Medical Informatics Europe 2002
G Surján et al. (eds)
IOS Press 2002

A process for specifying integration for multi-tier applications in healthcare

Juha MYKKÄNEN, Jari PORRASMAA, Mikko KORPELA

Computing Centre, HIS Research and Development Unit University of Kuopio, Finland

Abstract. Integrating heterogeneous application systems in healthcare is needed to support clinical work, patient-centric care, regional interoperability and utilization of both valuable legacy systems and new technologies. The integration process is quite complicated, and must be supported by flexible integration processes and methods. The integration methods should support evaluation and specification of different integration approaches and technologies on many different interoperability levels. We introduce a process for specifying many integration decisions for a given integration situation. The process is part of an integration method, which is validated in PlugIT project in Finland.

1. Introduction

The integration of healthcare software systems has remained one of the most prominent issues in healthcare software development [1]. Many workflows in healthcare facilities involve more than one application [2], and patient-centered care, regional healthcare networks and changing work in healthcare require integrating these systems. The application architectures in healthcare are evolving towards distributed, component- and service-based and web-enabled systems [1, 3, 4, 5]. The heterogeneous environment, legacy systems and pressure to deliver quality software rapidly add even more pressure on systems integration. Furthermore, there are many complementary and also overlapping technologies and standards available for integration. Selecting appropriate approaches for each integration need is a complex task, and systems integration in healthcare requires defining more specific processes for the integration [6, 1].

In this paper, we propose a process for specifying integration in multi-tier applications in the healthcare domain. The process is part of an integration method we have specified. The methods for application integration have focused on the functional integration and interface specification, or the integration activities have been tightly coupled with the development process (e.g. [7, 8]). Our approach includes additional interoperability levels, and a dedicated process for creating an integration profile. The work is based on the literature and our initial experience in integrating multi-tier component-based and legacy systems [9].

2. Interoperability levels

Applications must be integrated on various heterogeneous levels. *Interaction protocols* are needed on these levels to specify the interoperability solutions between systems. In our method, the interaction protocols contain definitions of used technologies and standards, definitions of the syntax of the interfaces between systems, and definitions of the semantics

of the interfaces. The seven-layer interoperability model, whose layers have been defined in [10], includes the following interaction levels:

1. *Technical interfaces (L1)*: technologies used to implement the interoperability,
2. *Technical infrastructure (L2)*: activation, error handling, calling conventions etc.,
3. *Application infrastructure (L3)*: application architecture conventions [11], interface style, design patterns [12], security conventions, such as user identification etc.,
4. *Functional interfaces (L4)*: the precise interfaces of the services, which are either data-oriented (such as HL7 message definitions [13]) or processing-oriented (such as the OMG healthcare service definitions [14]),
5. *Semantics (L5)*: the knowledge about the exact meaning of the functional interfaces. For example, different measurement units in systems may cause need for mediation,
6. The *functional reference model (L6)*: the system-internal design or implementation factors that affect the interoperability, such as the common functionality, classifications and terminologies,
7. The *development lifecycle (L7)* interoperability extends the interoperability between systems to include requirements, analysis, and design phases of the systems in addition to the development.

The first six of these interoperability levels (L1-L6) must be addressed to achieve interoperability. There are also higher levels, which include setting up or mediating between different care practices, policies and strategies between the participating organizations. These policy-level issues, as well as the most technical networking technologies are not considered in our method.

3. Positioning integration points in the application architecture

In our previous work [15], we have used the *business component architecture* [10] in specifying a component-based architecture and migration paths for healthcare information systems. In the architecture, several tiers are used to separate the responsibilities of different parts of the system. The execution environment (infrastructure) provides different services in different tiers. In multi-tier [e.g. 16] or client/server systems, the tiers of the model can usually be identified, at least as logical layers, some of which may be implemented using healthcare-specific integration platform or middleware [17, 14]. We use the tiers to locate *integration points* between systems (Figure 1):

