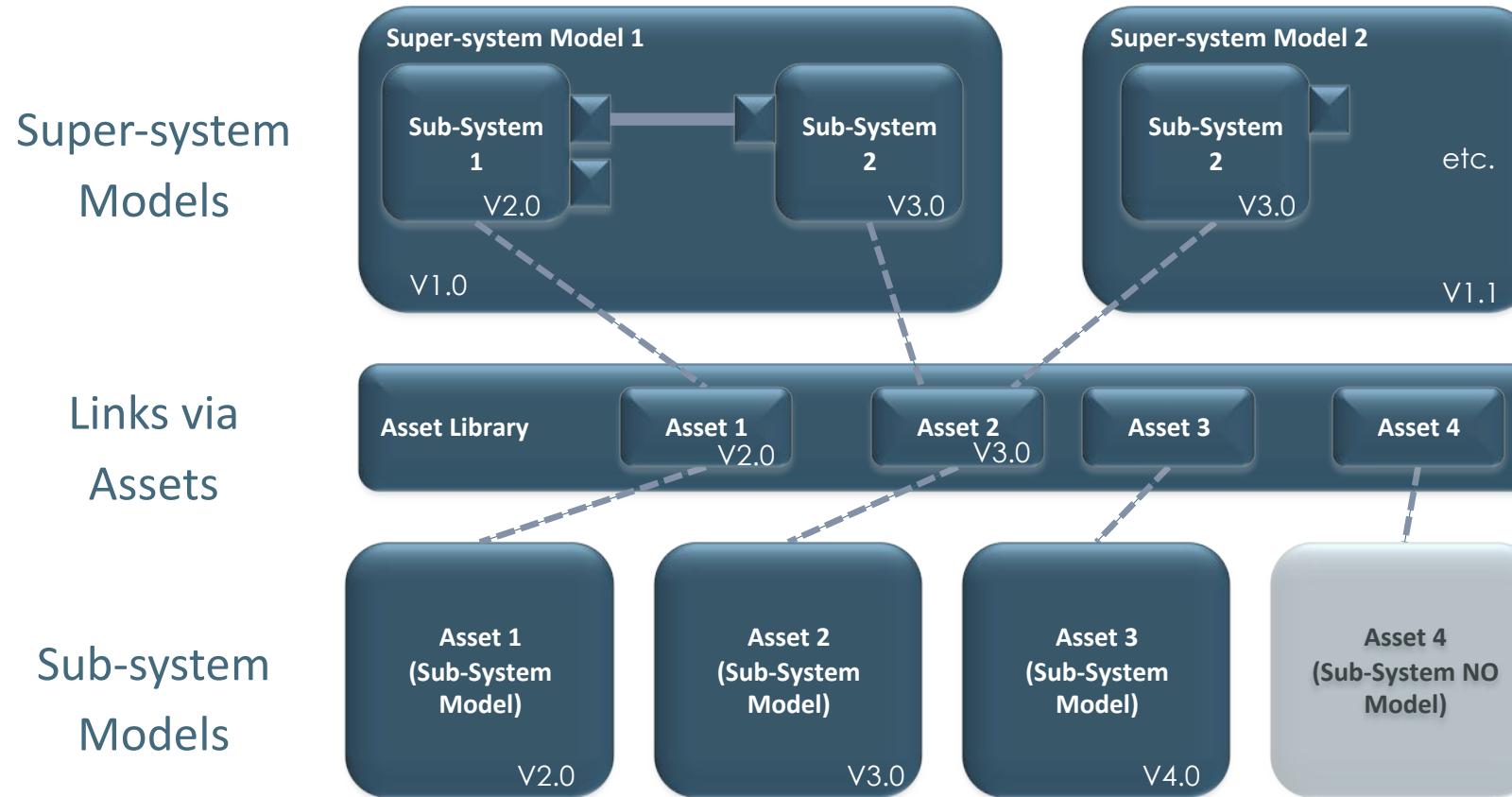
ASSET-BASED MODULAR DESIGN

- Use Sub-system from PTC Integrity Asset Library in Super-system Model
 - Reuse interfaces, requirements, operations, parameters, constraints, etc.



