Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online
CONFERENCE PAPER ARCHIVE

IFAC-PapersOnLine 48-3 (2015) 288–293

Towards a Cloud Manufacturing systems modeling methodology

A. Talhi [*] JC. Huet [**] V. Fortineau [*] S. Lamouri [*]

[*] *Arts et métiers, ParisTech, France*
(e-mail: asma.talhi@ensam.eu
virginie.fortineau@ensam.eu
samir.lamouri@ensam.eu)
[**] *ECAM-EPMI, Université Paris Seine,*
Cergy, France
(e-mail: jc.huet@ecam-epmi.fr)

Abstract: In this study we present an adaptation of ASDI (Analysis-Specification-Design-Impelmenation) methodology to the Cloud Manufacturing domain. Cloud Manufacturing is an emerging paradigm in which dynamically scalable and virtualized manufacturing resources are provided to the users as services over the Internet. In order to implement a Cloud Manufacturing platform that will map manufacturing users and providers we propose a modeling methodology named ASDI-Onto. It uses ontologies as modeling approaches to model the Cloud Manufacturing domain and introduce the outlines of the future work.

Keywords: Ontology, Cloud Manufacturing, ASDI, modeling

1. INTRODUCTION

Cloud Manufacturing (CM) is a new paradigm where all resources and abilities involved in the whole lifecycle -Hardware/software- are provided to the users in a pay-as-you-go manner. Based on novel technologies like Service Oriented Architecture (SOA) and Cloud computing, CM is an emerging solution where users can request services ranging from product design, manufacturing, testing, management and all other stages of a product lifecycle. The CM term was taken as reported in the literature. Although it gives the impression of treating the manufacturing level, theoretically, CM all phases of lifecycle that are provided as services.
The Cloud Manufacturing system is based on multi-layer framework that includes(Xu (2012)):

- Manufacturing Resource Layer (MRL): contains the resources needed during the lifecycle;
- Virtual Service Layer (VSL): where identified manufacturing resources are virtualized and packaged as a service;
- Global Service Layer (GSL): responsible for locating, allocating, and monitoring the manufacturing resources;
- Application Layer/User Domain (UD): serves as an interface between the user and manufacturing cloud resources.

Our objective is to define a CM architecture that will map users and providers that deal with industrial information systems within PLM (Product Lifecycle Management) context. PLM is an approach that allows the management of the product's data in such environments. According to Terzi (2005) it is an integrated, Information and Communication Technology (ICT) supported, approach to the cooperative management of all product related data along the various phases of the product lifecycle. As a result, there is a need of using a structured methodology in order to achieve our goal.

However, papers that deal with Cloud Computing and CM domains do not provide a methodology that explains in steps how to implement a robust architecture efficiently.

In this study we present a modeling methodology aiming to built a CM platform for mapping users and service providers in order to achieve PLM projects with best delays and costs.

The rest of the paper is organized as follow: section 2 presents a synthetic review of the Cloud Manufacturing models and modeling approaches followed by the introduction of the chosen methodology: ASDI (Analysis-Specification-Design-Impelmenation). Sections 3 and 4 depict the adaptation of ASDI methodology to our domain and a proposition of CM model and its validation. Finally section 5 concludes this study and outlines the future work.

2. RELATED WORK

2.1 Cloud Manufacturing models and modeling approaches

We have shown in a previous work (Talhi et al. (2014)) that form the global service layer point of view there is no model that fits our needs. Indeed, the models found in the state of the art we conducted focus on the "provider layer" in order to model resources and operators in shop floor or to provide framework of knowledge integration in networked manufacturing. Our aim is to provide a

2405-8963 © 2015, IFAC (International Federation of Automatic Control) Hosting by Elsevier Ltd. All rights reserved.
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2015.06.096

platform where every phase of the product lifecycle is provided as a service. This makes our model service-centric since it is the concept that links users and providers. Vincent Wang and Xu (2013) developed data models for manufacturing service, service provider and service request using EXPRESS. Since that EXPRESS provides portability with STEP data models, it is not ideal for processing heterogeneous data generated from multiple resources over the Internet. Lu et al. (2014) use an ontology within a Cloud management engine to mange user-defined clouds. The ontology is used to instantiate companies on which the user executes a customised rule to create a Cloud and define users authorized to access this Cloud.

