Research Article

Service Oriented Integration of Distributed Heterogeneous IT Systems in Production Engineering Using Information Standards and Linked Data

Navid Shariat Zadeh,¹ Lars Lindberg,¹ Jad El-Khoury,² and Gunilla Sivard¹

¹Royal Institute of Technology (KTH), Production Engineering, Brinellvägen 68, 10044 Stockholm, Sweden

²Royal Institute of Technology (KTH), Mechatronics Division, Brinellvägen 83, 10044 Stockholm, Sweden

Correspondence should be addressed to Navid Shariat Zadeh; navidsz@kth.se

Received 1 September 2016; Revised 18 November 2016; Accepted 5 December 2016; Published 19 March 2017

Academic Editor: Feng Xiao

Copyright © 2017 Navid Shariat Zadeh et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

While design of production systems based on digital models brings benefits, the communication of models comes with challenges since models typically reside in a heterogeneous IT environment using different syntax and semantics. Coping with heterogeneity requires a smart integration strategy. One main paradigm to integrate data and IT systems is to deploy information standards. In particular, ISO 10303 STEP has been endorsed as a suitable standard to exchange a wide variety of product manufacturing data. One the other hand, service-oriented tool integration solutions are progressively adopted for the integration of data and IT-tools, especially with the emergence of Open Services for Lifecycle Collaboration whose focus is on the linking of data from heterogeneous software tools. In practice, there should be a combination of these approaches to facilitate the integration process. Hence, the aim of this paper is to investigate the applications of the approaches and the principles behind them and try to find criteria for where to use which approach. In addition, we explore the synergy between them and consequently suggest an approach based on combination of them. In addition, a systematic approach is suggested to identify required level of integrations and their corresponding approaches exemplified in a typical IT system architecture in Production Engineering.

1. Introduction

One of digital factory's main goals is to improve the design process that supports innovation by modelling, simulation, and visualization of manufacturing systems to get an understanding of each domain interdependencies between domains. The splitting of the overall design process into specific domains such as layout designs, logistics, and material flow analysis and process planning results in a situation where process, product, and manufacturing resource properties are distributed among a set of models, and it is trivial that these properties are interrelated and dependent on each other [1]. These dependencies can also be thought of as interrelationships between digital models or files encompassing models. Digital design of production systems without managing such interrelations results in data inconsistency throughout the system. Moreover, different disciplines in each domain use different IT tools based on their specific requirements.

Each IT tool supports particular design and development tasks. Hence, it must be possible to share and communicate digital models among systems as well as different disciplines involved in the design process. However, the models cannot be exchanged or shared between IT systems if they are not interoperable. A definition of interoperability is mentioned in [2], as “a measure of the degree to which diverse systems, organizations, and/or individuals are able to work together to achieve a common goal.” Therefore, it is a crucial feature of any system to be able to interact with other systems without losing information during its mission. The interoperability for computer systems is typically defined in terms of syntactic interoperability and semantic interoperability. One aspect of semantic issues is the usage of different vocabulary by design disciplines to address one concept; for instance, using operation, task, and activity to describe a transformation activity to realize a product feature. Another aspect of semantic issue is addressing different concepts using same

terminology. For example, a flow analyst may include the loading and unloading time of a part in calculation of cycle time while a process planner excludes those times as set up times. The syntax defines grammar, symbols, and rules to construct facts about the desired domain.

For two decades, many research initiatives have been conducted to manage the interoperability problem in this area by developing ontologies and information standards [3]. These approaches essentially aim to solve the syntactic and semantic interoperability by representing information in a single unified format and storing data in a centralized or distributed repository. In particular, ISO 10303 (standard for the computer-interpretable representation and exchange of product manufacturing information) labelled “STEP” has been endorsed as a suitable standard to exchange a wide variety of product manufacturing data. The representation and integration of product, process, and production resources information using different STEP application protocols are well covered and extensively investigated by many researchers. Although these approaches have shown rigorous potential to integrate heterogeneous data from different IT application with different level of detail, they have the following drawbacks:

- (i) They are complex and they require deep knowledge of certain areas of mathematics, programing, and communication protocols.
- (ii) Sometime they add unnecessary level of detail to the information that must be shared.

However, making two systems interoperable does not necessarily require the integration of data from different disciplines using a common information model. That means making IT systems interoperable can be done; for instance, via industrial initiatives such as Linked data and Open Services for Lifecycle Collaboration (OSLC). OSLC focuses on IT system integration by standardization of the most common concepts that must be shared among humans and applications. Its aim is to assure data consistency across the IT applications. However, it is not a remedy for all kinds of required integration in an organization. Therefore, harmonizing the two approaches of (1) using one standardized information model with (2) the paradigm of managing information, residing in a heterogeneous IT environment, is a vital task. Thus, there will be an optimal mix of these approaches in practice to integrate information and IT systems.

Furthermore, the data that can be acquired during the operation phase (feedback data) can help different actors to improve the design process. Therefore, the ability to access, process, and communicate these data is from high importance and interoperability solutions must consider integration of this information with the data from the design phase. When it comes to collect and integrate the run time data, in addition to standardized data, a standard data transfer protocol is also required.

For these reasons, this paper investigates the integration and interoperability principles, guidelines, and their applicability in the digital factory domain in the manufacturing system lifecycle. In particular, we explore the combination

of linked data with STEP based data exchange to achieve interoperability in digital factory domain.

Our major contributions are listed as follows:

- (i) Comparing and documenting loosely coupled integration principles versus integration using information standards in particular STEP standard and OSLC in the production engineering domain.
- (ii) Suggesting a methodology to integrate heterogonous information from different sources and manufacturing lifecycle phases by using these approaches synergistically to assure data consistency cross IT applications.
- (iii) Developing computer applications as prototypes that can be used as templates for industry and end user programmers.

In this work discrete event simulation (DES) and factory layout design are considered as the representatives of digital factory for our discussion and implementation. This is because of the fact that they require information regarding plant layouts, products, processes, and manufacturing resources that are building blocks of the digital factory. Moreover, they represent a typical scenario where information is often not consolidated but resides in different CAX software tools and databases in a factory.

On this account, first the paper describes the information standards in particular ISO 10303 STEP, their purposes, and applicability. Secondly, it describes fundamental of OSLC and Linked data for IT systems integration purpose. Then, it compares OSLC with STEP standard according to their functional and nonfunctional properties. Next, it introduces a generic industrial IT system reference architecture which will be used later to show the applicability of these two approaches. Later it explains main tasks that must be performed to design and develop the IT system architecture before deciding about the integration approaches. Then, it describes a case study to clarify and verify the applicability of suggested approach in this context. Finally, the paper is concluded with suggestions for future work. Our focus is on integrating IT systems and making them interoperable in production engineering domain. However, we try to specify guidelines and criteria to identify where to use which approach. We target the manufacturing communities as the audience of this paper who are familiar with STEP to some extent but are not familiar with service oriented OSLC integration.

2. STEP Standard

Among different available information standards to represent material flow data and layout related data, ISO 10303 (STEP) has exposed a rigorous potential to structure and integrates heterogonous data from different information sources. Using STEP to exchange 3D geometrical model, product structure, manufacturing process plans, material flow data, kinematic data, and so forth has been shown and proven in [4–10]. STEP is intended to handle product data throughout its entire life cycle. The STEP information models are represented in different application protocols (APs). These Aps belong to

FIGURE 1: STEP standard architecture.

different industrial domains such as AP239 (ISO 10303-239), “Product Life Cycle Support” (PLCS), and AP242 “Managed model based 3D engineering.” Figure 1 depicts the architecture of the STEP standard. The Application Reference Models (ARM) of STEP identify the information requirements in the standard and define the terminology of various domains of interest (Universe of discourse). The Application Interpreted Model (AIM) is the data integration mechanism of the STEP standard. The Integrated Resources (IRs) define highly generic information types, and the AIM is the result of a mapping of the specific information types of the ARM to these Integrated Resources [11–16].

STEP describes how to represent and exchange the digital product information by defining a specific file format. The beauty of this standard is that it encompasses a wide variety of information domains concerning product, process and resource structure, property, geometry, documents, classification, organization, versioning, skill, stochastic properties, and so forth. Moreover, STEP modular architecture in different application protocols (APs) facilitates the integration of heterogeneous information. However, these standards are designed in a generic way to be able to represent a wide variety of product and manufacturing data. Therefore, they are typically used together with a concept model or other ontologies to add semantics to the exchanged data and guide the instantiation of the standard.

