



Analysis of the OSLC Specs and the PLM Reference model in the context of SE Scenario #1

V0.7
December 7th 2010
Gray Bachelor
Mike Loeffler
OSLC PLM Workgroup





Acknowledgements

- OSLC PLM Workgroup

- Mike Loeffler

- Gray Bachelor

- OSLC RM Workgroup lead

- Ian Green

....and other OSLC Workgroup members
and leads





Contents

- Background
- Motivation
- PLM Reference Model (proposed) overview
- Analysis of the OSLC Specs and PLM Reference Model in the content of the selected Scenario
- Next steps



Background

- Open Services for Lifecycle Collaboration are community developed interface specifications
- The OSLC PLM Workgroup aims to promote and pursue the use of the OSLC Resource models (Specs) to support collaboration between Software Application Lifecycle Management and Product Lifecycle Management
- Due to the lack of available and agreed reference or domain information the PLM Workgroup sponsored work to support investigation and subsequent recommendation by selecting and building up a representative scenario and PLM Reference model
- The current focus is on analysis and comparison with the aim of publishing some findings that can guide current usage and potential extensions



Overview of the OSLC specs

<u>Scenarios</u>	<u>Topics</u>
<u>Collaborative Lifecycle Management</u>	<u>Change Management</u> - integrations with work item and change management repositories. <u>Quality Management</u> - integrations in quality management and testing. <u>Requirements Management and Definition</u> - integrations in requirements management and requirements definition tools. <u>Asset Management</u> - integrations between asset management and other lifecycle tools. <u>Architecture Management</u> - integrations related to models and other artifacts used during analysis, design and construction that help maintain architectural integrity. <u>Software Configuration Management</u> - integrations targeted at version control, change sets, baselines and other resources <u>Automation (Build/Deploy)</u> - integrations involving automation tools like build, deploy, and analysis tools.
<u>Software Project Management</u>	<u>Estimation and Measurement</u> - integrations with estimation tools and performance, project, and portfolio management
Product Lifecycle Management	<u>PLM and ALM</u> - integrations across product lifecycle management and application lifecycle management tools
<u>OSLC Common Architecture</u>	<u>OSLC Core</u> - Essential elements, patterns for common concerns <u>Reporting</u> - a common set of features that RESTful services should implement to enable reporting.

What is the status of the SPECS ?



<u>Scenarios</u>	<u>Topics</u>	Rel	Pre
Collaborative Lifecycle Management	Change Management - integrations with work item and change management repositories.	V1	V2
	Quality Management - integrations in quality management and testing.	V1	V2
	Requirements Management and Definition - integrations in requirements management and requirements definition tools.	V1	V2
	Asset Management - integrations between asset management and other lifecycle tools.	V1	
	Architecture Management - integrations related to models and other artifacts used during analysis, design and construction that help maintain architectural integrity.		V2?
	Software Configuration Management - integrations targeted at version control, change sets, baselines and other resources		V1
	Automation (Build/Deploy) - integrations involving automation tools like build, deploy, and analysis tools.		
Software Project Management	Estimation and Measurement - integrations with estimation tools and performance, project, and portfolio management		V1
Product Lifecycle Management	PLM and ALM - integrations across product lifecycle management and application lifecycle management tools		
OSLC Common Architecture	OSLC Core - Essential elements, patterns for common concerns		
	Reporting - a common set of features that RESTful services should implement to enable reporting.		V1



Motivation



Enterprise context

- The business opportunities from SMARTer products and services are well understood
- However the operating environment for software and systems becomes more interdependent and complex
- Enterprises in many industries need to reduce the cost and time to achieve the target level of quality and cost for complex software and systems
- Enterprise with less mature lifecycle support consume excess time, cost and resource to manually align Software Application activities and deliverables with those related to Products
- Those who have achieved greater efficiencies from direct integration have experienced delays and excess cost for subsequent business change projects such as during acquisitions, alliance ramp up, new product lines and product line consolidation due to the specific codification of the software, products and lifecycle in the interfaces
- Enterprise seek to apply lower cost open web integration techniques, with the potential to reduce cost and time for integration, driving aspects of consistency across the enterprise with the aim of achieving new competitive from greater agility



Goal

- To reduce the cost and time to provide process support for the product and service lifecycle by reducing the complexity of enterprise integration



Strategy

- To enable aggregation and combination of heterogeneous services and information
 - Greater support for preferred representations e.g. OSLC
 - Support for aggregation and combination
 - Support for prototyping