- The *user tier (U)*, which typically contains a graphical user interface, located in end-user workstation. The user tier may integrate several services or components from lower tiers.
- The *workspace tier (W)*, which contains application logic for supporting the work of one user, often also the state and the clinical context of the application. Location of the workspace tier may be together with the user interface in the user workstation (fat client) or it may be located on a server (e.g. web server, thin client).
- The *enterprise tier (E)* provides distributed application logic of the component or the application, typically used over the network. This tier is critical for many quality attributes of distributed applications, and the infrastructure may contain many domain-specific or domain-independent services to support applications (e.g. events, transactions, person identification). The physical location of the enterprise tier is typically an application server.
- The *resource tier (R)* typically contains the persistence aspects of the application, or encapsulates legacy systems. It is preferable to integrate applications on the upper tiers rather than on data storage or the resource tier, which can risk the data integrity

controlled by business rules in upper tiers. The functional reference model (L6) often has strong binding to the database of the application.

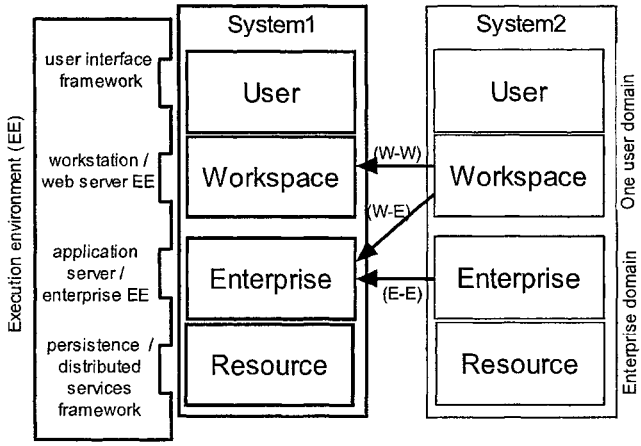

Fig. 1: Application tiers and the execution environment (adapted from [10]).

The application integration can be achieved within one tier or between different tiers. In our approach, integration solutions between any tiers are possible. For example, a web server application may use services of an application server (W-E), which are implemented by wrapping the user interface of a legacy system (E-U). Some standards (e.g. HL7 and XML) are somewhat neutral in relation to which tiers they are used in (E-E, E-W). Other standards are usable in some tiers only, for example CCOW [18] for integration in workspace tier (W-W). (See Figure 1).

4. Proposed process for creating a protocol profile for integration

In this section, we propose a process for specifying an integration protocol profile for healthcare applications using the interaction levels and reference architecture of previous sections. The proposed process has the following steps (See Figure 2):

1. *Modeling the integration domain.* Use the functional requirements and existing functionality to decide on what data or functionality must be shared. Methods for specifying this level include investigating or creating use cases and scenarios as well as defining analysis and design models for the integration domain [19, 13, 2]. As a result, integration points in the workflow and the functional contents of the systems (L6) and semantic mediation (L5) solutions [20] should be defined. This step is crucial in integration, as it sets foundation for the following phases.
2. *Examining the application architecture.* Use the non-functional and functional requirements and the existing application architecture to decide whether the integration is user-centric or enterprise-centric. In addition to the core functionality in the integration, also security, error handling etc. must be considered. Specify the integration points in the different distribution tiers in the application architecture and possible mediation between different architectural approaches in applications (L3).
3. *Examining the infrastructure.* It is necessary to utilize the existing technical infrastructure in addition to introducing new approaches. Consider the specific technologies and infrastructure used in applications as candidates for implementing

the integration (L1, L2). In legacy systems, the technologies may not be suitable for implementing new integration solutions. Building adapters or wrappers to different systems can alleviate the situation [10].

4. *Identifying the functional interfaces.* Select the style (e.g. processing-oriented operations or data-oriented messages) and basic contents of the functional interfaces (L4). As a result, a platform- or technology-neutral definition of interfaces should be produced.
5. *Choosing the integration technologies.* Use suitable assessment methodologies ([21], [7]) to specify technical standards, including the introduction and selection of new technologies for the integration point (L1, L2). Assess also different tools and methods for a given technology. Consider also the tools and technologies in the existing infrastructure.
6. *Specifying functional interfaces.* Specify the precise, technology-specific functional interfaces or message definitions (L4) using the technical standards and architectural decisions on levels L1, L2 and L3 and platform-neutral definition of the interfaces defined in step 4.
7. *Choosing tools and products.* The resulting protocol profile from steps 1-6 gives guidelines for acquiring additional tools and building adapters or wrappers and implementing the interoperability in the integration project. It can also be used as a guideline for separate teams or companies, whose products need to interoperate, in selecting solutions for the development of the systems, or in supporting acquisition of integration platforms or middleware [13, 14, 18].