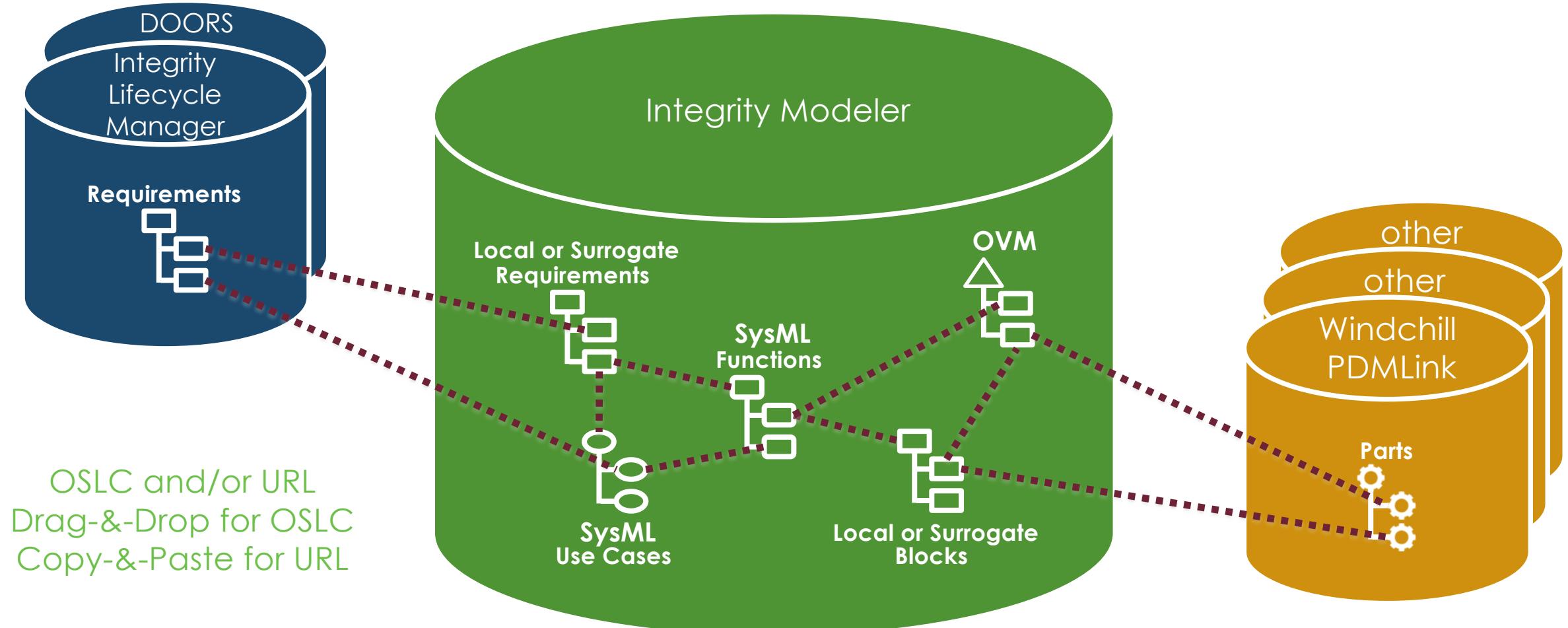
ASSET-BASED MODULAR DESIGN

- Super-system Model = Configuration of