The studies above depict models that describe the related domains without any methodology to explain how to efficiently use these models. We want to use our ontology in an efficient way to benefit from semantic advantage and inference mechanism to built a robust CM platform. Then, there is a need of a methodology that explains explicitly how to deal with the studied domain in order to achieve our goal and implement a CM platform. Zellner (2011) defines a methodology as a set of five mandatory elements:

(1) Procedure model: order of activities to be fulfilled when employing the method.
(2) Technique: way of generating results; supports an activity.
(3) Results: an artifact (e.g. a document, etc.) created by an activity.
(4) Role: the one who carries out the activity and is responsible for it.
(5) Information model: consists of the above-described elements and their relationships. Information models are also used to represent the results.

We have found in the literature a methodology that fulfills these features and meet our needs: ASDI (Analysis-Specification-Design-Implementation). This methodology is described below.

2.2 ASDI Methodology

Gourgand and Kellert (1992) proposed a methodology called ASDI (Analysis-Specification-Design-Implementation) used for the design and the implementation of modeling, simulation and piloting software environments dedicated to a domain (class of systems). To obtain knowledge models of complex systems, ASDI recommends Chabrol et al. (2008) a systemic decomposition of the studied system in three communicating subsystems inspired by Le Moigne (1992): (i) the Physical Subsystem (PSS) defines the physical entities set (which could concern different fields, such as production, storage, handling and transport), their geographical distribution and the links between them; (ii) the Logical Subsystem (LSS) (called also informational subsystem in some simulation methodologies) represents the flows of entities which have to be handled by the system, along with the set of operations concerning these flows, and the nomenclatures which refer to this set; (iii) the Decision-making Subsystem (DSS) contains the management and working rules of the system. The Analysis and Specification phases allow us to obtain the generic modeling of a domain and the Design and Implementation phases allow us to create a library of reusable software components of this domain.

Recently, several authors propose adaptations for different utilizations of ASDI:

- Féniès et al. (2006) see the Health Care Systems as a Health care Supply Chain (HSC). They define HSC as an open set, crossed by human, material, informational and financial flows. They propose a modelling methodology for Supply Chain Evaluation named ASDI-HSC.
- Chabrol et al. (2008) develop a methodology to design decision making aid tools based on various resolution methods (mathematical formalization, simulation, etc.), which all start by the formalization of a knowledge model (knowledge formalization) related to the studied system. They present the different steps to follow from the knowledge model to the creation of decision making aid tools, along with two applications concerning a surgical unit and an obstetrical unit.
- Chabrol et al. (2006) propose to use ASDI for the Traffic Multi-Agent (ASDI-mi), their goal is to build a software environment and a decision aid making for Urban Traffic Systems. Their methodology has been developed to use Model combination (in order to obtain what is called a combined or a hybrid model). Models can represent different levels of detail and different temporal horizons. They use mainly UML language.
- El Haouzi et al. (2008) explain that the requirements for manufacturing control evolve from traditional centralized approaches where decision making is hierarchically broadcast to more complex distributed control architectures involving autonomous entities and processes. In order to evaluate these new architectures, they wanted to use a discrete-event simulation. However, the complexity of distributed architectures and Demand Flow Technology (DFT) standardization requires the introduction of modularity and reusability in the modelling process. Thus, they proposed a methodological approach, based on ASDI, to develop a library of generic simulation components that can be instantiated as automatically as possible into a modular simulation model. The most important improvement they propose is to enhance the systemic decomposition with a diagram which describes the autonomous processes. They use also UML language.
- Huet et al. (2013) propose a methodology inspired by the component-based approach, the paradigm of system of systems (SoS) and the methodology ASDI. This methodology makes possible to build an action model with the aim of evaluating the performance of a Holonic Manufacturing System. Their goal is also to manage the decentralized and centralized decision making process. UML diagrams are also use.
- Huet and El Abbassi (2013) use this methodology on Green Cloud Computing (GCC) system. Due to limited energy, environmental problems and the fast growth of computer power consumption, the design, study and management of a GCC system are increasingly difficult and expensive as existing configurations are multiple and complex. Several approaches exist for modelling and simulating GCC. Current software environments try to take into account as many phenomena and facilities as possible according

