Analysing software integration scenarios: the case of telecommunications operations software

Oleksiy Mazhelis¹, P. Tyrväinen¹, Erkki Viitala

¹ University of Jyväskylä, P. O. Box 35, FI-40014 Jyväskylä, Finland
² Comptel Corporation, P. O. Box 1000, FI-00181 Helsinki, Finland

(Received December 22 2007, Accepted March 18 2008)

Abstract. Telecom operators deploy a vast number of software systems to support their operations. Vendors of this software often integrate in their products several software systems, in order to enable innovations, minimize customer’s integration efforts, etc. Different integration scenarios can be envisioned, and the issue of identifying more beneficial scenarios is of a great importance to the vendors. This paper aims at analyzing different software integration scenarios from the viewpoint of their support for innovations, and focuses on the context of telecommunications operations software. For each scenario, the overall modularity of the set of software systems is evaluated, and the expected benefits of the scenario are modeled in terms of the modularity gain it provides. The paper introduces a modularity-based approach to the integration scenario analysis, and applies it to compare the benefits that can be obtained from each scenario. As a result of the analysis, the scenarios which are likely to be more beneficial for the vendors are identified.

Keywords: telecommunications software, operations support systems, business support systems, vertical integration and disintegration, system complexity, modularity index

1 Introduction

The telecommunications industry has accumulated a large number of heterogeneous systems each serving as a part of access networks and control networks. The so-called operations support systems and business support systems (OSS/BSS) are responsible for monitoring these networks, as well as for managing their performance, quality of service, faults, configuration, roaming, accounting, customer relationships, frauds, etc. [8, 10].

The OSS/BSS heavily rely on software which, depending on the functional responsibilities of the software systems, can be grouped into several vertical functional groups (such as fulfilment, assurance, billing, etc.) and into several horizontal functional groups (such as customer relationship management, service management and operations, resource management and operations, etc.)[1]. Together, the set of OSS/BSS software systems can be seen as a single large-scale system (a system of systems), wherein individual OSS/BSS software systems can be seen as subsystems or modules which are either operating independently or in concert with others.

Software vendors engaged in OSS/BSS software development often provide a set of software products, while making these products pre-integrated with each other as well as with other related OSS/BSS products.
For the OSS/BSS software vendors, it is critical to assess, prior to large-scale development and integration efforts, products are likely to benefit from integration.

In order to answer the above question of which subsystems to integrate, the reasons for such integration need to be considered. A not exhaustive list of such reasons includes:

- **Sustaining innovation potential.** According to previous studies focusing on biological systems, an attempt to improve individually one of the interconnected modules in a complex system is unlikely to bring benefits at the system level, since changes to this module bring about chains of changes in dependent modules, and the improvements in performance of one module are likely to be outweighed by decreased performance of some of the dependent modules[1]. Thus, in order to succeed in innovation, the organization should coordinate innovation efforts across closely interdependent modules, and hence should in-source and therefore integrate these modules[6].

- **Customer-specific integration effort minimization.** Customers of the OSS/BSS software - telecom operators - strive to minimize their costs by avoiding the integration job, if possible. Software vendors, in turn, aim at minimizing their integration costs to gain higher margins. Therefore, to achieve a win-win scenario, the vendors are likely to provide products or subsystems which they have already integrated (or pre-integrated).

- **Sharing similar knowledge base** as well as other design and implementation related assets needed for implementing the products. Whenever commonalities exist across ready products and new products, by reusing earlier experience in implementing the new products or their sub-systems, the vendors also improve the quality of the implementation as compared with a “do-it-from-scratch” approach. Therefore, the vendors are likely to integrate in their offerings the products whose implementation requires similar knowledge.

This paper is aimed at identifying likely integration scenarios based on the expected change in innovation potential (positive or negative) that the integration may incur. Meanwhile, it should be noted that other factors affecting the likelihood of integration, such as minimisation of customer-specific integration efforts, are not considered in this paper, but are left for further work.

From the innovation potential viewpoint on the integration, whenever a system consists of several modules and some of them are interdependent, innovations are more likely to be successful in i) those modules that are rather independent of the others, or in ii) an interdependent module if the set of interdependent modules are under the control of a single organization thereby allowing it to coordinate the innovation activities spanning several modules. Therefore, as mentioned above, in order to succeed in innovation, the organization should out-source relatively independent modules, but it should integrate (in-source) closely interdependent ones[6].