Versioned Sub-systems

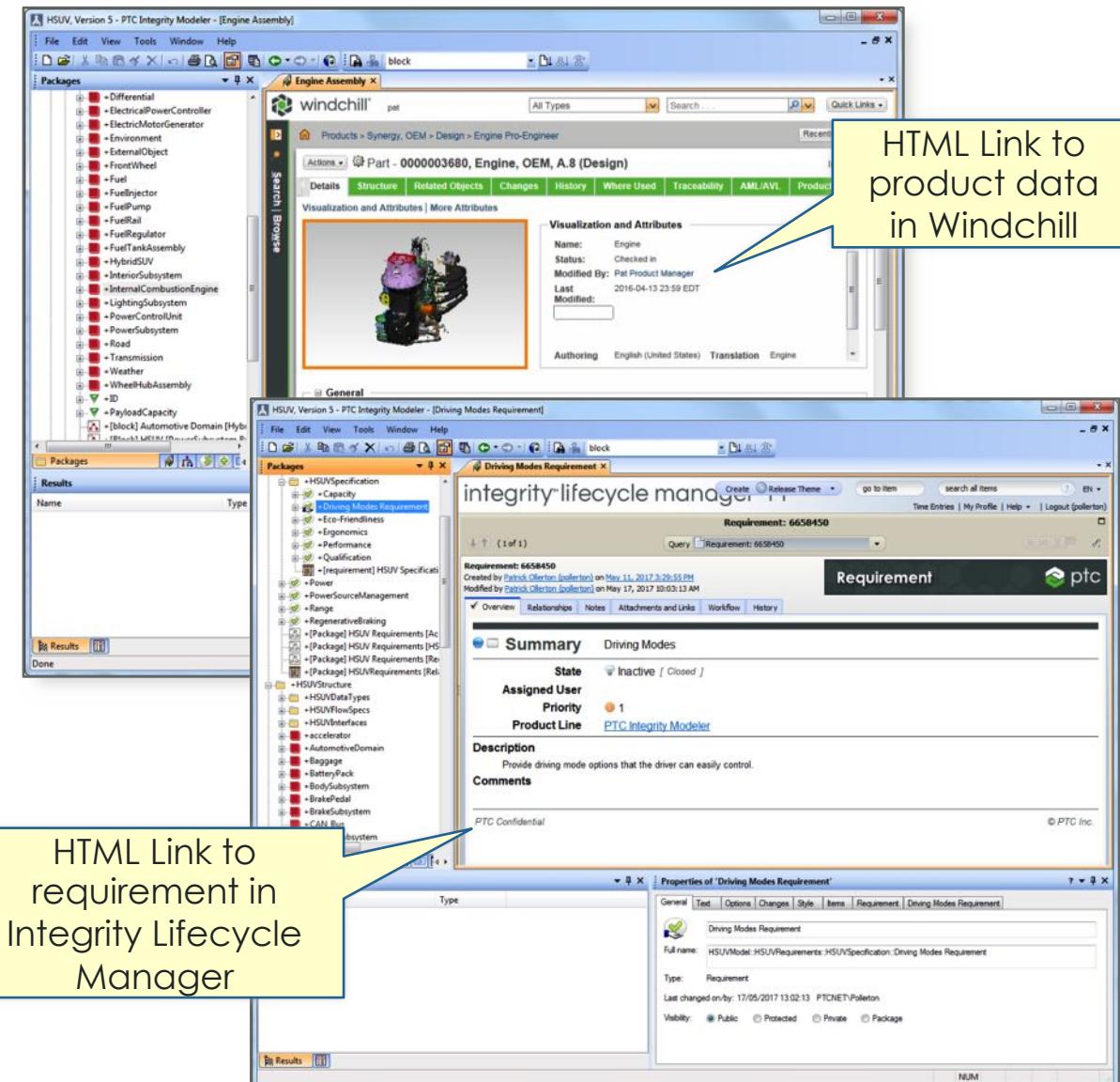
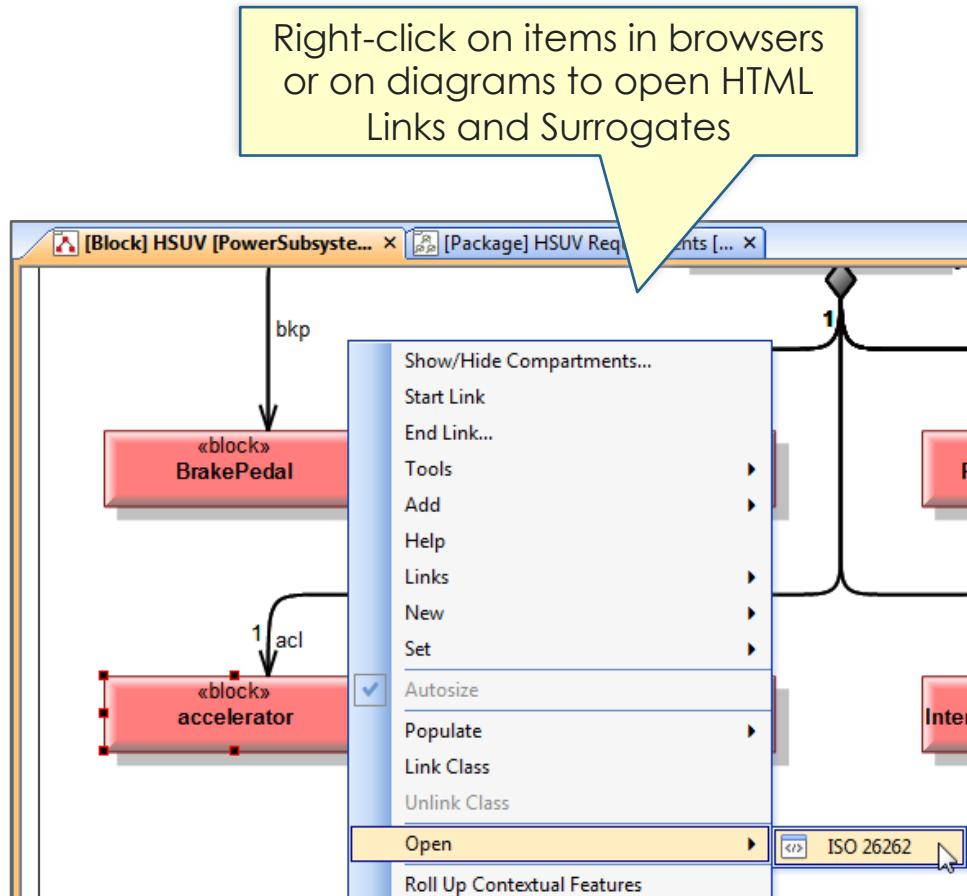


