An Information System Architectural Framework for Enterprise Application Integration

André Vasconcelos
CEO - Centro de Engenharia Organizacional, INESC
Rua Alves Redol 9
1000-029 Lisboa, Portugal
andre.vasconcelos@ceo.inesc.pt

Miguel Mira da Silva
Instituto Superior Técnico
Av. Rovisco Pais
1049-001 Lisboa, Portugal
mms@dei.ist.utl.pt

António Fernandes
Instituto Superior Técnico
Av. Rovisco Pais
1049-001 Lisboa, Portugal
antonio.fernandes@gsi.inesc.pt

José Tribolet
CEO - Centro de Engenharia Organizacional, INESC
Rua Alves Redol 9
1000-029 Lisboa, Portugal
jose.tribolet@ceo.inesc.pt

Abstract

Information system (IS) architectures have not paid enough attention to integration in the past because integration was not important to build ISs from scratch. However, with the variety and number of ISs in medium/large organizations increasing, including ERP systems, the need for integration is bigger than ever. Furthermore, most organizations now want to integrate their ISs with those belonging to other organizations. In this paper we propose an extension to our previous proposals for representing IS architectures in order to properly support a large variety of integration scenarios between IS, including intra and inter organizations. In particular we support manual and automatic, synchronous and asynchronous integration. We also present an example to illustrate the proposal with real world IS integration needs.

Keywords: Information System Architecture, Information System Integration, CEO framework, Enterprise Architecture, Enterprise Application Integration.

1. Introduction

Integration between software components has always been a fundamental part of any information system. Recently, its importance has been growing due to the need of integrating diverse information systems, both within and between organizations. The move towards ERPs in the last 10 years has not reduced the need for integration, but it has even increased it. And integrating diverse information systems to react online to external events is a necessary condition for e-business [1].

Information system architectures (ISAs) have not paid sufficient importance to integration because they assume that enforcing the existence of a single database eliminates the need for integration.

Although this might be true, within certain proprietary IS, the fact is that more and more organizations are installing a number of incompatible information systems (some are ERPs, but most are specialized for a specific task) that cannot share a single database but need to share data. So the need to integrate information systems cannot be avoided anymore, and this leads to new challenges in terms of information system architectures.

In this paper we build on previous research performed by our research group (CEO) in this area and complement this research with a proposal to incorporate integration aspects into an ISA.

The paper starts with an overview of information system architectures and presents our own CEO framework that had already identified high-level concepts for representing integration. In particular, a concept called "IS Service" can be used to represent integration between two information system components. In our previous work however, nothing had been proposed to represent integration at the application or technological levels.

We then present a brief introduction to the most important concepts in integration, in particular to show how much richer services integration can provide for than the current RPC-like synchronous services. These RPC-like services, for which Web Services [2] are but the latest incarnation, can be used to integrate software components, but are clearly inappropriate to integrate information systems – especially across organizations.

Assuming the limitations of current ISA to represent integration are clear, we then propose a set of new concepts, namely the IT Integration Block and the IT Integration Service, which together can describe a variety of real-world integration scenarios in ISA. The "Service" concept is not limited to synchronous integration anymore. In particular, we propose that integration should be classified according to automation level (manual or automatic) and role type (source or target). We also propose that integration services should be characterized according to their technological, synchronism, and organizational level.

Finally we present a real-world example taken from a project on food safety in which we participate. This example illustrates how the concepts proposed in the paper can be used to represent integration between information systems, both at the IS and IT levels. The results obtained are then discussed and compared against other common approaches.

2. Overview of Information System Architecture

The Information System Architecture (or ISA, for short) represents the structure of the components, their relationships, principles and directives [3] with the main purpose of supporting business [4].

In the 80s, a software architecture and ISA were considered synonymous. But in the 90s emerged the need for manipulating concepts that exceeded the description of how a system was internally built. The Zachman Framework [5] can be considered the first important signal that software architectures were not enough.

While software architectures represent internal system details (using, for example, E-R and DFD diagrams) ISA focus on the high-level business processes [6], [7]. Using the “city” as a metaphor, we can use the concept of “IS urbanization” to emphasize the need for models to guide the evolution of IS independently of current technological trends [8].

An ISA can be divided into three levels [9]:

- Informational (or Data) Architecture – represents main data types that support business;
- Application Architecture – defines applications needed for data management and business support;
- Technological Architecture – represents the main technologies used in application implementation and the infrastructures that provide an environment for IS deployment.

2.1. Informational Architecture

The major purpose of the Informational Architecture is to identify and define the main data types that support business development [9], [10]. For example, data (the support of the informational architecture) can be categorized according to different dimensions, including: primitive vs. derived, private vs. public, and historical vs. operational vs. provisional [11].

