Capability Profile for Enterprise Application Integration

Abdelhadi Belfadel, Jannik Laval, Chantal Bonner Cherifi, Nejib Moalla
University Lyon 2, DISP Laboratory
Lyon, France
{name.surname}@univ-lyon2.fr

***Abstract*— Open source applications are becoming more and more important and many companies use these applications in their own work to realize projects faster, better and cheaper. But in some cases, the entire use of an open source application is not necessary, with the need to use only one functionality of an open source application to link or integrate it with another application. This paper presents a positioning for a framework to use open source application decomposition in specific functionalities (features) as conceptual view for enterprise applications integration. The decomposed feature will be exposed as an Application Programming Interface (API) to support enterprise integration. Our solving approach uses a standardized methodology for manufacturing software interoperability through capability profiling provided by the ISO 16100 series, and which is also applicable and usable for developing general software applications including enterprise applications.**

Keywords: Open source, enterprise application integration, API, ISO 16100, capability profile.

I. INTRODUCTION

Open source software (OSS) products have made successful inroads into many information systems segments [20]. Many companies are investing in open source projects, and lots of them are also using such software in their own work. An OSS, is computer software with its source code made available with a license in which the copyright holder provides the rights to study, change, and distribute the software to anyone and for any purpose [16].

The open source software has received enormous attention in the last several years. It is considered as a fundamentally new way to develop software [12], [19], because of its several advantages: faster adoption of technology, increased innovation and reduced costs and time-to-market [17], [10]. These benefits had a major impact on software industry by reducing the amount of development, and the way of building systems changed from developing all software units, to building system by combining software units which are provided by various open-source communities or vendors. The literature discusses several possible benefits of OSS adoption [13], for example cost cuts (by reducing the license fees, hardware requirement, scaling costs), independence from vendors of proprietary products, software reuse, community support (free maintenance and upgrades of the software). But there are also risks related to adopting OSS but not all organizations consider them [13], we could have hidden costs due to time-consuming to evaluate the selected solutions, lack of products with specific functionality, and the customization needs of the selected solution, to fit the context in which it will be used.

Most of these OSS solutions are heterogeneous and are deployed on different platforms, and there is a need for interoperability between the multiple disparate solutions and ad-hoc applications which are developed for a specific issue. For this reason of interoperability and reusability of an open source application within a European project, we wanted to use a single feature (functionality) of an open source application and to integrate it with other applications, we realized the difficulty to find a methodology letting us to achieve this objective.

This paper proposes a Framework to resolve this issue, letting us to decompose functionalities (features) of an open source application as conceptual view, and expose the wanted feature as an Application Programing Interface (API) to support enterprise integration. Our proposal identifies the ISO 16100 series as a Framework to describe capabilities of manufacturing software in capability profile.

The first section of this paper presents some definitions about interoperability and integration and the difference between them, we present some standards that addressed these issues. In the second section, we expose some positions regarding the standards and why we have chosen to work with ISO 16100. Section three describes our Framework letting us to decompose features of an open source application in order to support enterprise application integration. Section four presents what we intend in the future works.

II. STANDARDS FOR INTEROPERABILITY

Before exposing the standards of the interoperability of heterogeneous systems, we will start by giving the definition of interoperability and integration and the differences between them.

A. *Interoperability and integration*

Regarding the definitions of interoperability, it have been well explained in [11]. In general, interoperability is the ability of two systems to understand each other and to use the functionalities of one another. The word interoperate implies that one system performs an operation for another system. From the computer technology point of view, it is the faculty for two heterogeneous computer systems to function jointly and to give access to their resources in a reciprocal way. Interoperability is considered as significant if the interactions can take place at least on three different levels: data, services and processes, with a semantics defined in a given business context.

According to ISO 19439 [3], enterprise integration is the process of ensuring the interaction between enterprise entities necessary to achieve domain objectives. Enterprise integration can be addressed in various manners and at various levels [11], for example: (i) physical integration (interconnection of devices), (ii) application integration (integration of software applications and database systems) and (iii) business integration (coordination of functions that manage, control and monitor business processes).

According to ISO 14258 [1]. Two systems are considered as Integrated if there is a detailed standard format for all constituent components. Interoperability is more related to the Unified approach where there is a common meta-level structure across constituent models, providing a means for establishing semantic equivalence or the federated approach where models must dynamically accommodate rather than having a predetermined meta-model.