THROUGH THE DEVELOPMENT LIFECYCLE

External Traces & Model Surrogates with Visual Model Trace Links



TRACING FROM REQUIREMENTS TO SYSML TO CAD



THINGWORX TRACE MANAGEMENT (SE-PE) DISPLAY



ThingWorx Trace Management (SE-PE)

Modeler Provider: PTC.OSLC.ResourceProvider.modelerconnector.arc.item

Trace: Realizes

Windchill - Parts

Number	Name	Version
00072	PowerSubsystem	A.1
00078	ElectricalPowerController	A.1
00075	FuelTankAssembly	A.2
00081	InternalCombustionEngine	A.1
00074	BatteryPack	A.1
00079	Differential	A.1
00080	Transmission	A.1
00086	CAN_Bus	A.1
00085	ElectricMotorGenerator	A.1
00077	PowerControlUnit	A.2
00073	accelerator	A.1

You define the Integrity Modeler types that are available in the ThingWorx Trace Management app

You define the valid link types for your organization

Details

Traces

View

Use Case

Field	
Id	PTC.OSLC.ResourceProvider.modelerconnector.arc.item:http://icenter
Name	Accelerate

Details

Traces

Trace	Name
Satisfy	Performance (HSUVModel::HSUVRequirements::HSUVSpecification)
Allocate	Power (HSUVModel::HSUVRequirements)
Implement	PowerControlUnit (HSUVModel::HSUVStructure)

WINDCHILL LINKS TO INTEGRITY MODELER



The screenshot shows the Windchill interface for a part named '00072, PowerSubsystem, OEM, A.1'. The 'Traces' tab is selected in the 'Details' view. A callout box labeled 'Integrity Modeler icons shown' points to the icons in the 'Traces' table header. Another callout box labeled 'Integrity Modeler type and trace link type displayed' points to the 'External Type' and 'Trace' columns in the table. The table lists 12 trace objects with columns for Number, Version, Server, Title, External Type, and Trace.