to a planning level (strategic, tactical or operational). In order to tackle with this complexity, they propose a modelling methodology aiming to build a software environment. They use UML as modelling language to model the management and control system.
- Royer et al. (2014) propose to use an other modelling language : Business Process Model and Notation (BPMN). Their aim is to build decision-making tools to help the pharmacists to reorganise the medication use-process in a hospital.

This methodology has two very interesting elements which are linked:

- The use of library components that can be reused,
- It enables the performance evaluation of a scenario and hence, the building of generic decision making tools.

3. PROPOSED METHODOLOGY

3.1 ASDI adaptation: ASDI-Onto

The proposed methodology ASDI-Onto is an extension of ASDI in order to support the establishement of CM platform. Hereafter we explain how we have adapted ASDI to our domain study and our objective. With the aim to build a CM architecture, using ASDI provides opportunity for studying the systems belonging to the domain. To do so, in our ASDI modeling we propose to deal with two abstraction levels : The analysis and desing/implementation levels for the domain of study. We do not take into account the specification step, where functions and behavioural model of system objects have to be described. Since we deal with the CM from the Global Service Layer point of view, then the behavior of the components is the charge of service providers. ASDI-Onto presented in this paper proposes the use of ontologies to develop the generic knowledge model, and further the implementation of this model using Cloud simulation systems to build the library components.

Figure 1 depicts the adaptation of ADSI to our domain. The CM ontology, is built during the analysis and specification step, it's our generic knowledge model. The phase of building the domain model is based on enumerating all the entities needed to describe the domain and the relationships between them (Analysis). Our ontology responds to these requirement since it includes all the common concepts, find in the literature, that can be used to describe the CM and their relationships.

Moreover, the library of components of ASDI-Onto is the CM architecture which is built using Xu (2012) definition of CM system. Our objective is to define a CM platform that will match providers and users in a way that the latter find the suited service to his request. The Design phase consists in modifying existing Cloud Computing implementation in order to add concepts related to CM and all the steps of the product lifecycle. The implementation is the final step, based on the Design one, it enables the implementation of CM architecture which is the action model in the ASDI methodology. Since in the CM everything is seen as a service, our implementation has

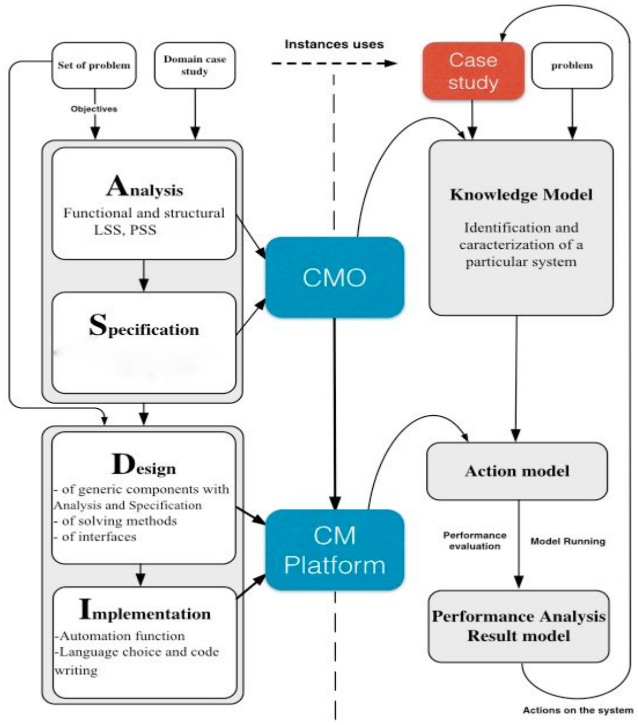


Fig. 1. ASDI-Onto

to guarantee the integration of various resources organized as a set of services offering a modularity and reusability advantages to the system.