Frenken[6] introduced the so-called indicator of modularity, which indicates how independent the system’s modules are from each other. The minimum modularity corresponds to the system in which each module has the pleiotropy (the number of functions affected by a change in the module) equal to the total number of functions in the system. The maximum modularity is achieved when one of the modules has the maximum pleiotropy, while all the others have pleiotropy equal to one; the module with the maximum pleiotropy plays in this case the role of a global interface mediating the interdependencies between all the other modules. Therefore, as mentioned above, in order to succeed in innovation, the organization should out-source relatively independent modules, but it should integrate (in-source) closely interdependent ones[6].

The calculation of the above indicator of modularity takes into account the degree of interconnectedness among the subsystems. In addition to interconnectedness, the modularity can be hypothesized to depend also on other factors, such as the complexity of interfaces between subsystems, and the number of interface variations that need to be maintained (see Fig. 1 below). Indeed, the existence of a complex interface between two subsystems suggests that these subsystems are tightly coupled; according to software system design principles, tightly coupled components are to be placed in a same module. The variability of an interface, on the other hand, suggests that at least one of the interacting subsystems is subject to a change, and therefore integrating these subsystems inside a module would transfer the management of the variation from the customer to the
software vendor. However, both factors - the complexity of interfaces and the variability of interfaces - are left outside of the scope of this paper.

The above hypothesized relationship between the successfulness of the integration strategy on one side, and the system modularity on the other side, is used in this paper in order to compare several integration strategies possible in the domain of OSS/BSS software. Namely, integration scenarios are modelled by using the connectivity matrix (which will be described below), and the corresponding system modularity values are calculated. The integration scenarios are then ranked according to the expected increase in the system modularity that they may incur, and the more likely integration scenarios are identified.

The paper is organized as follows. The next section introduces our approach to the analysis of integration scenarios. In Section 3, this approach is applied to evaluation of integration strategies of the OSS/BSS software, and the results of the evaluation are then discussed in Section 4. Finally, conclusions to the paper are provided in Section 5.

2 Using the connectivity matrix to assess modularity of OSS/BSS software

In this section, our approach to the analysis of integration scenarios based on the comparison of modularity index values is introduced, and the process of applying this approach to the OSS/BSS software is described.

2.1 Using modularity index to compare integration scenarios

In our approach, alternative integration scenarios are compared based on their system modularity values. Alternative scenarios are modeled by using the connectivity matrix, where the content of cell \((i, j)\) indicates whether changes in subsystem \(j\) have an effect on subsystem \(i\). For each scenario, the subsystems to be integrated are identified, and the corresponding columns and rows of the connectivity matrix are merged.

\[
M = 1 - \frac{P}{N},
\]

where \(N\) is the number of modules in the system, and \(P\) is a system pleiotropy evaluated as

\[
P = \log_F \prod_{n=1}^{N} P_n.
\]
Here, \( P_n \in \{1, F\} \) denotes pleiotropy of subsystem \( n \); and \( F \) designates the number of functions performed by the system (we assume that each subsystem performs one function, i.e. \( F = N \)).

The value of the system modularity is calculated independently for each of the alternative integration scenarios, as well as for the null scenario where no integration is present (in order to model the system prior to the integration). The integration scenarios are then ranked according to the expected change in the system modularity that they may incur. Finally, the top-ranked scenarios are selected as the most likely ones, since they are expected to incur the highest gain in the system modularity and therefore also in the innovation potential.

### 2.2 Analyzing modularity of OSS/BSS software

In order to assess the modularity of OSS/BSS software, we studied the interdependencies between different OSS/BSS subsystems, using the documentation made available by the TeleManagement Forum (TMForum). In May 2000, the TMForum established a New Generation Operations Systems and Software (NGOSS) program. This program contributes to the harmonization and interoperability of OSS/BSS software through the elaboration of a standardized telecom process model called enhanced telecom operations map (eTOM). NGOSS also develops a shared information and data (SID) model serving as a common information definition for NGOSS applications, and further, a technology neutral architecture (TNA) aimed at standardizing component-based integration among OSS/BSS systems.

Another TMForum’s technical program, called the OSS through Java (OSS/J) initiative, is focusing on the development of application programming interfaces (APIs) aligned with the TMForum’s NGOSS framework. In the frame of this initiative, a roadmap of APIs is elaborated, and for each OSS/BSS application area, a software specification, a reference implementation, and a technology compatibility kit is (to be) provided.

