Toward Manufacturing System Composability Analysis: A Use Case Scenario

Boonserm (Serm) Kulvatunyou, Evan Wallace, Nenad Ivezic, Yunsu Lee

Systems Integration Division, NIST, Gaithersburg, USA
{serm,ewallace, nivezic, yun-su.lee}@nist.gov

Abstract. Smart manufacturing system will be able to quickly adapt to new and changing requirements, implying that software and hardware components of the manufacturing system need to be easily recomposed. In addition, provision of software applications and components is trending toward distributed, heterogeneous, and cloud-based. However, engineers who need to compose a software system will have difficulty finding and using software components with the right functionality and compatibility without a standard to describe these software components. This paper identifies the need for a reference functional ontology to provide a common way to describe a software component functionality. Such an ontology would lead towards the needed standard to describe the software components. The objective of this paper is to discuss high-level requirements for such a functional ontology and provide an initial use case to illustrate how the functional ontology may enable composability analysis.

Keywords: computer integrated manufacturing, service-oriented manufacturing systems, industrial automation, smart manufacturing, functional ontology

1 Introduction

Smart manufacturing (SM) systems are characterized by the ability to quickly adapt to new and changing requirements induced by disruptions and disturbances [1]. For this reason, SM systems are expected to be dynamically composed from network-connected devices and software components. This would allow SM systems to gather and analyze data from these devices and software components and adapt themselves to respond to disruptions and disturbances [2, 3].

The ability to dynamically compose software components is complicated by the proliferation of network-connected devices (also known as Internet of Things) and the increasingly available software in cloud marketplaces such as Oracle Marketplace, SAP HANA Marketplace, and Nimbis Marketplace[1]. With such marketplaces, virtually any individual can develop and provision software components. Such an open market environment will create heterogeneous, yet overlapping offerings of software

[1] https://cloud.oracle.com/marketplace;http://marketplace.saphana.com/; https://www.nimbisservices.com/marketplace/

components in terms of their functionalities (the things components can do) and compatibility (the technical and pragmatic aspects of components that affect how they can work together). To effectively deal with such heterogeneity we propose to develop a reference functional ontology that leads to a standard that will facilitate common understanding of software component offerings. The objective of this paper is to outline requirements and methodology to develop such a reference functional ontology. Through a use case illustration, the paper demonstrates the application of the ontology to composability analyses and identifies future research direction.

Composability analysis supports the finding and evaluation of components that can be functionally and technically combined to perform a desired task. We envision that tools and methods for composability analysis will be developed to assist system and software engineers in composing SM systems. This analysis may also provide additional information, such as identification of additional components necessary for their interoperation.

For the purpose of clarity, we introduce a few working definitions of essential concepts that will be used in the rest of the paper. Components can be either software or hardware; however, for this initial investigation, this paper focuses only on software components. In this research, service orientation [5] is adopted as a predominant paradigm for manufacturing system composition. That is, software components expose their functionalities through services. For this reason, service is the focal point in the rest of the paper (rather than component).

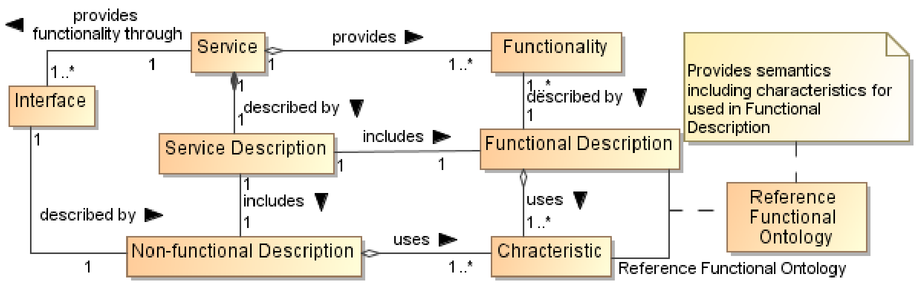

provides functionality through
Service
provides
Functionality
Provides semantics including characteristics for used in Functional Description
Interface
described by
Service Description
includes
Functional Description
described by
includes
uses
Reference Functional Ontology
described by
Non-functional Description
uses
Chracteristic
Reference Functional Ontology
1
1..*

Fig. 1. A service-oriented concept diagram

Fig. 1 illustrates the concepts related to service that will be used in the rest of the paper. A service provides functionalities. Functionality is the ability to do things that is described by a functional description. Functional description (e.g., validate engineering change order) is typically included in the service description that also includes other technical details and constraints (e.g., message exchange pattern, communication and security protocols) about how to access the service through an interface. We label these technical details non-functional description. Functional and non-functional descriptions are described with a set of characteristics that are essentially a set of properties. In this paper, we stress the importance of the functional description part that is much less developed than the non-functional description part (although both should be formalized). Therefore a reference functional ontology should be developed that provides characteristics or properties for the functional description.