2.2. Application Architecture

The second architecture level defines the main applications needed for data management and business support [10]. This architecture defines the major functional components of the architecture to guarantee access to the data in acceptable time, format and cost [9]. However, it should not be a definition of the software used to implement the information system. Spewak also proposes a methodology – Enterprise Architecture Planning (EAP) – to define an application architecture from informational and business requirements [9].

More recently, several authors have adapted Zachman’s framework and Spewak’s EAP to better address their needs, including several proposals know as the American Federal Government [12], Joint Technical Architecture [13], and the Treasury Enterprise Architecture Framework [14].

2.3. Technological Architecture

This architecture defines the technologies that provide an environment for application building and deployment. At this level, the major technological concepts are identified, such as technologies to implement applications, inter-process communication, data storage, and so on [9].

At Technological Architecture level, EAB (Enterprise IT Architecture Blueprints) is a reference landmark [15]. Boar confirmed that IT architectures do not have a repeatable, coherent, non-ambiguous and easily perceptible representation [15], proposing a set of blueprints for defining IT architectures in a systematic, coherent and rigorous way. However, all these proposals introduce new notions and icons, not supported by any rules or standards. As a result, potential users are reluctant to adopt these proposals because they are forced to acquire a high-level knowledge and experience before actually defining any IT architecture.

2.4. Comparison with Software Architectures

In the 90’s, software architecture had similar concerns. In particular, there was no consensus in software architecture concepts [16]. As a result, the IEEE formed a taskforce that defined a standard called “Recommended Practice for Architectural Description of Software-Intensive Systems” to provide a conceptual framework for software architecture [6].

Based on this IEEE standard, the Open Group proposed the TOGAF (The Open Group Architectural Framework) framework for ISA design and evaluation [17]. This framework provides not only a methodology for ISA development but also provides a taxonomy, architectural principles and standards for ISA, mostly at the technological level.

In addition, TOGAF proposes a technical reference model that defines a taxonomy for coherent, consistent and hierarchical description of the services provided by the application platform such as data management, network, operating system, transaction processing, and system administration. Finally, TOGAF also presents several architecture qualities that are inherent to the architecture definition, such as performance, availability, usability, adaptability, and portability.

However, the TOGAF framework has several limitations. The most import limitation is that the focus is mainly technological, not addressing either the informational or application architectures. Another limitation resides in the fact that only a set of IT notions and principles are proposed, not concrete modeling blueprints. This makes TOGAF interesting for thinking on ISA from a technological viewpoint but clearly inadequate for modeling ISA in a global and coherent way.

2.5. The CEO Framework

In order to address the issues explained above, the Organizational Engineering Center (or CEO, for short, in Portuguese) proposed the CEO framework [18] for modeling enterprises using a restricted set of business objects. The CEO framework was defined as an UML profile [19] and evolved from recent research [20], [21].

Although the CEO framework cannot be used to define a complete ISA, it presents some interesting extensions to represent dependencies between businesses and systems. The business objects defined in the framework are *goals* for strategy modeling; *processes* for business process modeling, *resources* for business resource modeling, and *blocks* for IS modeling. The CEO framework also ensures consistency, easy of use and provides mechanisms to maintain integrity with the ultimate goal of reducing the “impedance mismatch” between business and IT architectures.

Recently, CEO framework founding concepts at Information System level where investigated and an UML profile for ISA modeling at informational, application and technological levels was proposed [22]. Figure 1 presents the current core concepts of the CEO framework (at ISA level).

