Flat Versus Hierarchical Information Models in PLM Standardization Frameworks

Sylvere Krima, Joshua Lubell

Flat versus Hierarchical Information Models in PLM Standardization Frameworks

Sylvere Krima and Joshua Lubell

National Institute of Standards and Technology
Gaithersburg, Maryland, USA
{krimas, lubell}@nist.gov

Abstract. Smart manufacturing requires digital product data to be shared and exchanged among numerous engineering applications and information systems. But no single product data standard can satisfy every integration scenario. Customizable standardization frameworks for Product Lifecycle Management (PLM) attempt to address this problem by allowing users to add new information structures to an existing data model in a controlled manner. A PLM information model may be either flat or hierarchical. We discuss two approaches. One is based on ISO 10303-239 as an exemplar for customizing flat models. The other is based on Open Application Group Integration Specification (OAGIS) as an exemplar for customizing hierarchical models. We evaluate the two approaches and observe that the type of model strongly influences how well the PLM standardization framework meets each evaluation criterion, and that the best choice is use-case dependent.

Keywords: Open Application Group Integration Specification, Product Life Cycle Support, Reference Data Library, Core Components, information modeling

1 Introduction

Smart manufacturing, which is information-intensive and requires advanced communication and network technologies [1], requires that digital product data be shared and exchanged among numerous engineering applications and information systems. Standardized information models for Product Lifecycle Management (PLM) aim to help manufacturers meet these requirements. Terzi et al. define PLM as "a product-centric, lifecycle-oriented business model," enabled by information technology, in which "product data are shared among actors, processes and organizations in the different phases of the product lifecycle for achieving desired performances and sustainability for the product and related services." [2] Manufacturers are under pressure not only to bring to market ever more complex products but also to bring them faster and cheaper. Doing so requires product information that can be used by many different participants in the product realization

process. To help meet these requirements, standards bodies, industry groups, and consortia have standardized a number of product information models targeted to a particular integration scenario. Examples include the Object Management Group (OMG) PLM Services [3, 4] and the ISO 10303-242 Business Object Model [5, 6], both of whose scope is limited to design engineering data.

To remain competitive, manufacturers are using PLM data to optimize their proprietary business processes throughout the product lifecycle. To maximize the benefits of PLM, manufacturers must map their proprietary processes to the standardized PLM models. Unfortunately, due to the complexity of today's products and processes, no single standardized PLM model can satisfy every use case. Moreover, the ever-evolving product data requirements pose a serious technological challenge. Why? Because currently, it is not possible to define, in advance, information structures flexible enough to meet such changing requirements [7]. Developing such structures requires a new approach that allows users to integrate new information structures into an existing information model. The need for that kind of capability has led to the development of frameworks that software implementers can use to customize interoperable PLM standards.

To enable the creation of such standards, a PLM standardization framework must include:

- An initial, generic information model capable of representing a broad spectrum of products and related industrial data. This model's semantics are too abstract to be used directly for a real-world scenario, but are intended to be consistent with any scenario-specific application [8].
- A methodology for customizing the initial information model to meet the requirements of a particular scenario-specific use case.

The initial information model may be *flat* or *hierarchical*. Although one can conceive of an initial information model having both flat and hierarchical parts, we know of no existing PLM standardization framework with this characteristic. Our definitions, adapted from Zimmermann [9], are as follows. We define a flat information model as having objects that are accessible from one another and are arranged as peers. For such models, the framework provides a means for referencing external classification taxonomies that refine the meaning of concepts in the model. The reference ensures compatible exchange forms, since all implementations share the underlying model (see left hand side of Fig. 1). Moreover, as the figure illustrates, constructing a model meeting the requirements of a specific business process requires two operations: identifying the appropriate subset of the initial model and referencing the appropriate business context information from the external information source.

We define a hierarchical information model as a tree-like structure where objects can contain other objects or collections of objects. For such models, the framework provides mechanisms for building extensions of the underlying model by adding new concepts and relationships to the initial information model. Creating an extension is usually straightforward since it does not require any additional modeling methodologies or implementation methods beyond those used to create the initial information model. However, because extensions add new concepts and relationships to the initial model, different implementations of the same extensions may be incompatible – when, for example, independent organizations doing similar work make different modeling choices. The right hand side of Fig. 1 shows a "business document" exchange form as

an aggregation of data elements. The business document is analogous to the union of the information model and business context shown on the left hand side of Fig. 1. The extension shown implements the addition of a new substructure that reuses an existing data element and includes additional, business-context information.