Authors have previously shown representation of factory layout related data and manufacturing process specification and their integration using STEP AP239 together with the process specification language (PSL) ontology [17]. Digital factory Building Block, a Swedish project executed by authors, also describes how to define models of manufacturing resources and processes in order to make the information consistent and reusable. The long term goal of this project was to create a system neutral repository of production information, a library of digital resource models, that can harmonize information from many different IT sources and can be used in different IT applications. Therefore, in this work we discuss the necessity and applicability of this approach in comparison with Linked data approach.

3. Linked Data

OASIS OSLC [18, 19] is an emerging interoperability open standard whose focus is on the linking of data from independent and heterogeneous software tools in order to support end-to-end life cycle processes. The standard assumes no centralized integration platform (such as a PLM software) and instead adopts the architecture of the Internet to achieve massive scalability and flexibility. OASIS OSLC is based on the W3C Linked Data Platform (LDP) and follows the Representational State Transfer (REST) architectural pattern [20, 21]. It provides users with platform-neutral usage of web technologies to integrate the tool chain. Moreover, it reduces the need for one tool to have knowledge of the internal data structure of another tool when they are integrated (loosely coupled integration). This helps system integrator and end users to be more agile in adapting their tool chain when introducing a change such as replacing an IT tool with a new one, or a change in the design process.

Figure 2 illustrates the overall structure of the OSLC standard, divided into the OSLC core specification, and a set of domain specifications [22]. The former builds upon the basics of LDP, REST, and HTTP to standardize the basic integration mechanisms and services, which each domain specification is in turn expected to adopt. For a given lifecycle topic (such as Requirements Management, Change Management, etc.), a domain specification specifies the vocabulary for the lifecycle artefacts needed to support a set of basic integration scenarios.

The services defined in OSLC core are outlined in Figure 2. OSLC defines the concept of Service Provider, which is the central organizing entity of a tool, under which artefacts are managed. Typical examples of a Service Provider are project, module, product, and so forth. It is within the context of such an organizing concept that artefacts are managed (created, navigated, changed, etc.). For a given Service Provider, OSLC allows the definition of two services (Creation Factory & Query Capability) that provide other tools with the possibility of creating and query artefacts, respectively. In addition, OSLC defines delegated

FIGURE 2: OSLC core concepts and relationships (OASIS OSLC Lifecycle Integration Core: June 2016).

UI (Selection and Creation) services that allow other tools to delegate the user interaction with an external artefact to the Service Provider under which the artefact is managed. This structure allows the discoverability of the services provided by each Service Provider, starting with a Service Provider Catalogue (which acts as a catalogue listing all available Service Providers exposed by a tool).

Following the REST architectural pattern, OSLC allows for the manipulation of artefacts, once accessed through the services, using the standard HTTP methods CRUD to Create, Read, Update, and Delete. In OSLC, tool artefacts are represented as RDF resources, which can be represented using RDF/XML, JSON, or Turtle.

4. Comparison of OSLC and STEP

This section focuses on the evaluation and comparison of STEP and OSLC to answer our first research question. A number of comparison aspects are introduced for discussing and contrasting OSLC and STEP. We have specified these criteria by conducting iterative literature survey, which was conducted over different information modelling languages, ontologies, and information standards within the production engineering domain. We consider both functional and non-functional characteristics. Here, we define functional properties as technical properties, architecture, domains, scope, and mechanism to integrate semantic and syntactic issues. Nonfunctional properties include scalability, consensus, and extensibility.

4.1. Functional Properties

4.1.1. Export/Import Models. If export/import functionality is required between a pair of tools (Source tool, destination tool), the dataset created in the first IT system must be translated to a new dataset compliant with the internal data structure of the destination tool. Hence, the destination can interpret the dataset, for instance, exporting a 3D solid model developed in a CAD system to a finite element method (FEM) application to perform the finite element analysis. Adopting STEP standard as a system independent format eradicates the need for developing application-to-application translators (point-to-point translators). Moreover, most CAX

vendors support 3D representation of items that is the most complex part of the STEP standard. This facilitates the implementation of other geometry related data such as kinematic and geometric dimensioning and tolerancing using STEP. In contrast, OSLC cannot be independently practiced to fulfill this requirement. It is important to note that an OSLC Service Provider does not represent a tool or tool instance, but it represents a “container” of resources that are hosted by an IT system, not the tool itself [23]. Another important issue is that, in service oriented architecture, tools expose both data and functionalities. However, OSLC is about service oriented integration that can be used in a service oriented architecture to expose data and create a tool chain.

4.1.2. Product Data Archiving/Persistent Storage of Data. Long time storage of product data is a very important requirement from sustainability and product liability viewpoints. In order to store product related data, rich metadata is required and as mentioned STEP provides a complete information model to represent a wide variety of product related data. All data are serialized with part 21 of STEP that is the data exchange format of the STEP standard [24]. It has ASCII structure and each instance of the data is represented in one line. STEP part 21 defines the encoding mechanism on how to serialize data according to EXPRESS schema. In contrast to STEP, long time archiving of product data is not in the scope of Linked data and OSLC, since OSLC focuses on the publishing and sharing of data across domains. In the other word, it is a solution for the system of systems interoperability.

4.1.3. Data Linking Mechanism. Integration of IT applications by linking a minimalistic set of data is the main principal of OSLC. It couples the concepts that are associated with or common in various domains. Hence, the need for duplication of data in different systems will be reduced, whereas STEP is about exchanging product and manufacturing system data by translating them to a system neutral physical file (part 21 of standard). Then other tools exploit these files or the data embedded in them.

4.1.4. Versioning Behaviour. Versioning behaviour is a property of a relationship between two classes of information (source item and related item). It controls the configuration

FIGURE 3: Versioning behaviour of parts when participating in a relationship.

of the source and related items when revising them. For instance, assume part one and part three are related to each in a product BOM (See Figure 3). The “fix” behaviour identifies that a new revision of part one must still be associated with the previous version of part three. Float behaviour specifies that if both part one and part three are revised the new revision of part one must be associated with the new revision of part three. OSLC domains such as change management and configuration management define the behaviour/actions that must be considered when a change occurs. However, they do not define rules to specify criteria when to use which behaviour. In contrast, STEP does not define any behaviour but includes schemas to represent product/resource PDM related data such as versions, states, actors effectivity date, documents, and so forth (See Figure 4).

4.1.5. Domain/Architecture. The Application of STEP 233 and AP239 for the purpose of requirement management, modelling simulation, product configuration, and maintenance feedback has been verified and validated by many researchers and research institutes such as Organization for the Advancement of Structured Information Standards (OASIS). OASIS provides users with guidelines, templates, and rules in order to use the PLCS information model to exchange data [25–27]. In summary, from the information modelling point of view, STEP encompasses the information models of different OSLC domains such as change management, configuration management, and so forth. Figure 4 partially depicts the OSLC data model for a change request on the left, including change title, person who contributed, the state of the change request, affected items, and so forth. To the right a simplified STEP representation of a product and its version is illustrated.

4.1.6. Openness. By openness, we mean the capability of a specification or standardized information models to refer

to information from other specifications or domains. STEP does not follow this principle, which means that the data instantiation must be carried out according to the standard information schemas and instantiation rules. In contrast, this is possible by adopting Linked data approach and OSLC specification. OSLC resources can be linked to other HTTP resources on the web and not only resources which are described as OSLC resources.

4.1.7. Product Definition. STEP part 41 (Integrated generic resource: Fundamentals of product description and support) defines “product_definition_schema” as a generic aspect of products, the categorization of products, definitions of products, and the relationships among them [28]. The “product_definition” entity data type represents a class of products or an aspect of it (general property, geometry, organization, etc.), for a specified life cycle phase. The life cycle phase of a “product_definition” may be further described by usage, by discipline, or by both. The ALM-PLM Interoperability group of OSLC is working on product definition specification that is in the draft state [29]. The current draft is inspired drastically by STEP product-definition-schema. In general, it is possible to use OSLC core together with OSLC configuration management and change management domains to represent any item in its entire life cycle.

4.1.8. Representation of Follow-Up Data (Data from Production Operation Phase). These data can describe how a physical product changes from its original state during its utilization phase. For a manufacturing resource, life cycle data include stop times, availability, and energy consumption. STEP AP239, Product Life Cycle Support, (PLCS) supports the distinction between three life cycle phases of a resource type including “as designed,” “as planned,” and a realized resource “as is.” OSLC is intended to represent life cycle information of any resource as well.

FIGURE 4: OSLC change request to left and product revision to the right.