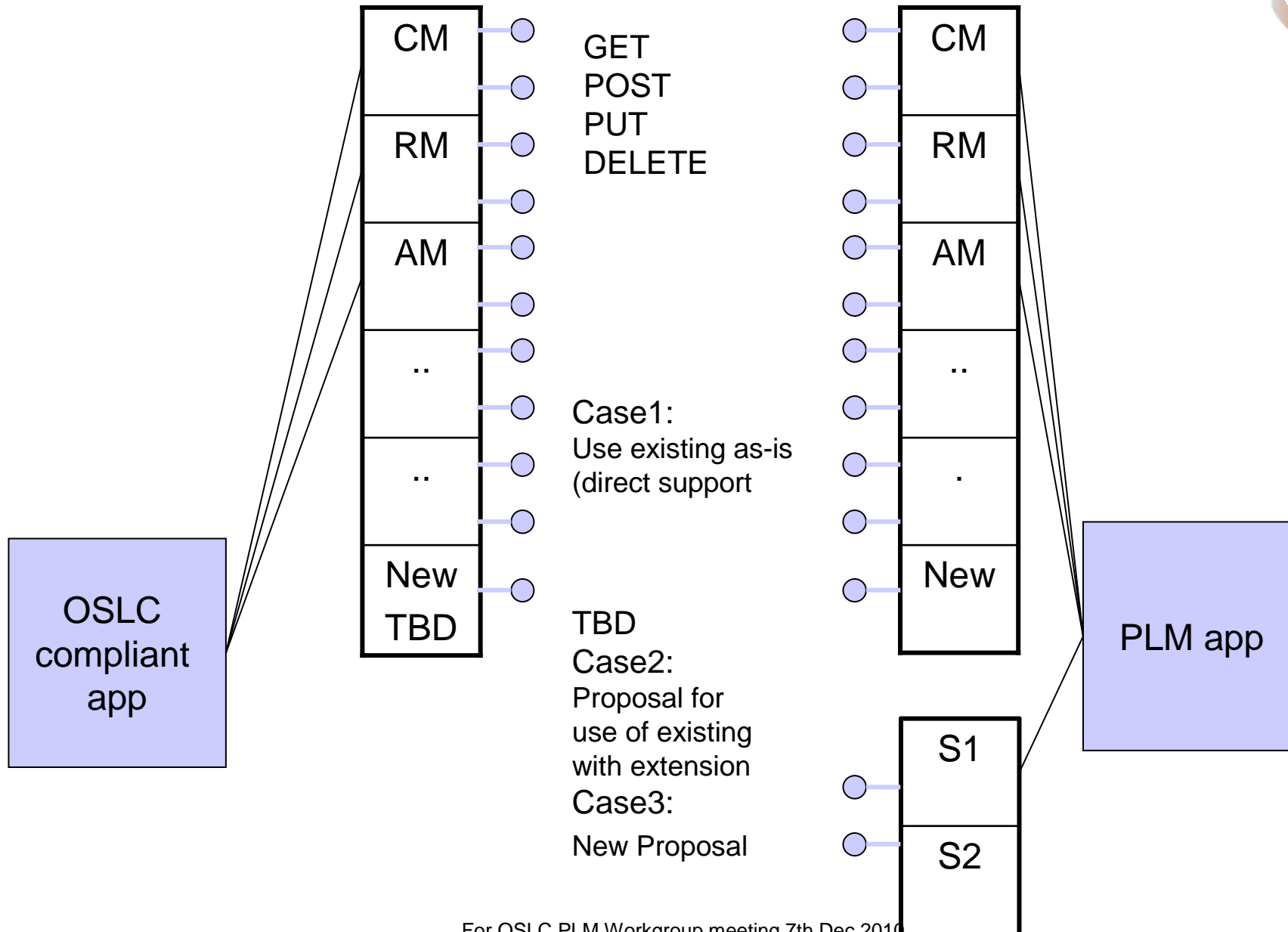


Objective

Towards the strategy of enabling aggregation and combination of heterogeneous services and information by providing greater support for preferred representations e.g. OSLC

- To make the use of the OSLC Resource model (aka OSLC Specs) viable for both OSLC and PLM Service Providers

