34 SUPPORT OF VIRTUAL ENTERPRISES BY AN INTEGRATION INFRASTRUCTURE

Erik van Busschbach, Bram Pieterse, Arian Zwegers
Baan, evbusschbach@baan.nl
THE NETHERLANDS

Enterprises cooperate more extensively with other enterprises in various forms. These collaboration forms need suitable infrastructures and supporting applications to realize the collaboration. This paper presents new ideas and requirements for such an integration infrastructure. The evolution of enterprise business systems is accompanied by an evolution in the integration domain. The former results in current c-commerce applications. The technology and functionality features that an integration infrastructure should offer to enterprises to engage in c-commerce activities are described. An integration infrastructure capability stack is defined that contains a dimension consisting of connectivity, transformation, routing, and process management layers, and a dimension of functional environments.

1. INTRODUCTION

One of the trends in the global market is the increasing cooperation among enterprises during the entire product life cycle. This is related to business drivers, such as the need for cost reduction, flexibility, focus on core competencies, and so on. The resulting collaboration form is anything from a rather stable alliance between partners as in a supply chain to a more transitory cooperation as in a virtual enterprise. The latter can be defined as a temporary alliance of enterprises that come together to share skills or core competencies and resources in order to better respond to business opportunities, and whose co-operation is supported by information and communication technology (ICT) [4]. When the final product and/or service has been delivered, the virtual enterprise dissolves.

Collaborative commerce (or c-commerce) is concerned with the cooperation among enterprises. Both stable cooperation forms, such as supply chains, and more dynamic cooperation forms, such as virtual enterprises, need suitable infrastructures and supporting applications to realize the collaboration. In our view, an essential element to realize c-commerce, is a platform or an integration infrastructure that provides communication and integration services, community management including modeling relationships among enterprises, and support for workflow. On top of the integration infrastructure, applications are needed for the monitoring, management, and optimization of the multi-enterprise collaboration.

The objective of this paper is to present new ideas and requirements for an integration infrastructure that should be able to support the set-up and operation of

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35585-6_68

L. M. Camarinha-Matos (ed.), *Collaborative Business Ecosystems and Virtual Enterprises*

virtual enterprises. The evolution of enterprise business systems and the integration domain are presented, and it is shown that new business requirements demand new integration services in order to enable emerging forms of collaboration.

This paper is organized as follows. The next section gives an overview of the evolution of enterprise business systems, and it discusses the background of and some issues within c-commerce. Section 3 provides an overview of the evolution of integration solutions, and it presents requirements for an integration infrastructure. In section 4, a framework is shown that represents different perspectives upon the services that an integration infrastructure should offer. A discussion concludes this paper.

2. COLLABORATIVE COMMERCE

2.1 Evolution of enterprise business systems

In order to obtain a better picture of new enterprise business systems, the historical development of these systems is outlined below. The reader is referred to (Aerts *et al.*, 2001) for a more detailed historical overview.

The first computation and data storage applications became available in the 1960s in order to support an enterprise hierarchy composed of self-contained functional units under central control. Software applications were introduced for calculations and data storage. For each function, a database plus self-contained applications became available. Computer technology supported computing and data storage, but communications were not affected much; messages were (manually) exchanged between the various functions in the form of paper listings or magnetic tapes.

Because speed – and therefore logistics – became more important, the 1970s witnessed some changes in the organizational structure for enterprises in manufacturing and distribution of physical goods. Special coordinators became responsible for order promising and goods flow control, initially supported by Material Requirements Planning (MRP) applications that were soon followed by Manufacturing Resource Planning (or MRP II) applications.

In the 1980s, enterprises changed their organizations towards business units. Business process re-engineering appeared, which stated that logistics (or in general the value-adding process) should be treated as the leading principle for organizing business, rather than the functional hierarchy. Workflow management systems resulted from this change. MRP II systems evolved into Enterprise Resource Planning (ERP) systems by linking financial consequences to logistical transactions and by extending the core MRP focus on manufacturing to other areas such as distribution, service, HRM, and warehousing. Standard ERP-software delivered standard integration between functions.

In the 1990s, supply chain processes were the focus of bilateral process reengineering efforts; an internal integration into a process-oriented enterprise is followed by an "external" supply chain integration, linking together several supply chain members. Local integration became more and more difficult in the case of multi-site systems. The approach widened from expanding and integrating applications based on a single, homogeneous software architecture, based on more

or less synchronous (instantaneous) communication, to the integration of applications based on different architectures. Communication between these components, often through Electronic Data Interchange (EDI), was necessarily asynchronous (message-based). Packaged enterprise application integration solutions came up offering the middleware to connect different applications.

