

The Dimensions of Embedded COTS and OSS Software Component Integration

Tuija Helokunnas

Tampere University of Technology, P.O.Box 541, 33101 Tampere, Finland
Nokia Research Center, P.O.Box 407, FIN-00045 Nokia Group, Finland
Tuija.Helokunnas@tut.fi

Abstract. This paper describes the dimensions of the integration of embedded Commercial-Off-The-Shelf (COTS) and Open Source Software (OSS) components in the telecommunication systems. The paper emphasizes a telecommunication system vendor view to COTS and OSS component integration. The paper is based on semi-structured interviews held both at component supplying and integrating companies in Finland. The following embedded COTS and OSS acquisition, integration and maintenance dimensions were identified: Vision and strategy, business and markets, software engineering processes, software engineering environments and collaboration approaches. The paper describes the main characteristics of each dimension. The paper focuses on the collaboration approaches and especially on the information and knowledge exchange between a system vendor and all of the component suppliers.

1 Introduction

The aim of this paper is to describe the dimensions of the integration of embedded Commercial-Off-The-Shelf (COTS) and Open Source Software (OSS) components in the telecommunication systems. Software component as a concept has attracted software community during the past thirty years. Principles of Component Based Software Engineering (CBSE) have been considered to provide the software product developing companies with means to manage complexity of the software systems. In addition to develop components in-house, the telecommunication system vendors have recently studied and deployed the acquisition, integration and delivery of the externally developed components, e.g., [1]. Currently an essential part of a telecommunication system planning is to perform make/buy/reuse decisions, i.e. select COTS, OSS and in-house developed components to be integrated into a system.

Meyers & Oberndorf [2] defined the COTS component as a product that is sold, leased or licensed to the general public in multiple, identical copies. It is offered by a vendor, which supports and evolves it and tries to profit from it. The COTS product is being used without internal modification from the customer. IEEE [3] defined Modified-Of-The-Self (MOTS) software to be similar to COTS software; however, the MOTS software is tailored to buyer-specific requirements by the component

This paper has been published under copyright of Springer-Verlag in the Proceedings of the PROFES 2002 conference, December 9-11, Rovaniemi, Finland
<http://www.springer.de/comp/lncs/index.html>

Copyright © Springer-Verlag

vendor. OSS software is different from COTS software in several ways including unlimited access to source code, free distribution of the software and its modifications without restrictions and not allowing discrimination based on users or usage [4]. This paper uses the term external component to cover COTS, MOTS and OSS software components.

Most of the embedded software development companies lack competent human resources. In addition, it is not feasible to develop all the needed software in-house. The main reason for companies to acquire external components is to speed up product development. Attempts to utilize external software components include the use of very small pieces of code such as a sorting algorithm implementation downloaded from the Internet to the integration of large subsystems like embedded database management systems and distribution management systems. However, the product processes and system architectures typically do not yet fully support the integration of external embedded component activities.

This paper emphasizes a telecommunication system vendor view to COTS and OSS component integration. The system vendor integrates in-house developed and external components into products and delivers the products mainly to telecommunication operators. The paper is based on information gathered during the last year. Information gathering included interviewing of 22 people working for a global telecommunications system vendor in Finland. The interviewed people had at least two years experience in working with COTS and OSS components. The interviewed people had technical, commercial and legal background. The rough questionnaire followed during the interviews was:

- How widely do you use COTS, MOTS and OSS components?
- What is a feasible size of a COTS, MOTS and OSS component?
- What are the changes caused by the use of COTS, MOTS and OSS components?

Typically the discussion originated by the third question was quite long. Several more detailed questions were presented during the discussion. The more detailed questions included:

- What are the changes needed to software development processes?
- What are the changes needed to software engineering environments?
- What are the changes needed to software architecture design approaches?

