Preprint version of an article published as: Business Process Management Journal, Year: 2005, Volume: 11, Issue: 5, Page: 517-531, ISSN: 1463-7154, DOI: 10.1108/14637150510619867 © Emerald Group Publishing Limited http://www.emeraldinsight.com/bpmj.htm

Towards a workflow-based integration architecture for business networking

Diogo M. R. Ferreira
INESC Porto, Portugal

J. J. Pinto Ferreira
Faculty of Engineering, University of Porto, Portugal

Keywords BPM, enterprise integration architectures, workflow technology, e-business

Abstract The Internet is the underlying network infrastructure that will allow enterprises to create new business structures and to reconfigure existing ones. This trend, which is known as business networking, requires enterprises to be able to integrate and coordinate business processes that extend across autonomous business partners. Traditionally, the problem of integration has been addressed by the field of Enterprise Integration, which underlines the importance of Workflow Management as a tool for business process integration. In the context of business networking, most of the principles of Enterprise Integration and Workflow Management are still useful but, definitely, a new kind of architecture is required. This paper proposes a decentralized inter-enterprise workflow solution, paving the way towards an integration architecture that will allow enterprises to automate their business relationships and to develop arbitrary business networks.

1. Introduction

The Internet fosters new business possibilities that will make enterprises reengineer their systems and reorganize themselves. As a globally connected and widely accessible network infrastructure, the Internet provides unprecedented access to potential business partners. In an increasingly connected world, it becomes easier to find business partners that can perform some of the work better, faster and cheaper. On the other hand, enterprises can associate with each other in order to provide products or services that none of them would be able to provide alone.

In the past, industry has improved its productive powers by *dividing labor* among a set of workers (Smith, 1776); now and in the future, as enterprises strive to become more competitive, they will necessarily adopt a strategy based on *dividing competencies* among a set of business partners. The development of appropriate integration architectures will allow enterprises to make effective use of the Internet in order to establish closer relationships with each other, creating new business structures. The act of developing business relationships supported by information technology is referred to as *business networking* (Österle et al., 2001). This means that the potential of the Internet to reshape the way companies conduct business with each other will eventually lead to a business landscape dominated by dynamic business relationships.

This scenario requires enterprises to be able to integrate and coordinate business processes that extend beyond their borders. For this purpose, enterprises need solutions that provide them with flexibility regarding the kind of business processes they can establish with each other, while respecting the autonomy of each business partner. Taking advantage of the Internet as a business platform is not straightforward because a lot of functionality is still missing to support the integration of business processes between enterprises. Up to now, only application integration has been successfully implemented by making use of technologies such as EDI, XML, message-oriented middleware or, more recently, Web Services (Huang and Chung, 2003).

But in order to achieve process integration, enterprises must be provided with process modeling and execution capabilities that will allow them to design and automate inter-organizational workflows (Aalst, 1998a).

In the past, paradigms such as Business Process Reengineering (BPR) and Computer-Integrated Manufacturing (CIM) have devised methodologies and architectures that have allowed enterprises to carry out enterprise-wide reengineering projects focusing on the integration of processes and resources. In the future, similar paradigms will be required to guide enterprises in integrating their processes and resources with those of their business partners. The Internet and its related technologies already provide an underlying infrastructure for enterprises to share and exchange information with each other. Now what is missing is a framework or architecture that combines the concepts, approaches and technologies that will allow enterprises to integrate business processes with each other.

2. Perspectives from Enterprise Integration

During the 80s and 90s, the field of Enterprise Integration (Vernadat, 1996) has been developed several integration architectures. These architectures provide tools and methodologies to break down systems into functions, to understand the relationships between those functions, and to identify generic building blocks that can be used to build any of those functions. Each enterprise integration architecture is a *reference architecture,* which defines a methodology to capture enterprise functions into models, and to devise particular system architectures that will implement those models.

2.1. Open System Architecture for CIM (CIMOSA)

Being the first major research initiative in Enterprise Integration, CIMOSA (AMICE, 1993) was also the first architecture to establish a set of concepts (such as that of *executable models*) and a model-based integration approach that would have an influence on other integration architectures, if not on the whole field of Enterprise Integration, for several reasons.