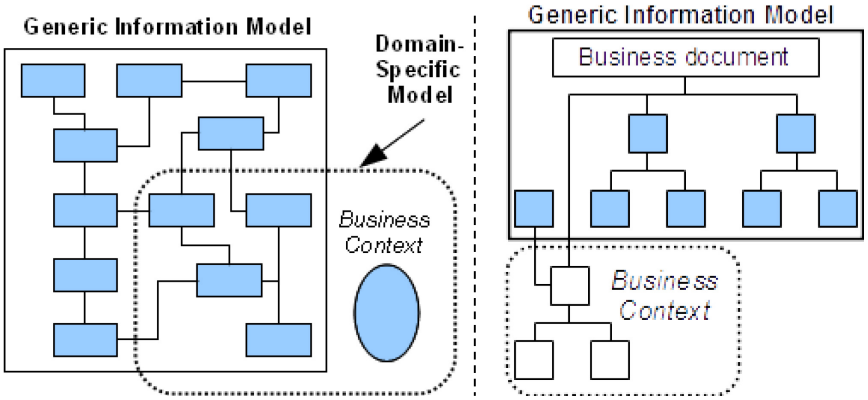

Fig. 1. Customization of flat (*left*) and hierarchical (*right*) models.

In this paper, we compare the flat and hierarchical PLM standardization frameworks, with a focus on customizability. We choose the Product Life Cycle Support (PLCS) framework [10] for developing data exchange specifications using ISO 10303-239 [11] as an exemplar of the flat approach[1]. We choose the Open Application Group Integration Specification (OAGIS) [12] as an exemplar of the hierarchical approach. In the following sections, we provide an overview of the customization approaches used in flat and hierarchical PLM standardization frameworks, justify our choices of PLCS and OAGIS as exemplars, evaluate the two, and make two observations. First, the type of model strongly influences how well the framework meets each evaluation criterion. Second, the best choice is use-case dependent. We note that this paper's focus is limited to customization. Other important characteristics of PLM standardization frameworks, such as the quality of the initial information model and impact of changes to the initial information on existing implementations are not discussed.

2 Related Work and Existing Customization Approaches

A key goal of PLM is to align engineering processes, such as design and manufacturing, with more business-focused activities such as sales, inventory control, and enterprise resource planning (ERP). In this section, we review previous efforts in classifying PLM standards and in harmonizing product information with electronic business ("e-business") information standards, and then describe the "Reference Data Libraries" and "Core Components" customization approaches.

[1] The PLCS framework, introduced in section 2, is not part of ISO 10303-239. However, both are commonly referred to as "PLCS" (a source of confusion).

2.1 PLM Standards Landscape and Harmonization

The PLM standards typology of Rachuri et al. [13] includes Type Two and Type Three standards. Type Two standards define information models specific to a domain of discourse. An example of a Type Two standard is the Systems Modeling Language (SysML) [14], a graphical language intended for (but not limited to) use in Systems Engineering applications. No single Type Two standard can represent "all of PLM." In our context, "all of PLM" includes all information pertaining to products, processes, and services that make up the entire product lifecycle – beginning with detailed design and ending with disposal. Type Three standards are architectural frameworks, which are standards for creating families of interoperable Type Two standards. The PLM frameworks we evaluate in this paper are Type Three standards.

Paviot et al. [15] determined that the ISO 10303-239 (Product Life Cycle Support, PLCS) [11] Type Two standard is – unlike many other Type Two standards – customizable by design. Since customization is inevitable, PLCS must be tailored to fit both the scope and the granularity of a specific PLM domain. This observation is critically important because, without such flexibility, a developer of a Type Two standard must choose between scope and granularity. The flexibility of PLCS enables the ISO 10303-239 information model – a Type Two standard – to serve as the foundation of a Type Three PLM framework. We choose PLCS as an exemplar flat framework (i.e., one with a flat information model) both because of its flexibility and because the PLCS framework includes a methodology for customization of the ISO 10303-239 information model using Reference Data Libraries, discussed in 2.2.

