Performance Evaluation and Improving Semantic Web Service Discovery for Advanced Manufacturing Collaboration in Cloud Platform

Prof. Rajesh Gaikwad
Computer Engineering
Atharva COE
India

Prof. Mahendra Patil
Computer Engineering
Atharva COE
India

Prof. Dhananjay Dakhane
Computer Science and Engineering
Sipna' COET
India

Prof. Satish Ranbhise
Computer Engineering
Atharva COE
India

***Abstract*—Today's Web provides static and structured information for its users. Most of its contents are stored in the database. From a machine point of view a information having associated information is very difficult to be processed and hence information on the web cannot be manipulated by a computer. This paper tries to find out technologies used to make web semantic in manufacturing Industries.**

Keywords—Performance evaluation,Semantic Web, Cloud manufacturing

I. INTRODUCTION

Semantic Web and Service Web

The current Web is aimed at providing information and services that are directly consumed by human beings. Although most of the content is stored in databases, the information is presented without the structural information found in databases. From a machine point of view, it is very difficult to process all this information, so the Web cannot be automatically manipulated by computers. The Semantic Web (SW) vision is an extension to current Web, where information has associated semantics that can be processed and understood by machines. Thus, using Semantic Web (SW) technologies, all this information can be automatically processed, extracting knowledge to feed algorithms, in order to allow autonomous interaction between computers, and to improve the cooperation between people and computers. Unfortunately, this vision has not yet succeeded because "[current web applications] have little ability to interact with heterogeneous data and information types". That original vision of the SW has been refocused in a Web of Data, which aims to use the Web as a large "traditional" database. Linked Data (LD) is a successful initiative within the Web of Data that consists of a number of principles for publishing, connecting and querying data, relying on SW technologies, such as Resource Description Framework (RDF), RDF Schema (RDFS) and OWL ontology languages, and the SPARQL query language. If the published data is freely available using an open license it is 4 1.1. RESEARCH CONTEXT commonly identified as Linked Open Data (LOD). Datasets that offer their information using LOD principles increased their number from 12 to 203 between 2007 and 2010, and, currently, there exist 328 LOD datasets. Furthermore, the triples that they offer also increased from 500 million to 26.9 billion in the same period. Web Services provide a more dynamic interaction compared to the current static, syntactic Web. According to the World Wide Web Consortium (W3C), "a Web Service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-process able format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards". Web Services (WSs) are the building blocks to provide a world of distributed computing on the Web, envisioning a Service Web where a large amount of stakeholders publish and consume services in order to dynamically access a concrete functionality taking benefit of Web infrastructures and Service Oriented Architectures. This scenario involves the use of mechanisms that allow to discover published services according to user requirements, and to rank those discovered services in terms of user preferences. However, Usage and integration of web services are important flaws in current WSs technologies. Thus, discovery and ranking mechanisms are supported by syntactical information descriptions, so these processes cannot take benefit of the SW. In order to realize that Service Web vision, there is a need for semantically aware descriptions and related technologies that allow for an automatic and flexible discovery of services.