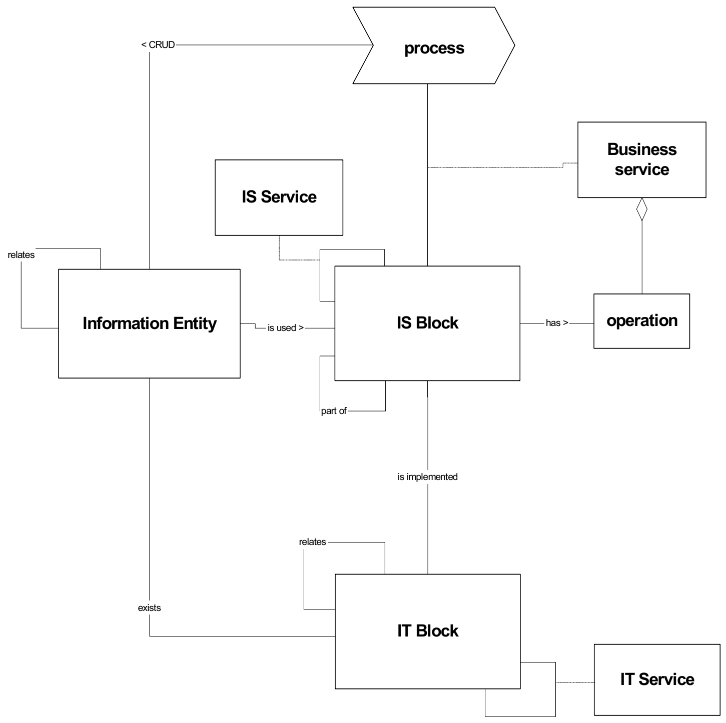

Figure 1. CEO UML Meta-model Extensions for ISA [22]

The core concepts in the CEO framework profile are:

- Business Process – a collection of activities that produces value to a customer;
- Information Entity – any person, place, physical thing or concept that is relevant in the business context and about which is possible and relevant (for the organization) to keep information;
- IS Block – a collection of mechanisms and operations organized in order to manipulate data;
- IT Block –infrastructure, application platform and technological/software component that realizes (or implements) an (or several) IS Block(s).

From a technological point of view the concepts proposed are (represented bellow in Figure 2):

- IT Infrastructure Block – represents the physical and infra-structural concepts: the computational nodes (servers, personal computers, mobile devices and so on) and the non-computational nodes (for example, printers, networks) that support application platforms;
- IT Platform Block – represents the collection of services needed for implementing and IT deploying applications.
- IT Application Block – represents the technological implementation of an IS Block. At this level it is relevant to consider the kind of IT Application Block (namely presentation, logic, data and coordination block), and its “technological principles” (for example, if it is implemented using components, modules, or objects), amongst other characteristics.

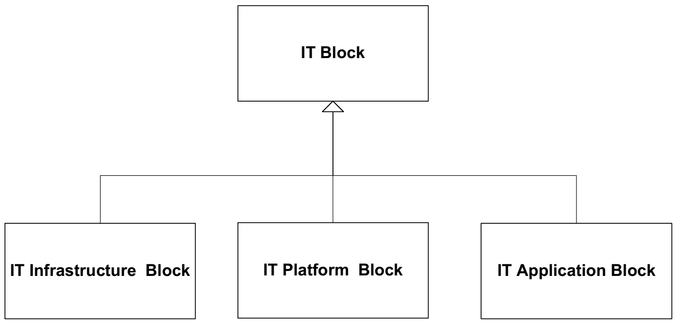

Figure 2. IT Block metamodel

It is interesting to note that in [22] some integration concepts are taken into account. Particular the Service concept is proposed as an aggregation of operations provided by an architectural block, organized in three different categories:

- Business Service – a collection of operations provided by IS Blocks that support one (or several) business process(es);
- IS Service – a set of operations provided by an IS Block (to others IS Blocks);
- IT Service – the technological services provided by application platforms (based on [17] research).

The Service is a core integration concept in ISA and will form the basis for our proposals presented in section 4.

2.6. Conclusion

This overview demonstrates that, currently, there is still no mechanism to properly represent integration concepts in ISA at all (informational, application and technological) levels in order to develop subsequent inspection and/or simulation of different business and technological scenarios. Though some recent approaches (for example CEO framework) provide a starting point for ISA modeling the integration concepts are not considered in detail.

3. Information Systems Integration

Integration – also known as EAI, for Enterprise Application Integration – was always an important part of any information system. The popularity of standard ERP packages in the 90s was supposed to solve the need for integration, but in fact only enlarged the problem; since the ERP cannot replace all operational IS, in particular the operational ISs that run the business, all these remaining ISs have to be integrated with the ERP.

Since most organizations nowadays are more or less satisfied with their IS, the main challenge became to integrate internal ISs with other external ISs. This integration between organizations – sometimes called B2B integration – just extends integration to ISs belonging to other companies, and technically is quite feasible. However, in terms of IS architectures, it becomes even more fuzzy to define the borders of an IS and even the borders for an organization [23].

On the other hand, although there are many kinds of integration, it is important to note that in the end of the day all these are based on exchanging data between two ISs [24]. The differences reside on how this data exchange occurs, what kind of data is exchanged, which guarantees are offered, and so on. For example, the method level exchanges data between two applications while the data level exchanges data between two databases.

Another source of confusion comes from the fact that integration is both a traditional technology but has also become very popular quite recently. In particular, Web Services promise to revolutionize both EAI and B2B even though the technology behind – remote procedure call – is nothing new. In fact, XML is just a data formatting language and solves only a small part of the integration problem. Without transactions, security, and performance, Web Services can be used to integrate applications inside an IS but are clearly not appropriate for integrating IS, and even less for B2B integration [25].