Use of OSLC Services to support collaboration between unlike tools





PLM Reference Model

Overview of the PLM Reference Model

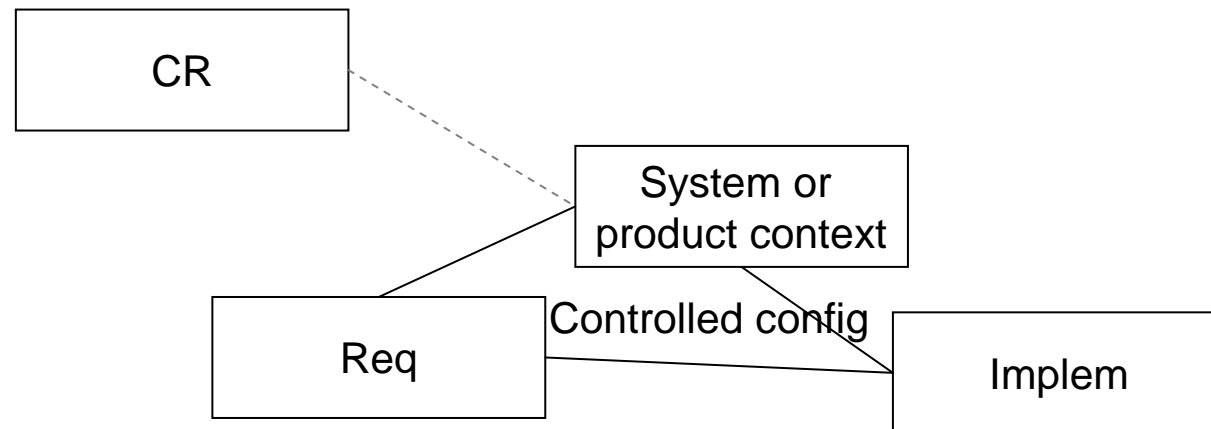


- The objective of the model is to provide a representative description of PLM information related to the scenario
- Certain standard representations have been selected to address the concerns of Scenario #1
 - SysML and STEP
 - Due to their ability to represent aspects of context, requirements and system implementation
 - Due to the motive to support modelling

The PLM reference model to support Scenario #1



- Primary concerns
 - CR – Change Request
 - Req – Requirement
 - Context – Product and System context e.g. classification, configuration, effectivity
 - Implem – Product and System implementation in models and documents





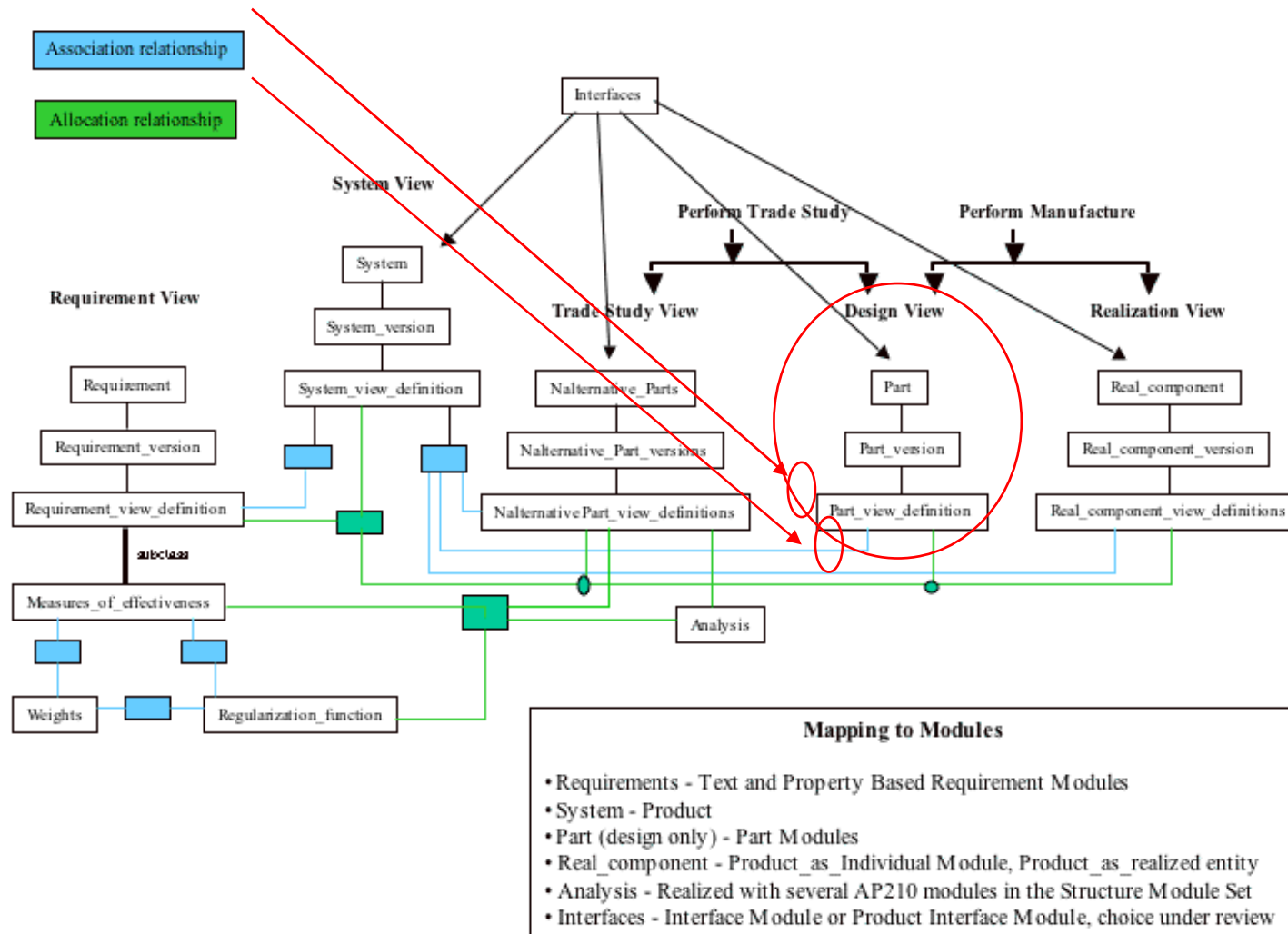
Overview of the proposed PLM Reference Model content