4.1.9. Representation of Real-Time Data. These data can be collected from sensors and machine controllers and can be used to monitor and optimize the production resources or production systems at real time. OSLC uses HTTP, which can also be used as communication protocol of devices and machineries in the shop floor. However, here we assume that manufacturing resources in the shop floor can use TCP/protocol to communicate. In addition, HTTP is text based protocol and can be a burdensome for some shop floor devices, for example, due to energy consumption aspect. These kinds of devices are referred to as constrained devices. Constrained devices protocol (CoAP) is a lightweight application protocol dedicated to constrained devices for the communication [30]. Since CoAP can be implemented of the REST pattern using HTTP, Linked data still can be used to represent sensor data. To achieve this, there should be gateways available that can be used in application layer to assist constrained devices to communicate with other devices or IT systems. The motive to use Linked data and HTTP to communicate shop floor data is that they are originally developed for the web interoperability; hence, they are aligned with new manufacturing paradigms such as industrial Internet of things. STEP AP238 (STEP) is a machine tool control language that extends the STEP standard with the machining model in ISO 14649 [31]. This STEP protocol can be used to collect machining information from shop floor like cutting tools, manufacturing features, dimensions and tolerances, and so forth. Another standard

in STEP family is ISO 15531 also known as MANDATE (Manufacturing flow management data: Data model for flow monitoring and manufacturing data exchange) is an International Standard for the modelling of data used in manufacturing management (except product and component data). The purpose of ISO 15531 is to create a standardized data model to represent manufacturing management data including resource, their capability, maintenance, constraint, and control information [32–34]. However, its scope is not collection of sensor data. In general, using STEP standard for communicating unstructured data such as sensor data from the shop floor data would lead to a lot of overhead data [35].

4.2. Nonfunctional Properties

4.2.1. Stakeholders. The main stakeholders of STEP standard are developers of CAX tools, PLM tools, end users, and schema developers. The stakeholders of OSLC are software engineers and schema developers. End users receive explicit attention of OSLC since it provides the users with delegated dialogs UIs in HTML page that are intended for human interaction. While the STEP P21 file, the physical exchange file of the standard, is intended to be interpreted by computer systems, hence, OSLC is more human understandable.

4.2.2. Scalability. Both specifications can be scaled to projects of different size according to their specific needs. Adoptions of some parts of the STEP or OSLC do not oblige system

integrators to adopt other parts. One can represent product definition related data using STEP and the logic of material flow using process specification language (PSL) ontology and integrate a CAD tool and a discrete event simulation tool using OSLC. Extensibility: thanks to modular architecture of both standards, they can be extended with new schemas and data types without affecting the other modules if it is not necessary.

4.2.3. Consensus. There is a wide spread community acceptance of both standards such as Boeing, Air Bus, Swedish Defense, IBM, and so on and PLM system vendors such as EURO-STEP and standardization organizations such as ISO SC4 working group and OASIS.

4.2.4. Implementation Technology. STEP defines the set of general standard data access interface (SDAI) methods in part 22 of the standard [36]. Then, the methods are implemented in specific programming languages by a language binding. These bindings are defined for Java, C, and C++ [37]. OSLC is built on W3C Resource Description Framework (RDF) and REST architectural style. Operations on resources are executed using HTTP protocol.

4.2.5. Specificity. Sometimes depending on the user's viewpoint using STEP standard adds unnecessary level of detail to data required to be shared. For instance, representing the probability distribution of the Mean Time Between Failures (MTBF) in STEP AP214 requires approximately 16 entities and their attributes to be instantiated. This requires 3 entities and their attributes to be instantiated in standard Core Manufacturing Simulation Data (CMSD) [38], while OSLC standardizes the data with the level of granularity that users decide.

4.3. Summary Comparison of OSLC and STEP

- (1) Separation of domain specific concepts and generic concept allows users to define their own semantic and terminologies while using the same generic schema. Both OSLC and STEP separate between core and domain specifications. OSLC core focuses on common resources and properties, as well as basic mechanisms and patterns that all domain specifications follow.
- (2) STEP is capable of representing wide variety of product data model and manufacturing system related data (Geometry, product structure, classification, process plan, property, change management, activities, skills, environment, dimensions, and tolerances). On the other hand OSLC supports lightweight integration mechanism (RESTful services) and cannot support export and import functionalities.
- (3) Modular architecture of the STEP allows consolidation of wide variety of heterogeneous data if it is necessary. By consolidation, we mean that all desired information is represented in one data model, whereas OSLC is based on linking data instead of

copying data from different sources to one unified data model. Hence, using OSLC (when applicable) results in minimization of data copied when integrating IT systems.

- (4) Reference data library and domain ontologies are available for both approaches and they can be used to add semantic to the shared data.
- (5) Long term archiving of product related data and product manufacturing related data is possible using STEP while OSLC is not designed for this purpose.
- (6) OSLC resources and properties can be linked to other web resources that are not part of OSLC specifications. In STEP standard, data must be exactly instantiated according to the STEP schemas and its rules.
- (7) STEP standard is more mature, industrially accepted, and adopted than OSLC.
- (8) OSLC can be used to integrate IT tools with Internet based devices (shop floor manufacturing resources that expose data with HTTP or other protocols).

From the performed comparison, we can evaluate the second question concerning the overlap of these approaches and when and where they are best used. Viewpoints and concerns expressed in the design process, as well as business and information requirements, determine whether adopting OSLC is sufficient or if STEP based data exchange is required. If a 3D model of a production resource developed in a mechanical CAD system must be exported to a CAD tool to design the layout or where data closely tied to geometry must be shared, the STEP standard is preferred. An example of geometry related data in layout design is the connection point of media to a production resource such as electrical cabinet or a machine tool coolant interface. OSLC is preferred where loosely coupled integration fulfills the users' requirements as it has less overhead than STEP. It is important to note that predefined criteria to select one of these approaches cannot be formulated as the choice is very dependent on context. For instance, consider a manufacturing system design task including two subtasks, plant layout design and material flow analysis. In the first scenario, a material flow analyst needs only to determine the size of different blocks in the layout and the distances between them. Hence, the flow analyst does not need to embed the layout geometry model into the discrete event simulation model. In this scenario, OSLC can be used to transfer the data to the flow analysis IT system with the help of an IT system adaptor. In the second situation, the flow of material through the actual layout is also required; thus, the flow analyst needs to integrate the 2D layout into the material flow model. In this scenario, STEP can be used to integrate the two software tools and make them interoperable. Figure 5 illustrates these two integration scenarios.

5. System Architecture Reference

Figure 6 illustrates the main components (or applications) in an IT system used to support the digital factory design

FIGURE 5: OSLC integration versus STEP integration.

FIGURE 6: IT system architecture reference to support digital factory.

process. This architecture is aligned with the IT architecture that ISA 95 standard (Enterprise-control system integration) introduces [39].

The top authoring applications (or AA) layer encompasses software tools like CAD, CAM, and so forth. The AA layer is where most of the design and development tasks are done. The optional Application File Managers (AFM) layer is next. If present, the AFM manages application files and their often-complex relationships. Usually 3D CAD systems require AFMs to manage the interdependency between the files and their file references while office programs, like MS Word and Excel, do not. Application file managers should

generally be closely bound to a particular vendor's authoring applications. The Product Lifecycle Management (PLM) layer is where all the design information is collected, configured, and maintained throughout the product lifecycle. To achieve this, a consolidated information model for the domain must be created to identify the data structure in the PLM system. This unified model is usually called *metadata*. To store data persistently in this layer a relational database is used along with a file system folder which stores and structures application files. Additional functions provided by a PLM system include user endorsement and permissions, revision and version control functionality, workflow management,

dependency management including imposing of referential integrity, and IT systems integrations [40]. The PLM level is sometimes a combination of applications (with functionality provided by the system vendor) and a development platform for various in-house database related applications. PLM manages not only product data but also factory layouts and manufacturing related data.

Next is the Enterprise Resource Planning (ERP) level that is a catchall for the systems surrounding the architecture, including production and service planning, inventory management, and sales. These external systems can include systems at vendors, customers, partners, and so forth.

The Manufacturing Execution System (MES) layer follows. This level controls all manufacturing information in execution time such as monitoring machines, robots, and employees.

Lastly, we have the Data Integration and Visualization (DIV) layer that stores and integrates the shop floor data. This layer collects large amounts of real-time data, typically used for monitoring, controlling, and planning on the shop floor. The run time information comes from various sources such as temperature gauges, pressure sensors, and machine and robots controllers.

A major difference between this layer and the PLM layer is the database technology used to store the data. In this layer, large amounts of data are collected and stored in an unstructured form in real-time databases. Real-time databases are temporal, meaning that each stored datum point is time specific and can evolve as time progresses. For instance, the individual stop times of a manufacturing resource are stored as raw data in this layer. When these raw data are processed and converted to information, they may be stored in a PLM or ERP system for continuous improvement purposes. However, since temporal data are stored with timestamps, data validity may decrease with time, reducing it simply to historical reference data.