Web Services discussed in a wider context become even more confusing, from an ISA point of view. For example, SOAP [26] – a standard for exchanging XML between two applications – can be considered the most important part of Web Services. Not only SOAP addresses a small part of the integration problem – neglecting security, document types, quality of service, workflow definition, and so on – but also there are still many problems to make SOAP compatible products to work together, e.g. Java and .NET.

Besides that, SOAP is basically an old-fashion, synchronous, non-transactional RPC and will suffer from the problems experienced previously with DCOM and CORBA [27]. And, in our opinion, the main differences in SOAP – basically the adoption of XML and being supported by most vendors – will not be enough to overcome these technological problems.

Fortunately, integration is much more rich and powerful than Web Services advocates want us to believe and offers nowadays a rich variety of dimensions that could and should be represented as part of an IS architecture. Some examples of issues around integration, in no particular order, are:

- Integration can occur at the data, method, interface, portal, and process level – this variety basically represents how the application “sees” integration, although all levels of course exchange data;
- Integration can occur inside a computer, inside an Intranet, inside an Extranet, or on the public Internet – each zone will have its own guarantees of

bandwidth, require different kinds of security, and so on;
- Integration can occur inside a department, inside an enterprise, inside an holding, inside a value chain, or between two (or more) unrelated enterprises – decisions can usually be imposed inside a company, but are more difficult inside an holding and even more difficult (or downright impossible) to impose on another company;
- Integration can occur inside the same country or between countries – for example, digital certificates issued in the USA cannot be used to sign digital invoices in Europe;
- Integration can be synchronous or asynchronous – asynchronous integration has no reply but has higher performance and is scalable;
- Integration can be transactional – guaranteeing that all integration steps take place (or none at all) and extending the transaction concept to the other IS;
- Integration can offer many levels of security, from zero to non-repudiation of reception. These different levels of security should be applied only when necessary (in particular, between companies on the public Internet) since they complicate integration, increase costs and reduce performance;
- Integration can be used to exchange bytes (e.g. TCP/IP), data structures (XML), documents such as orders and invoices (EDIFACT or UBL), workflows (ebXML) or business processes – most integration projects these days are based on XML, but the real problems start when documents are exchanged based on workflows that represent business processes;
- Integration can be performed directly between two ISs (e.g. peer-to-peer) or indirectly via an intermediary (e.g. a message broker) – most asynchronous integration products also use an intermediary to store messages, but only at the implementation level; a broker offers more added value services, such as converting data between two different formats, defining and executing workflows, and so on.

Of course, some of these issues are more important for some levels of IS architectures than other issues:
- The informational architecture defines what kind of data types (high level, such as orders and invoices, not integers and strings) are exchanged between two ISs. Although these days XML seems so important, this level is not interested whether the document is formatted according to EDIFACT or XML.
- The application architecture defines which applications exchange data, what kind of data they exchange and how they exchange that data. For example, exchange can be synchronous or asynchronous, manual or automatic, and so on.
- The technological architecture defines which technologies are used to exchange data, such as XML for formatting data structures, HTTP for communication protocol, and digital certificates for security. This is the level most computer experts are familiar with, but it address only a small part of the integration equation and is only relevant to those writing software.

This paper focus on the application and technological architectures using both existing and novel concepts:
- The existing IS Block and IS Service concepts (proposed in [22]) can be used to represent the operations an information system depend on another.
- The novel IT Integration Block (a specialization of IT Block) and IT Service concepts (proposed in [22]) can be used to represent which applications exchange data and how they exchange data.

4. Modeling Integration in ISA

The previous sections emphasized the inexistence of any praxis, mechanism or language for modeling integration concepts in ISA.

This section proposes an original collection of concepts (including their graphical representations) that allow the semantic manipulation of integration in ISA.

4.1. Integration at IS level

The representation of a concept is critical for its discussion and abstraction. In this paper, in compliance with the CEO framework introduced in Section 2.5, we propose a set of extensions to the UML modeling (standard) language [19] in order to accommodate the new integration concepts.

In fact, the CEO Framework did not define properly the concepts (and corresponding UML stereotypes) for Integration modeling in ISA, although these concepts are crucial in any ISA.

We propose that the «IS Service» concept should be used as the core concept for modeling integration at the IS level because the IS Service already describes how the operations, belonging to an IS Block, are aggregated and made available to other IS Blocks. Although no new stereotype is proposed at the application level, the IS Service is a foundation for modeling integration in ISA at the application level and can be easily extended later if really necessary.