2.2 Current drivers and inhibitors

Nowadays, three major movements put additional requirements to enterprises: globalization, outsourcing, and customization. Organizations expand their scope to become really global, and differentiate their patterns of cooperation to encompass collaborative activities. Outsourcing and a focus on core competencies requires better collaboration, synchronization of processes, and appropriate handling of time and distance constraints. Customization demands make-to-order manufacturing, better demand visibility, and more flexibility in general in order to execute faster and more efficiently. Closer collaboration with partners is required by globalization, outsourcing, and customization.

However, the trend towards closer collaboration is hindered by a number of factors. Firstly, companies tend to adopt an enterprise-centric perspective. They are preoccupied with their own internal business processes, and pay little attention to inter-enterprise processes. They tend to optimize their own performance, not seldom at the expense of their suppliers. For example, some large car manufacturers have a history of squeezing their suppliers.

Even if companies look outside their own enterprises, they are inclined to optimize the relationships with their closest suppliers and customers in a one-to-one fashion. Symptoms of these paired relationships are excess inventory, long product development cycles, and missed opportunities to serve customers. Supply and demand data is not readily available across the entire value chain, so the inventories of suppliers, manufacturers, and distributors increase as they try to guess one another's actual capacity and demand. Product development is hindered – for instance – since it takes contract manufacturers several weeks to react to an Engineering Change Notice. Finally, inter-enterprise customer service fails because companies cannot integrate their systems with the applications of new partners.

Popular solutions which are available in the market today, such as Supply Chain Management applications, typically address cooperation within paired relationships. Only the cooperation between an enterprise and its closest suppliers or customers are considered. The supplier's supplier and the customer's customer are not taken into consideration. Although the logistics management of an enterprise towards its direct neighbors might be optimized, the overall supply chain is far from optimal.

2.3 Collaborative commerce defined

Despite the problems as described above, it is apparent that enterprises will have to adopt approaches such as "collaborative commerce" (or "c-commerce") to remain competitive in most industry segments (AMR, 2001; Forrester, 2001; Gartner, 2001). Gartner defines c-commerce as follows:

> "C-commerce is the collaborative, electronically enabled business interaction among an enterprise's internal personnel, business partners, and customers

throughout a trading community. This trading community can be an industry, industry segment, supply chain or supply chain segment." (Gartner 2001)

C-commerce should be considered as a business model rather than a solution that can be offered by vendors. It benefits an enterprise by extending the enterprise's visibility and cooperation throughout the value chain, thereby contributing to the realization of virtual enterprises.

Perhaps the most essential element of c-commerce is the extension of an enterprise's knowledge assets to include those outside the enterprise. When intellectual capital is leveraged across enterprises, the benefits of c-commerce can be realized. Sharing intellectual capital and combining core competencies with partners are the major ingredients of collaboration.

C-commerce requires systems that enable enterprises to share information and collaborate in communities of interest. The next generation of ERP systems, sometimes called "ERP II", and the technologies that support it should provide the flexibility to enable the real-time virtual enterprise through the integration of disparate business systems.

2.4 Issues in collaborative commerce

Before an enterprise can leverage c-commerce for external collaboration, it must have the applications, business processes and organizational structures internally which can cope with the demands of external transaction processing and collaborative people-to-people interaction. In the following subsections, we briefly discuss a number of issues that have to be addressed before an enterprise can be a true c-commerce player. Although these issues are quite interrelated, this paper classifies them into three main areas:

- organizational issues, including people-related issues,
- business processes,
- applications and technology.

2.4.1 Organizational issues

Several organizational issues play a role in the move to c-commerce. Amongst others, enterprises have to establish proper relationships with partners, establish trust among partners, and deal with contract management issues.

Unlike a four-wall, enterprise-centric supply chain planning situation, in which the constraints are known and resources can be controlled, successful c-commerce initiatives depend upon the development of collaborative relationships. These collaborative processes are focused on streamlining communication between supply chain partners, while improving trust and breaking down barriers to sharing information. So-called "channel masters" control the extended supply chain. This position defines them as the natural leaders for development of communication standards and collaborative pilots. Enterprises operating as channel masters can either force channel partners into collaborative business processes, squeezing profitability from trading partners, or develop win/win processes. These enterprises need to act as benevolent dictators, with a focus on the efficiency of the entire supply chain, and establishing proper relationships among its partners that benefit all.

The issue of trust is recognized by various authors. For instance, a four-level risk-trust hierarchy was developed by CTP to help enterprises understand and manage the interplay of risk, interdependency, and trust in the networked economy (CTP 2001). Another four-level model of trust by Sabherwal (1999) has been adopted by Van den Berg and Van Lieshout (2001), who discuss trust in e-commerce and present measures to increase it.