- Based on the OMG SysML education example with enhancements for PLM
- Main elements
 - SysML model representations
 - Requirements Diagram
 - Block diagrams
 - STEP text, xml and OWL representations
- Example instance
 - Hybrid SUV
 - Based upon the OMG SysML model with extensions to support the scenario and to provide a viable reference model

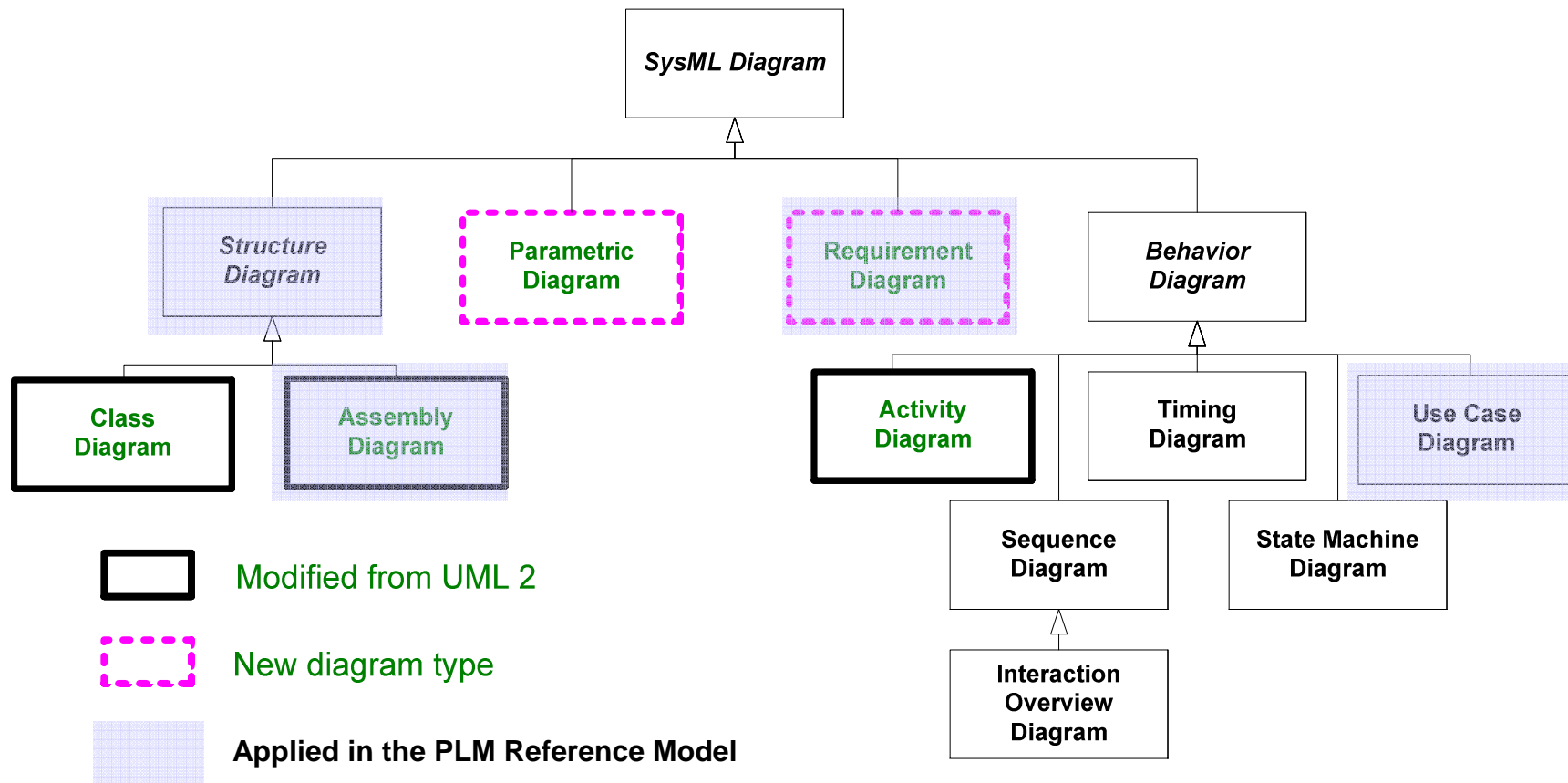
STEP supports PLM representation of System & Product decomposition