To summarize, interoperability has the meaning of coexistence, autonomy and federated environment, whereas integration refers more to the concepts of coordination, coherence and uniformization [11]. From the point of view of degree of coupling, fully integrated systems are tightly coupled, it means that the components are interdependent and cannot be separated, and interoperable systems are loosely coupled, it means that the components are connected by a communication network and can interact, they can exchange services while continuing locally their own logic of operation. Thus, two integrated systems are inevitably interoperable; but two interoperable systems are not necessarily integrated [11].

B. ISO 14258 Industrial automation systems - Concepts and rules for enterprise models

It was the first standard that addressed the problem of interoperability of heterogeneous systems [15]. The intention of the standard is to describe the enterprise in models and use those models to identify and resolve interoperability issues. It identifies requirements for standards on model interoperability and defines three ways in which models can be related to each other [18] :

- The integrated approach where there is a standard format for all constituents of the system. The two key points in this approach are the existence of a standard model form, which is ideally produced from the standardization of a large number of models and that each party translates its original representation to this standard form. The problem with the integrated approach is it requires an enormous effort to standardize large numbers of models and results in the most restrictive form of interoperability.

- The unified approach assumes that there exists a template that provides a common meta-level structure across constituent models, providing a means for establishing semantic equivalence. This template is then the basis for a metamodel. The metamodel is not in an executable form as it is in the integrated approach. Using the metamodel, any model can be translated into any other, however loss of some semantics is possible.

- The federated model scenario exists if one assumes that no agent successfully or globally can impose requirements for semantic equivalence across all models of an enterprise or across enterprises.

The federated approach seems most promising to networked enterprises where model and applications need to dynamically adapt and accommodate, in particularly in the virtual enterprises environment. Indeed, using federated approach to establish interoperability allows the enterprises in question to keep their own identity, methods of work, tools and applications, and possibly with reduced time, limited changes and costs to establish interoperability.

C. ISO 15745 -Industrial automation systems and integration - Open systems application integration frameworks

ISO 15745, defines an Application Integration Framework (AIF) which is a set of elements and rules for describing integration models and application interoperability profiles (figure 1) [2]. The first part of ISO 15745 defines the generic elements and rules for describing integration models and application interoperability profiles (AIPs), as well as their component profiles, process profiles, information exchange profiles, and resource profiles. These profiles may describe profiles based on particular technologies and therefore makes this standard applicable to industrial automation environments [14]. Environments such as discrete manufacturing, process automation, electronics assembly, semiconductor fabrication, wide-area material handling, and other automation and control sectors such as utility automation, agriculture, off-road vehicles, medical and laboratory automation, and public transport systems. Parts 2, 3 and 4 of ISO 15745 define the technology specific elements and rules for describing both communication network profiles and the communication related aspects of device profiles based upon particular fieldbus technologies.

The middle section of figure 1 shows the AIP, consisting of one process profile, one or more resource profiles, and one or more information exchange profiles. Underlying the AIP are the relevant integration models which represent the

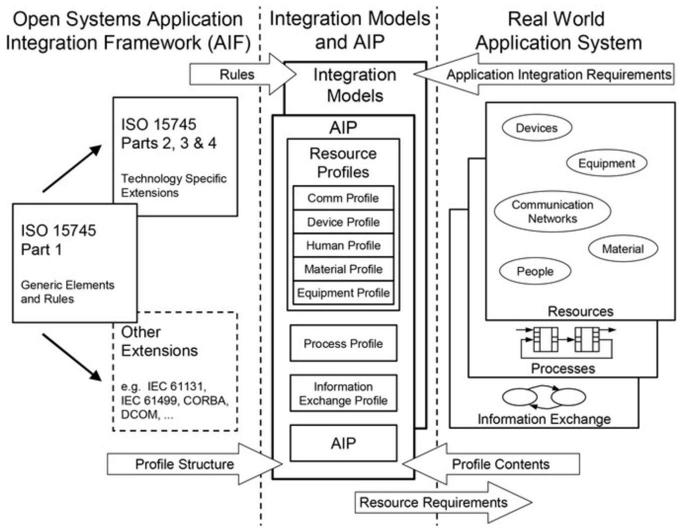

Fig. 1. Context of ISO 15745 [2]

application requirements [15]. With its focus on shop floor application, figure 1 shows the next level of details for the resource profile consisting of communication, device, human, material and equipment profiles. The standard represents in detail the most used shop floor communication technologies in the form of device profiles.