Successful deployment of PLM requires both product metadata standards and e-business standards [4]. Fiorentini and Rachuri [16] investigated methods for sharing PLM data among engineering and business software applications. Their research focused specifically on the OMG PLM Services [3], a Type Two standard. Fiorentini and Rachuri selected OAGIS [12] as the e-business standard with which to harmonize the OMG PLM Services. OAGIS is a critical standard for application-to-application and business-to-business integration [17]. By successfully mapping portions of the OAGIS Engineering Change Management concepts to OMG PLM Services concepts, their research demonstrated the feasibility of harmonizing product design data standards with OAGIS. Since the PLCS scope is a superset of the OMG PLM Services, and both have information models based on ISO 10303 [4], it follows that (1) portions of PLCS and OAGIS can be harmonized and (2) portions of PLCS implementations and OAGIS implementations can be made interoperable with one another. Based on this conclusion, as well as the broad scope and widespread adoption of OAGIS relative to other e-business frameworks [18], we choose OAGIS as the exemplar for e-business frameworks with a hierarchical model.

2.2 Customization and Reference Data Libraries

The Reference Data Library (RDL) approach aims to enable controlled customizability without sacrificing breadth. A RDL is an externally-defined, controlled vocabulary for specializing concepts in an underlying schema [19]. For example, consider a concept `Person` defined in a generic information model and the `Person`'s

specialization in an accompanying RDL. The RDL specifies a taxonomy that enables specialization of a `Person` instance as a `Customer`, an `Employee`, or other concepts. The RDL approach assumes the existence of an underlying information model with a wide scope – the generic information model. Such a model, however, is too abstract to be verified by subject matter experts associated with a specific integration scenario. That makes the model difficult to implement, and use.

To overcome these difficulties, the RDL approach allows users to work with any subset of the original information model by defining scenario-specific subsets of the underlying information model. These subsets use *templates* to define how information-model entities and their attributes will be instantiated. A template is a predicate with a signature specifying arguments and their types [20]. Templates are critical elements of the RDL approach because they apply an integration scenario and an externally-defined controlled vocabulary directly to the underlying schema. A template also may invoke other templates, providing a means of modularizing and combining integration patterns. Templates should not be seen as a means of customizing an information model, but rather as a way of customizing its use.

The PLCS framework, as proposed by the Organization for the Advancement of Structured Information Standards (OASIS) PLCS Technical Committee (TC) [21], employs the RDL approach. The underlying information model for PLCS is ISO 10303-239. The TC has developed guidance for defining RDLs using the Web Ontology Language (OWL) [22] and templates as SysML block and parametric diagrams[2] [3].

2.3 Extensions using Core Components

The Core Components approach to extension is based on the Core Components Technical Specification (CCTS) [23] standard, which provides the foundation for several XML-based e-business standards, including OAGIS. CCTS-based e-business schemas use a hierarchical modeling pattern. Customization involves 1) identifying the relevant components, 2) associating them with a selection of the components contained in the generic document, 3) interpreting the components to their business-specific use, and 4) selecting from the generic fields those fields that can represent the business-specific information units that describe those components. Customization can also add components to the document, or add fields to a component. This process creates new artifacts, based on a business-specific terminology and representation of the information. This customization is known as *extension* because users can extend the initial artifacts, in addition to restricting some and leaving others unused.

Revisiting our previous RDL example, we specify an extension by applying the four steps from the preceding paragraph. First, we determine that (1) a `Customer` concept is necessary to represent our business information requirements. We next determine

[2] This guidance is not an OASIS standard, but rather a Committee Specification that has been approved only by the members of the TC and not by the entire OASIS membership.

[3] The PLCS TC uses SysML for this (atypical) purpose because they found the SysML notation useful for representing templates and their relationships with other templates. They also wanted to take advantage of existing SysML software tools.

that (2) `Person` is the CCTS concept closest to `Customer`. We then determine that (3) a `Customer` needs, in addition to a `Person`'s fields, an `id` field. Finally, we (4) extend this component by adding an `id` field.