Additional interviews were performed in three embedded software component vendor companies located in Finland. The interviewed people included the managing directors and R&D directors of the companies. The questions presented include:

- What is your business logic model?
- What are the legal requirements?
- How is the collaboration and knowledge exchange with customers performed?

2 COTS and OSS Software Component Dimensions

The interviewed people at the system vendor side estimated that the amount of COTS, MOTS and OSS software in lines of code varies from nearly zero to about 75 % of a mobile network element subsystem. Typically, COTS, MOTS and OSS software had

quite a minimal portion compared to 70 or even 80 % share that external software was identified to have in the most network management systems. Most of the COTS, MOTS and OSS software components were found from the reusable communication software platforms. Typical components were operating systems like Linux, database management systems and distribution management systems.

One of the interviewed people who had six years experience in the external component acquisition and management argued that the feasible size of an external component equals to 2 to 4 man years development effort. Otherwise the additional work caused by the acquisition of the component is typically not covered by the decreased development efforts. This is in line with the literature, e.g., Szyperki [5] felt that too small components with a variety to choose from, e.g. hundreds of different implementations of stacks and queues, are not likely to increase software development productivity.

2.1 Changes Caused by the External Component Integration

The interviewed people identified the following areas to be affected by the acquisition and maintenance of external components (Figure 1): Vision and strategy, business and markets, software engineering processes, software engineering environments and collaboration approaches. The identified areas are close to those mentioned by Lim [6] when describing areas related to software reuse.

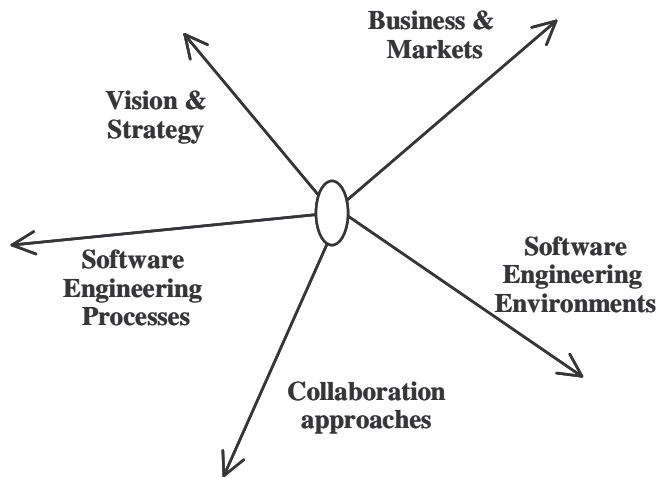


Fig. 1. Areas affected by the acquisition and maintenance of external components

Vision and strategy give the fundamental reasons for the external component integration activities. Several interviewed people said that the acquisition and integration approaches of the external components should be supported by the

strategy of the company. When a company replaces a part of in-house software development with the integration of external components, the nature of the company changes from a producer to a consumer organization [2]. The top management of the company shall be committed to this change. Strategy planning and communication provides the top management with means to lead the change. For example, instead of stating that the company shall be the best mobile software house in the next two years, the top management defines that the company shall become the best mobile software integrator.

Business and markets include value net analysis, revenue logic analysis, market analysis, planning of control points and caring of stakeholders. A system vendor as well as a component supplier needs to describe and analyze *the value net* of products and services. According to a rather general view, the concept of value can be regarded as the trade-off between benefits and sacrifices [7]. Some authors define value purely in monetary terms, whereas others use a broader definition, which also includes such non-monetary revenues as competence, market position, and social rewards [8]. In this paper the broader definition of value is applied covering also the non-monetary aspect. The concept of “networks of organizations”, i.e. networks, is understood as any group of organizations or actors that is interconnected with direct/indirect exchange relationships [9]. These business networks cannot be designed and managed by any single actor. The concept of “network organizations”, i.e. *nets*, refers in here to a more narrow concept than the term network. In other words, whereas networks are viewed from macro-level, nets are viewed from the micro-level, usually from a viewpoint of a single actor. This kind of business net is characterized often by the importance of a focal firm or a hub-company that is able to drive the formation and management of the net, e.g., [10]. However, in speaking of networks and nets, the important issue is to understand that both of these concepts include not only actors that compose the entity, but also different kinds of interrelated activities and resources, e.g., [9].

