Knowledge Management in a Large Organization: a Practical Case Study

Antonio Ballarin¹, Spartaco Coletta¹, Daniela Principi⁴, Giulio Concas², Marco Di Francesco³ and Katiuscia Mannaro²

¹Sogei Spa, v. M. Carucci n. 99, 00143 Roma, Italy
²Department of Electrics and Electronics Engineering, University of Cagliari, P.zza d'Armi Cagliari, Italy
³FlossLab Srl, v.le Elmas 142, Cagliari, Italy
⁴General Administration, Personnel and Services Department, Ministry of Economy and Finance, v. XX Settembre, 97, 00187 Roma, Italy

Abstract. In this work we present an approach, based on a Knowledge Federation, for the management of the information regarding the life-cycle of Software Application ad IT services for organizations' operations. We need many information about software and IT to study the software product's life-cycle management, and to understand the life-cycle of information, which is based on four phases: introduction, growth, maturity, and decline. We need a knowledge base about the products and their costs to understand how to manage the single software products. We will focus our attention on mapping software applications, Cloud Infrastructure, application maintenance, cost tracking and management in a whole system. Our approach is based on dynamic, real-time extraction of data from existing repositories, and on its dynamic retrieval, depending on the needs to be addressed at strategic level. This level of knowledge can be used to support decision making.

Keywords: Software life-cycle Management, PLM, Knowledge Management, Software Engineering, Cloud Computing, IaaS.

1 Introduction

Large organizations have to manage hundreds, often thousands, of different applications that are the key for the organization's operations. Such a management of them is a core issue for these organizations, and software applications assume the role of assets where the organization’s knowledge is synthesized [21 - 23]. Product life-cycle Management (PLM) is generally defined as a strategic business approach for the effective management and use of corporate intellectual capital [31]. Software products differ from other products for many characteristic related to life-cycle [30]; a software have to be evolved with the organization and the new features, but it can be changed or updated relatively easy by using patches or released updates. The release frequency of a Software Product is very high; the evolution of the processes modifies the software. The maintenance phase is very large in the software's life-cycle. The custom applications have to be developed and maintained by the organization.

Sogei manages highly sophisticated services, projects, technology and project management consultancies on behalf of the Italian Ministry of Economy &Finance (MEF) and other central, local, health services Administrations. Its activities cover two main areas: Management and development of IT services for the MEF, through technical and project consultancies Implementation, on behalf of the MEF, of the Program for the Rationalization of Public Expenditure in Goods and Services through the use of information technologies and innovative purchasing tools. The Cloud Platform in Sogei is based on the paradigm [32] of Infrastructure as a Service (IaaS).

Infrastructure as a Service is a model that provides a full computer infrastructure, hardware, Middleware, OS and virtualization software. Moreover, these cloud deployment models are Private in Sogei. They can't use a Public or Hybrid Cloud because Sogei has security and privacy constraints for the managed data. In this private model, the company realizes a cloud computing environment that allows to store the data within their operational structure, with obvious advantages in terms of security and privacy. In this case, the cloud services are accessible only by authorized end-users. This model allows to enjoy the benefits of public model, especially in terms of costs and scalability, and at the same time to have the guaranteed standards of management and security, typical of the private models.

The management of these applications can be used at the tactical level where we consider the management of specific aspects of the application portfolio, such as functionalities dictionary, application maintenance, mapping between software and Cloud Infrastructure where the software resides, cost tracking and management, and the like. This portfolio contains a complex software [1-5] with subsystems or modules so that loads and features are strategically
distributed, typically according to power-laws, among different parts of the software [6-9], [33]. We need to manage also other information to have comprehensive views of the general situation, the overall costs and functionalities, and their trends. This strategic information must be obtained also at the department level. Moreover, the definition of the strategic information is not static, but new needs must be addressed continuously, so the system able to get it must be highly dynamic and configurable. Typical examples are software systems organized as software networks, where software units are connected in a software graph, and software metrics are power-law distributed [12-13], [16-17], [19-20].

In this paper, we present an industrial application for a Software Application Management, based on Knowledge Federation, that we implemented for the Information Systems of the Italian Ministry of Economy and Finance. This application extracts the data from existing repositories and the data is manipulated based on the knowledge needs of the organization [21-26].

These studies resulted in a prototype system, called System Map Pilot, developed to demonstrate the feasibility of the approach, and which is currently in use in the organization.