OAGIS facilitates integration of disparate business systems by defining a standardized architecture for representing Business Object Documents (BODs), the messages to be exchanged. OAGIS has historically been solely XML-based. OAGIS 10, the newest version of the standard, encourages a more model-driven approach, allowing for alternative methods for specifying and implementing BODs [1][4]. BOD data contains the message content, represented as a verb-noun pair. The verb identifies an action performed on a noun. The noun identifies the business-specific information that is exchanged. Nouns are made of extensible building blocks called components, as in CCTS.

Despite the large number of nouns and components, OAGIS BODs by themselves cannot support every possible message exchange. Therefore, OAGIS provides a variety of customization mechanisms. Component Open Extension, introduced in OAGIS 10, is a simple mechanism that does not require any changes to the OAGIS XML schema definitions. Overlay Extension, an XML implementation of the CCTS customization mechanism, provides OAGIS users with more flexibility – but requires modifications to the BOD schema definitions.

3 Use Cases and Evaluation Criteria

In this section, we discuss two use cases for PLM standards: data exchange and data sharing. We then identify two PLM-related requirements and assessment criteria to evaluate the PLCS and OAGIS frameworks with respect to the use cases. *Data exchange* enables the transfer of information from one processing entity to another. Exchange requires translating that information from the source schema into an instance of a target schema. Successful exchange means that the translation must reflect the source information as accurately as possible [24]. Because it typically involves few, if any, time constraints, data exchange is often considered to be a batch operation. *Data sharing*, on the other hand, requires real-time access to the information source [4]. Data sharing's technical requirements differ from those of data exchange in that the information provider must expose data requested by the consumers on demand. A PLM standardization framework can provide the pieces needed to standardize interfaces for product data sharing.

Having discussed our proposed use cases, we now consider two capabilities a PLM standardization framework needs to best support them. These capabilities all facilitate interoperability, which Ray and Jones [25] define as the ability of disparate software applications to share digital technical and business data efficiently and without errors. Chen et al. [26] developed a more expansive characterization enumerating the following interoperability concerns: data, services, processes, and business. As

[4] However, since the developers of OAGIS have not yet provided customization guidance for implementing such an approach, this paper's discussion of OAGIS extension is XML-specific. Non-XML BOD extension guidance is planned for future OAGIS versions.

discussed in 2.1, PLM frameworks are Type Three standards used to create families of interoperable Type Two standards. PLM framework customization methods must address interoperability concerns while also allowing family members' data models to retain business-specific terms and definitions.

Controlled customization is a process intended to limit the introduction of inconsistencies and to facilitate interoperability. Data exchange requires controlled customization to maintain data quality during translation. Controlled customization limits the possibility of introducing inconsistency – and breaking interoperability – when tailoring an initial information model for implementation. Controlled customization accomplishes this goal by restricting the set of potentially customizable concepts from the initial information model to those that minimize the likelihood for inconsistency. Since a PLM standardization framework's initial information model is very large, and never used as a whole, customization is a necessary and often complex process. Controlled customization requires defining a subset of the original concepts that can be customized. Defining this subset requires an understanding of the information requirements of the downstream lifecycle processes that will use the results of the customization.

Unlike data exchange, data sharing happens within the scope of a specific context: the business transaction the data sharing supports. Because of data sharing's ephemeral nature, guarding against long-term inconsistencies is not an issue. Therefore, data sharing does not require controlled customization. It does, however, require the development of software interfaces specific to a particular business domain. Such interfaces can be specified as a collection of standardized *business objects*.

Business objects, when combined with standards for product metadata and recent advances in service-oriented architecture (SOA) technology, create new integration possibilities [4]. Each business object encapsulates all of the product information in a specific transaction. A business object model [27] results from an implementation method that uses a domain-specific, transaction-oriented vocabulary, which hides the complexity and reduces the granularity of the underlying information model.

A business object instance automatically instantiates the underlying generic concepts and their relationships. These instantiations result from an invertible mapping between the business object model and the underlying information model. This mapping is defined unambiguously and is computer interpretable, enabling interoperable business object model implementations.