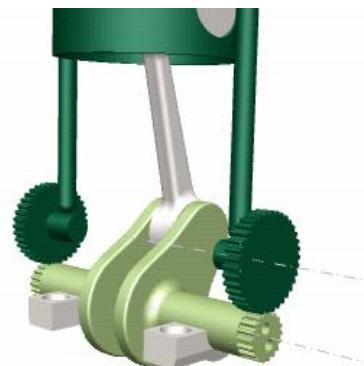
Number	Version	Server	Title	External Type	Trace
fc4f8cec...	000000	model...	EPAFuel EconomyTest	Activity	References
eacb06a...	000000	model...	PowerSubsystem	Block	Realizes
8f2ab98...	000000	model...	PowerControlUnit	Block	Implement
4358bcb...	000000	model...	[Package] SySim Custom Controls	BlockDefinitionDiagram	References
1f66cff8...	000000	model...	PowerControlSoftware	Class	Implement
e805dab...	000000	model...	Power Control Class Diagram	Class Diagram	Visualizes
47e1a18...	000000	model...	Interface1	Interface	Realizes
944a18b...	000000	model...	Range	Requirement	Allocate
03968a3...	000000	model...	[Package] HSUV Requirements [Ac...	RequirementDiagram	Visualizes
37e213c...	000000	model...	Accelerate	UML Activity Diagram	References
44453a7...	000000	model...	Accelerate	Use Case	Realizes
d432327...	000000	model...	HSUVUseCases [Operational Use ...	Use Case Diagram	Visualizes

PHYSICAL INTERFACES

Interfaces are controlled boundaries between modules, components or parts

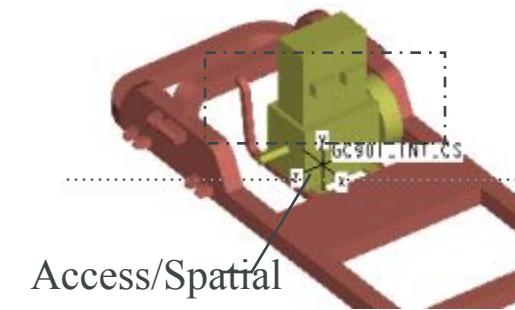
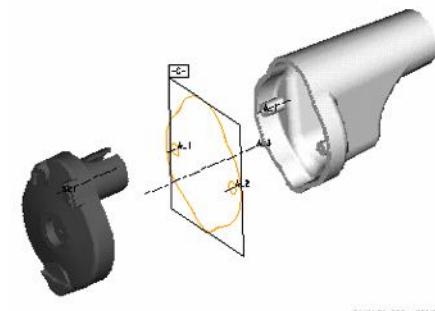
Types include:

- Attachment, Spatial (envelope)
- Transfer (e.g. power)
- Communication
- User Interface