2 The data sources

The study starts with the analysis of the available information in the organization.

There are many software applications used in the organization (almost one thousand in our case). All these applications are customized, i.e. the specific role of the organization here considered (the Ministry), does not easily allow to use and integrate standard software as they could be components off-the-shelf. There are also many information about the life-cycle of the applications.

The information about the applications are managed by various systems, under different perspectives:

1. Application maintenance and contractor activities monitoring (BIG): there is a ticketing system tracking and managing the applicative bug fixing and evolutionary maintenance requests.

2. Configuration management (CMS, CMA): the hardware farm is based on multi-core processors, with extensive use of virtualized environments; this systems tracks how applications are replicated and mapped to the hardware, to virtual systems, to databases and to the network.

3. Functionalities management (INFAP): the Organization keeps a list of all applications and of all their functionalities and size (in Function Points), together with their history. This repository has many uses, one of these concerns the size definition of an applications , i.e. its value.

4. Cost and project management (SIGI): the costs of the various applications and maintenance activities is tracked through an ERP system.

5. Contracts (DePF): the contracts made with the various suppliers are kept in a document management system (DMS), accessed through a dedicated portal.

6. Test factory (LINCE): the applications must be tested on their functionalities before releasing and this repository contains test plans defined during previous phases.

Fig. 1 shows a schematic view of the existing relations between the various systems holding the above described data. These systems hold real data, subject to continuous updating and also to frequent schema modifications. From time to time, even new databases are introduced, to address new needs or to substitute existing ones. For instance, an inventory of functions and services (INVAPP system) will be added to the sources list, but it is not available yet. Two large classes of problem have to be addressed for such software management in general: refactoring [13-15] and software maintenance for fault detection, since bugs may largely affect more complex software [10-11]. Due to lack of space, we will not deal with them in this work. We will only consider these systems as the source of the information we used as a starting point for our work.

3 The information management

From a conceptual point of view, the information we need could be summarized with the architecture represented in Figure 2.
To define this knowledge we analyzed the original databases, chose the relevant data, and built macro-schema to reconcile the data. These data are tagged with meta-data carrying the relevant information about their meaning, so that they can be aggregated and analyzed dynamically. The data extracted from the various systems are linked through common identifiers, used throughout the systems.

The consolidate information make a very useful knowledge base of information about the applications and their evolutions, costs and infrastructure.

Fig. 3 shows the output produced by a query to System Map Pilot system and its information flow. Heterogeneous data sources, on the right side, cooperate to obtain the relevant business information of the left side.

Fig. 4 shows a conceptual map of System Map Pilot and of its information flows. Each main block shares a list of metadata, allowing to link one block to any other block plugged into the system.

4 The Prototype of System Map Pilot

We implemented a prototype version of System Map Pilot, that is presently working and is used within the organization. It is a prototype because its user interface is still quite rough, and because the intervention of a programmer is still needed to build new query capabilities in it. The system is used mainly to aggregate data at department level, to manage departments' budget and to optimize maintenance at department level.

Fig. 5 shows a specific query: the time trend of Function Points developed for a specific Department, with and without considering the reuse level. Note the restructuring performed in 2010, that reduced the deployed FPs, with no decrease in the total number of available Fps.
5 Conclusions

In this paper we described an Application Portfolio management system developed as a knowledge base to manage the information about hundreds, often thousands, of different applications that are the key for the organization's operations. We explored two main levels. We considered the management of specific aspects of the application portfolio, such as functionalities dictionary, application maintenance, mapping between software and hardware where the software resides, cost tracking and management. We considered also the comprehensive views of the general situation, the overall costs and functionalities, and their trends. This strategic information must be obtained also at the department level.

The System Map Pilot developed is based on the extraction of data from the existing repositories depending on the needs to be addressed at strategic level and the organization.

We showed how it is possible to tag data with meta-data carrying the relevant information about their meaning, so that they can be aggregated and analyzed dynamically, analyzing the original databases, choosing the relevant data, and building the view of the interesting knowledge.

We showed how the data extracted from the various systems are linked through common identifiers, used throughout the systems. While this emphasizes a real-time access to data, it is also endowed with a cache holding the most recently obtained data, that is periodically updated. In particular, we found how this approach reduced the deployed Function Points, with no decrease in the total number of available Function Points.

6 References


conference Object-oriented programming languages, systems, and applications (OOPSLA).