NoSQL databases, such as graph databases, are more suited to the collection of real-time data due to their scalability and the capability of doing millions of data transactions per second. The main difference between the design of an IT system to support digital factory design and other software development projects is that software development projects start more or less from a clean state, while designing a digital factory framework is a combination of selection, configuration, and adaptation of a reasonably high-level commercial application. The data maintained in the IT system represents a large part of the Intellectual Property of the organization and needs protection in terms of secure repositories, controlled access, and authorized and authenticated users. Managing the configuration of items, essentially through a combination of managing revisions and relationships, is very important. Hence, the data changes must be coordinated between applications. For instance, an approved Change Order in the PLM system may trigger a change to a CAD model, which in turn requires changes to the process plan. These changes must be managed as a unit, (e.g., released together to production). To support the design process efficiently, the components of the IT system need to be integrated on some level. The complex data structures of

the authoring applications mean that these integrations also tend to be complex.

Now that we have defined the IT system architecture, we address the third question, the evaluation of using Linked data to avoid managing product data by adopting a single central platform. We also assess the role of the PLM IT system in a distributed architecture.

One main task of the PLM system is configuration management through a combination of managing revisions, managing relationships, and maintaining referential integrity (i.e., referenced item cannot be modified or deleted independent of the item referencing it). This is a feature that may need to be managed centrally by a single IT tool, the PLM software tool. Nevertheless, this does not necessarily mean that all artefacts that belong to a configuration need to be managed by this same tool. They can be linked from external authoring tools by using OSLC since a distributed architecture can tolerate the presence of a somewhat central system such as the PLM system. However, in this scenario the implementation and customization must be performed in a way that considers the following issues:

- (i) Enforce business rules, most importantly when releasing and changing product definition data. One typical rule is that all constituent parts must be released before an assembly of these can be released.
- (ii) Define, implement, and control the relationship behaviour of resources.
- (iii) Update the links between different resources when a change occurs according to the defined behaviour.
- (iv) Store the previous configuration in one or more tools (if necessary for traceability).
- (v) Authenticate users and authorize them to perform operations on the items.

Developing suitable mechanisms to update links between resources in the IT tool chain and synchronizing a heterogeneous tool chain when introducing an engineering change is not a trivial task. It requires implementing the business logic across the tool chain which in turn demands significant amounts of customization and configuration. Moreover, this raises two questions: which tool should store the different configurations of resources and is the tool capable of storing the RDF triples persistently?

CAD/CAM software tools, as main components of the digital factory IT system, do not have engineering change management and configuration management features. Nevertheless, if an in-house solution provides versioning, life cycle management, permission and identity management, behaviour management, dependency management, and change management and if it can store RDF triples in a graph database in the cloud or on a specific server, then basically a PLM system has been created; although, conceptually the need for an IT system for configuration management has not been eliminated. If a PLM system is deployed in a distributed IT environment, it is in principle possible to implement any type of integrations, (e.g., OSLC or STEP) as the business demands. However, it is obviously

not possible to enforce the same set of rules in an integration scenario as it is in the close world assumption scenario under which a PLM system operates.

After defining the architecture of the digital factory IT system, we investigate the applicability of STEP and OSLC for the integration of different components in the architecture. Our focus is to make systems interoperable by using OSLC and STEP in a complementary and synergistic manner. In each step, we try to define the main questions requiring answers before taking the next step.

6. Integration Approach

Adopting a suitable integration strategy requires both an understanding of the business side (from the business benefits down to the detailed working procedures) and the IT side (from hardware infrastructure to application software). The design of the IT system architecture mainly includes making decisions concerning the IT tools used to support business processes and how these IT tools are integrated. The following sections summarize the most important actions to identify the integration strategies, scenarios, technologies, and file formats.

6.1. Activity Modelling. An activity model must reproduce the main processes executed in the organization that support the design and development of the manufacturing systems. The activity model must start from the top-level process and be broken down to detailed procedures, (e.g., how to release a document or how to do the feasibility study of a change request). Since these processes are usually complex and varied, simplification is needed.

An activity model is a communication tool between domains in the design process. Three questions must be answered in this step:

- (1) What does the design team want to accomplish?
- (2) What workflow should be followed?
- (3) Which software tools are used or can be used to support the design process?

The focus of an activity model is to identify operational workflow and to capture the functions and information requirements. To develop activity models, the overall design process is divided into domain specific disciplines. The properties of the product, manufacturing resources, and processes are distributed in various digital models. Information sources and requirements are specified in this step. Different viewpoints participating in the design process must be taken into consideration to identify the processes' interactions in terms of information.

Activity modelling can be developed using all established languages such as Integration Definition for Function Modelling (IDEF0), UML activity modelling [41], and Structured Analysis and Design Technique (SADT) [42]. Using languages such as UML sequence diagram or communication diagram can help to identify the sequence of messages exchanged between different actors as well as software tools for some specific scenarios. This can help to determine the

route of integration, which will be discussed in Section 6.3 of this chapter.

It is important to note that legacy data and legacy IT tools in an organization may dictate a particular workflow to the design process. Hence, the activity model should be compliant with these constraints.

6.2. Information Modelling. The activity model helps to identify information that should be shared among disciplines involved in the design process. An information model structures this information and can be used to define the preferred terminology within an organization to facilitate communication between both humans and computers. The focus should be on the information that must be shared and communicated among different disciplines in different domains.

Consider a machining center as a manufacturing resource. Concepts relevant to the process planning include the resource's parts and properties (e.g., table, axes, and work zone) for the selection of resource type, interface information for the selection of tools and fixtures, process liquid usage and control system information for the creation of NC code, and the process operations and accuracy that the resource can perform. Concepts useful for factory planning include outer dimensions of the machines for the layout, media interfaces, and interfaces to external equipment for the installation. Concepts relevant for the material flow analysis are machine availability, cycle time, scheduled maintenance, tool change time, failure information (such as mean time between failure and mean time to repair), product routing and material flow logic, and events. As mentioned these data have different scope, levels of detail, characteristics (temporal, nontemporal, geometry, and nongeometry) and come from different lifecycle stages of a manufacturing system.

Structuring the data in one step can be very tricky. To overcome this, the authors suggest structuring data in three layers [43]. The first layer is created by carefully analyzing domain-dependent and independent concepts. The analysis of individual domains helps to specify the concepts that must be shared as well as their semantics, properties, and relationships. Subsequently, a set of domain concepts or ontologies is attained (level 1). It is the best practice to use domain specific languages ontologies for modelling a particular domain. For instance in the context of a digital factory, the following standards have shown the reach model to represent different domain:

- (i) IFC2x3 standard can be used to represent building design data [44, 45].
- (ii) STEP AP214, AP242 for representing 3D geometry model of product and manufacturing system.
- (iii) STEP 239 for product lifecycle data.
- (iv) Process Specification language to represent the logic of the material flow in the shop floor.

On the second level, a unified information model is created. This level incorporates data from the first level and structures it into a single information model to identify the information that must be communicated and the dependencies among

them. All actors must reach a consensus on the shared concepts, level of granularity, semantics, and structure of concepts.

If import/export functionalities between applications are required, the STEP standard can be used (level 3). PLCS (AP239) is a suitable choice in our approach thanks to its generic nature and availability of reference data libraries to add necessary semantics to the shared data (see Section 6.3.5).

Similar to information model, a data model is about structuring the metadata. In this instance, data model refers to a data structure that is compliant with or similar to the internal data structure of the software tools or application. For instance, most of the commercial PLM systems use a relational database for the persistent storage of data. Therefore, it is convenient to structure this data with an entity relationship diagram (ER model) in the second level.

6.3. System Integration. In this task, we should identify how software tools in the architecture should be integrated. The following sections identify the questions best used in the decision making process.

6.3.1. What Information Should Be Maintained in the PLM System? There is no explicit answer to this question but the following can help to determine what information should be kept in a PLM system:

- (1) Is the entity version controlled? (E.g., maintains history of released versions.)
- (2) Is the entity access controlled? (E.g., only some users may update a property, but all may read it.)
- (3) Is the entity globally searchable? (E.g., to find any production equipment of a certain type. Note that we include reporting here, e.g., to generate a list of all machines being installed during the last year.)

6.3.2. What Types of Integration Are Required? We can identify four types of integration between a PLM system and authoring IT tools:

- (1) Integration of files: this means the PLM system manages files through a checkout, check-in approach but cannot interpret the file content. Thus, files are managed as bulk data but some administration data such as creator, time and date of creation, and modification are coordinated with the PLM system.
- (2) Integration of properties: in this case, some properties are exchanged between software applications and the PLM system. For instance, lengths and widths of different blocks in a block layout model are integrated to their corresponding entity and properties in the PLM system.
- (3) Integration of structure: in this case, the internal structures of a file created in the applications are instantiated in the PLM system. An example would be the integration of the hierarchical structure of a layout to its corresponding structure in the PLM system.