e.g. AP233



PLM Reference model can be further built out to support model driven development



Base diagram from OMG

Summary of the enhancements made to OMG Hybrid SUV example



Base OMG SysML model	Extension for OSLC PLM Reference Model	Notes
Automotive Domain Breakdown diagram	1. Automotive domain block is versioned	Container not versioned in the model (How implemented in Topcased)
Operational Use case diagram	No change	Not include on-boarding of energy e.g fuel or battery charging
Diagram HSUV Specification Requirements diagram	<ol style="list-style-type: none"> 1. Version annotation to each element – requirement, block 2. Variant requirements 3. Variant requirements associated with implementation variants 4. Variant expressions 	<ol style="list-style-type: none"> 1. Satisfies by was in separate diagram 2. There is no recognised standard for variant expressions
HSUV Breakdown (Block) diagrams	<ol style="list-style-type: none"> 1. Versioning at the block level 2. Block level annotation for the xml extract 	
Power Sub-system Block Definition Diagram (BDD) & Internal Block Diagrams (IBD)	<p>Three variants with appropriate decomposition</p> <ol style="list-style-type: none"> 1. Version annotation to each element – requirement, block 2. Alternative implementation of the Power Control Unit block 	The IBD is named Combined Motor Generator
Power Control Unit Breakdown diagrams	<ol style="list-style-type: none"> 1. Break out the Calibration and Software to be part of the assembly 2. Version annotation to each element – requirement, block 	To reflect current practice. Compatibility not addressed e.g. effectivity or explicitly



Note for future meeting

- The PLM Reference model is proposed for adoption by the OSLC PLM Working Reference as a 1.0



Analysis



Objective

- To identify the ability of the current Spec to support the scenario
- To identify extensions of the current Spec to support the scenario



OSLC SPEC analysis

- Analysis of existing specs
 - Initial focus on Core and RM
- Traceability scenarios from RM team
 - Link
- Identify “same as” or “equivalent” using OWL
 - Capture gaps and issues
- Investigate possibility to make a basic transformation
- CM, AM next iteration

What might the output of analysis look like ?



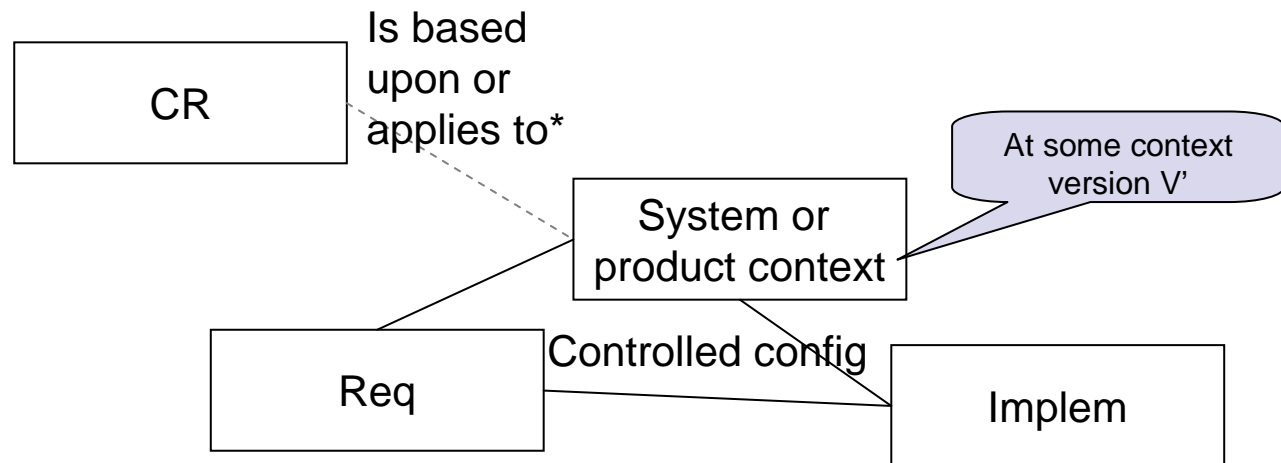
- A set of statements about entities and relationships compared across the OSLC Spec and the PLM Reference Model
- A list of resources with examples of GET, PUT, POST
- A data graph of entities and relationships with resources identified
- A taxonomy for entities, relationships and resources showing origin or equivalence

Summary of the scenario by way of the key business entities & their relationships