D. ISO 16100 Industrial automation systems and integration - Manufacturing software capability profiling for interoperability

ISO 16100 series which is titled Manufacturing Software Capability Profiling for Interoperability, consists of six parts: Part 1: Framework [7]; Part 2: Profiling Methodology [4]; Part 3: Interface services, protocols and capability templates [5]; Part 4: Conformance test methods, criteria, and reports [6]; Part 5 : Methodology for profile matching using multiple capability class [8]; Part 6: Interface services, protocols for matching profiles based on multiple capability class structure [9].

The ISO 16100 standard series targets the representation of a software capability profile [7]. The standard provides a method to represent the capability of manufacturing software relative to its role throughout the lifecycle of a manufacturing application, independent of a particular system architecture or implementation platform. Software interface requirements are characterized as Manufacturing Software Units (MSU) with capability elements and rules. A manufacturing software unit is a class of software resource, consisting of one or more manufacturing software components, performing a definite function or role within a manufacturing activity while supporting a common information exchange mechanism with other units [7].

The schema to describe software capability unit is the capability profile template of a manufacturing software unit (see figure 2). This schema denotes the manufacturing function, resource, and information handled by the

- Common Part
  - Template ID
  - Capability Class Name and Reference CCS
  - Software Unit ID
    - Vendor Name
    - Version Number & History
    - Computing Facilities Required
      - Processor
      - OperatingSystem&Options
      - Language
      - RuntimeMemory
      - DiskSpace
      - MultiUserSupport
      - RemoteAccess
      - AddOns&PlugIns
    - Measured Performance of the Unit
      - ElapsedTime
      - NumberOfTransactionsPerUnitTime
    - Reliability Data of the Unit
      - UsageHistory
      - NumberOfShipments
      - IntendedSafetyIntegrityLevel
      - CertificationBody
    - Support Policy
    - Price Data
  - Capability Class Reference Dictionary Name
  - Number Of Profile Attributes
  - Number Of Methods
  - Number Of Resources
  - Number Of Constraints
  - Number Of Extensions
  - Number Of LowerLevels
  - Number Of Subtemplates At Next Lower Level
- Specific Part for Capability Class
  - Reference MDM Name
  - MDD Description Format
  - MDD Description
    - Set Of MDD Objects
    - List Of MDD Objects
    - Time Ordered MDD Objects
    - Event Ordered MDD Objects
  - List Of Capability Class Attributes
  - List Of Capability Class Methods
  - List Of Capability Class Resources
  - List Of Capability Class Constraints
  - List Of Capability Class Extensions
  - List Of Capability Class Lower Levels
  - List Of Capability Class Subtemplates

Fig. 2. Conceptual structure of a capability profile template (ISO 16100-5) [8]

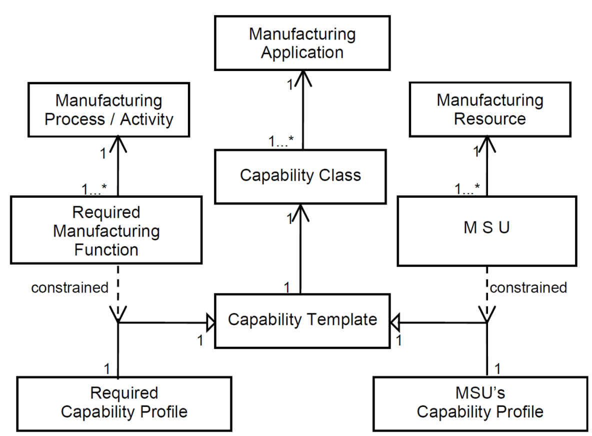

Fig. 3. Class diagram of capability (ISO 16100 part 3) [5]

manufacturing software unit.

Figure 3 shows the relation between capability class, capability template and capability profile. The structure of manufacturing software unit capability profile template is derived from the manufacturing capability class structure, and capability profile for manufacturing software unit is an instance of capability template filled with appropriate information.

III. POSITION REGARDING THE RELATED WORKS

ISO 15745 and ISO 16100 are used for integration and interoperability issues, and both standards use profiles to describe the information needed to identify the capabilities of the entities that should interact. The process of profile creation on ISO 15745 is called application interface profiles construction and it is composed from 5 steps, whereas on ISO 16100, the steps of profile creation is seen as capability profiling process and it is composed by 3 steps.

According to K. Kosanke [14], the need for harmonization is necessary because the two standards are different and any potential user, who wants to use these standards, will be confused by their difference. The author concluded by a limit in the use of these standards with respect to the human aspects of interoperability, where information about the internal structure and the dynamics of the application can be more important than information about the potential exchange itself.