Fig. 2 summarizes the dependency relationships discussed in the preceding paragraphs between use cases and PLM standardization framework capabilities.

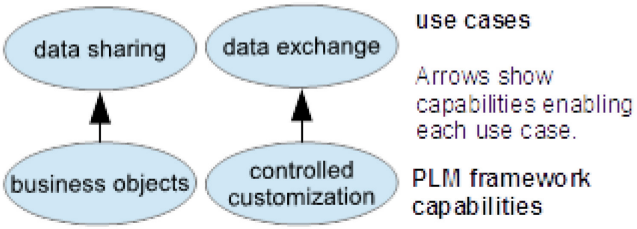

Fig. 2. Use cases and capabilities.

4 Evaluation of PLCS and OAGIS

We now assess the PLCS and OAGIS frameworks with respect to their business object creation and controlled customization capabilities. We favor native support of a capability by the information model because lack of native support often results in a new layer of complexity, in the form of implementation-specific guidance or a parallel information model. For illustrative purposes, we use representation of a Bill of Material (BOM) as a recurring example. Our example uses the OAGIS `BOM` noun and the PLCS `PhysicalBreakdown` template. ISO 10303-239 defines a physical breakdown as "the partitioning of a product into a set of related physical elements so as to form explicit, parent-child views that comprise the product elements."

The PLCS framework uses templates for encapsulating ISO 10303-239 concepts into business objects. Templates are used in conjunction with the Platform Specific Model (PSM), an implementation model derived from the ISO 10303-239 information model. The PSM is available from PLCSlib [10], an online environment created for the development and use of PLCS templates. Templates are defined using SysML diagrams[5].

Unlike PLCS, OAGIS has a native mechanism for representing business objects, namely the OAGIS BODs. Because BODs are composed of nouns representing business objects and verbs representing actions performed on business objects, the OAGIS BODs are well-suited for representing engineering and business processes[6] [4]. The OAGIS framework follows the CCTS methodology and uses standardized components as building blocks for defining BODs. BOD developers extend low-level components to support domain-specific information, combining them together to create domain-specific objects, the OAGIS nouns.

For example, consider the OAGIS `BOM` noun. As shown on the left hand side of Fig. 3, this noun comprises four elements: a header (`BOMHeader`), the item data (`BOMItemData`), the product option(s) (`BOMOption`) and classifiers of the product option(s) (`BOMOptionClass`). `BOMHeader` is partially expanded to show child elements used in an extension. In this figure, the BOM is a set of part descriptions, where each part is identified by a `BOMItemData`, and the additional elements provide specific information about the part as it is used in this structure.

The right hand side of Fig. 3 shows how a BOM might be represented using the PLCS PSM. The thick arrows indicate cross-references between PSM objects. `ExchangeContextClassLibrary` points to a RDL. `ExternalOWLClass` points to an RDL class. The rest of the PSM objects result from following the guidance specified in the PLCSlib `PhysicalBreakdown` template. The other PSM objects represent generic concepts. These concepts include cross-references to other `ExternalOWLClass` objects (omitted from Fig. 3 to reduce clutter).

[5] We refer readers to PLCSlib for details on the PLCS template methodology.

[6] Because the BOD schemas contain many weakly-typed optional elements, OAGIS users are encouraged to add constraints to their BOD implementations. The OAGIS standard provides guidance on adding constraints to XML schema definitions.

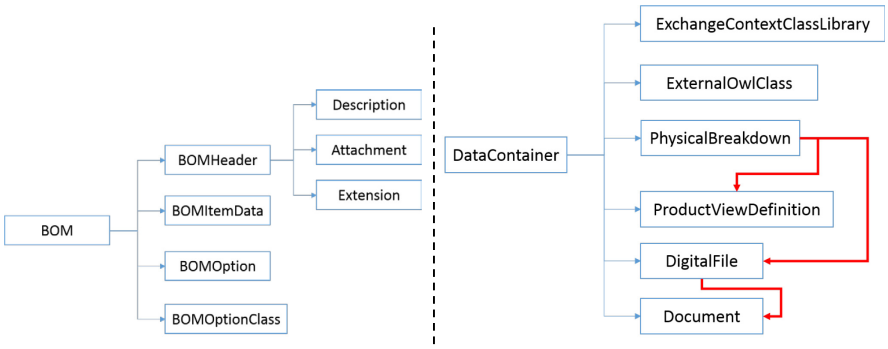

Fig. 3. BOM represented in OAGIS (*left*) and using PLCS PSM (*right*).