In the following, we focus on the analysis/specification of the domain in order to provide a generic knowledge model: the CM Ontology.

3.2 Domain analysis

The Generic Model Knowledge is built during the Analysis phase and is dedicated to the domain. It gathers all the systems by identifying their common points according to the entities they contain and their interactions. Simon (1962) defines a complex system as a system made up of a large number of parts that interact in a non simple way. We believe that the Cloud Manufacturing system is a complex one since it maps a set of actors who, according to their statutes, are providers or users of resources virtualized in services that, in addition, handles all stages of product life management. According to the decomposition of Le Moigne (1992), our ontology includes the PSS and LSS since it models the different resources, and the services offered by resources's owners or used by the consumers. The DSS is not included in the ontology since it must be as generic as possible and in our case rules management are specific to each enterprise. Indeed, management rules can be for example the company's decision concerning the visibility of its services to the system's actors. This company can restrict access to its service if the user is a competitor and the choice of confidentiality rules are not always the same for all the domain's system.

The choice of ontologies as a modeling tool is motivated by the following reasons:

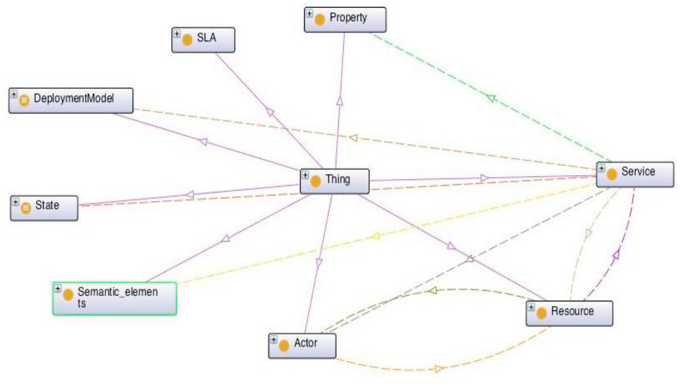


Fig. 2. Top level of CM ontology

- Ontologies provide a simplified and understandable view of the domain since they represent explicitly the meaning of concepts and their relationships. Moscato et al. (2011) explain that different Cloud systems and vendors have different ways to describe and invoke their services, to specify requirements and to communicate. Therefore, the definition of a CM ontology allows us to overcome this problem because it represents the common concepts suitable to be used by the different cloud providers.
- Thanks to the inference mechanism, the ontology automates the discovery services by performing a semantic map- page between the user query and services described. Tsai et al. (2008) argue that, the reasoning abilities provided by the ontology systems; in oriented-service frameworks, can facilitate the service matching process, and provide a certain level of flexibility by returning the most compatible services when a perfect match cannot be found, and reduce the manual work for a user.
- Fortineau et al. (2013) explain that inference ontologies deal with non canonic data, which makes it possible to have individuals in different classes at the same time (and also to merge different points of view of the same object). This is another advantage of inference ontologies in our study: Let the individual X be a design software proposed by some provider, and let the concepts "DesignaaS" (Design as a Service), "Saas" (Software as a Service) be in the ontology. X must belong to the two classes so that the mediator returns this service as a response if the user employs one of these terms in his request.