Operating in virtual enterprises necessitates enterprises to rethink their way of working regarding contract management. The characteristics of virtual enterprises requires enterprises to change their way of working from that of tightly connected legal entities bound together by volumes of contractual documents to ad hoc casual relationships, still with contractual requirements, but ones which are more definitive of intellectual property protection and valuation. A range of legal issues are beginning to emerge. This is mainly due to the lack of solid contractual basis which govern the electronic exchange of information and documentation within and between such virtual enterprises. Examples of emerging legal issues are proof of receipt of electronic data (such as drawings and emails), ownership of information, intellectual property and access rights, and company versus project information. EC project eLEGAL (IST-1999-20570) addresses these types of issues and aims to define a framework for specifying legal conditions and contracts to enable a legally admissible (exclusive) use of information and communication technology in project business (Hassan *et al.*, 2001). Vendors such as diCarta and I-many already provide commercial contract management solutions.

2.4.2 Business processes

The inter-enterprise business processes in a virtual enterprise need to be defined, and reference models need to be developed. In an enterprise-centric supply chain management situation, the constraints are known and resources are controlled. In a c-commerce environment, however, it is crucial to develop collaborative relationships, among which the definition of collaborative processes. These inter-enterprise processes are focused on streamlining communication between virtual enterprise partners.

Vendors need to supply solutions that support the business processes in specific industries. They need to provide reference models and supporting technology for inter-enterprise business processes per industry and perhaps even per industry segment.

In addition to the definition of inter-enterprise business processes, workflow management applications are required to manage the execution of these processes. However, different enterprises deal with workflows differently. The way an enterprise processes a sales order might be different from the way another enterprise deals with it. Nevertheless, partners in a virtual enterprise share the responsibility for the execution of inter-enterprise business processes. When enterprises distribute a process between multiple enterprises, the resultant workflow logic could be very conditional. Managing workflows within a virtual enterprise will require an event-driven process rather than a synchronized sequencer.

2.4.3 Applications and technology

C-commerce requires new applications and new technology. New applications need to provide real-time information and visibility, and need to have a network-centric focus.

Visibility into plant operations is needed for the execution of inter-enterprise business processes. Without that visibility, the synchronization of production activities in the larger manufacturing value network can result in faulty business processes. For example, to give an adequate 'capable-to-promise' statement, visibility into real-time plant capacity based on actual and planned shop floor utilization is required. Applications need real-time input and processing capabilities. These applications should be capable of real-time event monitoring and handling alerts. Current initiatives in Supply Chain Event Management aim to provide these capabilities.

The current role of ERP systems will change in a c-commerce environment. Current ERP systems act as the back-office transaction processing systems which focus on enterprise optimization in a domain which is usually restricted to manufacturing and distribution. In a c-commerce situation, the role of ERP will be to provide the information in such a way that inter-enterprise business processes can be realized and the enterprise can participate in a virtual enterprise. ERP systems need to be transformed to be able to function as a main source of (real-time) information that can be made available to "partners" such as employees, business units, suppliers, and customers. Other applications, such as collaborative planning, collaborative project management, and collaborative product commerce applications, use the information supplied by the ERP system to manage and optimize collaboration with partners.

An integration infrastructure provides the technology that is needed to expose internal data to the outside world, to enable inter-enterprise business processes, and to connect to legacy applications and systems, such as ERP. Sections 3 and 4 provide a further discussion on the integration infrastructure.

2.5 Collaborative commerce environment

In this section, an overall picture of an envisaged c-commerce environment is shown (see **Figure 1**).

A c-commerce setup as shown in **Figure 1** is typical for an organization that allows other enterprises to use its collaborative applications and infrastructure. It is likely that c-commerce applications will be controlled and maintained by enterprises that allow other enterprises to access these applications and/or that use these applications for the benefit of all partners in a virtual enterprise. In **Figure 1**, this enterprise was a main contractor, though it will more likely be a Collaboration Service Provider (CSP) due to the fact that enterprise will place more trust in an independent third party. Essential is that the channel master or the CSP provides an infrastructure that can be used by other partners for the exchange of information, for the definition of the virtual enterprise itself, and for workflow support. On top of the integration infrastructure, applications are provided for the monitoring, management, and optimization of the multi-enterprise collaboration.