The flat structure is a consequence of the RDL approach, which requires that the initial information model be abstract and that business-specific classification be done in an external class library. This external classification gives rise to fewer composition relationships, but more association relationships. For example, the OAGIS `BOM` noun is composed of a header, item data, and information regarding options. However, the PSM `PhysicalBreakdown` object is too abstract for the initial model to make any assumptions about its composition. In PLCS, as in any other RDL-based framework, a less-abstract concept such as a BOM is defined in a RDL rather than in the initial information model. Additionally, the BOM object must be linked to the elements of its composition, which are modeled as independent (external) objects. As we will discuss in section 5, information model flatness in the RDL approach has advantages that could offset the impact on business object complexity.

To achieve controlled customization, RDLs customize concepts in the initial information model using information external to that model. Instances of customizable concepts contain links to external references through a property designed specifically for the purpose. A RDL-based framework, such as PLCS, controls customization at the information model level by providing only a limited set of concepts with the "external references" property.

Extension is a customization method that enlarges the initial information model to support new requirements. Extensions introduce new concepts and relationships. Controlling the use of extensions requires policies limiting extension to specific parts of the initial model and prohibiting extensions elsewhere. To do so, a CCTS-based framework such as OAGIS must control editorial rights of its information artifacts. This control cannot be done at the information model level; instead, it must be done at the implementation level using an implementation-dependent method. OAGIS specifies its information artifacts as XML schemas spanning a directory tree that contains multiple directories and files. OAGIS allows only certain definitions in certain files to be modified. Moreover, OAGIS provides XML-specific rules on how to specify the modifications.

To summarize, PLCS supports specialization using external references and controls customization by having hooks in the PSM for pointing to an RDL. OAGIS supports extension but not external references. It controls customization through XML-specific and directory structure-specific policies that allow only certain concepts to be extended.

Because the PSM natively controls customization, controlled customization in PLCS is not tied to a specific implementation method, as is the case with OAGIS.

Table 1 presents our evaluation results:

Table 1. Summary of evaluation results.

	RDL based	Core Components based
Business Objects	Represents business objects as templates, in the form of SysML diagrams in the case of PLCS.	Hierarchical XML element representation is naturally amenable to creation of business objects.
Controlled Customization	The information model is designed in a way such that only a certain set of its entities can be specialized within the RDLs.	Lower level concepts cannot be extended, but higher-level concepts can be. OAGIS provides a variety of XML-based extension methods.

5 Native Support as a Metric for Framework Capabilities

Based on our assessment in section 4, we observe inherent tradeoffs that depend on whether the PLM standardization framework's information model is flat or hierarchical. If the model is flat, as is the case with the RDL approach, then it controls customization directly. A flat information model limits the possibilities for redundancies or inconsistencies when exchanging data. For example, a flat file will not have two real-world products (individuals) with the same product model each containing a separate copy of that product model in their information content. In PLM, the individual and the product model are both first class objects. Therefore, a flat representation is advantageous for keeping the product model metadata and the individual model's metadata separate from one another. However, a flat information model is not natively a business object model. To support a business object model implementation, additional guidance is needed. In the PLCS framework, the template methodology provides this guidance, but it increases the complexity of standards development and deployment. Complexity increases from the additional difficulties in the creation of business object models.

On the other hand, if the PLM standardization framework's information model is hierarchical, as is the case with CCTS-based e-business frameworks, then it is natively a business object model. A hierarchical information model supports business objects "for free" because they require less cross-referencing. However, in the e-business frameworks, the individuals rather than their product models are the primary focus. As a result, redundant or inconsistent product models are possible. Also, the information model does not natively control customization, so additional implementation-specific, controlled customization methods must be provided. For OAGIS, these methods include the Component Open and Overlay Extensions.