4.2. Integration IT level

The integration process can be divided into three parts (represented bellow in Figure 3): a source (the system that calls the service or sends the message), a target and the integration port itself representing the relation between source and target. At the IT level we propose to split the characteristics associated exclusively to the source, target, and those associated to the relation.

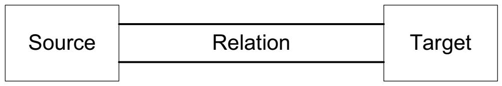

Figure 3. Integration Process

Considering only the characteristics associated exclusively to the system (source or target), the integration can be described along two dimensions:

- Automation Level – the integration services executed in the source or target system are accomplished Automatically (no human interference) or Manually (implies human interaction);
- Role Type – the system may be the source or the target of data. For example, in a web service, the source is the client; in a messaging product, the source is the IS sending the message.

Using the IT concepts presented in Figure 2, we propose that IT integration should be adopted as a novel concept to encapsulate both the platform (e.g., J2EE, .Net, CORBA, etc.) and/or the IT Application. Figure 4 presents our proposed «IT Integration Block» in the scope of Figure 2.

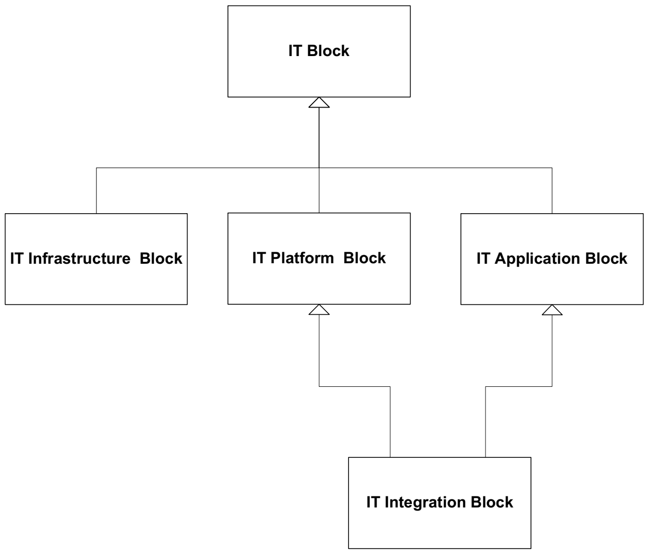

Figure 4. Proposed UML extension for modeling integration concepts in ISA

Figure 5 presents the attributes proposed above for the IT Integration Block UML stereotype (in detail).

« IT Integration Block »

Automation Level: {Automatic,, Manual}
Role Type: {Source, Target}

Figure 5. IT Integration Block in detail

The IT Integration Block is not further specialized to accommodate the large diversity of concepts and the continuously progression in this area. However, depending on the objectives and the target audience, the IT Integration Block can be specialized to model integration specific concepts such as message broker, WebServices, and so on. The case study, presented in the next section, exemplifies these issues.

The IT Integration Service (proposed in [22]) can be used to model the relation port of the integration process as presented in Figure 3. We propose this component be characterized in terms of:

- Technological Level –if integration takes place inside a computer, inside an Intranet, inside an Extranet, or on the public Internet.
- Synchronism Level – integration between IT Blocks may occur synchronously (as in RPC, for example), or asynchronously (usually with no reply, scalable and with higher performance).
- Organizational Level – distinguishes integration between a department, inside an enterprise, inside an holding, inside a value chain, or between two (or more) unrelated organizations.

Figure 6 presents the proposed UML extensions in detail.

«IT Integration Service»

Technological Level: {Computer, Intranet, Extranet, Internet}
Synchronism Level: {Asynchronous, Synchronous}
Organizational Level: {Department, Enterprise, Enterprise Group, Value Chain, General Public}

Figure 6.IT Integration Service

The next section applies all these concepts to a concrete real world example in order to validate these proposals.

5. Case Study: SafeFood

The main objective of the SafeFood project is to create an information system that supports retail company's efforts to guarantee the quality of their food products through the continuous exchange of (almost) real-time data about those products.

The project involves not only a perishable products distribution company but also many other external organizations, mainly suppliers. All these entities already have their own information systems that must be integrated. For example, the Control Quality department is responsible for the products acceptance or rejection. The products storage and their distribution to the stores are performed by the Logistics Department. The Stores are responsible for selling products to the customer. The

Agricultural unit is responsible for contract management with the Producers Organization (named ahead as OP), which commits to delivery the perishable products on the negotiated dates.

In Figure 7 the entire ISA at the application level is presented. The dependencies between the IS Blocks are presented using «IS Service». The arrows mean the dependencies between the IS Blocks. For example, the IS Block "SafeFood System" depends from the service provide by the IS Service "Control Quality API".