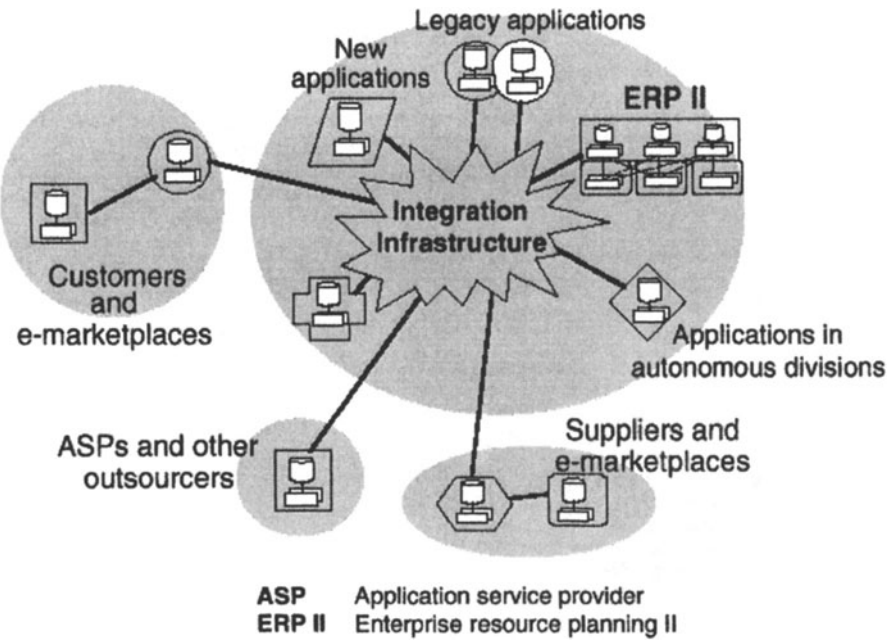

Figure 1 - C-commerce environment (Gartner, 2001)

3. INTEGRATION INFRASTRUCTURE

3.1 Evolution of the integration domain

Roughly every decade brings a new IT paradigm. In section 2.1, it is illustrated that the previous couple of decades saw a new generation of enterprise business systems. The evolution in enterprise business systems is accompanied by an evolution in the enterprise application integration domain (see **Figure 2**).

In the 1970s, most automation happened with homegrown systems, which turned out to be too inflexible to deal with business change and too complex and costly to maintain. These systems represented functional silos, which were integrated with other systems only at a high cost. Integration of these islands of automation was practically non-existent.

The focus on business process re-engineering and the desire to remove the islands of automation led to large-scale adoption of ERP systems in the eighties. ERP systems replaced homegrown systems often at the penalty of loosing functionality, which was custom-made for a specific enterprise. ERP systems were quite monolithic applications, again with only few connections to other systems.

To support particular business functions in more depth with richer and more specific functionality, ERP systems were extended in the 1990s with bolt-on applications, such as customer relationship management (CRM) systems, warehouse management systems (WMS), advanced planning and scheduling (APS)

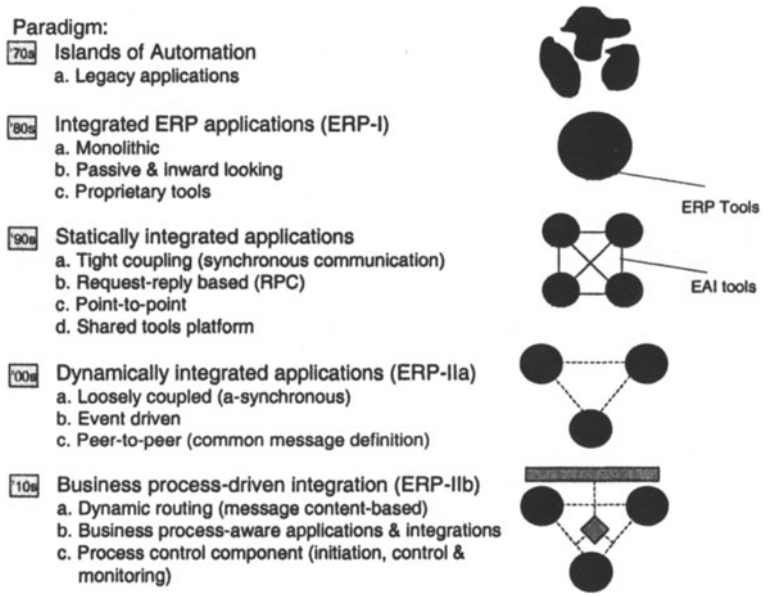

Figure 2 - Evolution of integration

applications, and transport management systems (TMS). Integrations between those applications and ERP systems were either provided out-of-the-box or were custom built, but usually through static point-to-point connections, with or without advanced connectivity tools.

3.2 Current situation

In striving for more flexible and modern solutions and a leaner IT organization, most companies have replaced their homegrown legacy systems with packaged enterprise applications. The next hurdle is to connect disparate systems from different vendors, whilst still retaining the flexibility to change the IT infrastructure according to business needs. Many companies find themselves literally caught in a spider web of systems, technologies and interfaces, which is increasingly hard to manage and maintain, but also incapable of adjusting to the requirements of today's dynamic business environment.