Transfer of Power

Direct/Attachment



Access/Spatial

Communication

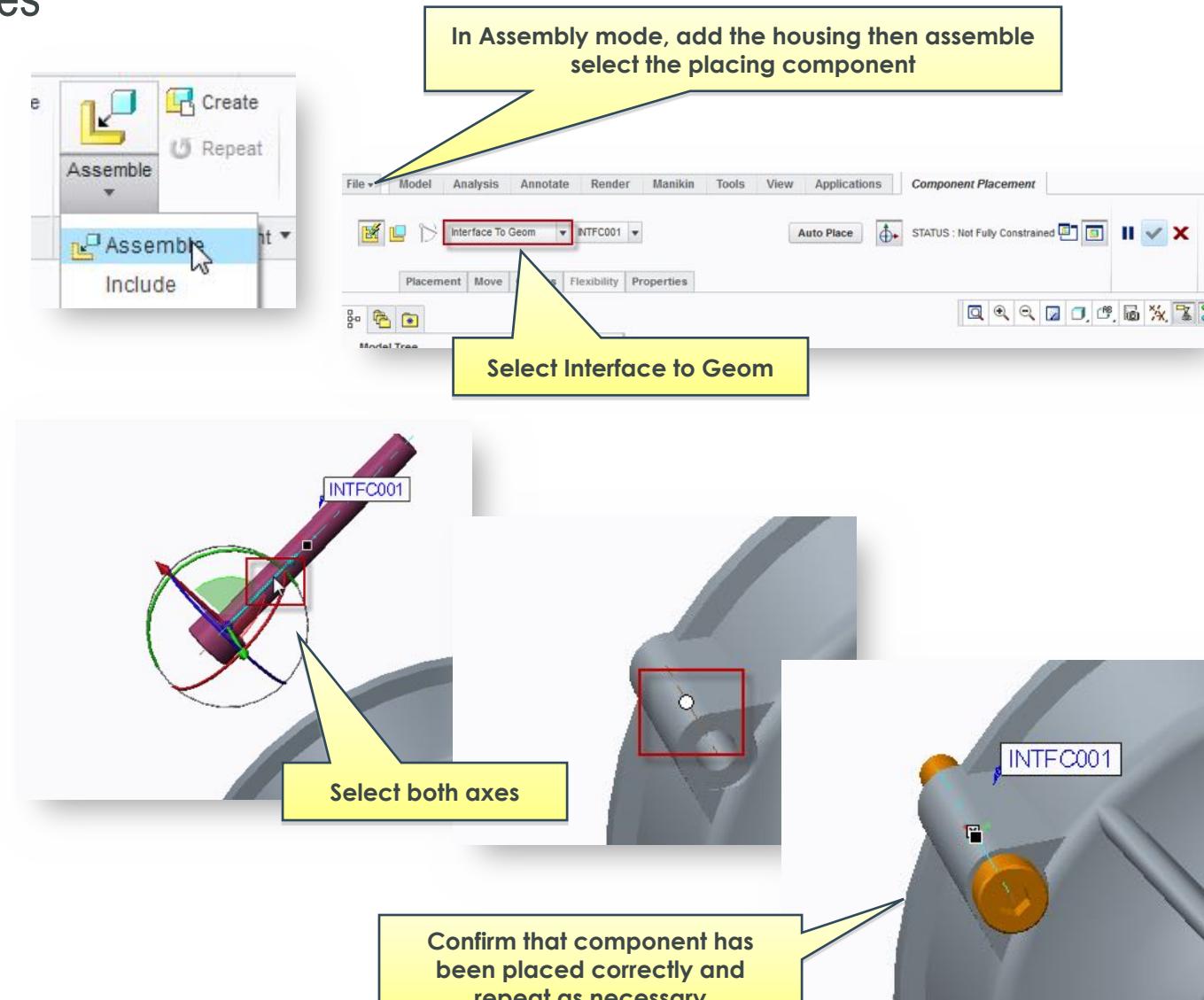
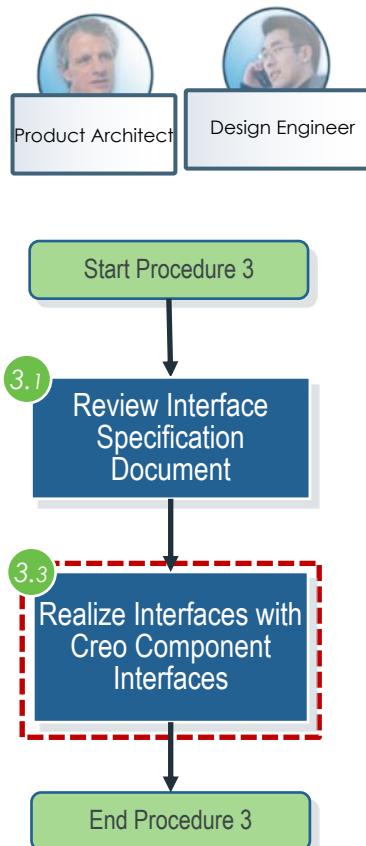