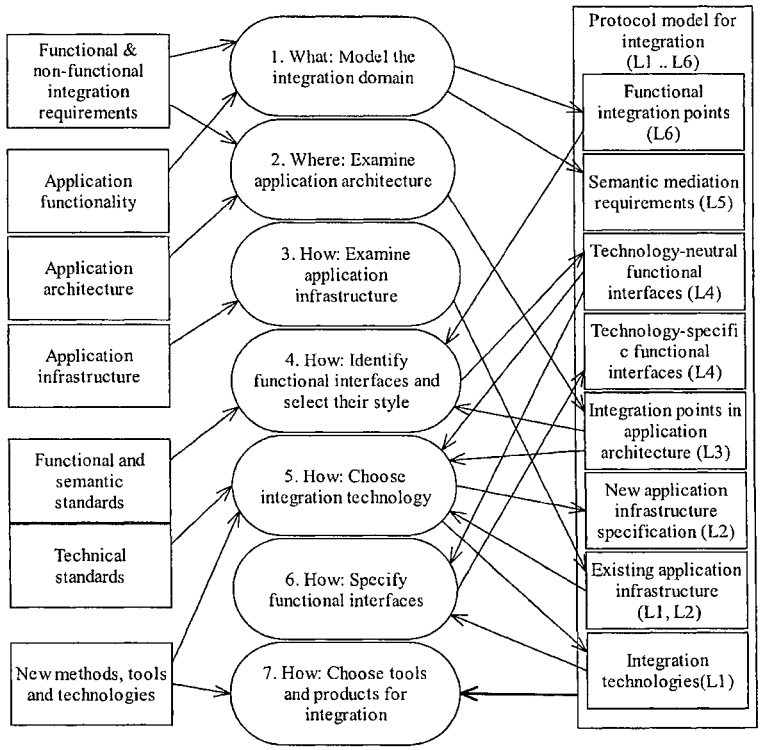

Fig. 2: Proposed process for specifying application integration in healthcare.

The phases of the process are partly overlapping, and the overall process should be conducted in a reasonable timeframe. The resulting integration profile should also be evaluated, and resource factors including money, know-how and time-to-market requirements should be considered as well. Several iterations of the process are usually needed to obtain a detailed model for the integration.

5. Discussion

Building applications with flexible functional reference model (L6) and infrastructure that includes open technologies (L2, L3) supports the process. Healthcare-specific standards are becoming more and more available on these levels, and many other levels can be addressed with domain-independent solutions. The integration effort reduces, if the specification is conducted in an early phase of the development process and specifications are shared between development teams (L7).

In comparison to other integration methods [7, 8], our method includes additional levels, such as existing application architecture and infrastructure [11]. It also includes a practical specification process, supporting efficient and systematic evaluation and selection of standards and tools. Integration platforms based on open architecture and standards (e.g. [17]) provide many technical and architectural solutions to different layers. However, when integrating systems within the platform, the process is still needed, and the platform simplifies the process by setting constraints to the selection of technologies. Furthermore, such platforms are not yet generally used [6].

In healthcare, the applications cannot usually be unified on architectural levels, and the integration is often peer-to-peer, not centralized, activity. Our process takes this into account by using flexible reference architecture and by enabling use of suitable methods in different phases and interoperability levels. Our method can be applied in integration of legacy systems and in integration during the development process. The method does not cover the assessment of business reasons or gathering the requirements for the integration situation. Existing products, tools and technologies, as well as policy, legislative and organizational issues usually affect the requirements.

6. Conclusions and future work

We presented a process for creating a protocol profile for systems integration in healthcare. The process is an essential part of an integration method we have specified. The method takes architectural and infrastructure considerations into account in addition to functionality and interfaces between the systems. It can be used to specify interaction between systems in integration or development projects, or to support acquisition of open platforms and infrastructure.