During the last five years, system integration has evolved from batch file transfer and custom-built interfaces to a higher level of sophistication based on middleware products and standards. Application vendors have started to open up their applications through XML and standard, application-level APIs. Middleware vendors have emerged to provide off-the-shelf tools to connect applications. These so-called Enterprise Application Integration (EAI) solutions provide tools for application connectivity, message transport, data mapping, and so on.

Due to inflexibility and high maintenance cost of integrations between ERP systems, bolt-on applications, and other applications, tools emerge to support loose coupling of applications through message-based or data-driven architectures, also termed peer-to-peer architectures. Internet communication models are inherently

peer-to-peer due to high latency, message-based communication and standardization of message definitions (such as RosettaNet and OAG).

Whereas EAI solutions used to focus on intranet environments, the scope of the integration problem has reached beyond the enterprise boundaries. With the rapid introduction of business-to-business electronic commerce using Internet, integration of systems across companies has become a challenge as well. As a result EAI vendors extend their offering to cover B2B integration capabilities and support for emerging communication standards as well.

Next to this, the business will show a need for more dynamic integrations, i.e. a flow of information between applications, which is governed through business rules and conditions. In other words, the flow and destination of information is determined dynamically based on the message content, rather than fixed upfront at configuration level. This requires a router as part of the integration infrastructure. Also a need emerges for a business process-driven approach to integration rather than a systems-level approach. This requires a process control component as part of the integration infrastructure, to initiate processes or activities (events), control processes across applications (e.g. splitting and merging of business objects during the process flow, managing the integrity of long-lived transactions, etc.), and monitoring the state and performance of the process.

3.3 Requirements

There are multiple ways applications can be connected to one another via integration technology. An integration infrastructure should be flexible in providing these various alternatives and leaving the choice to the enterprise and the situation. This subsection provides a background on the consolidated requirements that form a basis for the definition of the integration backbone. **Figure 3** provides an overview of the technology and functional features that an integration infrastructure should provide to an enterprise. The enterprise should determine which features it uses, e.g. synchronous or a-synchronous communication, real-time or batch, and so on.

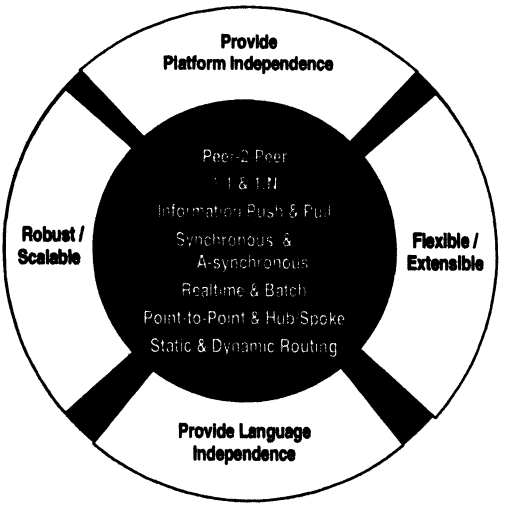

Figure 3 - Integration alternatives providing customer flexibility

3.3.1 Technology Features

An integration infrastructure should offer enterprises four technology features, namely platform independence, language independence, robustness/scalability, and flexibility/extensibility.

Provide Platform Independence
The evolution of enterprise systems has natural dependencies with the operating systems platforms and hardware systems. From an integration perspective, this results in the challenge to link applications that are built on various platforms in a seamless manner. Therefore, one of the technology features of the integration infrastructure is to provide platform independence, hence facilitating cross-platform communication between systems.

Provide Language Independence
Programming languages and software engineering paradigms have evolved in different dimensions over the past 40-50 years. As a result, in a typical integration case examples can be found of each of the stages of this evolution. In order to facilitate communication between this hodgepodge of applications, the integration infrastructure should provide facilities to remove any dependence on any specific language.

Robust / Scalable
The integration infrastructure is basically at the center of all communication between all connected applications. As more and more applications will depend for their successful execution on the contribution of other applications, the integration infrastructure should be reliable. Therefore, features like high (if not 100%) uptime, distribution of risk (also known as removing single points of failure), and extension of load without a drop in performance become more critical to make the system robust and scalable.

Flexible / Extensible
Related to the previous point, but from a more functional angle, is the ability of the integration infrastructure to be ready for changes. Over time, more applications will rely on the integration infrastructure and become network-oriented. Therefore, it is likely that extensions will be needed, both in the area of new applications that need to be added and new services that are built inside and on top of the integration infrastructure. This imposes requirements on flexibility and extensibility.

3.3.2 Functional Features

Various functional features that an integration infrastructure should offer are presented below.