Typesetting conventions used in this paper are: **Class**, individual and *property*. The top level of the ontology that constitutes the generic knowledge model (Figure 2) is composed of **Actor** class represents the actors that interact with the Cloud Manufacturing system. **Deployment Model** for the Cloud Manufacturing deployment's mode, this class contains the following instances: Community, Hybrid, Private, Public. **Semantic_elements** class contains all the elements needed to allow and facilitate the communication with the service and its use. It consists of: **Data_language, Programming_language, Protocol**. **SLA** class includes all the elements needed to define the SLA. **State** describes the state of the service and contains the following individuals: free, unavailable, used. **Property** class is divided into two subclasses: **functional_property** and **Non_functional_property**. The **functional_property** class describes the elements needed to run a service and **Non_functional_property** class describes properties that will be considered, in addition to functional ones to decide which service is most suited for a particular user such like **Availability, Qos, reliability**. **Resource** class describes the resources that will be packaged as service and contains two subclasses **Hard_resource** that combines **Manufacturing_equipment** and **Computational_resources**, and **Soft_resource** that include non material resources like **Software, Experience, Skill**. **Service** is the Cloud system's centric entity. In Cloud Manufacturing all the resources are virtualized and packaged as a service. The **Service** class contains subclasses **PLMaas, ServiceModel**. The **PLMaas** class includes PLM phases like: Design as a service (**DaaS**), Manufacturing as a service (**MFGaas**), Simulation as a service (**SIMaas**), etc. The **ServiceModel** class describes the service's categories: **Saas, Paas, Iaas, Holonaas** where the latter is used to describe the service that encompass manufacturing resources. Indeed, according to the holon's architecture introduced by Bussmann (1998) we notice that a holon has a "logical" part that allows the control and the communication with the physical part. We believe that this definition is suitable for the "manufacturing" part of the Cloud Manufacturing since our aim is to make physical entities interact with other services and remotely accessible.

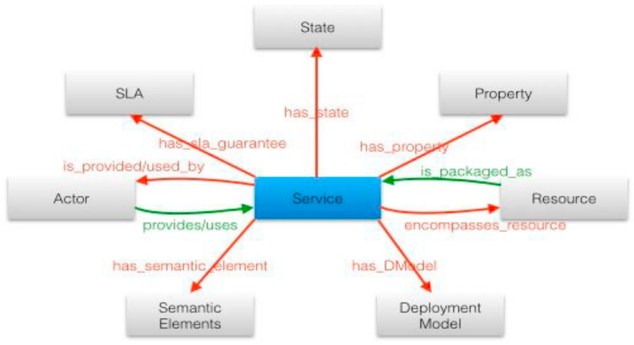


Fig. 3. Semantic links between the top-level concepts

Figure 3 depicts explicitly the semantic links between the top-level concepts.

The model is service-centric since the service is the most important entity that maps the users and the providers. The **Actor** individuals can either provide or consume a service **Service**. The **State** individuals specify the state of the **Service** whether it is used, free or on maintenance. The **Property** classe and subclasses define the **Service** functional and non-functional characteristics. The **Deployment_model** adresses the deployment mode of the Cloud (public, private or hybrid) where the **Service** is provided. **Semantic_elements** specifies the means of communication with the **Service** like the data format handled and generated. Resources belong to an **Actor** (enterprise for instance) and are virtualized and packaged as a **Service**.

In addition to semantic relationships between concepts, the CM ontology includes also SWRL rules. SWRL (Semantic Web Rules Language) is language that completes the expressivity of OWL with rules based on the RuleML paradigm Fortineau et al. (2014). These rules make explicit the implicit links between instances using logical axioms. In our case, between a resource, a service, and a supplier: *is_owned_by(?y, ?x), is_packaged_as(?y, ?z)* $\rightarrow$ *is_provided-_by(?z, ?x)*

This rule means that if a resource "y" belongs to a supplier "x", and the resource "y" is virtualized as a service "z" then service "z" is provided by "x".

4. MODEL VALIDATION

The application use case is a CM plateforme as a part of a project in collaboration with a french company. The validation of our ontology is a first step towards the achievement of this objective. We have performed unit tests based on industrial scenarios to validate the CMO. Our generic knowledge model is being validated by populating the ontology with a number of instances and performing inference mechanism on them. This mechanism allows us to infer implicit information with few explicit knowledge. This is one motivation for choosing ontologies since this mechanism allows to reduce the modeling commitment and thus providing time saving and facilitate futur model integration. Here after we present one of the unit tests performed on the ontology.