To solve our problem of open source application decomposition in specific features, we must take into account the fact that there are some open source applications that are not sufficiently documented, sometimes with the obligation to access the source code to understand how some feature works, so we were also looking for a Framework to describe the capabilities of an application, which has to be human readable, letting us to know what kind of features an application offers, and how these features work. For all these reasons, we decided to work with ISO 16100 because it focuses on the interfacing requirements for interoperability which interests us in this case, instead of ISO 15745 which identifies a larger set of elements needed to support interoperability between application components. Also, we have chosen ISO 16100 for its simplest representation of a software capability profile which is human readable and computer interpretable.

At this stage, an open source application can be decomposed and described using capability profiles provided by the ISO 16100 series, but this is not enough because the difficulty now lies in switching from the logical level of capability profiles to an application programming interfaces letting us to use the desired features, and then integrate them in order to be used in our information system. In what follows, we will present an overview of our Framework allowing to solve these problems and reach our final goal.

IV. CAPABILITY PROFILE FOR ENTERPRISE APPLICATION INTEGRATION

Each application is represented as an activity tree structure that is hierarchical (see figure 4), and an activity (which is a feature in our case) is considered as a software unit.

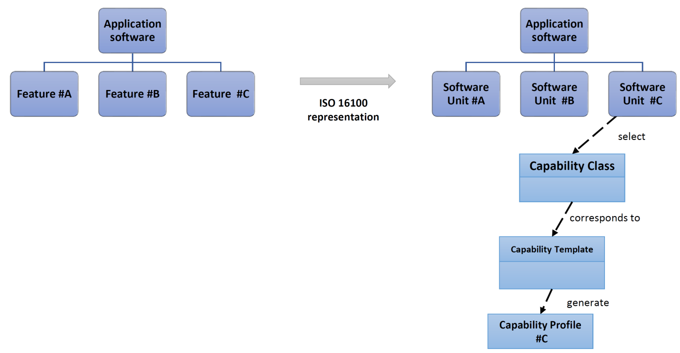

Fig. 4. Activity tree structure

As shown in figure 5, an application may have several features (here an example of two applications, with features noted from #A to #E), for the purpose of using a specific features (e.g. #A and #D) for enterprise application integration, the procedure to be followed is as follows :

For each feature, we will identify a capability class structure by using the methodology of ISO 16100 (part 2), and get the corresponding template to this capability class. The selected template will be filled and a validation process will be applied using the conformance test methods provided by ISO 16100 (part 4), to ensure that the profile is well formed and valid according to the specifications of ISO 16100 (part 3).

A capability profile contains a lot of information, an extraction pattern will be applied to lighten the information, extracting only the capability definition needed to develop an application programming interface allowing us to access and use this feature. An example of data that interests us on the capability profile are input / output data from the specific part of the profile, input data types, output data types.

Once the extraction done, we develop our APIs based on these data which are considered as functional and technical specifications. The generated APIs will be managed on an API Gateway to expose their endpoints, and manage the security issues.

These APIs could be called or used to resolve the interoperability issue by accessing their services directly through endpoints exposed by the API Gateway, or they could be integrated together using enterprise application integration patterns and exchange data with other applications using the enterprise service bus.