In the OSS/J document describing the software API specification and implementation roadmap, the APIs to be implemented are listed, and their dependencies are illustrated\[^9\]. For instance, the dependencies of the customer management API are illustrated in the Fig. 2 below.

![Customer Management Interactions](image)

**Fig. 2. Customer Management Interactions (adopted from [9])**

For the purposes of the analysis, the diagrams illustrating different API and their dependencies were merged, and the overall diagram produced was used in order to construct the connectivity matrix, where the content of cell \( i, j \) indicates whether changes in API \( j \) have an effect on API \( i \).

It should be noted that the elements involved in charging and billing appear to be underrepresented in the OSS/J roadmap (the billing API is listed as an item for further work in the roadmap). Therefore, in order...
to complement the connectivity matrix with the dependencies present in the billing- and charging-related modules, 3GPP specifications devoted to charging and billing were employed. Also, individual judgment was applied in order to reconstruct the dependencies with other modules present in the matrix. Both the overall API inter-dependency diagram and the connectivity matrix are illustrated in Fig. 3 and Fig. 4 (note that the added billing- and charging-related modules are not shown in the diagram in Fig. 3 for simplicity).

The connectivity matrix shown in Fig. 4 was in turn employed in order to study how different integration scenarios affect the resulting system modularity. For each of the integration scenarios, the modules to be integrated were identified, and corresponding columns (rows) of the connectivity matrix were merged. After that, the system modularity of the transformed matrix was calculated, and the produced values were used for ranking the scenarios and identifying the most likely ones. The details of the scenarios and the results of their comparison are provided in the next section.

3 Comparing OSS/BSS software integration scenarios

The OSS/BSS software is aimed at supporting the telecom operators’ business processes, which are documented in the TMForum’s eTOM. The processes specified in the eTOM are split into three major process areas: i) operations, ii) strategy, infrastructure and product, and iii) enterprise management. In this study, we focus on the software supporting the operations process area, since it includes the core customer operations processes and represents “the focal point of the eTOM framework”[4]. Within the operations process area, eTOM defines several vertical and horizontal process groupings shown in Fig. 5. There are four horizontal process groupings in the operations process area:

- Customer Relationship Management (CRM) software manages customer interface, and provides ordering, problem handling, service level agreement (SLA) management, and billing functionality.
- Service Management and Operations (SM & O) software is responsible for service configuration, problem management, performance analysis, and rating.
- The software in the Resource Management and Operations (RM & O) segment includes resource provisioning, trouble management, and performance management software.
- Finally, Supplier / Partner Relationship Management software implements operational interfaces with operator’s suppliers and partners.

Similarly, there are four vertical process groupings:

- Fulfillment software provides the customer with the products they ordered.
- Assurance software ensures, through proactive or reactive actions, the quality of the services delivered to the customers.
- Billing software is responsible for collecting usage statistics and issuing bills to the customers, as well as provides a customer interface for billing-related inquiries.
- Operations Support and Readiness (OS & R) software offers management, administrative, and logistics support to the other groups.

These groupings were used as a basis for selecting the integration scenarios to be compared. In total, the following seven integration scenarios were selected (information about Supplier / Partner Relationship Management software was not available in the OSS/J API roadmaps, and hence this grouping was excluded from the list of scenarios):

- Vertical integration of Fulfillment software
- Vertical integration of Assurance software
- Vertical integration of Billing software
- Vertical integration of OSR software
- Horizontal integration of CRM software
- Horizontal integration of SM & O software
- Horizontal integration of RM & O software

*MSEM email for contribution: submit@msem.org.uk*
Fig. 3. Diagram of OSS/J API inter-dependencies
The above integration scenarios were compared based on the change in the system modularity they incur. The produced system modularity values for these scenarios are shown in Tab. 1 below along with the number of modules being integrated. The first row in the table, denoted as “None”, corresponds to the null scenario.

Besides the results of integrating all the modules belonging to each grouping, Tab. 1 also shows the results of integrating only higher-pleiotropy modules within each grouping (in this study, modules with $P_n > 2$ were considered as higher-pleiotropy modules). The results of integrating only higher-pleiotropy modules are shown...
in Tab. 1 in the “Best modularity index” column. The values in bold indicate that the modularity is greater as compared with the no-integration scenario. Finally, the graph in Fig. 6 compares the values of modularity index for different scenarios, as well as for the no-integration case.