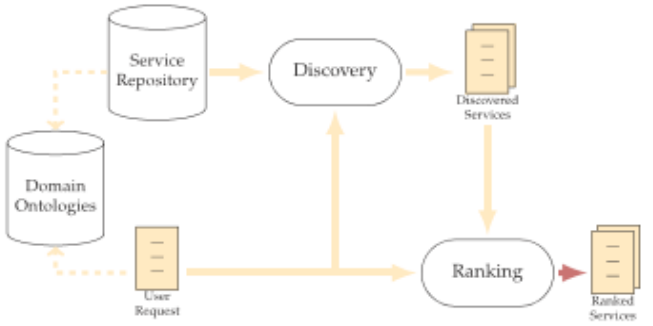

Figure 1: SWS Retrieval Activities
Source:PhD theisis Jose Maria Garcia

Advances in the field of cloud computing and networking have led to rapid development and market growth in areas such as online retail, gaming and health care. These advancements have helped in creating economic benefits by transforming investment in infrastructure. Enterprises can now rent infrastructure on-demand without the hassle of frequent maintenance or upgrades. They can also access high performance computing and elastic resources to collaborate with their peers and improve service delivery to customers [1]. An exemplar use case of cloud services adoption can be seen in a health care industry study [2], where a secure cloud environment was leveraged to manage information and foster collaboration between emergency health care professionals. In the field of advanced manufacturing however, the adoption and benefits of cloud computing has been significantly lesser than expected. This is due to limitations in cloud platforms that are not capable of fostering community engagement for advanced manufacturing design (e.g., fluid/thermal analyses), which typically requires collaborative work among multi-site engineering experts. To leverage cloud services and minimize cost, there is also a need to transform traditional workflows that feature data-intensive computation and networking during design and development of new domain applications. A new cloud-based architecture that provides Platform-as-a-Service (PaaS) management capabilities, for delivering Software-as-a-Service (SaaS) "Apps" to their customers should be adopted by advanced manufacturing community to address their problems. Such an architecture should aim at supporting an "App Marketplace" that thrives on agile development, organic collaboration and scalable sales of new manufacturing Apps requiring easy access to simulation and modeling resources. Our envisioned cloud architecture is broadly illustrated in Figure 2 that enables client requests to be provisioned from any one of the cloud providers such as Amazon, IBM, Google or GENI [3]. To deploy an App development environment, we need to be able to gather requirements, allocate resources and provide a means for the clients to keep track of resource usage. We identify three core platform services that are essential to perform these tasks, namely:

- Ontology Service,
- Resource Brokering Service, and
- Accounting Service.

These services can deploy a development environment for engineers to collaborate with their clients, and allow them to use this environment for agile development of manufacturing Apps. When multiple Apps that work together are developed (e.g., WheelSim that needs to work with TruckSim), they can be integrated within an App Marketplace.

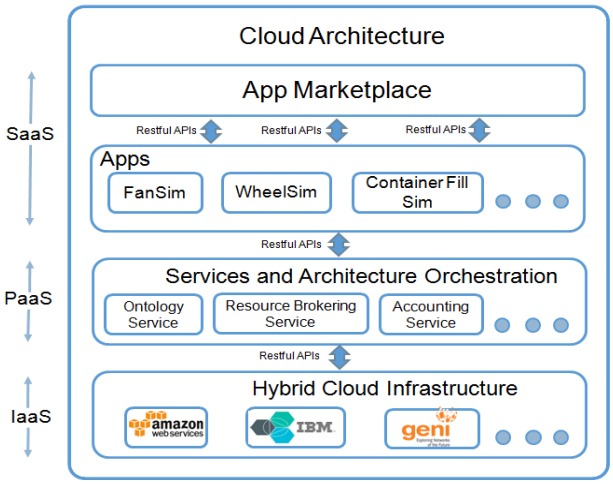

Figure 2: App Marketplace Cloud Architecture
Source: J. Yu, J. Ni, "Development Strategies for SME E-Commerce"

II. PROBLEM EVALUATION

In today's dynamic and rapidly evolving cloud services market, cloud platform providers employ proprietary technologies for common service related tasks and operations. This trend makes it challenging for an App developer to easily request for a cost-effective and suitable resource configuration when choosing between multiple providers. It is therefore critical to develop cloud services with improved data processing and resource customization with inclusion of semantics and relationships for better knowledge exchange of user experience and other performance requirements. Moreover, the conflict is the consumption of the App and development of the App may need a different service configuration by customers. For e.g., multi-expert collaboration for manufacturing design will need higher remote access requirements to cloud-hosted software/-data, versus the functional App will need to be consumed in a scalable manner by customers with different end-devices (e.g., PC, tablet) accessing from distributed locations with mobility For addressing the above problems, an ontology-based approach would be suitable to foster common understanding of the platform requirements and enable mapping of these requirements to the desired App's underlying cloud resource capabilities. Ontology-based approaches have been successfully used in many application domains to bridge the gap between heterogeneous concepts, and achieve the desired level of semantic interoperability. However, one of the critical requirements for successfully adopting an ontology approach is to have commonly accepted and openly accessible ontology trees within the application community. Unfortunately, cloud service providers such as Amazon or even GENI do not have fully developed ontologies, and their latest ontologies are constantly evolving. New ontologies are being proposed and are being developed such as WSMO (Web Service Modeling Ontology) in the W3C community that aim to address use cases in Amazon; GENI community also has published a compute resources ontology [3] that is being refined in efforts related to

Resource Specification (RSpec) for Future Internet experiments. Nevertheless, the challenge is to consistently use both App as well as cloud resource ontologies such that resource configuration can be made easier, and also users can take advantage of market competition in inter-cloud scenarios for more cost-effective resource access