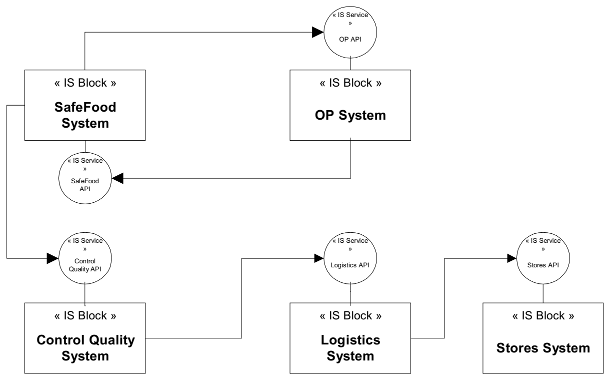
« IS Service » OP API
« IS Block » SafeFood System
« IS Block » OP System
« IS Service » SafeFood API
« IS Service » Control Quality API
« IS Service » Logistics API
« IS Service » Stores API
« IS Block » Control Quality System
« IS Block » Logistics System
« IS Block » Stores System

Figure 7. ISA at the application level

The dependencies between IS Blocks represent points of integration between those systems. In Figure 8 the dependencies between two concrete IS Blocks are represented in detail.

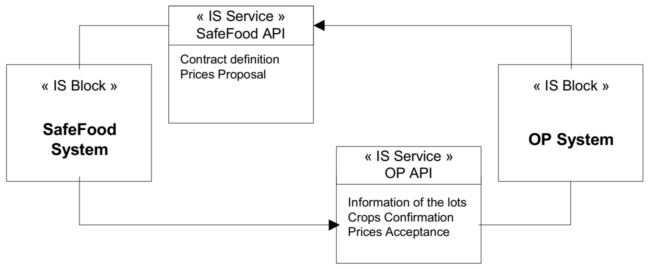

Figure 8. Dependences between SafeFood System and OP System

The IS Block "SafeFood System" could be further decomposed into three information systems (Agricultural Management System, Commercial Management System and Procurement Management System) as presented in Figure 9. Each of these IS Blocks are implement by an IT Block.

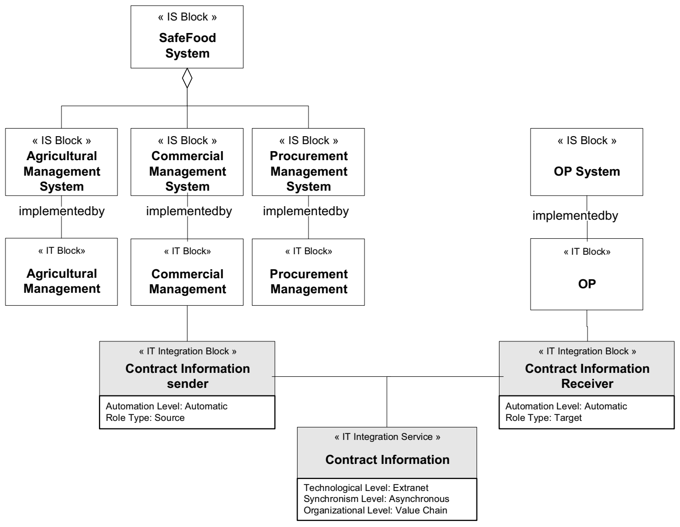
« IS Block » SafeFood System
« IS Block » Agricultural Management System
« IS Block » Commercial Management System
« IS Block » Procurement Management System
« IS Block » OP System
implementedby
implementedby
implementedby
implementedby
« IT Block» Agricultural Management
« IT Block» Commercial Management
« IT Block» Procurement Management
« IT Block» OP
« IT Integration Block » Contract Information sender
Automation Level: Automatic
Role Type: Source
« IT Integration Block » Contract Information Receiver
Automation Level: Automatic
Role Type: Target
« IT Integration Service » Contract Information
Technological Level: Extranet
Synchronism Level: Asynchronous
Organizational Level: Value Chain

Figure 9. Integration between IT Blocks in detail

The integration between the IT Block "Commercial Management" and the IT Block "OP" is performed through two IT Integration Blocks and an IT Integration Service. In this example, the "Contract Information" is a data exchange between two organizations belonging to the same Value Chain. This exchange is asynchronous and takes place inside an extranet (for example, a VPN on the Internet).

The integration between the commercial management IS and the procurement management is described in Figure 10. This integration is accomplished via a intermediary system broker. The IT Integration Block from the IT Block Message Broker corresponds to the Adapters usually used in the System Broker.