In the following sections, we briefly introduce the service-orientation. Then, high-level requirements for the ontology are provided in the form of competency questions. We outline the vision of how the ontology should be developed and maintained. Finally, an integration use case which illustrates the usage of a reference functional ontology is provided before giving the conclusion and describing future work.

2 Service Orientation

For software components to work together, the trend is to allow access to *functions* provided by these software components as services (hence a service can be viewed as a wrapper of a function or functions). This trend has accelerated recently with the arrival of cloud computing, which virtualizes computing and communication resources. That is, service consumers do not need to know how service providers offer their services – from where, by which, or by how many software components.

The service-oriented paradigm is essential to enable SM systems as it emphasizes visibility and semantics that enable (1) matching between needs and functionalities, and (2) composition of service functionalities to address those needs. The visibility and semantics are enabled by service descriptions and service contracts that capture the essential information the service consumers and providers need to be aware of and agree upon [5].

3 Functional Requirements for Functional Ontology

This section discusses high-level functional requirements for a reference functional ontology. For practicality, we divide the requirements for semantically rich functional description via the reference functional ontology into three increasingly-capable classes: composability analysis, change management, and automated composition. Change Management is the ability to use a functional description to automatically reconfigure an existing composition as a result of some changes in the participants. While some of the related works described earlier such as OWL-S aimed at automated composition, our goal is first set upon the composability analysis. For that specific goal, requirements for a reference functional ontology can be expressed with the following competency questions [4].

1. Does a provider's service functionality semantically match, at least partially, the functionality desired by the service consumer's goal?
2. Are the service's non-functional characteristics desired by the service consumer compatible with those of a provider's service, and if not what are the incompatibilities? (The non-functional characteristics include communication protocols, security constraints, pre-conditions, processing capacity, etc.)

4 Vision for Functional Ontology Development

Manufacturing is a large domain, even when considering only software functionalities and not hardware functionalities. In the ISA-95 manufacturing system control architecture [6], functionalities can range from process, sensor, equipment, cell, plant to enterprise levels. They can also vary by industries, types of processes, and products. Top-down, closed-effort ontology development has not prevailed; in the past decade, bottom-up, collaborative ontology development has garnered increased interest. This trend is notable from the following efforts. Linked Data (also known as Linked Open Data or LOD) [7] allows Web documents, Web data, and Web ontology to be linked, annotated, and queried in a structured way. FreeBase is a Web-based open platform that allows the publication of a Web ontology[2]. One of the most popular Web ontology development environments[3] has recently developed the Web version of the tool for collaborative ontology development.

Ontology development should be moving to include a significant bottom-up component [8]. Work in other industries have started to address this need, as reference models are developed to be tailored and/or extended for use by stakeholders [10]. In one of our past works [9], a framework is outlined for evolving a reference model for manufacturing capability information by applying structural canonicalization and semantic gap analysis methods using design patterns and inputs from proprietary models. Such a framework needs to be adapted for the reference functional ontology development.

5 Use Case Illustration of Composability Analysis

This section presents an integration use case as an initial illustration of the composability analysis idea using the envisioned functional ontology (the elements of the functional ontology are shown in italics throughout the tables in this section). The intention is to inspire further research rather than to provide a solution here. The use case sets the stage for gathering, analysis, and validation of requirements for the functional ontology. Ultimately, we will be harvesting this and other use cases to identify the requirements. In the paper, we focus on showing the use of the envisioned functional ontology. The intended use of ontology terms are shown in the relationships and constraints supporting composability analysis (see tables in this section). To enable ontology-based composability analysis, these relationships and constraints will be formalized as ontology axioms in future work, supported by an analysis of required representation and reasoning framework [11].

The industry use case is based on the need to streamline the engineering change management (ECM) process which may be composed as part of other larger processes such as a regular design update process and product fault monitoring process. It is also driven by the change in a company's applications landscape. Let's assume that a company initially only had a manufacturing application, namely MA, to manage product data for design and manufacturing. At this point, the ECM process is *streamlined* within MA. Later on, the company purchased a new application to manage the whole product life cycle information, namely PLMA. PLMA expanded product data management functionality to track not only as-designed and as-manufactured product data but also as-maintained data. The company decided to use PLMA as the master

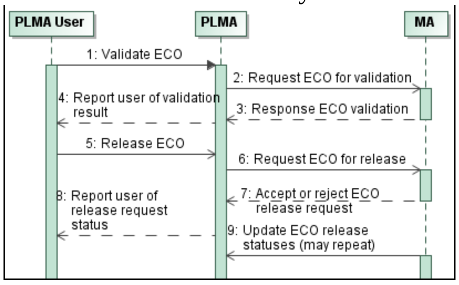
[2] See http://www.freebase.org

[3] See http://protegewiki.stanford.edu/wiki/WebProtege

application for product data management. Therefore, PLMA needs to be integrated with MA in order for the ECM process to be *streamlined* again.