- (4) Integration of object-relationship: this case is equivalent to the integration of structures. However, it includes integrations between the PLM system and other systems in the product life cycle like shop floor hub, real time database, and so forth.

6.3.3. Should Integration of Authoring Applications Be Point-to-Point or via a PLM System? Now we have to determine which applications should be integrated and decide whether the integration should be point-to-point or through the PLM system. As the receiving applications generally dictate the data requirements, these criteria must also be considered. Once we conclude that a PLM system is in the architecture, we have to determine the route of integrations, either through the PLM system or point-to-point.

The answer depends on the context and the life cycle of the design process as well as the users' requirements and viewpoints. For instance, in the early design stages, the flow analyst conducts many iterations to develop different flow concepts based on the current layout. Then, these concepts go through a feasibility study until one is selected. If there is no need for the traceability of the embedded layout in the flow simulation model, the integration can be directly between the flow simulation tool and layout tool instead of integrating them via a PLM system. This is very valid when there is no need to trace which version of the flow simulation is associated with which version of the conceptual layout in the early phase of design.

6.3.4. What Is the Timing in Integration and Synchronization of Changes? Timing is a critical facet of any integration strategy. The following are the three main types of timing considerations for the integration and synchronization of changes in the IT tool chain:

- (1) Time interval: integration is based on specific time intervals. For instance, mean time between failures (MTBF) of a manufacturing resource is typically updated annually or semiannually.
- (2) Event driven: whenever a change occurs (data and documents), for instance, when a reversion of a product BOM is released in a PLM system, it must be integrated with the ERP system.
- (3) On demand (Pull): it is when a user of a MES system makes a query about the last revision of a work instruction in a PLM system.

6.3.5. How to Use Information Standards and OSLC Specifications to Integrate IT Systems and Data to Benefit from Both? According to the comparison made in Section 4, actors can specify if Linked data is enough to make tools interoperable or STEP based data exchange must be deployed. Deploying STEP schemas (according to view points and scope) is recommended to specify OSLC resources, their properties, their structure, and their dependencies and for reuse of the OSLC core as is to provide users with CRUD services.

With this approach, system integrators and end users can benefit from a STEP rich information model and OSLC

FIGURE 7: Integration based on OSLC and STEP in a complementary way.

services at the same time. Another advantage of this approach is that it makes possible to link both structured and unstructured data. Figure 7 demonstrates a summary of the idea by creating OSLC resources based on STEP while also creating CRUD services by using an OSLC core. These resources can be either the same artefacts in two applications or different pieces of associated information in two software tools. An example of the latter case is linking a process plan in a CAM application and its associated product in a PLM system or a CAD tool using proprietary adaptors. The synchronization engine is responsible for the propagation and for adjusting required changes of artefacts in different systems when one artefact is initially modified in only one system. As mentioned, this can be implemented using PLM systems. Time or event based procedures are required to harmonize changes of integrated data throughout the tool chain. For instance, when a process plan is changed and a new cycle time is calculated, it changes the material flow analysis and may change the configuration of cells in the layout. In cases where point-to-point integration is applied, data and tool ownership must be explicitly defined, and suitable mechanisms must be implemented to keep the links updated while saving old resource configurations. A common data model is used for reporting and long term archiving of products, processes, and manufacturing resources based on STEP standard AP239 (PLCS).

6.4. Implementation. This is the configuration and customization step. Here configuration means using the available mechanisms in the applications to tweak its appearance and behaviour to our needs, while customization means writing code to better meet the information and integration requirements. Developing plugins for a CAD tool to integrate technical data from a production resource or developing a tool adaptor to expose OSLC data are some examples of customization tasks.

To apply the suggested approach, OSLC support must be added to the IT tools. There are three methods for adding OSLC support to an IT tool:

- (1) Native approach.
- (2) Plugin approach.
- (3) Adaptor approach.

The native approach is suitable when system integrators have access to the source code of the IT tools; therefore, it is primarily suitable for IT tool vendors.

The plugin approach is for users who want to add OSLC support to their IT tool COTS and requires system integrators who have detailed knowledge of the tool's API. In this approach, the IT tool should be modified in a way that all information in the tool has a URL and is OSLC compliant.

The adaptor approach is similar to the plugin approach but it does not change the tool. Hence, the adaptors present the information hosted by the tool as an OSLC resource. The adaptor approach is best suited to our needs.

7. Case Study

The aim of our case study is to verify data and IT tool integration based on the proposed approach, for factory layout design using commercial software tools. The intention is to integrate software tools and data so that various users or applications can utilize data concurrently to accelerate feedback among activities in the layout design process. A Research Concept Vehicle (RCV) is the product in the case study. The RCV is a testing laboratory for vehicle research into sustainable transport systems at the Royal Institute of Technology (KTH). The vehicle is a platform where research results can be implemented and evaluated in real-life [46].

7.1. Activity Model. An activity model was developed to identify plant layout design processes and the flow of

FIGURE 8: Factory layout activity modelling using IDEF0 (breakdown structure of the activity “Design the detailed layout”).

information among activities. This activity model serves as a reference to specify the information requirements and our IT system architecture. The layout development process is usually divided into a conceptual phase and a detailed phase [47].

In the conceptual phase, main areas required for the production system are created including machining areas, assembly areas, transport corridors, and warehouses. Functional interfaces between manufacturing resources, media, material handling system, and buildings should be devised in order to calculate the required media according to the total needs of the machines. Then, the detailed layout is created by populating the manufacturing resource models in the block layout with their physical connections, buffers in line, ventilation system, electricity layout, safety layout, and so forth. Finally, the entire layout is verified by integrating different 3D layout models like machinery layout, media layout, infrastructure layout, and so forth, to detect static and dynamic collisions and other rules and legislations.

We partition the design process according to three disciplines, process planning, product routing analysis, and layout planning and design. The activity model is developed in three hierarchical levels using IDEF0. Figure 8 illustrates the “layout detailed design” activity in the second level and its break down activities.

The break down structure of this activity depicts the different information requirements of a manufacturing resource according to the presented viewpoints. In this activity, the flow analyst decides on the number of machines and their

configuration on the shop floor (cell or line), balances the production line, calculates the buffer sizes, and levels the product mix to be fed into the production line. According to the activity model, the flow analyst requires the sequence of operation for each product, their cycle times, set up times and Takt time, and so forth. The cycle time that the flow analyst needs is the aggregate of times for each machine set up. Detailed times for each product feature are not required as this information is generated by the process planners.

The layout designers then check if the required space for the cells, stations, and buffers is available and whether the minimum distances between each pair of machines can be fulfilled or not. Next, the media layout designers select the required facility to provide the production line and machines with electricity, compressed air, fluids, and so forth.

7.2. Information Requirement. Authors have described information requirements for the representation of the logical relationship of manufacturing process specifications such as product routing and layout related data (reference to my licentiate). However, in this case study we exclude the representation and integration of the material flow logic in our scenarios, assuming that this type of information resides in the simulation tool.

Another issue is the level of granularity of integration between the layout software tools with the flow simulation tools. In this case, the layout model is not intended to be integrated with a DES model. Hence, the layout information that must be shared with the flow simulation tools includes

FIGURE 9: The system architecture of the use case.

the boundary size of the production resources and spatial dimensions of the equipment in the layout. A UML class diagram has been developed to structure the data that must be integrated or communicated in our domain of interest. A light version of this model implemented in the Innovator PLM system is illustrated in Section 7.3.

7.3. IT System Architecture. The system architecture considered in this case study is depicted in Figure 9. A heterogeneous distributed IT system architecture was designed to test and discuss this approach. At the application layer, we use Autodesk Inventor for 3D layout design, AutoCAD architecture for 2D layout design, SimJava, and Siemens Plant Simulation for material flow simulation. The 3D layout models include complex relationships with other CAD files that are referred by a layout CAD model. For this reason, it is more convenient to use the application file management dedicated to the layout tool, in this case Autodesk Vault.

ARAS Innovator is used as our PLM system to represent the information needed on levels of granularity ranging from “documents” (large data sets) to objects and the properties of these objects. The Thingworx platform is selected as the IoT platform in the architecture.

As a starting point, 3D models of the RCV bill of material that have already been modelled in a Solid Edge CAD system have been exported as CSV files which have been imported into Innovator by a method developed using Innovator API.