The value net analysis performed by the system vendor shall focus on the changes that the transition from a producer to a consumer organization causes to supplier and customer core value, added value and future-oriented value [11] creation activities and resources. Typical areas to be considered are the management of the value net, measuring of a software component value and pricing of the software components. The main subject is to understand the role, activities, resources and relationships of the company currently and in the near future. However, value net analysis was quite unfamiliar to the interviewed people. It was not mentioned spontaneously during the interviews. On the other hand, several interviewed people mentioned that pricing of the external software components is a key topic.

One of the interviewed people mentioned *control points*. A control point is a technology or a competence that gives the system vendor a competitive advantage such as the competence to integrate Digital Signaling Processing (DSP) software into base station. The planning of the control points is based on the value net analysis. It provides the company means to manage the customer relationship.

A system vendor should evaluate and prioritize the *revenue logic* approaches of software component suppliers. The revenue logic evaluation is needed for the cost/revenue analysis of the end product. Especially, the maintenance costs of external

component based products might be remarkably high. One way to minimize the maintenance costs is to acquire only those components that have a second source. Revenue logic analysis and especially having the second source was identified to be important by half of the interviewed people.

2.2 Processes and Architecture

COTS and OSS integration requires that the general in-house component development based **software engineering processes** are modified. The most important modifications are related to make/buy/reuse decision-making, agreement negotiations with component vendors, technical, commercial and legal evaluations and software architecture including the external component integration approaches.

The management of the end product shall make the *make/buy/reuse decisions*. The decision-making is based on the negotiations and evaluations results. The decision-making is supported by the decision criteria. Especially, the criteria include formula for the financial comparison of the integration of external components versus the development of in-house components. Most of the interviewed people considered the decision-making process to be quite demanding.

One of the interviewed people mentioned that *the technical, commercial and legal evaluations* shall be carried in parallel to reduce the duration of the evaluations. This idea was supported by all of the interviewed people. Before the evaluations shall be performed the requirements and the evaluation criteria of the components to be acquired shall be described and analyzed. The end product management shall accept the evaluation criteria including various aspects such as the maturity of the vendor, the price of the component, the applied pricing model, the partnership agreement conditions, the maintenance of the component and the required integration work with the component.

It is the **software architecture** of a system that defines the structure of a component based system. The architecture shall support the integration and especially the maintenance of the external components. One of the most comprehensive definitions of the term software architecture was given by Meyers & Oberndorf [2]: "Architecture is a representation of a system or a subsystem characterized by functionality, components, connectivity of components and mapping of functionality onto components". The software architecture shall define the COTS and OSS component interfaces. Especially, if the COTS or OSS component interface is not an industry or official standard, glue software in multiple layers shall be developed on the top of the component. The maintenance of the end product requires that the changes in the interface are minimized and strictly controlled. Often the integration of COTS components causes delays during the testing phases because the testing-correcting cycle duration is longer with an external component than with an in-house developed component. One of the interviewed people argued that "OSS components are superior to binary code COTS components during the testing and maintenance phases because if the integrator has access to the source code, the integrator is able to correct high priority errors when needed".

Software engineering environment shall support the distribution of acquired COTS components and related documentation inside the organization. Especially, the environments shall support external and internal OSS component development and maintenance. Internal OSS, i.e., corporate source management system provides the company with means for reusing source code. Controlling of the interface of internal OSS components is the main issue.