We have defined an **Enterprise** type individual named EnterpriseA, a software individual: Mgsoftware that is **Manufacturing_software**. We have also defined individuals service3 and User1 without specifying their types in order to verify whether the inference mechanism will figure out that they are respectively **service** and **Consumer** to validate the CMO. The individuals are defined as follows:

- EnterpriseA *owns_resource* Mgsoftware
- User1 *use_cm_service* service3
- service3 *encompasses_software* Mgsoftware

We have also used the rule presented in section 3.2 and finally we defined two classes as :

- **Consumer** $\equiv$ **Actor** and (*use_cm_service* some **Service**)
- **Manufacturing_aas** $\equiv$ **Service** and (*encompasses-_resource* some **Manufacturing_resource**)

Figures hereafter show the information inferred using the definitions above. For exemple, for the individual service3 we have only specified that it *encompasses_software* Mgsoftware and we see from Figure 4 that the information inferred is :

- service3 $\in$ **Manufacturing_aas**
- service3 $\in$ **Saas**
- service3 *is_used_by* User1
- service3 *encompasses_resource* MgSoftware
- service3 *is_provided_by* EnterpriseA

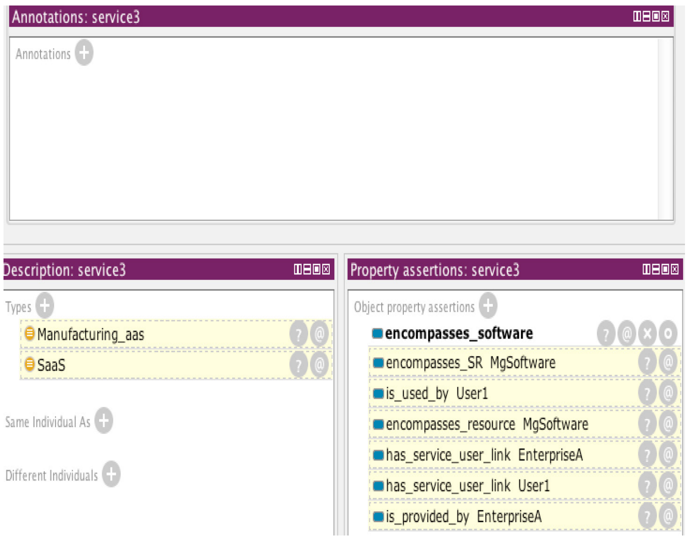

Annotations: service3
Annotations
escription: service3
Types
Manufacturing_aas
SaaS
Same Individual As
Different Individuals
Property assertions: service3
Object property assertions
encompasses_software
encompasses_SR MgSoftware
is_used_by User1
encompasses_resource MgSoftware
has_service_user_link EnterpriseA
has_service_user_link User1
is_provided_by EnterpriseA

Fig. 4. Inference results on individual Service3

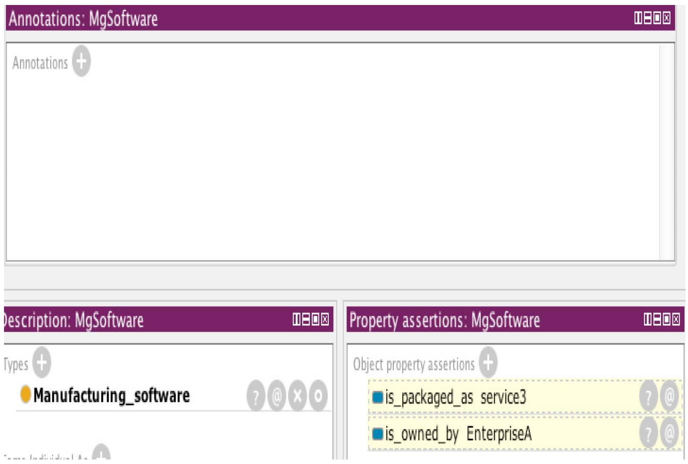

Annotations: MgSoftware
Annotations
escription: MgSoftware
Types
Manufacturing_software
Property assertions: MgSoftware
Object property assertions
is_packaged_as service3
is_owned_by EnterpriseA