As can be seen from Tab. 1, when all modules in a vertical/horizontal are integrated, only one integration scenario - Horizontal-CRM - improves modularity, while the others decrease it. Meanwhile, when only higher-pleiotropy modules are integrated, most of the integration scenarios increase the system modularity, with the only exception being the Vertical-Fulfilment scenario.

The results obtained suggest that the value of the system modularity greatly depends on whether all modules in a vertical/horizontal grouping, or only the modules with higher pleiotropy are integrated. According to Tab. 1, the modularity is significantly higher in the latter case. This is in accordance with[6] suggesting that the highly interconnected modules (i.e. modules with higher pleiotropy) need to be integrated so that system modularity would improve.

<table>
<thead>
<tr>
<th>Integration strategy</th>
<th>Modularity index (all modules)</th>
<th>Number of modules in grouping</th>
<th>Best modularity index (modules with (P_n &gt; 2))</th>
<th>Number of modules with (P_n &gt; 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>0.706</td>
<td>0</td>
<td>0.706</td>
<td>0</td>
</tr>
<tr>
<td>Vertical - Fulfillment</td>
<td>0.684</td>
<td>6</td>
<td>0.701</td>
<td>3</td>
</tr>
<tr>
<td>Vertical - Assurance</td>
<td>0.700</td>
<td>9</td>
<td>0.717</td>
<td>6</td>
</tr>
<tr>
<td>Vertical - Billing</td>
<td>0.701</td>
<td>7</td>
<td>0.711</td>
<td>5</td>
</tr>
<tr>
<td>Vertical - OS &amp; R</td>
<td>0.699</td>
<td>5</td>
<td>0.710</td>
<td>3</td>
</tr>
<tr>
<td>Horizontal - CRM</td>
<td>0.738</td>
<td>7</td>
<td>0.749</td>
<td>5</td>
</tr>
<tr>
<td>Horizontal - SM &amp; O</td>
<td>0.679</td>
<td>9</td>
<td>0.708</td>
<td>5</td>
</tr>
<tr>
<td>Horizontal - RM &amp; O</td>
<td>0.601</td>
<td>16</td>
<td>0.711</td>
<td>7</td>
</tr>
</tbody>
</table>

Fig. 6. Values of modularity index for different scenarios

Table 1. System Modularity for Different Integration Scenarios
4 Discussion

In the paper, an approach to identifying likely software integration scenarios based on the system modularity has been introduced and applied in order to find likely integration scenarios in the context of the telecommunications operations software. This modularity-based approach assumes that software subsystems interacting with each other are more likely to be integrated in software vendors’ offerings. In order to verify the validity of this assumption, we used the TM Forum’s Product Directory (http://www.tmforum.org/browse.aspx?catID=2528) which provides examples of software products corresponding to different segments of the TM Forum’s Telecom Applications Map.

For each vendor, its products were analyzed; whenever products of the same vendor were present in more than one segment, it was interpreted as an indication of these products being integrated or pre-integrated. Having compared the Product Directory with the interconnectivity matrix, we found that those subsystems which are inter-dependent according to the interconnectivity matrix, are more likely to be (pre-) integrated by the vendors in their product offerings. More specifically, 31.8% of companies were found to integrate dependent subsystems, while 24.4% of companies integrate independent subsystems; the difference is statistically significant at the level of 0.0054 (**) in a two-sample t-test assuming unequal variances. This suggests that the information about interconnections present in the interconnectivity matrix can be used to identify the connected subsystems, and therefore to identify the subsystems that are more likely to be integrated.

In the previous section, the values of the system modularity were evaluated for seven scenarios of OSS/BSS software integration. It was found that the integration may result in an improved modularity, but it may also make the modularity lower. As shown in Fig. 6, integration is the most beneficial in the case of the Horizontal-CRM integration scenario, followed by the Vertical-Assurance, Horizontal-RM & O, and Vertical-Billing scenarios. The Vertical-OS & R and the Horizontal-SM & O scenarios introduce the least increase in modularity, while the Vertical-Fulfillment scenario gives a lower modularity than the “no integration” scenario. This suggests that the three latter scenarios are less likely to occur since they are less likely to bring benefits in innovation potential to the organizations following them.

A related study on the strategies of OSS software vendors[5] provides some support for these observations. According to the available OSS market data[2], during 2005, the number of companies providing software products in multiple market segments in the horizontal CRM layer - i.e. applying horizontal strategy in the CRM layer - was equal to 25 companies, which is more than the double of the number of companies applying horizontal strategies in the RM & O or the SM & O layers. This is in alignment with the hypothesis that the OSS vendors, while defining the scope of their market offering in the CRM domain, follow the modularity maximization approach described in this paper.