3 Collaboration Approaches

Schrage [12] defined that collaboration is the process of shared creation: two or more individuals with complementary skills interacting to create a shared understanding that none had previously possessed or could have come to on their own. Sonnenwald and Pierce [13] further emphasize that collaboration includes the completion of tasks with respect to a mutually-shared superordinate goal and takes place in a particular social or work setting. It is a primary mechanism to generate intellectual capital. The interviewed people were especially interested in finding out the practical requirements for successful collaboration. According to Marmolin et al. [14] collaboration consists of information sharing, knowledge integration, communication and co-working. These collaboration activities are conducted when a system vendor joins the OSS community or deals with a COTS vendor. A temporal joining of the OSS community for a software download, further development and upload or the acquisition of a widely used COTS component present a loosely coupled form of collaboration. On the other hand, a deep partnership relation with a COTS supplier is an advanced form of tightly coupled collaboration.

During the recent years the most common ways to collaborate with a supplier for a software product development have been body shopping and software project subcontracting. However, the future communication software development will be based on the Internet Protocol (IP). This means that the possibilities to use the general IT software components in the traditional communication software development field will increase. Therefore, the communication software field needs to learn new ways to collaborate with component suppliers. On the other hand, the previous software project vendors need to adopt new ways to approach the system vendors.

It is especially the management of the relationships with OSS and COTS sources where the system vendors need to focus on (Figure 2). Joining of the OSS community means that the integrator needs to actively not only download and further develop software but also upload the modified software for the others to use. Most of the interviewed people emphasized that a system vendor should avoid the making of software branches that are only further developed by the integrator itself. In addition, the system vendor whose aim is to produce reliable products has to very carefully evaluate the risks of using unknown source of software. In most cases the system vendor shall have a relationship with a reliable supplier. The supplier, i.e., a risk broker provides the system vendor with the external components that originally were free software, public domain software, shareware, OSS or COTS. It is the supplier

that is responsible for the maintenance of the components. In addition, the supplier delivers the modified OSS components back to the OSS community.

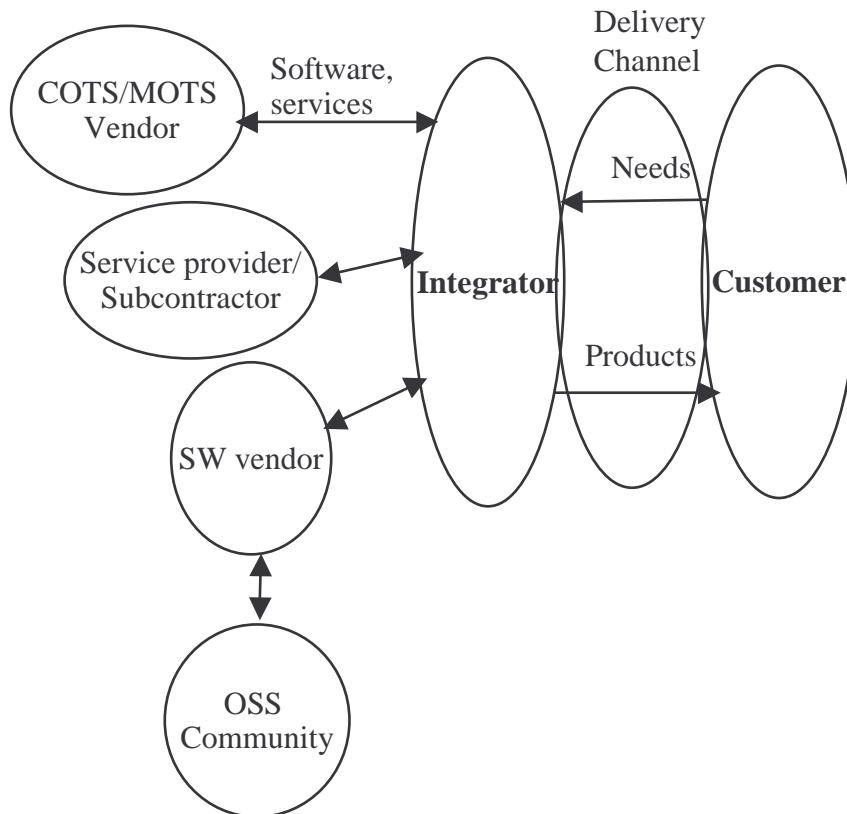


Fig. 2 Ways to acquire external software components to be integrated into communication software