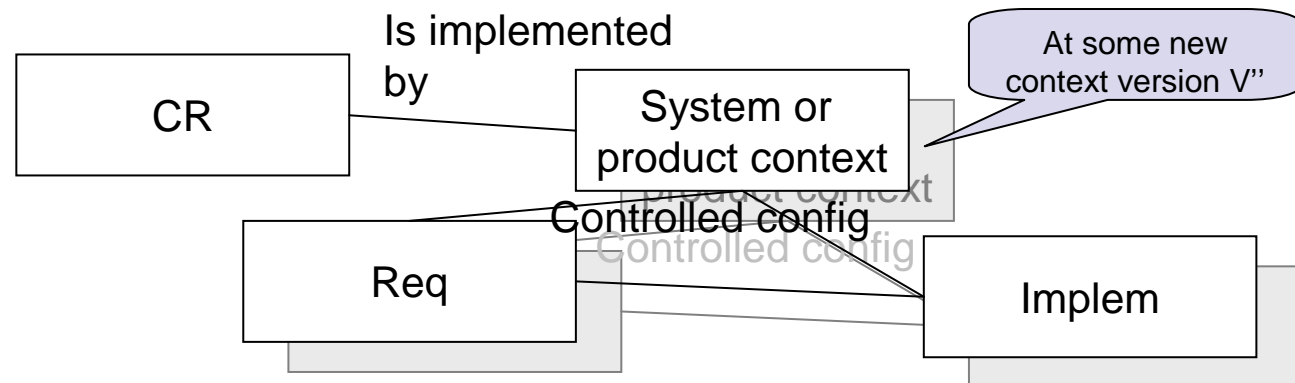


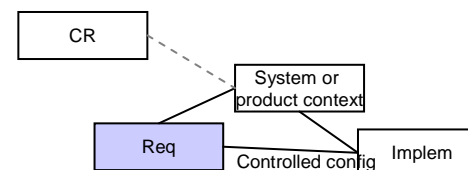
Pre-condition (Before

* Assuming basic triaging has been done prior to the start of the scenario




Post-condition (After

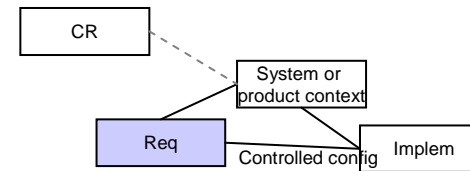




Requirement 1 of 2

Question	OSLC Answer  Rich Text Document	PLM Reference model Answer
How is a requirement defined ?	Requirement is a type. A Requirement resource has a shape which prescribe a set of mandatory attributes	Three primary entities of Requirement, Requirement version and Requirement view definition
How is a Requirement uniquely identified	Globally by a URI	By an id within a context
What determines the rules for representing a requirement ?	Meta-model rules (RDF) Resource shape	FILE_SCHEMA (('AP233_SYSTEMS_ENGINEERING_ARM_LF'));
What is the visibility of the requirements description ?	Global	? Header includes the names, time stamp, org
How is requirements meta-data defined ? E.g. organisational ownership	Title is mandatory plus optional properties	? ID, name and description ? Validate the usage of the Req view definition (effectivity only ?)
How is the relationship between collection and other resources ?	A special named relationship properties is “uses”	For requirements: #1230=REQUIREMENT_COLLECTION_RELATIONS HIP(", 'isComposedOf', ", #720, #1220) “the descriptor isComposedOf is optional”
How is the relationship between requirements and other resources defined ?	Named relationship properties available for use	#10700=REQUIREMENT_VIEW_DEFINITION_RELATIONSHIP('10700', 'DeriveReq', 'DeriveReq1', #3220, #8720);
How to version a requirement ?		3 level structure with the version defined through the REQUIREMENT_VERSION

Requirement 2 of 2



Question	OSLC Answer	PLM Reference model Answer
<p>How are groups of Requirements organised ? Identifying as a group</p> <p>Treat as a group e.g. Approve, implement, assign to a block or an organisational unit as group e.g. a black box approach “satisfiedby” Short hand</p>	<p>URI of the Requirements collection Dcterm: “Selected requirements for HSUV release XYZ”</p> <p>Can have a collection of collection etc A collection is looser grouping of elements that happen to have a common locator</p> <p>OSLC lacks the explicit “isComposedBy”</p>	<p>SysML package holds the Requirements as a container (as opposed to a collection)</p> <p>Any sub-tree denotes a group</p> <p>The Requirements are defined in isolation within the scope of a package and then associations are made buy way of e.g. #4530=REQUIREMENT_COLLECTION_RELATIONSHIP(“’,’isComposedOf’,”,’#820,’#4520); SysML does not have an external class for composition</p>
<p>Identify interdependency</p>	<p>Uses: URI (not titles) (Uses is not well defined e.g. to mean “isComposedby” Uses is a reference to another resource (as opposed to the strong decomposition inherent in UML)</p> <p>Open set (not supported) Tracelink here</p> <p>e.g. for an external link</p>	<p>#36=EXTERNAL_CLASS('http://www.omg.org/spec/SysML/Current/SysML-profile#DeriveReq','DeriveReq','The "derive requirement" relationship relates a derived requirement to its source requirement.',#34);</p> <p>#38=EXTERNAL_CLASS('http://www.omg.org/spec/SysML/Current/SysML-profile#Refine','Refine','The refine requirement relationship can be used to describe how a model element or set of elements can be used to further refine a requirement.',#34);</p> <p>#1230=REQUIREMENT_COLLECTION_RELATIONSHIP(“’,’isComposedOf’,”,’#720,’#1220) “the descriptor isComposedOf is optional and arbitrary isComposedOf is derived from the decomposition paradigm of SysMI modeling.</p>