Terzi [2] observed that product development and ERP, which are both within the scope of PLM, have fundamentally different information requirements. Product development is iterative, recursive, and requires a detailed and precise representation of the product model. The ISO 10303-239 information model and its PLCS PSM derivative are based upon the concepts of *product* and *activity* [15]. A product can

either be an individual real-world product, such as a manufactured automobile, or it may be a model of a (to-be-manufactured) product. An activity describes the occurrence of an action such as a design, manufacturing, or support operation or process. Using these two concepts, the ISO 10303-239 information model is able to represent assemblies, lifecycle information, product history, process plans, and schedules. ERP, on the other hand, involves a chain of repetitive operations and requires *transactional data*, defined by McGilvray [28] as data associated with an event or business process. OAGIS represents these repetitive operations as verbs. The OAGIS BODs encapsulate transactional data natively as business objects.

A concept in an information model cannot be both flat and hierarchical. Therefore, the same concept cannot natively support both controlled customization and business objects. To overcome this difficulty, a PLM standardization framework needs to provide additional implementation guidance, which results in added complexity for users. With respect to the two capabilities – business objects and controlled customization – we observe that there is no perfect framework. Consideration of native support, combined with Fig. 2 and Table 1, can help prospective users to determine the right framework to meet their requirements.

6 Conclusion

In this paper we discussed PLM standardization frameworks and their customization mechanisms. To represent business-specific information in a multitude of integration scenarios, the framework must enable customization and interoperability simultaneously. Our literature review identified two recurrent customization mechanisms. The first is extension, which adds to the initial set of concepts and relationships of the standard information model. The second is specialization, which uses classifiers from external sources to refine generic concepts into business-specific concepts. We also identified two primary approaches, RDL and CCTS, and we investigated an exemplary framework for each approach, PLCS and OAGIS respectively. We then described two key capabilities that PLM standards frameworks should support in order to meet requirements for the use cases of data exchange and data sharing. Fig. 2 summarized how the capabilities relate to the use cases.

We conclude that 1) choice of framework should take use case into account, 2) no single framework is best for both use cases, and 3) it is better for a framework's information model to natively support a capability than for the framework to require additional technology to implement the capability. As shown in Fig. 2, data sharing depends on support for business objects. Therefore, a CCTS-based framework such as OAGIS, with its native support for business objects, is a good choice to support data sharing. Likewise, an RDL-based framework such as PLCS with its native support for controlled customization, is a good candidate to support data exchange.

A significant limitation of the research is the lack of an industrial example with realistic PLM data. Applying such an example to our evaluation of PLCS and OAGIS would add more rigor to our conclusions. Another follow-on to the research discussed in this paper would be to expand upon Fiorentini and Rachuri's harmonization and integration work. Their research covered only one use case – engineering change

management (ECM). Pilot implementations of additional use cases exploiting other PLM disciplines where engineering and e-business concerns meet - such as logistics support and maintenance - could lead to useful lessons learned. Experience gained could not only result in improved metrics for evaluating PLM standardization frameworks, but also enable improvements to the frameworks themselves. Other possible follow-ons include evaluation of the RDL and CCTS approaches with respect to how well they support additional use cases such as long-term data retention, and exploration of the feasibility of combining both approaches within a single framework.

Acknowledgments. We wish to thank Jay Ganguli, Peter Denno, Albert Jones, and KC Morris for their insightful feedback on earlier drafts of this paper.