One of the main drawbacks of the integration of external components has been identified to be the long-lasting negotiations and evaluations. One of the interviewed people said that the negotiations have lasted even longer than one year. While the negotiations have been ongoing, the technical requirements have often changed. Especially, this has been the case when the requirements have not been thoroughly analyzed and understood. In some cases when the commercial and legal negotiations were finally finished the component to be acquired was already outdated. Therefore, long-lasting and deep partnership relations with a few preferred COTS supplier are needed to speed up the negotiations. In addition, partnership relation is required for developing shared and feasible software component roadmaps.

Currently there are many software technology areas in the communication software field such as in the base station software development where there are only a few or even none software components available. A system vendor may launch horizontal component markets by ordering components as a subcontracted development work and allowing the vendor to sell the components on the markets. However, a supplier moving from a software project business to software product business faces a need for a change. Often it is more practical to establish a new company for the product business than to try to change the existing company operation mode.

Several information and knowledge management items were brought up during the interviews. One issue that was frequently discussed with the interviewed people was the ability to manage *software requirements*. For example, one of the interviewed people at the system vendor side said that experience has shown that the careful analysis of the requirements reduces the time needed for the component evaluations. A shared concern was that when the relationships between the vendor, end customers and suppliers become wider and wider and more intermediaries occur between your company and the customer, it becomes harder to identify and manage the customer requirements. Interviewed companies had a need for shared practices and knowledge management tools to help them in such problematic situations as when the customer is far away from the company either physically or culturally, when there are several intermediaries involved, e.g. in ASP business, or when the customer base is very wide, including several different customer groups with different needs, concerns and purchasing logics.

4 Nokia Flexifamily™ Platforms Case

Nokia Flexifamily™ platforms provide a carrier-grade foundation for All-IP mobility systems. The aim of all-IP architecture is to enable growth beyond the present voice-centric service paradigm. The all-IP architecture combines multiple media streams for rich call, messaging, and browsing services into a single session. The packet-based traffic is carried over a single network. Target is to provide optimized quality of service, total scalability, and thus the lowest cost per bit. All-IP complements GPRS and EDGE/WCDMA and it supports add-on capabilities and capacity for increasing traffic. In order to offer data services to everyone, industry-wide standardization of interfaces is required, through forums like 3GPP, WAP Forum, Wireless Village, and IPv6 Forum. [15]

The development of Nokia Flexifamily™ platforms included integration of external COTS and OSS components. For example, one part of the platforms is The Nokia FlexiServer™ in the network domain. FlexiServer is a carrier-grade server platform using the Linux® operating system. FlexiServer will be the foundation for core network products with functions such as session control and registers. In radio access, FlexiServer is used for managing the signaling plane of mobility control functions, including common radio resource management. [15] Experiences in FlexiServer development raised a need for COTS and OSS component related process

development. Especially, the decision-making process for the evaluation and integration of an external component was further developed.

During the platforms development effort Nokia invited other industry players to share and standardize open specifications for interfaces of All-IP platforms. The Carrier Grade Linux Working Group under Open Source Development Lab [16] and Service Availability (SA) Forum industry initiatives [17] supports adoption of new technologies. Nokia contributes to standardization initiatives and implements them in products, using a "Networked Product Creation" product development processes and environments. The aim of Nokia is to create an ecosystem with economies of scale and leveraging capabilities - enabling industry innovation for all parties in the value net. Nokia argues that the use of a standards-based system, platforms, software components, and mainstream hardware reduces development, deployment, and operating costs to enable mobile data services to take off. The aim of Nokia is to focus on new key competence areas and leverage industry innovation by combining Nokia's own R&D with industry player partners. [15] This means that Nokia has had to develop and integrate collaboration processes into "Networked Product Creation" product development processes. This paper is strongly based on experiences gathered during the collaboration process development. Currently Nokia has agreed to follow the several collaboration issues covering processes such as a process for COTS and OSS component acquisition and maintenance and a process for glue software development.