Fig. 5. Inference results on individual MgSoftware

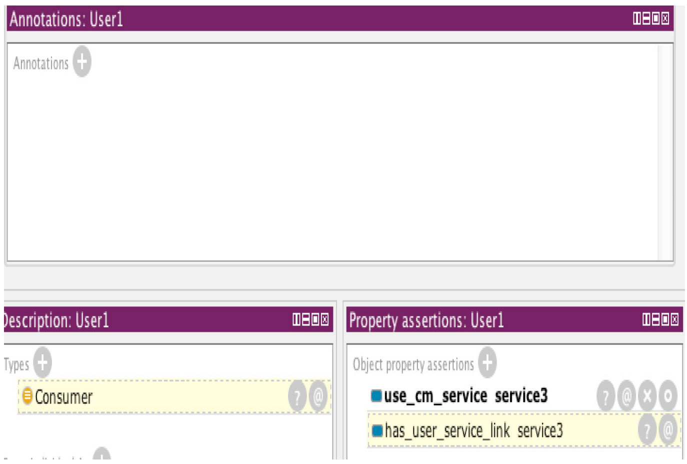
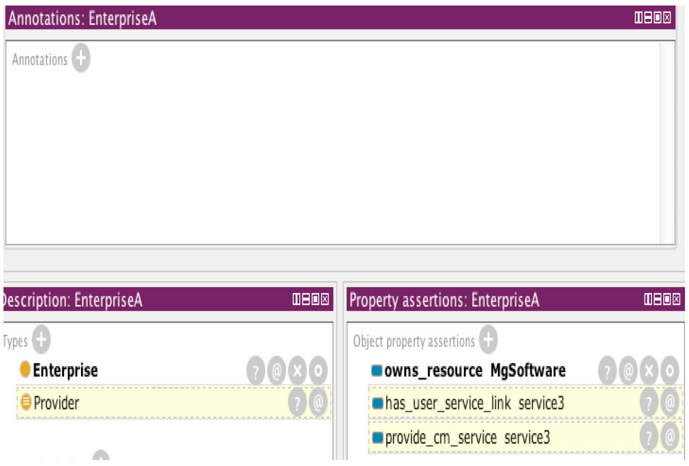

Annotations: User1
Annotations
escription: User1
Types
Consumer
Property assertions: User1
Object property assertions
use_cm_service service3
has_user_service_link service3

Fig. 6. Inference results on individual User1

5. DISCUSSION AND CONCLUSION

In this study we presented an adaptation of ASDI methodology ASDI-Onto with the objective to implement a Cloud Manufacturing Platform. The ASDI-Onto is based on two major steps : domain Analysis and Specification, Design



Enterprise
Provider
owns_resource MgSoftware
has_user_service_link service3
provide_cm_service service3

Fig. 7. Inference results on individual EnterpriseA

and Implementation. We have used ontologies to model our domain, the Cloud Manufacturing. We intend that our ontology will be as generic as possible, however more concepts can be added. This ontology is being validated using instantiation process. With the inference mechanism offered by ontologies, we demonstrated that the ontology has inferred implicit information with few knowledge. At this level, the choice of ontologies helps us to built a Generic Knowledge model that depicts CM concepts and the relationships between them taking into account the semantic aspect of this approach. The future work will focus on the design and implementation phase in order to provide a library of reusable components. This library in our case will represent the CM architecture components that will be used to implement a CM platform. This platform will map CM users and providers in a way that a user car choose between a set of service which ones will meet its needs. ASDI-Onto will further be validated by applying it on a real case scenario in order to help companies in the process of moving towards the CM.

REFERENCES

Bussmann, S. (1998). An agent-oriented architecture for holonic manufacturing control. In *Proccedings of the 1st international workshop on Intelligent Manufacturing Systems, IMS-Europe.*

Chabrol, M., Gourgand, M., and Rodier, S. (2008). A modeling methodology and its application to the design of decision-making aid tools dedicated to the hospital systems. In *RCIS'08:International Conference on Research Challenges in Information.*

Chabrol, M., Sarramia, D., and Tchernev, N. (2006). Urban traffic systems modelling methodology. *International Journal of Production Economics*, 99, 156 – 176.