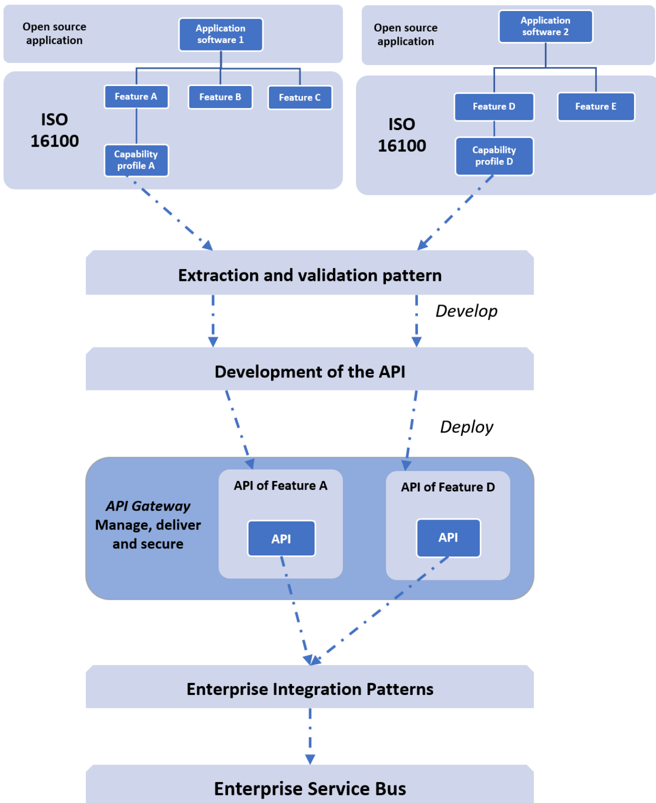

Fig. 5. Proposed Framework

V. CONCLUSION AND FUTURE WORK

In this paper, a positioning for an open source application decomposition in specific features based on the standard ISO 16100 series has been presented, aiming to expose features as an application programming interfaces, to be consumed to support enterprise integration. We intend in the future works to set the extraction pattern by selecting the suitable data from a capability profile letting us to develop and generate an API, and implement a proof of concept in a European project by applying our Framework on an application from Fiware project [1].

VI. ACKNOWLEDGMENT

This paper presents work developed in the scope of the project vf-OS. This project has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement no. 723710. The content of this paper does not reflect the official opinion of the European Union. Responsibility for the information and views expressed in this paper lies entirely with the authors.

[1] https://www.fiware.org/

REFERENCES

[1] Iso 14258, industrial automation systemsconcepts and rules for enterprise models, iso tc184/sc5/wg1, april 14, 1999.

[2] Iso 15745, industrial automation systems and integration open systems application integration frameworks, iso/tc/184/sc5, 2000.

[3] En/iso 19439, enterprise integration framework for enterprise modelling, 2003.

[4] Iso 16100-2:2003 industrial automation systems and integration manufacturing software capability profiling for interoperability part 2: Profiling methodology, 2003.

[5] Iso 16100-3:2005 industrial automation systems and integration manufacturing software capability profiling for interoperability part 3: Interface services, protocols and capability templates, 2005.

[6] Iso 16100-4:2006 industrial automation systems and integration manufacturing software capability profiling for interoperability part 4: Conformance test methods, criteria and reports, 2006.

[7] Iso 16100-1:2009 industrial automation systems and integration manufacturing software capability profiling for interoperability part 1: Framework, 2009.

[8] Iso 16100-5:2009 industrial automation systems and integration manufacturing software capability profiling for interoperability part 5: Methodology for profile matching using multiple capability class, 2009.

[9] Iso/cd 16100-6:2010 industrial automation systems and integration manufacturing software capability profiling for interoperability part 6: Interface services, protocols for matching profiles based on multiple capability class structure, 2010.

[10] Andrea Bonaccorsi and Cristina Rossi. Comparing motivations of individual programmers and firms to take part in the open source movement: From community to business. *Knowledge, Technology & Policy*, 18(4):40–64, 2006.

[11] David Chen, Guy Doumeingts, and François Vernadat. Architectures for enterprise integration and interoperability: Past, present and future. *Computers in industry*, 59(7):647–659, 2008.

[12] Chris DiBona and Sam Ockman. *Open sources: Voices from the open source revolution.* " O'Reilly Media, Inc.", 1999.

[13] Øyvind Hauge, Daniela Soares Cruzes, Reidar Conradi, Ketil Sandanger Velle, and Tron André Skarpenes. Risks and risk mitigation in open source software adoption: bridging the gap between literature and practice. In *IFIP International Conference on Open Source Systems*, pages 105–118. Springer, 2010.

[14] Kurt Kosanke. Iso standards for interoperability: a comparison. In *Interoperability of enterprise software and applications*, pages 55–64. Springer, 2006.

[15] Kurt Kosanke and R Martin. Enterprise and business processes-how to interoperate? the standards view. In *Workshop on Standards for Interoperability*, 2008.

[16] Andrew M St Laurent. *Understanding open source and free software licensing: guide to navigating licensing issues in existing & new software.* " O'Reilly Media, Inc.", 2004.

[17] Lorraine Morgan and Patrick Finnegan. Benefits and drawbacks of open source software: an exploratory study of secondary software firms. In *IFIP International Conference on Open Source Systems*, pages 307–312. Springer, 2007.

[18] Balázs Pataki, László Kovács, Claudia Guglielmina, and Alessandra Arezza. Abilities to support a federated architecture based interoperability bus with groupware and multimedia. In *Enterprise Interoperability II*, pages 415–426. Springer, 2007.

[19] ES Raymond. The cathedral and the bazaar [page web] http://www.tuxedo. org/esr/writings/cathedral-bazaar. *June*, 20:2005, 1999.

[20] Georg Von Krogh, Stefan Haefliger, Sebastian Spaeth, and Martin W Wallin. Carrots and rainbows: Motivation and social practice in open source software development. *MIS quarterly*, 36(2):649–676, 2012.