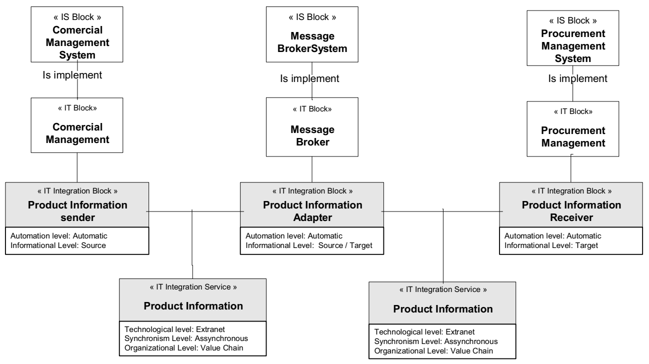
« IS Block » Comercial Management System
« IS Block » Message BrokerSystem
« IS Block » Procurement Management System
Is implement
Is implement
Is implement
« IT Block» Comercial Management
« IT Block» Message Broker
« IT Block» Procurement Management
« IT Integration Block » Product Information sender
Automation level: Automatic
Informational Level: Source
« IT Integration Block » Product Information Adapter
Automation level: Automatic
Informational Level: Source / Target
« IT Integration Block » Product Information Receiver
Automation level: Automatic
Informational Level: Target
« IT Integration Service » Product Information
Technological level: Extranet
Synchronism Level: Assynchronous
Organizational Level: Value Chain
« IT Integration Service » Product Information
Technological level: Extranet
Synchronism Level: Assynchronous
Organizational Level: Value Chain

Figure 10. Integration using a System Broker

In the example the Commercial System uses a Broker to send the products information to the Procurement System. A message broker is an integration intermediary that can be used to exchange messages between other information systems. This case study is particularly interesting not only because message brokers are positioning themselves as alternatives to both ERP and application servers, but also because message brokers play both the role of an information system and of an integration system.

6. Discussion

The case study presented in previous section illustrates possible scenarios where our proposals for representing integration in the ISA are explored. Further, the case study exemplifies how to represent the ISA at informational, application and technological levels, having integration as its main focus.

The proposed extensions to the CEO Framework provide the conceptual tools and visual modeling primitives (supported in a standard modeling language – UML) to model several integration concepts. The major gap between our approach and existing research is on addressing integration not only from a technological perspective but also from an informational and application perspectives based on an organizational framework.

Integration is sometimes only explored from a software-oriented perspective. This is usual the case when using a pure software development approach (as waterfall [28] or RUP [29]), where the focus is on a particular system and on its implementation, not addressing the "big picture" – the relation between the new system and other systems (from an informational, application and technological perspectives). These software approaches are still valid when implementing a system, however they do not provide to the system architect the global perspective for planning and discussing integration concepts in the global ISA (that should precede the implementation of a particular software system).

Other approaches have a more general aim. For instance, Zachman framework [5] provides the conceptual tools for organizing and classifying the concepts that should be addressed when planning the ISA (and latter the enterprise architecture). Zachman framework provides a more general view over the enterprise and the IS, however it does not propose any notation for representation of ISA or integration concepts. Our approach, on the other hand, integrated in an enterprise modeling framework (CEO framework) and defining the notation for integration modeling in ISA (based on a standard modeling language) addresses these issues.

[15] proposes a set of blueprints for modeling information systems at a technological level (as presented in section 2.3), introducing new notions and icons, not supported by any tools or standards. Our approach, as presented, is supported on the universal modeling language (a standard supported in several tools) and addresses integration not only from the technological point of view.

Another import approach in ISA is the TOGAF framework. When comparing TOGAF and our approach one can notice that TOGAF has a different focus – developing other issues, not addressed in our approach as the architecture development method (ADM), but disregarding others. Namely, TOGAF does not address integration issues in ISA at informational or application levels (it focus at IT level); TOGAF also does not concern about the notation used to represent the ISA (it address only the concepts in a ISA).

7. Conclusion

In this paper we first presented an overview of information system architectures and then concentrated on their lack of support to properly represent integration. We then proceeded with a brief introduction to the variety of integration models that exist in the real world, trying to demystify the idea that all integration problems can be solved with Web Services.

The main contribution of this paper is an extension to our previous proposal for representing ISA in order to include a number of integration models at both the application and IT levels. In particular, we proposed that integration should have a number of characteristics (e.g. manual or automatic) and not be limited to synchronous services.

The paper also presented a real-world case study (taken from a project in which we are involved) in order to illustrate the proposal with concrete integration problems between information systems.