References

1. Ivezic, N., Kulvatunyou, B., Srinivasan, V.: On Architecting and Composing Through-life Engineering Information Services to Enable Smart Manufacturing. Procedia CIRP. 22, 45–52 (2014).
2. Terzi, S., Bouras, A., Dutta, D., Garetti, M., Dimitris Kiritsis: Product lifecycle management – from its history to its new role. International Journal of Product Lifecycle Management. 4, 360–389 (2010).
3. OMG Product Lifecycle Management Services, v2.1. Object Management Group (2011).
4. Srinivasan, V.: An integration framework for product lifecycle management. Computer-Aided Design. 43, 464–478 (2011).
5. ISO 10303-242:2014. Industrial automation systems and integration – Product data representation and exchange – Part 242: Application protocol: Managed model-based 3D engineering.
6. Katzenbach, A., Handschuh, S., Vettermann, S.: JT Format (ISO 14306) and AP 242 (ISO 10303): The Step to the Next Generation Collaborative Product Creation. In: Kovács, G.L. and Kochan, D. (eds.) Digital Product and Process Development Systems: IFIP TC 5 International Conference, NEW PROLAMAT 2013, Dresden, Germany, October 10-11, 2013. Proceedings. pp. 41–52. Springer Berlin Heidelberg, Berlin, Heidelberg (2013).
7. Krima, S., Feeney, A.B., Foufou, S.: Dynamic customisation, validation and integration of product data models using semantic web tools. International Journal of Product Lifecycle Management. 7, 38–53 (2014).
8. Fenves, S.J., Foufou, S., Bock, C., Sriram, R.D.: CPM2: A Core Model for Product Data. Journal of Computing and Information Science in Engineering. 8, 014501–014501 (2008).
9. Zimmermann, T.: Information Architecture, http://www14.informatik.tu-muenchen.de/konferenzen/Jass05/courses/6/Papers/03.pdf.
10. PLCSlib, http://www.plcs.org/plcslib.
11. ISO 10303-239:2012. Industrial automation systems and integration – Product data representation and exchange – Part 239: Application protocol: Product life cycle support.

12. Open Applications Group Integration Specification (OAGIS) Release 10.1. Open Applications Group (2014).
13. Rachuri, S., Subrahmanian, E., Bouras, A., Fenves, S.J., Foufou, S., Sriram, R.D.: Information sharing and exchange in the context of product lifecycle management: Role of standards. Computer-Aided Design. 40, 789–800 (2008).
14. OMG Systems Modeling Language (OMG SysML). Version 1.3. Object Management Group (2012).
15. Paviot, T., Cheutet, V., Lamouri, S.: A PLCS framework for PDM/ERP interoperability. International Journal of Product Lifecycle Management. 5, 295 (2011).
16. Fiorentini, X., Rachuri, S.: STEP-OAGIS Harmonization Joint Working Group: PDM Subgroup Interim Report. (2009).
17. Y. Lu, K. C. Morris, S. Frechette: Standards landscape and directions for smart manufacturing systems. In: 2015 IEEE International Conference on Automation Science and Engineering (CASE). pp. 998–1005 (2015).
18. Lampathaki, F., Mouzakitis, S., Gionis, G., Charalabidis, Y., Askounis, D.: Business to business interoperability: A current review of XML data integration standards. Computer Standards & Interfaces. 31, 1045–1055 (2009).
19. Price, D., Bodington, R.: Applying semantic web technology to the life cycle support of complex engineering assets. In: The Semantic Web–ISWC 2004. pp. 812–822. Springer (2004).
20. ISO/TS 15926-7:2011. Industrial automation systems and integration — Integration of life-cycle data for process plants including oil and gas production facilities — Part 7: Implementation methods for the integration of distributed systems: Template methodology.
21. OASIS Product Life Cycle Support (PLCS) TC | OASIS, https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=plcs.
22. OWL 2 Web Ontology Language Document Overview (Second Edition). (2012).
23. UN/CEFACT Core Components Technical Specification Version 3.0. United Nations Centre for Trade Facilitation and Electronic Business (2009).
24. Fagin, R., Kolaitis, P.G., Miller, R.J., Popa, L.: Data exchange: semantics and query answering. Theoretical Computer Science. 336, 89–124 (2005).
25. Ray, S.R., Jones, A.T.: Manufacturing interoperability. Journal of Intelligent Manufacturing. 17, 681–688 (2006).
26. Chen, D.: Enterprise Interoperability Framework. In: Proceedings of the Open Interop Workshop on Enterprise Modelling and Ontologies for Interoperability. , Luxembourg (2006).
27. Hunten, K.A., Feeney, A.B.: Business Object Models for Industrial Data Standards. Presented at the ASME 2011 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference (2011).
28. McGilvray, D.: Executing data quality projects: Ten steps to quality data and trusted information (TM). Morgan Kaufmann (2010).