El Haouzi, H., Thomas, A., and Pétin, J.F. (2008). Contribution to reusability and modularity of manufacturing systems simulation models: Application to distributed control simulation within dft context. *International Journal Production Economics*, 112, 48–61.

Fortineau, V., Fiorentini, X., Paviot, T., Louis-Sidney, L., and Lamouri, S. (2014). Expressing formal rules within ontology-based models using swrl: an application to the nuclear industry. *In International Journal of Product Lifecycle Management (IJPLM)*, 7(1), 75–93.

Fortineau, V., Paviot, T., and Lamouri, S. (2013). Improving the interoperability of industrial information systems with description logic-based models?The state of the art. *Computers in Industry*, 64(4), 363–375. doi: 10.1016/j.compind.2013.01.001.

Féniès, P., Gourgand, M., and Rodier, S. (2006). Interoperable and multi-flow software environment: Application to health care supply chain. In J. Eder and S. Dustdar (eds.), *Business Process Management Workshops*, volume 4103 of *Lecture Notes in Computer Science*, 311–322. Springer Berlin Heidelberg.

Gourgand, M. and Kellert, P. (1992). An object-oriented methodology for manufacturing system modelling. In *Summer Computer Simulation Conference*, 1123–1128. Reno, Nevada, USA.

Huet, J.C. and El Abbassi, I. (2013). Green cloud computing modelling methodology. In *2013 IEEE/ACM 6th International Conference on Utility and Cloud Computing (UCC)*, 339–344. doi:10.1109/UCC.2013.73.

Huet, J.C., Paris, J.L., Kouiss, K., and Gourgand, M. (2013). A new reengineering methodology for the product-driven system applied to the medication-use process. *Decision Support Systems*, 55(2), 599 – 615.

Le Moigne, J.L. (1992). *La modélisation des systèmes complexes.*

Lu, Y., Xu, X., and Xu, J. (2014). Development of a Hybrid Manufacturing Cloud. *Journal of Manufacturing Systems*. doi:10.1016/j.jmsy.2014.05.003.

Moscato, F., Martino, B.D., and Munteanu, V. (2011). An Analysis of mOSAIC ontology for Cloud Resources annotation. In *Proceedings of the Federated Conference on Computer Science and Information Systems (FedCSIS), IEEE, Szczecin, Poland,*, 973–980.

Royer, J., Chabrol, M., and Paris, J.L. (2014). Process modelling and simulation for medication-use process. In *28th EUROPEAN Conference on Modelling and Simulation (ECMS)*, 7 pages. Brescia, Italy.

Simon, H.A. (1962). The architecture of complexity. In *Proceedings of the American Philosophical Society*, 6, 106(6) :467–482.

Talhi, A., Huet, J., Fortineau, V., and Lamouri, S. (2014). Sohoma'14: Internation workshoptoward an ontology-based architecture for cloud manufacturing. In *Service Orientation in Holonic and Multi-Agent Manufacturing. Nancy, France.*

Terzi, S. (2005). *Elements of Product Lifecycle Management: Definitions, Open Issues and Reference Models.* Ph.D. thesis, Université Henri Poincaré, Nancy-I.

Tsai, W., Sun, X., Huang, Q., and Karatza, H. (2008). An ontology-based collaborative service-oriented simulation framework with Microsoft Robotics Studio®. *Simulation Modelling Practice and Theory*, 16(9), 1392–1414. doi:10.1016/j.simpat.2008.07.007.

Vincent Wang, X. and Xu, X.W. (2013). An interoperable solution for cloud manufacturing. *Robotics and Computer-Integrated Manufacturing*, 29(4), 232–247.

Xu, X. (2012). From cloud computing to cloud manufacturing. *Robotics and Computer-Integrated Manufacturing*, 28(1), 75–86. doi:10.1016/j.rcim.2011.07.002.

Zellner, G. (2011). A structured evaluation of business process improvement approaches. *Business Process Management Journal*, (2), 203–237. doi: 10.1108/14637151111122329.