In Fig. 7 below, the integration scenarios considered are marked in the preferred order of integration, based on the change in modularity (positive or negative) they incur. As mentioned above, the most significant improvement of modularity was achieved when the CRM applications were integrated. This is due to the fact that the CRM applications are closely interrelated. As can be seen from Fig. 4, the CRM applications, which include customer management, order management, problem handling, customer SLA management, and billing & collections management, have a significant number of interdependencies with each other. Therefore, innovating in either of these systems requires an organization to have a control over the other CRM applications, thus making this integration strategy worthwhile.

The relatively small modularity improvement of the Vertical-OS & R scenario can be attributed to the high integration of the OS & R applications already in the null (no-integration) scenario. Indeed, following the NGOSS framework, OSS/J inventory API standardizes the integration with (i.e. access to or communication with) different inventories. As written in “OSS Inventory API Overview” on Inventory API, “all Inventory functions share common abstractions (e.g., entities, associations, entity specifications) and common base interaction model (e.g., queries based on traversal of relationships, atomic update procedures etc.),” though “Each function has its specific set of inventory entities and relationships, its specific business logic and interacts with different subset of OSS functions.” Thus, the OSS/J roadmap:

- standardizes inventories within horizontal layers, resulting in a single inventory per layer as opposed to a set of distinct inventories at each layer used by different applications within this layer, and

MSEM email for contribution: submit@msem.org.uk
integrates the abstractions and interaction models to be used in different inventories.

The first item above is incorporated already into the null scenario shown in Fig. 4, while the second item is modelled as the Vertical-OS & R scenario. Since the null scenario already involves a significant degree of integration, there is not much room left for additional integration in the Vertical-OS & R scenario.

In summary, this work contributes to the analysis of complex software systems by introducing a new approach to modelling software integration scenarios on the basis of the system modularity, and demonstrates the feasibility of this approach by applying it to model integration scenarios in the domain of the telecommunications operations software. The evaluation of the system modularity in this work is based on a connectivity matrix indicating inter-module dependencies, as opposed to the generalized element-function matrix in [6] where dependencies between modules and system functions are reflected.

It should be noted that the connectivity matrix which was created in this study, is likely to be incomplete. Indeed, such OSS/BSS elements as revenue assurance, interconnect billing, resource domain management, etc. are not represented in the OSS/J roadmap, and hence are also missing from the matrix. Furthermore, some of the connections in the matrix may be erroneous, either due to inconsistency in the OSS/J documentation, or due to the errors in modelling the dependencies between subsystems involved in the charging and billing processes. These mistakes in the matrix may negatively affect the internal validity of the results of the analysis. The problem of reconstructing higher-level design abstractions from source code and available documentation, known as the design recovery problem, is studied in the domain of reverse engineering [3]. Therefore, in future work, the advances in the domain of reverse engineering needs to be employed in order to revise the information in the connectivity matrix.

Another limitation of this work stems from the fact that the connectivity matrix models the existence of dependencies among modules, but it does not utilise the information about the complexity of individual interfaces nor the information about complexity of the modules themselves. Such information may be useful in further prioritizing the potential integration scenarios. Therefore, in future work we are planning to equip the proposed approach with the possibility to utilise this information.

5 Conclusions

Contemporary software often consists of numerous subsystems operating either independently or in concert with others. Due to numerous interdependencies between these subsystems, successful innovations in one subsystem are unlikely, if the software vendors do not have control over interdependent subsystems. Therefore, in order to support innovations in interdependent subsystems, software vendors often integrate some of
the subsystems in their products. We assume that successful integration of software subsystems should result in an increased modularity of the overall system, and therefore, likely integration scenarios may be identified based on the increase of the system modularity value.

Based on the above assumption, a modularity-based approach to comparing software integration scenarios has been introduced in this paper, and this approach has been applied in the context of telecommunications operations software. Several potential scenarios of integrating telecommunications operations software have been compared in the paper on the basis of the gain in the system modularity they incur. According to the obtained results, the CRM software subsystems were found to be the primary candidates for the integration, followed by the subsystems in the Assurance vertical and by the subsystems in the RM & O horizontal. In further work, the modularity-based approach needs to be extended by taking into account the information about complexity of interfaces between modules and about the number of interface variations. Besides, the connectivity matrix which serves as the source information for evaluating the telecommunications operations system modularity needs to be refined.

References