In the first step of the integration, a high-level integration process is captured as part of the requirement documentation. Part of the process is illustrated through the UML sequence diagram in **Fig. 2**.

It should be noted that because this is an existing process, all the relevant functionalities and associated logic are known, i.e., there is no composition to create a new functionality in this case. Therefore, only the information necessary to invoke services with those functionalities on the MA side and the information necessary to inform PLMA of the result need to be identified (as opposed to the case when the integration involves building a new functionality wherein information necessary to achieve the new functionality is also necessary to be identified). It is determined that the Engineering Change Order (ECO) entity/object supports all the information necessary to both invoke the functionality on the MA and PLMA systems.


PLMA User
PLMA
MA
1: Validate ECO
2: Request ECO for validation
3: Response ECO validation
4: Report user of validation result
5: Release ECO
6: Request ECO for release
7: Accept or reject ECO release request
8: Report user of release request status
9: Update ECO release statuses (may repeat)
Fig. 2. UML sequence diagram of the integration requirement

PLMA native service	**MA native service**
Export ECO outbound	Process ECO inbound
Import ECO inbound	Process ECO response outbound

Table 1. PLMA and MA native services

For a composability analysis, details of required services from each application are derived from the integration process requirement. Examples of service requirements are summarized in the last column on the right of **Table 2** and **Table 3**[4] (the first column indicates the characteristics of the service). The functional ontology should support representation of this service requirement. It should be noted that only few characteristics of the services are illustrated here. Additional characteristics will need to be modeled such as supported transmission protocol and other essential functional characteristics. Essential characteristics are artifacts that are expected to evolve over time as demanded by the users of services and the technological evolution.

Table 1 illustrates PLMA's and MA's native services. These are existing services on the two applications that need to be adapted to the service requirements. In a (service-oriented) smart manufacturing system environment, the service descriptions of these native services should be registered in a commonly accessible registry using the reference functional ontology to describe their functionalities. This is illustrated in the

[4] Note that only application to application interactions are accounted for. Also, we assume the following mapping from **Fig. 2** terminology to the last column of **Table 2** and **Table 3** terminology: Release → Create and Update → Notify. This mapping reflects presumed different terminologies used in integration and service requirements identification stages.

second column of **Table 2** and **Table 3**[5]. These tables contain a row for each of the services needed from the components in focus for the ECM process in our example. Below each service row are 3 more rows that describe characteristics of the interface provided by the component to support the service. The first of these rows indicates whether the service supports or requires list operations (List Oper). The second row indicates the kind of message exchange pattern (MEP) supported or needed. The final interface row describes the type of message (Msg) supported by the service.

Table 2. Composability analysis of PLMA services

Service Characteristic	**PLMA's native service registration**	**Δ**	**PLMA Required service**
Service	Export ECO outbound ⊃ *Validate ECO request outbound*	√	*Validate ECO request outbound*
List Oper	Y	≥	N
MEP	*Request Only*	√	*Request Only*
Msg	plmaECO ⊃ (*Validate ECO* ∪ *Validate ECO Response* ∪ *Create ECO* ∪ *Create ECO Response* ∪ *Engineering Change Notice*)	≥	*Validate ECO*

Table 3. Composability analysis of MA services

Service Characteristics	**MA's native service registration**	**Δ**	**MA Required service**
Service	Process ECO inbound [@action = 'simulate'] = *Validate ECO request inbound*	√	*Validate ECO request inbound*
List Oper	Y	≥	N
MEP	*Request Only*	√	*Request Only*
Msg	maECO ⊃ (*Validate ECO* ∪ *Validate ECO Response* ∪ *Create ECO* ∪ *Create ECO Response* ∪ *Engineering Change Notice*)	≥	*Validate ECO*
Service	Process ECO response outbound [@action = 'notify'] = *Notify ECO outbound*	√	*Notify ECO outbound*
List Oper	Y	≥	N
MEP	*Async Request Response* || *Request Only*	≥	*Request Only*
Msg	maECO ⊃ (*Validate ECO* ∪ *Validate ECO Response* ∪ *Create ECO* ∪ *Create ECO Response* ∪ *Engineering Change Notice*)	≥	*Create ECO*

The second column of each of these tables is used to describe native services provided by the components. The description in this column identifies the particular service variant used, the message used to carry information for the service; and then maps these to descriptions created from elements of a reference functional ontology.

[5] In this paper, we do not attempt to formalize any semantics of relationships between services and characteristics including the relationships in the Δ column.