First, the CIMOSA modeling framework, which comprises the *function view*, the *information view*, the *resource view* and the *organization view*, which is still one of the most comprehensive approaches to the development of enterprise models. Second, the distinction and relationship between the *system life cycle* and the *product life cycle* is fundamental in order to distinguish between supporting the engineering and supporting the operation of the integrated enterprise. Third, CIMOSA immediately recognized the need for an *integration infrastructure*, which provides the mechanisms to integrate heterogeneous enterprise resources. This integration infrastructure comprises several kinds of services – such as naming services, message queuing, location transparency, replication and concurrency – which seem premonitory when compared to distributed middleware architectures available today, such as CORBA. In addition, the CIMOSA integration infrastructure introduced the innovative possibility of executing enterprise models, and it defines a set of business services which are astonishingly similar, both in purpose and operation, to current workflow management systems.

2.2. Generalized Enterprise Reference Architecture and Methodology (GERAM)

Realizing the differences and similarities between major enterprise integration architectures, the IFAC/IFIP Task Force on Architectures for Enterprise Integration has developed a generalized architecture called the Generalized Enterprise Reference Architecture and Methodology (GERAM) (IFIP-IFAC, 1999). GERAM identifies a generic and recursive set of five entity types that play an important role in any enterprise integration architecture. As suggested in table 1, these entities can be given a special meaning in connection with business networking.

Enterprise entity type	**Original purpose**	**Equivalent purpose in a business network**
Strategic Enterprise Management Entity (type 1)	identifies and defines the need for starting an enterprise engineering or integration effort	deals with a set of business goals and identifies the market opportunities that require the linkage of several competencies into a business network
Engineering Implementation Entity (type 2)	performs the integration project defined by an entity of type 1	provides the means for an enterprise to integrate itself with other enterprises
Enterprise Entity (type 3)	represents the new way the enterprise operates after integration	represents the way an enterprise will interact and operate with other enterprises within a business network
Product Entity (type 4)	represents the products or services provided by the enterprise	corresponds to the end products or services produced by a business network
Methodology Entity (type 5)	represents a task-based methodology to be employed by any of the previous entities during its operation	represents a task-based methodology to be employed by each enterprise in order to integrate its business processes with those of its business partners

Table 1. The purpose of GERAM enterprise entities

Each of the GERAM enterprise entities has its own life cycle. The way their life cycles are connected to each other can be regarded as a generalization of the relation between *system life cycle* and the *product life cycle* in CIMOSA. GERAM extends this concept to the relation between the life cycles of the first four entities, with the fifth entity representing the methodology that supports the operation of both the Engineering Implementation Entity (type 2) and Enterprise Entity (type 3).

Now, if the Methodology Entity is based on business process modeling and execution techniques, then enterprises could use workflow management in order to describe and control their interactions. On one hand, workflow management could support the operation of business networks, by coordinating the interaction between an enterprise and its business partners. On the other hand, workflow management could also support the engineering of business networks, by allowing enterprises to describe, through the development of process models, how they will interact with each other. The main advantage of using workflow management as the Methodology Entity is that, should an enterprise change the way it interacts with its business partners, then the process models that describe those interactions could be redesign and released again for execution.

This is somewhat in contrast with some of the current business-to-business frameworks (Shim et al., 2000), such as OBI and RosettaNet, which specify fixed interaction models, which enterprises should comply with. But when a particular interaction model is assumed, the architecture can hardly be employed in a different scenario, hence the enterprise is not allowed to improve, change or redesign the interaction model. In addition, alternative solutions will have to be developed for other scenarios, increasing the number of different – and therefore incompatible – solutions being implemented. There is certainly an interaction model for every B2B scenario, but no particular model should be taken for granted. An integration architecture for business networking must leave room for defining the business processes that the trading partners will execute. But then, supporting the definition and execution of business processes is precisely the purpose of workflow management systems.

2.3. Enterprise Modeling Execution and Integration Services (EMEIS)

The framework of Enterprise Modeling Execution and Integration Services (EMEIS) is another generalized view of enterprise integration architectures, but whereas GERAM organizes enterprise integration concepts, EMEIS focuses on system-level requirements. Basically, EMEIS defines and characterizes a set of system-level services that are required for an enterprise to be able to develop and execute enterprise models. These services are divided according to three types.