Peer-to-Peer
One of the key characteristics of the integration infrastructure is that it should be able to provide equal services to all connected (enterprise) applications. In turn, this

should allow the applications to treat each other as peers. This is different from integration technology that is purpose-built to provide a generic interface to one enterprise application. Often, and especially in the case of ERP systems, integration technology was built with the assumption that ERP would only act as a server, responding to requests of remotely integrated clients.

1:1 versus 1:N
Applications can be connected through a dedicated link in a 1:1 fashion. However, as multiple applications start using the same infrastructure, they can also more easily link to one another through already available connections with the infrastructure. For example, suppose that applications from one vendor have been linked into an integration infrastructure. If a connector to the integration infrastructure is made for another vendor's application, the latter application can be used by the former applications immediately.

Push versus pull
In a pure event-based environment, a request or update is being pushed out or published by the initiating application. The event can be a remote procedure call (RPC) to a specific server application, or it can be an open publication to which other applications can subscribe. Pulling implies that the integration infrastructure is leading rather than the individual applications. It is usually time-triggered.

Synchronous versus a-synchronous
Applications can be connected through tight coupling or loose coupling. Tight coupling is similar to API-based integration, in which the client calls the server application directly through a stateful connection based on an RPC protocol. Loose coupling equals message-based integration, in which the connection is stateless and the use of messages decouples the client and server application. Tight coupling is inherently synchronous, whereas loose coupling is inherently a-synchronous.

Batch versus real-time
Some application communication is initiated by events, particularly in case the information is mission-critical and allows minimal latency. Pure real-time behavior is usually triggered by the client application, for instance at requests for an Available-to-Promise or sales price. Near real-time behavior usually applies to time-triggered synchronization of data, initiated by the integration infrastructure and based on minimal intervals, which results in continuous polling. The latter is obviously less efficient, but might be enforced by lack of activation or event behavior on behalf of the applications. Batch synchronization of data usually takes place in less mission-critical environments such as daily production planning, weekly financial reporting, or monthly financial budgeting.

Point-to-point versus hub-spoke
A next level of sophistication to the previous subdivisions is point-to-point versus hub-spoke. This addresses the functional level of integration, rather than the technical level. It is therefore a dimension that can be positioned orthogonally to the 1:1 and 1:N dimension. Point-to-point means that the client and server application are fully aware of each other, without an abstraction layer in between. It usually

implies that the client application is modified to suit the server application, and renders the integration proprietary to the combination of the two applications. The hub-spoke model introduces an abstraction layer or common object model that each application can plug into. This way, not only a connection can be reused across integrations, but also the mapping of the application model into the common object model can be reused. The hub-spoke model gives an exponential increase in integration efficiency, and makes integrations much more flexible, since integrations can more easily be added or replaced without impacting the overall environment.

Static versus dynamic routing
In a statically integrated runtime environment, integrations have been compiled or configured in such a way that they have a fixed communication line. Little notion is given to the fact that the business context is usually more dynamic than that. Imagine a multi-site environment in which each site has its own ERP implementation. The company has a single web store front through which customers submit orders. Depending on the items ordered, the geography of the customer and the availability of inventory, the most suitable production or distribution site is being assigned to fulfill the order. Depending on the outcome, the order needs to be routed to a different ERP application (instance). Therefore, a routing component adds dynamic behavior to an integration based on business rules and conditions, which are evaluated against the content or properties of a message. The values of the properties determine the destination of the message and the subsequent process flow. In a publish-subscribe scenario, some basic routing takes place, since applications can selectively subscribe to messages. However, this is best qualified as filtering, since routing is more deterministic in establishing the destination of messages.

4. INTEGRATION INFRASTRUCTRUE CAPABILITY STACK

4.1 Overview

Following the trends and requirements described above, this section defines a framework or capability stack that allows to identify the services an integration infrastructure should offer. The previous sections are concerned with what an integration infrastructure should provide. The framework presented in this section shows how that can be achieved. Compared to most current EAI solutions, content based routing and business process management are added as functional layers. **Figure 4** shows the complete capability stack.

In order to structure the services needed for the complete integration backbone, the capability stack is divided into two dimensions:

- Functional layers
  Four layers, i.e. Connectivity, Transformation, Routing, and Process Management, classify the various services from a functional perspective;
- (Functional) Environment
  Three environments, i.e. Integrated Development Environment, System Management Environment, and Runtime Environment, classify the various services from a life-cycle perspective;

The following subsections discuss both dimensions in more detail.