A. Ontology Integration

In our ontology creation phase, we assume that a manufacturing user is not necessarily an expert in cloud resource configurations and will take assistance from a cloud engineer for the actual service implementation. There are two main Ontologies that are created to address the semantic interoperability issue. The requirements for meeting the development and hosting needs of a given manufacturing App is described by the 'App Spec Ontology'. For our understanding purposes, we conceive the 'App Spec Ontology' in Section V-A based on the use case of TotalSim, a small business advanced manufacturing company with experts in the USA and UK. Next, we consider a separate 'Platform Capabilities Ontology 'that corresponds to the publicly available ontology from a cloud provider. More specifically, Figure 3 shows the schema diagram of the resources provisioned by GENI that we use to create the 'Platform Capabilities Ontology'. Hence these two Ontologies will have to be integrated to generate a merged Ontology which can be queried to extract specifications. These specifications form the "Spec Template" which is useful to deploy the App environment.

B. Need for Ontology Update

The ontology integration methodology described earlier results in a merged Ontology which is then queried to generate a "Spec Template" that is used to deploy the App environment. This may lead to an assumption that the ontology is static. However as discussed earlier, ontologies need to handle dynamic information and need to constantly evolve to remain useful. Specifically, at any stage of this process, a change in App's requirements or in the Cloud Service Provider's resource capabilities would render the merged ontology useless because querying it would present unreliable and inaccurate results. To overcome this issue, we incorporate a 'feedback loop' in our final workflow step to update the templates. This feedback can also enable reusing or recycling successfully used prior ontologies for either integration or provisioning of new Apps. As the ontologies are updated, the Spec Templates are versioned and stored for future use of the App or for adaptation to provision new Apps.

III. PROPOSED FRAME WORK FOR ONTOLOGY SERVICE

In this section, we describe the Ontology Integration Service and discuss its various components.

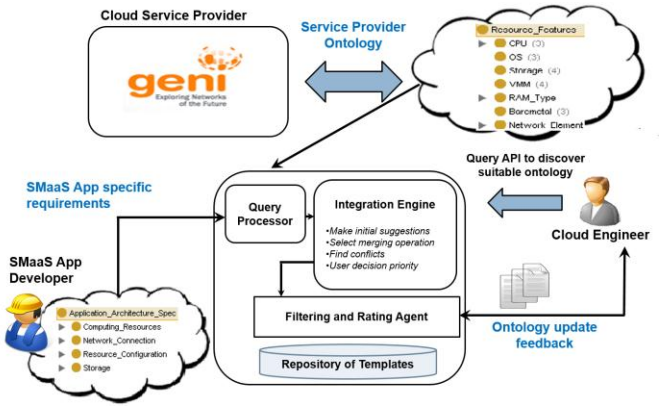

Figure. 3. Ontology Integration Workflow Components

Ontology Integration Service Components

This Ontology Integration Service essentially converts the requirements of the App developers into a usable App- Resource ontology. It accomplishes this using two primary functions:

(i) Combining the App's ontology and Infrastructure Service Provider's ontology to a App-Resource ontology which consists of a taxonomy of concepts of different cloud services to comply with the App-specific requirements, and

(ii) Finding the best PaaS service options.

Figure 5: Class hierarchy for Manufacturing Application Requirements Ontology

Figure 5 illustrates how the requirements are gathered from App developers and recorded in the App-specific requirement file which is forwarded to the service components that work together to generate the App ontology. This is then merged with the Service Provider's ontology and ultimately the unique App- Resource ontology is output. The cloud platform engineer can enter queries through a web portal containing the service name and App requirements. Cloud engineer update the ontology based on performance engineering experiments in co-ordination with the App developer. The components of the Ontology Integration Service are shown in Figure 5, which we now describe in detail.

1) Query Processor (QP):

The parsing of query to read the App-specific requirements is done by Query Processor. It analyzes the requirements and creates the suitable App ontology from existing Spec Templates or Catalog of ontologies specific to the particular manufacturing domain. The resultant App ontology output is forwarded to the Integration Engine.

2) Integration Engine (IE):

Suggestions for merging and mapping of App and Service Provider ontologies are analyzed by Integration Engine. It then aligns the App ontology with the Service Provider's ontology using one of the many ontology Mapping & Merging Tools
available such as the Prot´eg´e framework [4]. If conflicts arise as a result of merging and mapping operations, the service can fix conflicts automatically or prompt the user for suggestion. The resultant App-Resource ontology is next sent to the Filtering and Rating Agent.