User Interface



REALIZING INTERFACES

► Develop and Propagate Interfaces





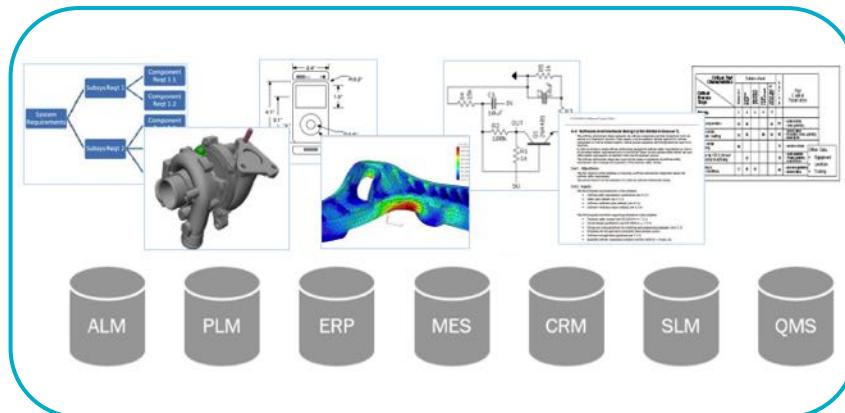
COLLABORATIVE AR/VR DESIGN



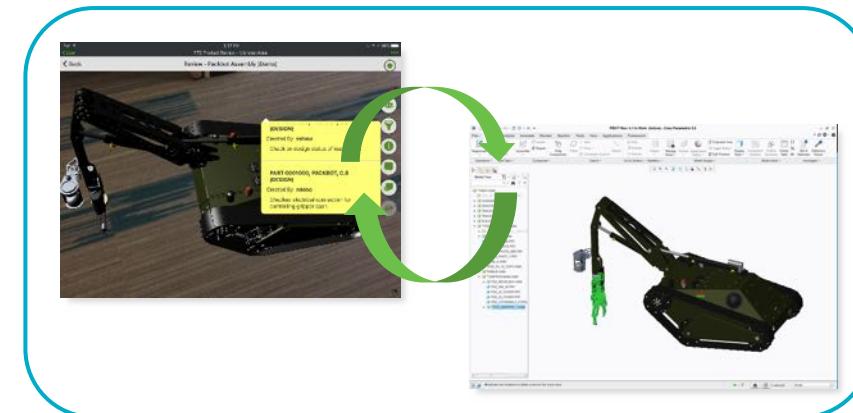
A Few Simple Steps from CAD to AR/VR



Collaborate Globally



Effortlessly Collect all Relevant Information

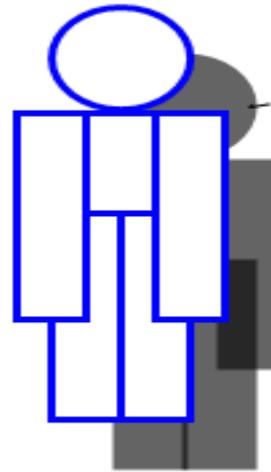


Closed-Loop Change Management

CONCLUSION



- Interface requirements start at the very beginning of development
- There are many ways to define an interface. The best one depends on particular circumstances and will change over time
- Interfaces can be traced from requirements through to architecture through to design and physical implementation
- Define common interfaces first in a collaborative environment.
 - This means they will be available when people need them.
 - They will also only be defined once
- Interfaces are where things usually go wrong so it is best to get them right.



Speaker

Thanks for your attention!



ptc