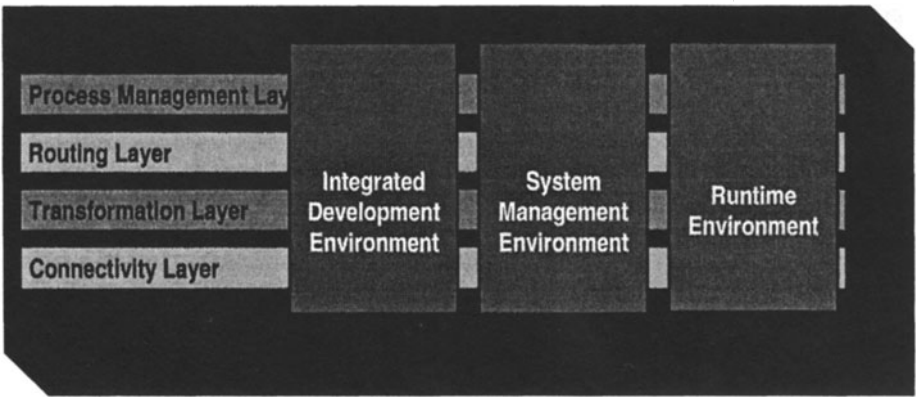

Figure 4 - Integration infrastructure capability stack

4.2 Functional Layers

Connectivity Layer

The bottom layer links applications in different (programming) languages, databases, middleware, (internet) protocols, and other technologies. The goal of the connectivity layer is to provide a transparent interface to collect, manipulate and distribute data and functionality of physically distributed application components.

Most applications have not been designed to interact with other applications. Built-in support for proactive interaction is often limited or not available at all. Therefore, services that extend the basic behavior of the applications, such as event generation services, are part of the connectivity layer as well.

For the message traffic between distributed applications, communication services provide functionality to reconcile low-level network and protocol differences, queuing for decoupled communication, and security. The connectivity layer also provides functionality for message format/envelope (re)formatting.

In the context of the trends described in this document, reusing usable existing components prevails relying on proprietary homegrown technology. In this case, industry strength message oriented middleware technologies are available. These products often provide services such as guaranteed delivery, load balancing, and multiplexing (one-to-many or many-to-one message distribution).

Transformation Layer

Whereas the transparency provided by the connectivity layer is primarily focused on removing technology dependencies, the transformation layer aims to reconcile the differences in the data and functions on a functional level. Services on this level resolve data model differences, data formatting differences, and data value differences.

Functional differences arise since enterprises want to have diverse local solutions, because this better suits their unique local conditions. This causes a tension between the obvious needs for cooperation among organizations (which would call for adoption of some common standards), and the suitability of certain

proprietary solutions that can more readily meet local conditions. This tension is an important factor in interoperability. In other words, even if there are global e-commerce standards sufficient for every business need, there will always be incompatible systems out there – either by choice or because of legacy. The integration infrastructure needs to integrate these systems with external e-commerce interfaces (ECIMF, 2001). Especially the transformation layer should provide the desired interoperability.

Routing Layer

The objective of the routing layer is to provide dynamic behavior that is based on the contents of a message, and that can be configured by a business analyst. Up to the transformation layer, the interaction and information exchange between the different systems has been rather static, even though connections could be changed through changing the configuration settings. However, there are two key disadvantages with this solution. Firstly, this mechanism only works on a connection level; the same configuration applies to all messages. Secondly, this is a task normally performed by a system administrator, not by a business analyst.

A reference to an implementation of content-based addressing is given by Schulz and Platte (2001). They refer to other references (Arnold *et al.*, 1999; Segall *et al.*, 2000), that start from the observation that it is usually the responsibility of the sender to direct a message. This requirement causes difficulty in situations where the sender does not know the destination, when it is constantly changing, or when the number of recipients varies. A common solution is to introduce an explicit agent at a known, fixed address to which the sender always delivers the message. This agent then handles the message distribution. The alternative suggested is to provide a means of content-based addressing, sending simple structured messages and allowing receivers to use a subscription language to select messages of interest (Arnold *et al.*, 1999; Segall *et al.*, 2000). A transparent router process takes over the role of the explicit third party. This indirect communication between a sender and a receiver can add additional flexibility in a dynamic and widely distributed workflow environment.

Process Management Layer

The process management layer adds capabilities allowing the end user to dynamically trigger, execute and monitor business processes. In turn, these processes dictate the flow of information between applications and other data sources. Perhaps the ultimate goal of this layer is to provide inter-enterprise workflow management services that support multiple dynamic workflows crossing organizational boundaries.

Lately, standardization consortia are actively formalizing and standardizing the business process definitions. Two examples are BPMI.org and ebXML (see (BPMI, 2001; ebXML, 2001)). Their respective works – Business Process Modeling Language (BPML) and electronic business XML (ebXML) – address complementary aspects of e-business process management. Whilst BPML is a meta-language for the modeling of business processes, just like XML is a meta-language for the modeling of business data, ebXML provides a standard way to describe the public interface of e-business processes. For this reason, BPMI.org provides a standard way to describe their private implementation.