Authors have shown how to control states and revisions of files containing the layout model via Autodesk Vault. The integration between the AFM and the CAD-layers works well for files, file references, and properties. We can for instance define a property (as a name, value pair) in Vault and view and modify it in the AutoCAD software tool, and, when the file is checked back in to Vault, we can search for the new value, modify it again, and so forth. In a similar way, we can modify files and file references.

Integration between AutoCAD and AFM is done by configuring the software and no coding or other customization is needed. Both are from the same family of software tools and the integration functionalities have already been devised by the vendor, Autodesk.

Autodesk 360 is used as a cloud to share 3D models of the machinery. Moreover, this cloud is available inside the Inventor tool and users can simply drag and drop these models into their layout. According to the information described in the previous section, the model illustrated in Figure 10 in the ARAS Innovator PLM system has been implemented. Note that the model is simplified and that we have eliminated the document class, which is the container of the file data items. The document can be associated with all entities in the model.

We have defined suitable lifecycle maps for versioned information entities. For instance, a document has states including “In Work,” “Review,” “Released,” and “Obsolete.” Similarly, a manufacturing resource may have the states “As Designed,” “As Planned,” “In Use,” and “In Service.” Moreover, the required properties on relationships between entities (source item and related item) have been defined to control the configuration of the source and related items when revising them.

Versioning behaviour also specifies in which life cycle stage of an item it must be promoted to new a revision. For instance, updating an item in the “In review” state may not affect the reversion (revision?), while updating it in the “Released” state generates a new revision of the item. The relationship model in Innovator supports regular configuration management tasks such as updating the relationships correctly when new revisions of related items are generated.

7.4. System Integration Using STEP and OSLC

7.4.1. STEP Implementation. The first case study included the development of an application for providing information for discrete event simulation Software. The aim was to support digital analysis through easy access to relevant and coordinated data. The goal was to define and implement models of production resources and manufacturing processes that can be reused in many different applications and which can also be updated based on feedback data. Figure 11 illustrates a simplified representation of a production system including a manufactured product, its process plan and production resource, and their properties using STEP application protocol 214 AIM level. The authors have provided a more detailed

FIGURE 10: Data model in Innovator (The PLM IT system, simplified version).

FIGURE 11: A simplified representation of a manufacturing process using STEP standard.

description of the data model and the implementation in [48].

7.4.2. Integration Using OSLC. In order to specify services and implement IT tool adaptors, several scenarios have been determined. Figure 12 illustrates parts of the developed UML sequence diagram that captures the order of messages between different domains in the case study. Thus, a minimalistic set of data together with their semantics is determined and illustrated in Figure 13. This terminology must be precisely defined and accepted by all the actors in a real industry case. The STEP model from the previous section is used to identify our OSLC resources and data exchange specifications.

7.4.3. Developing IT Tool Adaptors. This scenario starts with the process planning of a new part or a change in an existing process plan. The products' bills of materials and their 3D models already exist in ARAS Innovator. Process planners acquire product information from the PLM system and develop the manufacturing processes. Then the data concerning operation sequences, cycle, and setup times are instantiated into our PLM system (ARAS Innovator).

To implement this scenario, we have developed the tool adaptor for Innovator to enable process planners to create, read, and update process plans as well reading services to get the product related data. These data are essential inputs for the material flow analysis. Our tool adaptors are based on the OSLC core (no schema, just services and core specification).

FIGURE 12: An UML sequence diagram to identify the interactions between domains in the case study.

Hence, we have defined the resource shape of the OSLC resources according to STEP standard manually. We have also defined two Service Providers for ARAS Innovator. The first one provides services to expose part information to the process planner or material flow analyst. The second provides

process planners and material flow analysts with CRUD services concerning sequence of operations, cycle times, and set up times.

The Lyo code generator technology is used to develop our tool adaptors. This technology allows the system integrator

FIGURE 13: Interrelationship among three domains (layout design, discrete event simulation, and process planning).

to develop OSLC4 Java programming using a code generator. Here, users model the adaptor functionalities according to the OSLC core or its specifications through an Eclipse Modelling Framework (EMF) from which most of the code will be generated. The EMF project is a modelling framework and code generation facility for developing IT tools based on a metadata model. Working with a model of the adaptor allows system integrators to focus only on the adaptor model while the external part of the adaptor is generated automatically.

Designing an adaptor model includes different activities such as defining information resources, their types, properties, and relationships with internal and external resources and finally developing services.

Figure 14 presents the Service Provider Catalogue of Innovator that is partitioned according to process plan related data for RCV parts. The catalogue includes services for each part of the process plan. A process plan is an OSLC resource with a global identifier and includes sets of operations that are OSLC resources. Finally, each operation uses a manufacturing resource that is also an OSLC resource with properties.

Manufacturing resources such as machine tools, robots, and transportation facilities are essential parts of the factory layout. The most important pieces of manufacturing resource information, from the layout design perspective, are resource dimensions, outer shape and placement of resource media interfaces, placement of doors and other openings, mass, center of gravity, and material loading points. This information serves as inputs to the layout design activities. One aim is to communicate machine models including 3D geometry, connection points, and technical data and combine everything into one layout. According to our activity model, we consider two scenarios for the integration. In the first scenario, 3D model of manufacturing resources is exported

to the STEP standard using any CAD software. Then, the machine data card is represented according to STEP AP214 using our developed software applications. The input for the application is the machine data card as an Excel spreadsheet. The first application translates the technical data to STEP AP 214 and the second application merges it to the geometry of the resource. Then, the information is stored in the Innovator system since we still need to be able to manage versions, permissions, and support processes.

Another scenario is the integration of the machine data card of a manufacturing resource with its corresponding block in Plant Simulation. A tool adaptor for Inventor (the layout design software) has been developed to provide users with machine data card related information.

Once the media connection point on the 3D model of a manufacturing resource is determined, it can be stored to the Autodesk 360 cloud for distributed layout design process. Here, another complication is the integration of a manufacturing resource 3D model in the cloud with the machine technical data.

The machine data card is represented using the OSLC core and linked to its 3D model in the cloud. Autodesk 360 is a web-based application to view models and drawings and provides rich visualization in a web browser. As previously mentioned, an OSLC resource or its properties can be referenced in by any HTTP resource in the web. In this case, the Inventor Service Provider links the equipment embedded in the layout as an OSLC resource and connects it to their 3D model in the cloud (Autodesk 360). Figure 15 illustrates a simplified example of a manufacturing resource as an OSLC resource that is integrated to the geometry of the model in Autodesk 360 using the tool adaptor.

One of the services that the Innovator adaptor provides is the creation of new equipment or a new station

FIGURE 14: OSLC core services of Innovator for process plan related information.

by specifying HTTP protocols. However, for some integration scenarios, using these protocols is not the best strategy.

Suppose a material flow analyst intends to evaluate whether the current assembly line has enough capacity to manufacture a new product and, if yes, calculate the new station/equipment utilizations. Ideally, the flow analyst would like to visualize the station information inside the simulation tool. Hence, we would like to allow our users to visualize equipment information from the simulation tool or associate the new information such as utilization with an existing

station/equipment if they already exist in Innovator. If we used the OSLC protocols to implement this integration scenario, we would have to create the user interface needed for the flow analyst to enter all the valid data of the new equipment or present a list of existing ones. This is not only a lot of work but also could cause a poor user experience because it is unlikely that users are aware of all the fields that the PLM system will demand or the detailed validation that it does when adding a new manufacturing resource. OSLC solves this problem by providing system integrators with an additional integration style.

FIGURE 15: Presentation, representation of a machine tool technical data, and integration of these data with the machine 3D model in the cloud.

This solution uses the concept of a dialog whose intention is that the SimJava tool asks Innovator to display its selection or creation dialog to the user instead of implementing the user interface for creating or selecting equipment/station in its interface.

In the case of a dialog to create new equipment, the Innovator tool can provide initial data to the dialog to create the new equipment. In the case of a selection dialog, the simulation tool gets back the URL of the selected equipment/station, which it can then reference from the simulation tool. Figure 16 illustrates the delegated UI of SimJava tool adaptor that is integrated with the Innovator Service Providers. As shown, SimJava users have access to the selection and creation dialogs of Innovator.

Another implemented scenario is the integration Siemens Plant Simulation tool with Autodesk Inventor. In this case, the simulation analyst can query the process plan of a particular part using Innovator Service Provider to visualize properties of a manufacturing resource in the Inventor tool. Since the Plant Simulation tool has an ActiveX interface, we need just to add a method and provide the URL of the required OSLC resources.

ActiveX is an object technology allowing users to activate objects (ActiveX controls or other COM objects) within a container application and to exchange data with these objects using defined interfaces. Component object model (COM) is a technology that allows objects to interact across process and computer boundaries as easily as within a single process. COM enables this by specifying that the only way to manipulate the data associated with an object is through an interface on the object [49]. The first OSLC resource is a process plan query and the second OSLC resource is the equipment. To implement the latter case, users must know the exact name of the resource to make a precise query.