In the future we intend to explore other integration concepts at the technological level, in particular how to map the whole variety of integration technologies currently available to a limited number of primitive concepts. We are particularly interested in Web Services and integration across organizations where reliability and security are key issues.

We are also interested to develop the technological tools to model integration, quantify different integration scenarios in information system architecture and help the system architect in defining a system architecture using different integration patterns.

8. References

[1] Kalakota, Ravi and Marcia Robinson, *E-Business 2.0*, Addison-Wesley Longman, Incorporated, 2000.

[2] W3C, World Wide Web Consortium, *Web Services*, 2001. http://www.w3.org/2002/ws.

[3] Garlan, D. et al., *Architectural Mismatch (Why It's Hard to Build Systems Out of Existing Parts)*, Proceedings 17th International Conference on Software Engineering, Seatle, WA, April 23-30 1995, pp.170-185.

[4] Maes, Rik, Daan Rijsenbrij, Onno Truijens, and Hans Goedvolk, *Redefining Business – IT Alignment Through a Unified Framework*, White Paper, May 2000. http://www.cs.vu.nl/~daan/

[5] Zachman, John, *A Framework for Information System Architecture*, IBM System Journal Vol.26 Nº 3, 1987, p.276 – 292.

[6] IEEE Architecture Working Group, *Recommended Practice for Architecture Description – Draft IEEE standard P1471/D4.1*, IEEE, December 1998.

[7] Zijden, Stefan, Hans Goedvolk, and Daan Rijsenbrij, *Architecture: Enabling Business and IT Alignment in Information System Development*, 2000. http://www.cs.vu.nl/~daan/

[8] Sassoon, Urbanisation des systèmes d'information, 1998 (in French).

[9] Spewak, Steven, and Steven Hill, *Enterprise Architecture Planning: Developing a Blueprint for Data, Applications and Technology*, Wiley-QED, ISBN 0-471-599859, 1992.

[10] DeBoever, L., *Enterprise Architecture Boot Camp & Best Practices: A Workshop*, Meta Group, 1997.

[11] Inmon, W. H., *Data Architecture – The Information Paradigm, QED Technical Publishing Group*, 1999.

[12] Federal Enterprise Architecture Framework, version 1.1., September 1999

[13] Department of Defense Joint Technical Architecture, July 2002.

[14] Treasury Enterprise Architecture Framework, July 2002.

[15] Boar, Bernard, *Constructing Blueprints for Enterprise IT Architecture*, John Wiley & Sons, 1999.

[16] *How do You Define Software Architecture?, Software Engineering Institute*, Carnegie Mellon University, December 2000. http://www.sei.cmu.edu/architecture/definitions.html

[17] Open Group, *The Open Group Architectural Framework (TOGAF) – Version 7*, November 2001.

[18] Vasconcelos, A., A. Caetano, J. Neves, P. Sinogas, R. Mendes, and J. Tribolet, *A Framework for Modeling Strategy, Business Processes and Information Systems*, Proceedings of 5th International Enterprise Distributed Object Computing Conference EDOC, Seatle, USA, September 2001.

[19] UML Proposal to the Object Management Group, 1997. http://www.rational.com/uml

[20] T. W. Malone et al., *Tools for inventing organizations: Towards a handbook of organizational processes*, Management Science, March 1999.

[21] Eriksson, Hans-Erik, and Magnus Penker, *Business Modeling with UML: Business Patterns at Work*, John Wiley & Sons, ISBN 0-471-29551-5, 2000.

[22] Vasconcelos, A., P. Sousa, and J. Tribolet, *Information System Architectures: Representation, Planning and Evaluation,* Proceedings of International Conference on Computer, Communication and Control Technologies Orlando, U.S.A., July 2003.

[23] Linthicum, D., *B2B Application Integration*, Addison-Wesley, 2001.

[24] Vernadat, François, *Enterprise Modeling and Integration*, London, Chapman & Hall, 1996.

[25] M. Mira da Silva. *Challenges for EDI Adoption by Small and Medium-size Enterprises (SME)*. Accepted to the IADIS International Conference e-Society, Lisbon, Portugal, 2003.

[26] Newcomer, Eric, *Understanding Web Services: XML, WSDL, SOAP, and UDDI*, Addison Wesley Professional, (ISBN: 0201750813), 2002.

[27] M. Mira da Silva, *Information Systems Integration* (In Portuguese), FCA, 2003.

[28] Boehm, Barry W., *Software Engineering Economics*, Englewood Cliffs, NJ, Prentice Hall 1981.

[29] Jacobson, Ivar, Grady Booch, and James Rumbaugh, *The Unified Software Development Process*, Adisson Wesley, 1999.