How can a PLM hosted requirement be made available via an OSLC Resource model



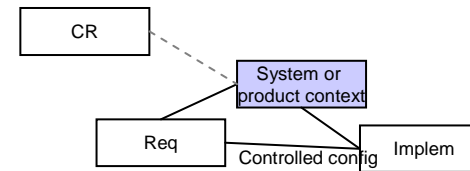
- A PLM tool makes available a view of the current state of a selected PLM Requirement resource
 - E.g. GET R1.2.1
 - E.g. GET R1.2.1. Version 2
 - E.g. GET R1.2.1, Version=* or History
- Current state is that available from the service provider which instantiates the OSLC Resource model
 - Id
 - Type = Req
 - State e.g.
 - Date of last change
 - Version
 - Lifecycle state
 - Approval state
 - Traceability status



Versioning discussion

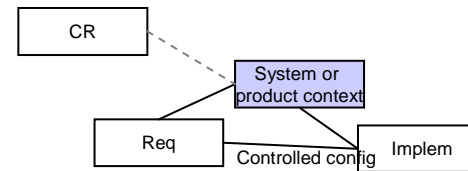
- For an entity like Requirement a version is a qualifier
 - To signal an update
 - To denote a derivative
 - To denote an alternative
- Query / view of Requirement version is just another example of an attribute or meta-data
 - E.g. date of last changed
- Version identities are just strings unless separate (external) control of version sequence or name structure is provided by an app
 - i.e not controlled by a schema
- Versions need not be linear (i.e. branch and merge, multiple valid versions in existence)
 - OSLC supports query of immediate predecessors and successors
- Versioning is a process
 - Make / spawn a new version

Context 1 of 2



Question	OSLC Answer	PLM Reference model Answer
How is the root context defined ? (Using a tiering concept for context)	Service Provider (may not be the authority)	STEP Filename and date created (as a snapshot for data exchange) Organisation ? Have parts been assigned to blocks ?
What constrains the context description ?		FILE_SCHEMA (('AP233_SYSTEMS_ENGINEERING_ARM_LF')); #20=ACTIVITY_METHOD('XSLT_Extract','XSLT Extract of STEP Part 21 Data File from Topcased SysML XMI','For initial creation of dataset');
In what context is a requirement valid ?	Valid everywhere Qualified by associations e.g. Query project name in a WI	See above

Context 2 of 2



Question	OSLC Answer	PLM Reference model Answer
How is project, product or system context defined ?	e.g. a WI within a Jazz project as a proxy for a new System release ? e.g. a name or property of a baseline / cfg ?	Identity, name and a version in reality this as an entry point to a config #23600=SYSTEM('23600','HybridSUV','HybridSUV System'); #23610=SYSTEM_VERSION('1','HybridSUV System Element Version',#23600);
How is product & system coding and classification supported ?	Not available except by tags or attributes to a thing – tags or attributes (see note below about requirements specifically)	Use the PRT or PRODUT or SYSTEM structure to define a taxonomy and then create associations
How is a requirement associated with a project, product or system coding & classification ?	Through a Requirement collection Query of link identified an external resource. E.g. a WI within a Jazz project for a new System release Or use attributes to explicitly hold tags e..g Rational, RM, Doors10 using dterms:subject (today changing a tag changes a requirement) (lose ability to look at history as not separately maintained)	#114700=REQUIREMENT_ASSIGNMENT('114700','Satisfy3',#1820,#30120); Where the associated reference already sits in a system structure



Other Qs

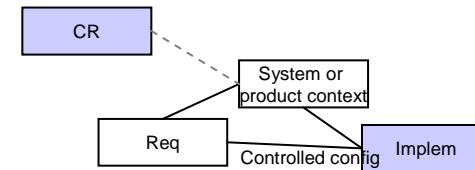
- How are changes to requirements identified ?
 - Versions
 - History
- How is a requirement located ?
 - Via a unique id – unique in a context
 - One from another
 - From a Product or System
 - From an element of a Product or System (Implementation)
 - By version
 - By relationship
 - IsComposedOf
 - IsDerivedFrom
- How is the scope of an identity defined ?
 - In SysML ? Open
 - In AP233 ? Done
- How can groups of requirements be organised ? Done
- How is information e.g. an associated document related to a requirement associated
 - with a requirement ?
 - with an organised group of requirements ?
- How is an implementation associated with a requirement ?
 - Version
 - Organised Group
- What is the context for a requirement ? Done