3) Filtering and Rating Agent:

The duplicate and nontrivial configurations obtained from the result are removed by Filtering Agent by sorting through the results. It stores the newly established ontologies into the database for the Rating Agent to use it as part of a knowledge base. The Rating Agent analyzes the merged ontology and calculates a rating called "popularity score" based on: (a) comparison of concept labels matching in the given query, (b) success of the development environment testbed, and (c) frequency of use of templates. Depending on the popularity score, it displays the ontology templates in an order of priority. Some templates may be less re-used, which does not necessarily reflect in the quality of the ontologies but may indicate a less common App specification. The Rating Agent could also consider user satisfaction with any resultant ontology, which can be obtained through feedback obtained from a cloud engineer or App developer. With the help of the resultant integrated App-Resource ontology, the cloud engineer can build the environment with a Resource Brokering Service. Ontologies can be very time-consuming and expensive to construct. As the use of ontologies for the representation of domain knowledge increases, so will the need for an effective set of tools to aid the discovery and re-use of existing knowledge representations. This is because a major advantage of using ontologies is their ability to be re-used as well as easily adapted to work with new knowledge bases of Apps which may have unique testbed requirements (compute, network, proprietary software etc.).

IV. CONCLUSION

This Paper has discussed the web technologies currently used or proposed for cloud manufacturing. Cloud architecture for advanced collaborative manufacturing has been presented in this paper that allows agile methods to be adopted in this domain. This architecture leverages public cloud infrastructure to provide scalable and on-demand resources in a cost-effective manner. It encompasses three services at the platform level: Ontology Service, Resource Brokering Service and Accounting Service. In our paper, we used ontology concepts to bridge the semantic gap between manufacturing App developers and cloud .service providers. We developed an Ontology Service that translates the App requirements of the manufacturing domain to the development and hosting environment specifications. The service functioned by merging the ontology of the App requirements with the ontology of the cloud provider resulting in the App-Resource ontology.

REFERENCES

[1] J. Yu, J. Ni, "Development Strategies for SME E-Commerce Based on Cloud Computing", International Conference on Internet Computing for
Engineering and Science (ICICSE), 2013.
[2] V. Koufi, F. Malamateniou, G. Vassilacopoulos, A. Prentza, "An Androidenabled Mobile Framework for Ubiquitous Access to Cloud Emergency Medical Services",Symposium on Network Cloud Computing and Applications (NCCA), 2012.
[3] GENI Wiki - http://groups.geni.net
[4] H. Sun, W. Fan, W. Shen, T. Xiao, "Ontology Fusion in High-Level-Architecture-Based Collaborative Engineering Environments", IEEE Trans. on Systems, Man, and Cybernetics: Systems, Vol. 43, No. 1, pp. 2-13, 2012.
[5] M. Saeki, "Ontology-based Software Development Techniques", ERCIM News, No. 58, 2004.
[6] NDL-OWL Models in ORCA - https://geni-orca.renci.org/trac/wiki/NDLOWL
[7] M. Hung, Y. Lin, H. Huang, M. Hsieh, H. Yang, F. Cheng, "Development of an Advanced Manufacturing Cloud for Machine Tool Industry based on AVM Technology", Proc. of IEEE CASE, 2013.
[8] H. Lan, "A Web-based Rapid Prototyping Manufacturing System for Rapid Product Development", Collaborative Design and Planning for Digital Manufacturing, 2009.
[9] T. Zhang, S. Ying, S. Cao, X. Jia, "A Modeling Framework for Service-
Oriented Architecture, Quality Software", Proc. of QSIC, 2006.
[10] N. Gobinath, J. Cecil, T. Son, "A Collaborative System to realize Virtual
Enterprises using 3APL, Declarative Agent Languages and Technologies IV", Springer Lecture Notes in Artificial Intelligence, 2006.
[11] M. Hung, Y. Lin, T. Huy, H. Yang, F. Cheng, "Development of a Cloud-Computing-based Equipment Monitoring System for Machine Tool Industry", Proc. of IEEE CASE, 2012.
[12] J. Cecil, R. Gunda, P. Calyam, S. Seetharam, "A Next Generation Collaborative Framework for Advanced Manufacturing", Proc. of IEEE CASE, 2013.