Model Development Services (MDS) support the development and analysis of enterprise models. These services provide assistance in model creation, model assessment, model management, and in maintaining a model repository. Model Execution Services (MXS) support the enactment, control and monitoring of enterprise operations according to executable models. MXS services comprise all the functionality required to handle events, to trigger processes, to schedule process execution, to interpret rules, to monitor conditions, to allocate resources, to retrieve information and to present information to resources. Shared and Base IT Services are a set of common services that provide fundamental capabilities to MDS and MXS. These services may include functionality such as naming services, reliability services, security services, encapsulation services and distribution services.

These kinds of services can be mapped directly to the capabilities of workflow management systems. First, each workflow management system provides its own set of Model Development Services to support the development of process models, although most workflow management systems are restricted to model creation and model management. Some workflow management systems are able to carry out model assessment (analysis). Second, each workflow management system provides its own set of Model Execution Services to support process execution. The way MXS are implemented depends on the particular architecture of each workflow management system. Third, workflow management systems must rely on Shared and Base IT Services in order to interact with resources, be it humans or applications. In the case of human resources, workflow management systems typically rely on e-mail, Web-based presentation services, or proprietary client applications; in the case of external application resources, they must rely on middleware integration services such as message-oriented systems.

3. Workflow management in inter-enterprise environments

The study of enterprise integration architectures and business-to-business frameworks shows the importance of workflow management systems as an integration tool. On the other hand, if the ability to describe and control business processes is one of the main issues in business networks, then workflow management systems are precisely the tools that will provide those capabilities. The fundamental difference is that whereas once the business processes extended across departments within a single enterprise, now they extend across enterprise borders and may involve several autonomous entities.

3.1. Autonomy and decentralization

Within a single enterprise, all resources perform their work under the hierarchical control of a single authority. But in an inter-enterprise environment, the execution of a business process that spans across several enterprises requires the coordination of a set of resources that are controlled by different authorities. In other words, the traditional workflow management approach – having a single point of control (the *workflow enactment service*) over the actions of all resources – does not work when each activity is performed by a resource that belongs to a different enterprise.

Each enterprise must be able to integrate its internal business processes with those of its business partners without being forced to comply with any kind a global, network-level business process. Therefore, workflow management must be achieved by linking the business processes running at different enterprises, i.e., by making each enterprise have its own workflow management system and by tying their workflow management systems together. This results in a decentralized architecture, where workflow capabilities are replicated at each node of a business network. Integrating business processes across enterprises then becomes a matter of defining the connections between processes running at different nodes. In general, these connections can be expressed as message exchanges between different workflow management systems.

3.2. The integration infrastructure

Enterprise integration architectures have since long emphasized the importance of having an underlying integration infrastructure that facilitates information exchange between all enterprise resources. Workflow management systems, on the other hand, have been focusing more on process modeling and execution capabilities, and less on the challenge of setting up such an integration infrastructure. Most workflow management systems either rely on a third-party solution (e.g. a database or a messaging system) or devise their own integration mechanisms, which often become a separate system on their own. But as EMEIS suggests, having an appropriate set of integration services is just as important as having modeling and execution services.

It is thus not surprising that the Internet and its related technologies have had such a strong impact on the capabilities of workflow management systems. On one hand, these technologies allow workflow management systems to distribute their functionality across the network. On the other hand, they allow resources to perform their tasks anytime, anywhere. In fact, without such a ubiquitous and easily accessible network it would be extremely difficult and expensive to come up with a common infrastructure which enterprises could use to integrate resources across their borders. But now that such a network infrastructure is available, it can and should be used to devise solutions that support business networking.

Existing technologies – e.g. message-oriented systems (Banavar et al., 1999), B2B frameworks (Shim et al., 2000), agent-based systems (Huhns and Singh, 1998), or even Web Services (W3C, 2004) – display different approaches towards taking advantage of the Internet and its related technologies. Most of these technologies, however, are either highly centralized or rely extensively on centralized services. An exception is agent-based systems using mobile agents, but these solutions have strong requirements and are prone to security issues, which makes it difficult to apply them in real-world business scenarios.

An option that fosters decentralized solutions without imposing very demanding requirements and which is has not been fully explored is peer-to-peer networking (Oram, 2001). By resembling the originally decentralized structure of the Internet, and by allowing network nodes to discover and interact with each other, peer-to-peer networking is a promising technology for the development of an underlying integration infrastructure for business networking.