5 Summary

This paper described dimensions of the development of COTS and OSS component based embedded software products. The following areas were identified to be related to the acquisition and maintenance of external components: Vision and strategy, business and markets, software engineering processes, software engineering environments and collaboration approaches. Business and markets include value net analysis and revenue logic analysis. The software component integrator needs to describe and analyse the value net of products and services. The value net description shall focus on the changes that the transition from a producer to a consumer organisation causes. Typical areas to be considered are the measuring of a software component value, pricing of software components and the management of the value net.

Collaboration activities shall be included in the value net management. Collaboration between the integrator and vendor is needed to minimize work effort and time spent on the technical, business and legal negotiations and evaluations. The lack of standardized software component interfaces makes the need for a well working relationship even more obvious. Software sourcing in the value net is a topic of a future research.

References

1. Helokunnas, T. & Nyby, M.: Collaboration between a COTS Integrator and Vendors. Proc. of the 7th European Conference on Software Quality. Helsinki, 2002
2. Meyers, B.C., Oberndorf, P.: Managing Software Acquisition. Open Systems and COTS Products. SEI Series in Software Engineering, Addison-Wesley (2001)
3. IEEE STD 1062-1993. IEEE Recommended Practice for Software Acquisition. USA (1998)
4. The Open Source Definition, version 1.9. <http://www.opensource.org/docs/definition.html> (2002), site visited 2002-08-03
5. Szyperski, C.: Component Software: Beyond Object-Oriented Programming. ACM Press & Addison-Wesley, 1997
6. Lim, W. Managing Software Reuse: A Comprehensive Guide to Strategically Reengineering the Organization for Reusable Components. Prentice-Hall, July 1998^{***}
7. Parolini, C.: The Value Net. A Tool for Competitive Strategy. John Wiley & Sons Ltd. (1999)
8. Walter, A., Ritter, T., Gemünden, H.G.: "Value Creation in Buyer-Seller Relationships", Industrial Marketing Management, Vol. 30, Issue 4, May (2001) 365-377
9. Axelson, B. and Easton, G., (eds.): Industrial Network. A New View of Reality. Routledge, London. (1992)
10. Doz, Y.L. and Hamel, G.: Alliance Advantage. The Art of Creating Value through Partnering. Boston, Massachusetts: Harvard Business School Press (1998)
11. Möller, K., Rajala, A., Svahn, S.: "Strategic Business Nets – Their Types and Management", submitted to Journal of Business Research (2002)
12. Schrage, M.: No More Teams! Mastering the Dynamics of Creative Collaboration. Currency/Doubleday, 1995
13. Sonnenwald, D., Pierce, L.: "Information behavior in dynamic work contexts: Interwoven situational awareness, dense social networks and contested collaboration in command and control". Information Processing & Management, 36(3) (2000) 461-479
14. Marmolin, H., Sundblad, Y. and Pehrson, B. : An Analysis of Design and Collaboration in a Distributed Environment, Proceedings of the Second European Conference on Computer-Supported Cooperative Work, ECSCW 91, Netherlands, 1991
15. Nokia (2002): Nokia Flexifamily™ platforms provide an open carrier-grade foundation for All-IP mobility systems. www.nokia.com, visited September 24, 2002
16. Open Source Development lab. <http://www.osdlab.org/projects/cgl/>, visited September 24, 2002
17. Service Availability Forum. <http://www.saforum.org/>, visited September 24, 2002