Figure 17 depicts the first scenario where the flow analyst runs the search method. This causes a graphical user interface (GUI) for querying a manufacturing resource to pop up. Through the GUI, the user can then query manufacturing resources and their properties.

The case study confirms that the integration of IT tools within the described IT system architecture can be done by deploying information standards together with OSLC specification and available technologies to integrate commercial IT tools and data to ensure data consistency cross IT applications

PLM:Aras Innovator

<http://pc113:8080/PLMAdaptor/services/serviceProviders/1/processPlans/KTPP001> version 0.1

Resource Presentation

Summary: <http://pc113:8080/PLMAdaptor/services/serviceProviders/1/processPlans/KTPP001>

Properties

ProcessPlanID: KTPP001
 ProcessPlanName: Upright body
 ProcessPlanVersion: A
 ProcessPlanOperation:

- <http://pc113:8080/PLMAdaptor/services/serviceProviders/1/operations/OP001>
- <http://pc113:8080/PLMAdaptor/services/serviceProviders/1/operations/OP012>
- <http://pc113:8080/PLMAdaptor/services/serviceProviders/1/operations/OP005>
- <http://pc113:8080/PLMAdaptor/services/serviceProviders/1/operations/OP004>

OSLC Tools Adapter Server 0.1 brought to you by [Eclipse Lyo](#)

Simulation Engine:

Utilization calculation

Status

PLANT Simulation Simulation of Plant A
 Created Nov. 25, 2014
 Updated Dec. 25, 2014
 Status OPEN

Description: This example shows how to link the result of a discrete event simulation system .

Add Link... Select Equipment to Link to... Create Equipment to Link to...

Related Equipments

- Equipment #2
- Equipment #3
- Equipment #4

Add Link

Name: Equip 1

URL: <http://localhost:8080/ArasAdaptor/>

Add Cancel

FIGURE 16: Delegated User Interface of Innovator to SimJava.

FIGURE 17: Plant Simulation uses the Activx interface to access to the Inventor OSLC.

in a typical IT architecture in the production engineering domain.

8. Conclusion

In the domain of production engineering modelling and simulation of material flow and plant layouts, process planning is often used to support the design process of manufacturing systems. These models generally represent different contexts and levels of detail and reside in distributed heterogamous IT systems using different formats and vocabulary and come

from different life cycle stages of the manufacturing system. Therefore, to live with a heterogeneous IT environment, it requires a good integration strategy to make systems interoperable. Interoperability solutions should deal with semantic and syntactic of data, presentation of data (human understandable), and data transfer protocols for communications. Standardization is a critical success factor to make IT systems semantically and syntactically interoperable.

ISO 10303, STEP, has been endorsed as a suitable standard to exchange a wide variety of product data and product manufacturing data. Hence, STEP based data exchange makes IT

systems integrated and consequently interoperable. However, IT systems can be integrated in other ways than sharing a common model, for example, via industrial initiatives such as OSLC and Linked data. Linked data approach is progressively being adopted for the integration of IT tools and creates tool chain, especially with the introduction of the Open Services for Lifecycle Collaboration (OSLC) on the IT tool interoperability. OSLC integrates only the concepts that must be shared across domains and presumes a loosely coupled integration of software tools and services. OSLC has created hope for system integrators and end users to bypass the need to adopt a central IT platform through which the product and production system data are coordinated. However, loosely coupled integration cannot fulfill all types of integration in the domain of production engineering and digital factory. For this reason, in practice there should be an optimal combination of these approaches. On this account, first we contribute to clarification of the applicability of these approaches, their integration mechanisms, their supportive technologies, and the principles they follow to make IT systems integrated. In summary, STEP and its rich data model can be used to exchange data and support import/export functionalities, especially when geometry or geometry related data must be communicated. OSLC is preferred when the most common concepts must be shared across the domains and not within one domain. In addition, with OSLC, it is possible to connect and use Internet based devices that communicate with HTTP protocol. Moreover, other HTTP resources on the web can be referred by any OSLC resource or its properties, thanks to the openness characteristic of OSLC. Another significant advantage of OSLC is that it is based on the service oriented RESTful architecture. This gives more flexibility to add or change components of the IT architecture since OSLC assumes a loosely coupled integration style.

However, according to the comparison made in Section 4, we realized that there are beneficial opportunities for system integrators or end users to combine STEP and OSLC with respect to information models, reference models and templates, services, and business logic. Therefore, we explored the combination of Linked data with STEP based data exchange to make IT systems interoperable. As a result, we proposed an approach that incorporates STEP to create domain information models of manufacturing systems and use OSLC core to provide the users with CRUD services. In other words, the OSLC resource shapes are constrained and formed by using the STEP schema. This approach not only facilitates the data communication and tool integration, but also reuses the existent STEP templates and its reference data libraries. It is important to note that adopting the close world information model of STEP that we suggest to be used to model the resource shape of OSLC is not contradictory with the open-world perspective of OSLC specification. The information that must be shared across the IT system may in general be connected to any HTTP resource on the web, but these resources or properties should be bound to the resource shapes mechanism of OSLC.

Furthermore, integration strategy is drastically affected by the IT architecture. Hence, we have outlined a reference IT architecture in production engineering. We have identified

practical information about the functionalities of each level in the architecture and surveyed the role of the PLM IT system in the distributed IT architecture. In summary, we state that if OSLC core and its domains such as configuration management and change management are adopted properly they can make IT tools interoperable and keep the shared data consistent throughout the IT tool chain. However, they cannot omit the need for the PLM system functionalities such as imposing referential integrity, managing dependencies, and storing the revisions of old configurations for the traceability purposes. Moreover, a distributed architecture tolerates the presence of a somewhat central system (such as PLM) as one of its components. In contrast to the distributed architecture, a PLM system is not typically designed to tolerate distribution of data, even though it might allow some point-to-point integration of other tools. As mentioned, a typical example/scenario is configuration management. This is a feature that may need to be managed centrally by a single IT tool, PLM system. However, this does not necessarily mean that all artefacts that belong to a configuration need to be managed by the same tool. They can be linked from external authoring tools. In this case, the required customization and configuration must be implemented in order to manage cross-domain dependencies and assure the data consistency.

Finally, a case study was conducted to verify data and IT tool integration based on the proposed approach in the domain of factory planning and design using commercial software tools. The results prove that the suggested approach is a pragmatic method and heterogeneity can be dealt with Linked data integration platform together with STEP based data exchange.

The suggested approach must be analyzed further and more case studies must be carried out to prove the concept in industrial cases. Creating a formal approach to map the STEP information model to resource shape of OSLC is considered as future work of this research. This can be accomplished by developing a STEP-OSLC toolkit.

Competing Interests

The authors declare that they have no competing interests.

Acknowledgments

The authors are grateful for support of VINNOVA (The Swedish Governmental Agency for Innovation Systems), Scania (Swedish truck manufacturing company), and XPRES (Initiative for Excellence in Production Research).

References

- [1] M. Törngren, A. Qamar, M. Biehl, F. Loiret, and J. El-khoury, "Integrating viewpoints in the development of mechatronic products," *Mechatronics*, vol. 24, no. 7, pp. 745–762, 2014.
- [2] N. Ide and J. Pustejovsky, "What does interoperability mean, anyway? Toward an operational definition of interoperability," in *Proceedings of the 2nd International Conference on Global Interoperability for Language Resources (ICGL '10)*, Hong Kong, 2010, <http://www.cs.vassar.edu/~ide/papers/ICGL10.pdf>.