Discussion on direction

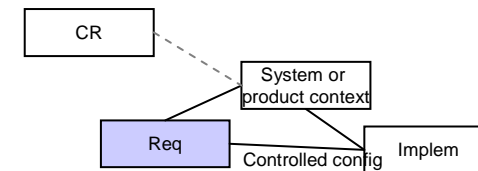
- Option 1:

- Complete approach for CR, Implementation and then the remaining associations



- Option 2:

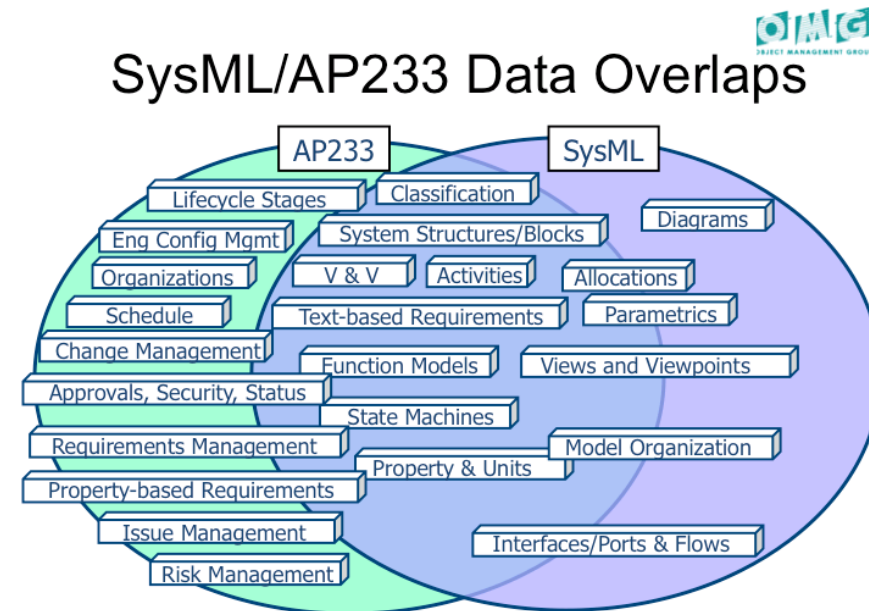
- Focus on the Requirements associations with CR and Implementation and drive through to more detailed mapping and then the first set of recommendations



Opportunity for fine-grained alignment with OMG



- OMG sponsored a mapping between SysML and AP233
 - Started in 2009
 - Last update 7/10
- Detailed OMG asset of side by side comparison
 - Use cases
 - Requirements
 - Blocks
 - Value properties
 - Activities
 - Constraint blocks
 - State machines
 - Packages and metadata
- XML/xmi assets for requirements and system structure
 - Path to RDF/OWL
- http://www.omgwiki.org/OMGSysML/doku.php?id=sysml-ap233:mapping_between_sysml_and_ap233#interactions_mapping



E.g. SysML to AP233 mapping for Requirements



SysML	AP233
Requirement	Requirement_view_definition → Requirement_version → Requirement
Containment	Requirement_collection_relationship
Allocate	View_definition_relationship + Classification ('Allocate')
Satisfy	Requirement_satisfied_by
Verify, Refine	View_definition_relationship where one end must be a requirement
Copy, Derive	Requirement_view_definition_relationship + Classification ('Copy', 'Refine')
Trace	View_definition_relationship + Classification ('Trace')
Trace between Requirements	Tracing_relationship
Text	Requirement View Definition ← Single_property_is_definition → Property_representation → Representation → String_representation_item



Next steps

- Detailed alignment of the RM Spec
 - tie out to the AP233/SysML
- Identify resources and how to make associations with the PLM Reference model
 - Part & part version as resources
 - Decision on Part view definition as a resource to support structure
- Write test cases for typical activity – Add, Delete, add associations
- Documents recommend to implement
 - Summarise findings and conclusions
 - Basis for prototyping
- Validate that the RM finding are indeed sufficient for CR
 - Primarily associate with version etc



Relevant OSLC Links

- Systems Engineering Scenario #1

- Systems Engineer Reacts to Changed Requirements
- <http://open-services.net/bin/view/Main/PLMSystemsEngineeringScenarioSystemsEngineerReactstoChangedRequirements>

- Proposed PLM Reference Model

- <http://open-services.net/bin/view/Main/PLMReferenceModel>

- Analysis of OSLC Specs and PLM Reference Model

- <http://open-services.net/bin/view/Main/OSLCSpecPLMReferenceModelAlignment>

- Reference model contribution

- <http://open-services.net/bin/view/Main/PLMScenariosGm>

- Existing Specs

- <http://open-services.net/bin/view/Main/WebHome>





For more information

- Open Services for Lifecycle Collaboration
PLM Workgroup

- <http://open-services.net/bin/view/Main/PlmHome>

- Contacts

- Dr Rainer Ersch, Siemens

- Gray Bachelor, IBM

- Mike Loeffler, GM