3.3. Aiming at reusable functionality

Business networking requires an integration architecture that should be fully decentralized in order to safeguard the autonomy of each enterprise. In addition, any integration architecture should exhibit the following general properties (Vernadat, 1996): it should be *open*, it should be *expandable*, it should be *modular*, its components should be *reusable*, and it should cater for *cooperative* solutions. This means that modeling and execution services that are intended to support business networking should be designed with these requirements in mind. Workflow management systems, however, do not usually exhibit all those properties; MENTOR (Wodtke, 1997), for example, which is modular and open, is not sufficiently expandable or reusable because the invocation of additional components must be pre-coded inside the workflow engine.

Furthermore, if each enterprise adopts its own workflow management system, then linking the business processes that run at different enterprises will become an insurmountable task. On the other hand, making all enterprises use the same workflow management system, besides creating the problem of choosing which one to use, may turn out to be impractical since each enterprise must tailor its system to its own needs, so it is likely that no single system satisfies all enterprises. Together with the requirements for an open, expandable, modular and reusable solution, these issues suggest that modeling and execution services should be obtained by assembling a set of reusable components, rather than developing an all-encompassing workflow management system. At the same time, a minimal set of common modeling and execution services should be provided so that the interoperability between different enterprises is guaranteed.

The solution is to provide each enterprise with a common, reusable workflow enactment service, which ensures that the workflow systems at different enterprises will be interoperable, and that each enterprise will still be able to extend its workflow system by plugging additional functionality into it. This reusable workflow enactment service is called the *workflow kernel*, and it is independent of any particular mechanism of invoking resources. The workflow kernel is able to carry out process execution by invoking external objects called *actions*, which in turn interact with local and remote resources over an integration infrastructure, as suggested in figure 1. The integration infrastructure facilitates information exchange between the workflow enactment service and the invoked resources, whether local or remote.

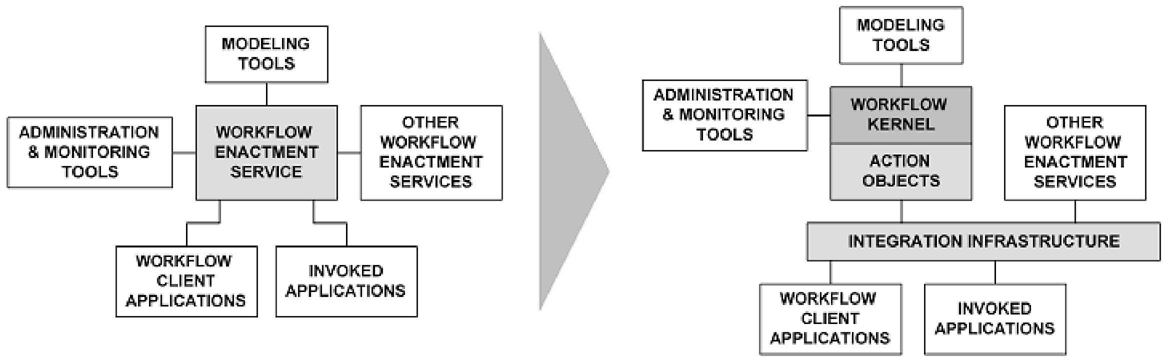

MODELING TOOLS
ADMINISTRATION & MONITORING TOOLS
WORKFLOW ENACTMENT SERVICE
OTHER WORKFLOW ENACTMENT SERVICES
WORKFLOW CLIENT APPLICATIONS
INVOKED APPLICATIONS
MODELING TOOLS
ADMINISTRATION & MONITORING TOOLS
WORKFLOW KERNEL
ACTION OBJECTS
OTHER WORKFLOW ENACTMENT SERVICES
INTEGRATION INFRASTRUCTURE
WORKFLOW CLIENT APPLICATIONS
INVOKED APPLICATIONS

Figure 1. Reworking the Workflow Reference Model

4. Proposed solution

Up to this point, we have seen that any solution that is intended to support process integration within business networks must be devised as an appropriate compound of modeling, execution and integration services. Modeling and execution services are provided by a local workflow enactment service available at each enterprise, allowing each enterprise to define and run its own business processes. Integration services are provided by a common, network-level integration infrastructure, which facilitates information exchange between enterprises and allows them to connect their business processes to one another.

In the previous section, we have seen that the integration infrastructure should be a fully decentralized platform, while the workflow enactment service should be provided as a reusable, extendable core of workflow functionality. In effect, the proposed solution presented in this section is the combination of a reusable workflow engine with a peer-to-peer integration infrastructure.