The elements of the ontology are italicized for clarity. A ⊃ indicates that the native component element provides a superset of (i.e., subsumes) the content or functionality required. The fourth column contains a description of the required service as defined in the ECM process. The **Δ** (third) column shows the relationships between the description of the registered native service in column 2 and that of the required service for each characteristic (in column 4). Two symbols are used to indicate the relationship. A '√' indicates that the native service or interface element can satisfy the required service or interface element. A '≥' indicates that the native component has greater capability than needed but can be made to match the element.

Taking the service registration in **Table 2** as an example, the first two columns indicate that the Export ECO outbound native service functionally subsumes the Validate ECO request outbound service; it is a list operation (List Oper), it supports the Request Only MEP; and its message definition (Msg), plmaECO, subsumes several messages including Validate ECO, Validate ECO Response, Create ECO, etc. In the Δ column, the '√' relationship on the Service row indicates that the Export ECO outbound native service and the Validate ECO request outbound matches; the '≥' in the List Oper row indicates that the native service is more capable than the required service; the '√' in the MEP row indicates that the native service matches with the required service and no adaptation is necessary; and the '≥' in the Msg row indicates that the message definition of the native service is more capable than that of the required service.

With these relationships established, the composability can be assessed. Firstly, the relationships between the Services and between the MEPs indicate that the native service can be directly piped into the required service without adaptation. Secondly, the required service does not need the list operation capability which means that there is no chunking or de-chunking required (in the mediator component that connects the PLMA with the MA service). Lastly, the fact that plmaECO subsumes the Validate ECO indicates that there will be adaptation needed – data transformation.

With the native service registration described in terms of the reference ontology, we expect that the relationship can be automatically determined. Determination of the relationship of some service characteristics will be more involved than others. For example, evaluating the relationship between messages of two services will generally require detailed semantic mapping analysis or semantic distance measurements.

The composability analysis should also be automatable. The algorithm will be specific to each service characteristic because of the differences in their underlying semantics. To enable such automation, additional semantics associated with expressing the integration requirement (e.g., required services) will need to be modeled. For example, although the List Oper row of the last service in **Table 3** has the same information as that in the service in **Table 2**, de-chunking in the mediator may be necessary in the former case, while it may not be necessary in the latter. This is because, in the former case, it may not be controllable at MA to send the notification for a single ECO at a time as this is an existing automated process, while in the latter case it may be the integration behavior that the design engineer will send a single ECO at a time for validation purposes.

6 Conclusion and Future Work

This paper proposes development of a reference functional ontology. The hypothesis is that the availability of such an ontology will allow for more efficient and effective manufacturing system composition. The ontology is expected to be evolving; and hence, new development mechanisms have to also be developed. The purpose of the reference functional ontology is to allow for expressive descriptions of service's functionalities. The reference functional ontology along with other reference ontologies for non-functional characteristics will enable composability analysis of services as illustrated in a use case scenario of engineering change management integration. The use case provides an initial substantiation of the composability analysis idea. In terms of future work, we plan to develop more use cases, especially those that involve lower level manufacturing control functions (in the ISA-95 layers) and those that involve hardware equipment. High-level concepts in the reference functional ontology are being developed; and encoding of ISA-95 and the SIMA reference activities into the ontology is also being experimented with. In order to precisely encode the ontology, we will formally define key concepts, such as composition, and develop measurement methods and metrics for the composability analysis.

Disclaimer

Any mention of commercial products is for information only; it does not imply recommendation or endorsement by NIST.

References

1. SMLC, Implementing 21st century smart manufacturing, Workshop summary 2011.
2. Pellet, J. 2013. Lessons learned from chief executive's manufacturing summit. Chief Executive Magazine, July/August 2013.
3. Floerkemeier, c. et al. 2008. Preface to the IOT Conference 2008, Springer
4. Gruninger, M., & Fox, M. 1994. The role of competency questions in enterprise engineering. Proc. of the IFIP WG5.7, pp. 212-221.
5. OASIS Reference Model for Service Oriented Architecture 1.0.
6. ANSI/ISA-95 (IEC 62264) Enterprise-Control System Integration - Part 1:2010, Part 2:2010, Part 3: 2013, Part 4:2012, Part 5:2013.
7. Bizer, C., et al. 2009. Linked Data—The Story So Far. International Journal on Semantic Web and Information Systems 5 (3): 1–22.
8. Recommendations for implementing the strategic initiative INDUSTRIE 4.0. 2013. Securing the future of German manufacturing industry. Acatech, Germany.
9. Lee, Y., & Peng, Y. 2013. A Framework for Developing Manufacturing Service Capability Information Model. Proc. Of IFIP Advances in Production Management Systems Conference, 414: 325-333.
10. CISCO Systems, Inc. Introduction to eTOM – White Paper.
11. Fiorentini, X., et al. 2010. An Analysis of Description Logic Augmented with Domain Rules for the Development of Product Models. *Journal of computing and information science in engineering*, *10*(2).