- [3] N. Shariatzadeh, *Manufacturing process specification [Licentiate thesis]*, Department of Industrial Engineering, Royal Institute of Technology, Stockholm, Sweden, 2014.
- [4] N. Shariatzadeh, L. Lindberg, and G. Sivard, "Rapid production changes through the coordination of factory layout models and activities," *Journal of Applied Mechanical Engineering*, vol. 2, article no. 126, 2013.
- [5] G. Sivard, "Digital factory building blocks," project within FFI Sustainable production technology, http://www.vinnova.se/PageFiles/751290057/2009-00996_publicrapport_EN.pdf.
- [6] T. Kjellberg, A. Von Euler-Chelpin, M. Hedlind, M. Lundgren, G. Sivard, and D. Chen, "The machine tool model—a core part of the digital factory," *CIRP Annals—Manufacturing Technology*, vol. 58, no. 1, pp. 425–428, 2009.
- [7] M. Hedlind, *Model driven process planning for machining [Ph.D. thesis]*, KTH, School of Industrial Engineering and Management (ITM), Production Engineering, Stockholm, Sweden, 2013.
- [8] Y. Li, M. Hedlind, and T. Kjellberg, "Implementation of kinematic mechanism data exchange based on STEP," in *Proceedings of the CIRP 7th International Conference on Digital Enterprise Technology*, September 2011, https://www.academia.edu/24270742/IMPLEMENTATION_OF_KINEMATIC_MECHANISM_DATA_EXCHANGE_BASED_ON_STEP.
- [9] A. von Euler-Chelpin, *Information modelling for the manufacturing system life cycle [Ph.D. thesis]*, Royal Institute of Technology (KTH), School of Industrial Engineering and Management, Stockholm, Sweden, 2008.
- [10] P. Falkman, J. Nielsen, B. Lennartson, and A. von Euler-Chelpin, "Generation of STEP AP214 models from discrete event systems for process planning and control," *IEEE Transactions on Automation Science and Engineering*, vol. 5, no. 1, pp. 113–126, 2008.
- [11] ISO 10303:242, "Industrial automation systems and integration—product data representation- and exchange—part 242: managed model-based 3D engineering," I. TS 184/SC4184, 2012.
- [12] ISO 10303-214, "Industrial automation systems and integration—product data representation and exchange—part 214: application protocol: core data for automotive mechanical design processes," TS 184/SC4184, 2010.
- [13] ISO 10303:214, "Industrial automation systems and integration—product data representation- and exchange—ARM index," I. TS 184/SC4184, 2009.
- [14] ISO 10303:214, "Industrial automation systems and integration—product data representation- and exchange—AIM index," I. TS 184/SC4184, 2009.
- [15] ISO, *ISO 10303-239 ISO/TC184/SC4: ISO 10303-239: Industrial Automation Systems and Integration—Product Data Representation and Exchange—Part 239: Application Protocol: Product Life Cycle Support*, I. TS 184/SC4184, 2nd edition, 2012.
- [16] "ISO 10303-41 Industrial automation systems and integration—product data representation and exchange—part 41: integrated generic resource: Fundamentals of product description and support," Geneva: I. TS 184/SC4184, 2005.
- [17] N. Shariatzadeh, G. Sivard, and L. Lindberg, "An approach for manufacturing process representation in product lifecycle management," *Key Engineering Materials*, vol. 572, pp. 239–244, 2013.
- [18] OSLC Core Specification Workgroup, "OSLC core specification version 2.0," Technical report: Open Services for Lifecycle Collaboration, 2010.
- [19] OASIS OSLC, 2015, <http://www.oasis-osl.org/>.
- [20] A. Ryman, *Ryman A. Linked Data, RDF, and OSLC Resource Shapes: Define REST API Contracts for RDF Resource Representations*, 2015.
- [21] T. Berners-Lee, "Linked data design issues," December 2009, <https://www.w3.org/DesignIssues/LinkedData.html>.
- [22] OSLC core, 2016, <https://tools.oasis-open.org/version-control/svn/osl-core/trunk/specs/osl-core.html>.
- [23] Open-services.net. OSLC Primer—Open Services for Lifecycle Collaboration, 2015, <http://open-services.net/resources/tutorials/osl-primer/>.
- [24] ISO, *ISO 10303-21: Industrial Automation Systems and Integration-Product Data Representation and Exchange—Part 21: Implementation Methods: Clear Text Encoding of the Exchange Structure*, International Standards Organization (ISO), Geneva, Switzerland, 2002.
- [25] "OASIS PLCS TC, DEXLIB for PLCS, Review," 2016 http://www.plcs.org/plcslib/plcslib/sys/dex_index_base.html.
- [26] OASIS PLCS TC, "Templates," 2016, http://www.plcs.org/plcslib/plcslib/sys/template_index_base.html.
- [27] OASIS PLCS TC, Reference data library, http://www.plcs.org/plcslib/plcslib/sys/refdata_index_base.html.
- [28] ISO 10303-41: *Industrial Automation Systems and Integration-Product Data Representation and Exchange—Part 41: Integrated Generic Resource: Fundamentals of Product Description and Support*, International Standards Organization (ISO), Geneva, Switzerland, 2005.
- [29] "OSLC Product Definition 1.1.," <http://open-services.net/specifications/product-definition-1.1/>.
- [30] I. Ishaq, D. Carels, G. Teklemariam et al., "IETF standardization in the field of the internet of things (IoT): a survey," *Journal of Sensor and Actuator Networks*, vol. 2, no. 2, pp. 235–287, 2013.
- [31] W. Yang and X. Xu, "Modelling machine tool data in support of STEP-NC based manufacturing," *International Journal of Computer Integrated Manufacturing*, vol. 21, no. 7, pp. 745–763, 2008.
- [32] ISO/CD 15531-1(2010), *Industrial Automation System and Integration—Industrial Manufacturing Management Data: Part 1: General Overview*, ISO/TC184/SC4/JWG8 N228: ISO/CD 15531-1, 2004.
- [33] ISO, *ISO/CD 15531-43, ISO/TC184/SC4/JWG8 N228: ISO/CD 15531-43: Industrial Automation System and Integration-Manufacturing Management Data Exchange: Part 43: Manufacturing Flow Management Data: Data Model for Flow Monitoring and Manufacturing Data*, ISO, Geneva, Switzerland, 2006.
- [34] "ISO/CD 15531-44, ISO/TC184/SC4/JWG8 N228: ISO/CD 15531-44: Industrial automation system and integration—Manufacturing management data exchange: Part 44: Information modelling for shop floor data acquisition," 2010.
- [35] A. Euler-Chelpin, *Information modelling for the manufacturing system life cycle [Ph.D. thesis]*, Royal Institute of Technology (KTH), Stockholm, Sweden, 2008.
- [36] ISO, "Industrial automation systems and integration—product data representation and exchange—part 22: implementation methods: standard data access interface," ISO 10303-22, International Standards Organization (ISO), Geneva, Switzerland, 1998.
- [37] *ISO 10303-27 Industrial Automation Systems and Integration—Product Data Representation and Exchange—Part 27: Implementation Methods: Java TM Programming Language Binding to*

- the Standard Data Access Interface with Internet/Intranet Extensions*, International Standards Organization (ISO), Geneva, Switzerland, 2000.
- [38] Y.-T. T. Lee, "A Journey in standard development: the core manufacturing simulation data (CMSD) information model," *Journal of Research of the National Institute of Standards and Technology*, vol. 120, pp. 270–279, 2015.
 - [39] ISA, "ISA95: Enterprise-control system integration, part 1: models and terminology," ISA Standard ANSI/ISA-95-2010, 2010.
 - [40] N. Shariatzadeh, L. Lindberg, and G. Sivard, "Rapid production changes through the coordination of factory layout models and activities," *Journal of Applied Mechanical Engineering*, vol. 2, article 126, 2013.
 - [41] P. Fettke and P. Loos, *Reference Modeling for Business Systems Analysis*, IGI Global, Saarbrücken, Germany, 2007.
 - [42] "Using the Structured Analysis and Design technique (SADT) in simulation conceptual modelling," in *Proceedings of the Winter Simulation Conference*, A. Tolk, S. Y. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, Eds., 2014.
 - [43] G. Sivard, N. Shariatzadeh, and L. Lindberg, "Engineering innovation factory," in *Proceedings of the 8th International Conference on Digital Enterprise Technology DET*, 2014.
 - [44] IFC 2x3, buildingSMART international, 2011, <http://www.buildingsmart.com>.
 - [45] "IFC 2x3 ISO/PAS 16739," buildingSMART international, 2011 <http://www.buildingsmart.com>.
 - [46] Royal Institute of Technology (KTH) Research Concept Vehicle (RCV), <https://www.itrl.kth.se/research/itrl-labs/rcv-1.476469>.
 - [47] N. Shariatzadeh, G. Sivard, and D. Chen, "Software evaluation criteria for rapid factory layout planning, design and simulation," *Procedia CIRP*, vol. 3, pp. 299–304, 2012.
 - [48] N. Shariatzadeh, G. Sivard G, and M. Hedlind M, "Material flow data representation and integration based on STEP," *Internetworking Indonesia Journal*, vol. 5, no. 1, pp. 11–19, 2013, http://www.internetworkingindonesia.org/Issues/Vol5-No1-B-2013/ijj_vol5_no1_b_2013_shariatzadeh.pdf.
 - [49] B. Rick, *ActiveX LabVIEW Advanced Programming Techniques*, CRC Press LLC, Boca Raton, Fla, USA, 2001, <http://www.unimiskolc.hu/~amper/progs/labview/manual%20-%20labview%20advanced%20programming%20techniques%20%28bitter%20et%20al%29.pdf>.