In addition to services for the monitoring, management, and optimization of inter-enterprise business processes, configuration and set-up tools are needed. Before processes within a virtual enterprise can be executed, the relations between the various partners have to be defined by means of tools for the set-up of virtual enterprises. These so-called extended relationship management (or XRM) services can be configured to cover a whole supply chain or virtual enterprise, possibly by an enterprise's partners themselves, so that a viral effect occurs (Forrester, 2001). Changing configurations of partners in a virtual enterprise necessitate a dynamic configuration of the integration with partners' IT systems, which has to be easily modified.

4.3 Functional Environments

Integrated Development Environment
The modeling and development environment is used to create integration content (such as business object definitions, mapping definitions, routing rules, process models, and so on). In addition, it contains tools for testing and code generation.

System Management Environment
This environment provides all required system management tools to install, configure, license, monitor, and manage integration infrastructure tools and content components.

Runtime Environment
This environment contains all services needed by the integration backbone at runtime. Examples of these services include services for event generation, connection, compression, encryption, transport, transformation, routing, and alerting.

5. DISCUSSION

This paper presents an initial study into the features and capabilities that should be offered by an integration infrastructure in order to support the setup and operation of virtual enterprises. It should be considered as a first step.

A subsequent step would be to elaborate upon the ideas presented. For example, a possible enrichment of the capability stack shown in **Figure 4** is the addition of a "foundation dimension". This third dimension would classify the integration infrastructure services from an architectural perspective. It could contain a base layer, a framework layer, and an integration foundation component layer. The base layer contains the (low-level) tools that the other layers of the integration backbone have in common. Examples are persistence, transaction, multi-language support, security, time, transport, and naming services. The framework layer contains the integration backbone frameworks. Generally, frameworks are designed to provide a rich yet clearly constrained foundation to ease further system development or system configuration. Frameworks can be built out of a compilation of a specific set of base layer components which have been extended for a specific purpose in one or more of the functional layers, the functional environments, or a combination. Common

examples are a compiler framework, a user interface framework, a system management environment framework, and an adapter framework. Finally, the integration foundation component layer contains the specific integration foundation components, which complete the other two foundation layers to form a complete integration service. These components are built as configurable plug-ins for the framework layer.

Furthermore, the ideas presented on the capabilities of an integration infrastructure need to be validated in practice. Current EAI tools provide extensive functionality in the connectivity and transformation layers, but lack proper routing and process management layers. More research is needed to start filling in these layers.

Finally, the integration infrastructure is just one component in an overall c-commerce environment (see also **Figure 1**). Other components, such as applications for collaborative planning, collaborative project management, and collaborative product commerce, need to be designed as well. And last but not least, current backbone systems, mostly ERP systems, have to be transformed in order to provide the information and functionality that is needed in a c-commerce environment.

6. REFERENCES

1. Aerts ATM, Goossenaerts JBM, Hammer DK, Wortmann JC. On the Evolution of Business, Software, and ICT Platform Architectures. Submitted to Information & Management, 2001.
2. AMR Research. Various documents on URL: http://www.amrresearch.com, 2001.
3. Arnold D, et al. "Discourse with disposable computers: How and why you will talk to your tomatoes". In Usenix Workshop on Embedded Systems (ES99), Cambridge Massachusetts, 1999.
4. Berg RJ van den, Lieshout JM van. Finding symbolons for cyberspace: addressing the issue of trust in electronic commerce. Production Planning and Control 2001; 12 (5): 514-24.
5. BPMI. Various documents on URL: http://www.BPMI.org, 2001.
6. CTP, Cambridge Technology Partners. Various documents on URL: http://www.ctp.com, 2001.
7. Camarinha-Matos LM. Trends in Virtual Enterprise Infrastructures. Lisbon: New University of Lisbon, 2000.
8. ebXML. Various documents on URL: http://www.ebXML.org, 2001.
9. ECIMF. Various documents on URL: http://www.ecimf.org, 2001.
10. Forrester. Various documents on URL: http://www.forrester.com, 2001.
11. Gartner. Various documents on URL: http://www.gartner.com, 2001.
12. Hassan TM, Carter C, Hannus M, Hyvärinen J. "eLEGAL: Defining a Framework for Legally Admissible Use of ICT in Virtual Enterprises". In Proceedings of the 7th International Conference on Concurrent Enterprising, Bremen, 2001.
13. Sabherwal R. The role of trust in outsourced IS development projects. Communications of the ACM; 1999; 42 (2): 80-86.
14. Schulz K, Platte KD. On Architectures supporting Interoperability of Enterprise Software. Discussion paper between the members of the EC expert group on Interoperability of Enterprise Software, 2001.
15. Segall B, et al. "Content based routing with elvin4". In Proceedings of AUUG2K, Canberra, Australia, 2000.