We are validating our method and the process in integrating systems in PlugIT project [9]. Future work also includes assessment of different integration technologies and standards in healthcare [4, 13, 14, 18] using the interoperability levels and the reference architecture.

Acknowledgements

This work is part of PlugIT project, which is funded by the National Technology Agency of Finland TEKES together with a consortium of software companies and hospitals.

References

[1] K.A. Kuhn and D.A. Giuse, From Hospital Information Systems to Health Information Systems: Problems, Challenges, Perspectives. *Methods of Information in Medicine* **4** (2001) 275-287.
[2] P.J. Toussaint and H. Lodder, Component-based development for supporting workflows in hospitals. *International journal of Medical Informatics* **52** (1998) 53-60.
[3] A. Klingler, An open, component-based architecture for healthcare information systems. In: A. Hasman *et al.*, (eds.), Medical Infobahn for Europe: Proceedings of MIE2000 and GMDS2000. IOS Press, Amsterdam, 2000, pp. 997-1001.
[4] Healthcare Information Systems Architecture Part 1 (HISA) Healthcare Middleware Layer. Comité Europeén de Normalisation / European Committee for Standardisation (CEN), 1997.
[5] B. Blobel, Application of the component paradigm for analysis and design of advanced health system architectures. *International journal of Medical Informatics* **60** (2000) 281-301.
[6] P. Knaup *et al.*, Integrating specialized application systems into hospital information systems - obstacles and factors for success. In: A. Hasman *et al.*, (eds.), Medical Infobahn for Europe: Proceedings of MIE2000 and GMDS2000. IOS Press, Amsterdam, 2000, pp. 890-894.
[7] M. Juric, I. Rozman and M. Hericko, Integrating Legacy Systems in Distributed Object Architecture. *ACM Software Engineering Notes* **2** (2000) 35-39.
[8] P. Kruchten, Rational Unified Process - An Introduction, Addison-Wesley, 2000.
[9] PlugIT project home page: http://www.uku.fi/atkk/plugit/
[10] P. Herzum and O. Sims, Business Component Factory. Wiley Computer Publishing, New York, 2000.
[11] D. Garlan, R. Allen and J. Ockerbloom, Architectural Mismatch: Why Reuse is so Hard. *IEEE Software* **6** (1995) 17-26.
[12] E. Gamma *et al.*, Design Patterns: Elements of Reusable Object-Oriented Software. Addison Wesley, Harlow, 1994.
[13] G.W. Beeler, HL7 Version 3 - An object-oriented methodology for collaborative standards development. *International journal of Medical Informatics*, **48** (1998) 151-161.
[14] CORBAmed Roadmap, Version 2.0 (draft). OMG Document CORBAmed/2000-05-01, OMG Healthcare Domain Task Force, 2000.
[15] J. Mykkänen *et al.*, From Legacy and Client/Server Systems to Components in Healthcare Information Systems in Finland. In: V. Patel *et al.* (eds.): MEDINFO 2001. IOS Press, Amsterdam, 2001, pp. 745-749.
[16] S. Chu and B. Cesnik, A three-tier clinical information systems design model. *International journal of Medical Informatics*, **57** (2000) 91-107.
[17] Y. Xu *et al.*, Integrating Medical Applications in an Open Architecture Through Generic and Reusable Components. In: V. Patel *et al.* (eds.): MEDINFO 2001. IOS Press, Amsterdam, 2001, pp. 63-67.
[18] The Clinical Context Object WorkGroup: It's Standards and Methods. Document CCOW98-02-16, The Clinical Context Object Workgroup, 1998.
[19] I. Jacobson *et al.*, Object-Oriented Software Engineering – A Use Case Driven Approach, 2nd edition. Addison-Wesley, Harlow, 1995.
[20] P. Degoulet *et al.*, Rationale and design considerations for a semantic mediator in health information systems. *Methods of Information in Medicine* **37** (1998) 518-526.
[21] P. Allen, Component specification and assessment. *Component development strategies*, **10** (2001) 1-16.