4.1. The workflow kernel

The workflow kernel is a reusable workflow engine based on Petri nets (David and Alla, 1992). There are several advantages of using Petri nets in workflow management systems (Aalst, 1998b). The main reasons are that (1) Petri nets have formal semantics that allow workflow processes to be described in a clear and precise way, (2) Petri nets have a solid mathematical foundation and several analysis techniques are available, and (3) Petri nets are a vendor-independent formalism. Within the workflow kernel, every process is represented as a Petri net, where each place is associated with a certain *action*, and each transition is associated with a certain *event*, as illustrated in figure 2. Every event is a result of a certain action. Whenever a new token is inserted into a place, the associated action is started. The following transition is associated with an event produced by that action, and it will be triggered as soon as that event occurs. Each action may produce more than one event, which may trigger different transitions, leading to different process paths.

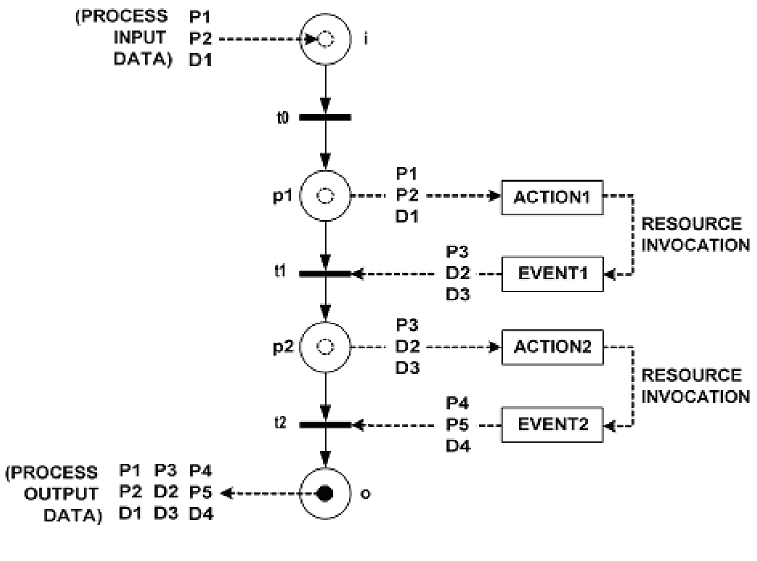


Figure 2. Process execution with the workflow kernel

Each action encapsulates the interaction with a particular resource (for example, a possible action is to add a new task to the worklist of a user). The input data for any action, as well the output data brought back to the workflow kernel by means of events, are represented as set of *properties* and *documents*. A property is a *name-value* pair, where *name* is a string and the *value* can be of any type. The value of one property can be assigned to the value of another; for example, Action2 in figure 2 is configured so that one of its input properties is assigned the value of property P3 before the action is effectively started. A document is a *name-location* pair, where *location* is the fully-qualified path or URL to a file. The location of a document can only be assigned the location of another document with the same name. In general, an action may take any number of properties and documents as input data. When the action is complete (for example, when the user has completed the task), an event may bring any number of output properties and documents back to the workflow kernel.

The workflow kernel exposes a set of interfaces that allow any external component to manipulate workflow kernel objects in order to create and run processes. The “actions” and their “events”, however, are implemented by external components, since they encapsulate the interaction with resources, which is application-specific. In order to allow actions to send events back to the workflow kernel, a special callback interface has been defined, which can receive any kind of object reference as input parameter. In fact, any external object that wants to be notified of any changes within the workflow kernel should implement this same callback interface. For example, if an external object is interested in being notified of any new token in a particular place, it may subscribe to notification events produced by that place. As another example, if an external object is interested in being notified of any new place within a particular process, it may subscribe to any notification events produced within that process. Internally, the workflow kernel also makes use of notification events in order to drive process execution. Every place subscribes from events produced by its associated action, so that whenever such event occurs, the place itself will trigger the appropriate following transition. The transition will then insert a new token into the following places, which in turn will start their associated actions.

The workflow kernel has been implemented based on Microsoft’s Component Object Model (COM) (Microsoft and DEC, 1995), using C++ as the programming language. The main reason for having adopted COM is that it provides several mechanisms (such as reference counting, connection points, and aggregation) that have greatly simplified the implementation of the workflow kernel.

4.2. The peer-to-peer integration infrastructure

The integration infrastructure has been built as a peer-to-peer, service-based platform on top of JXTA (Sun, 2001). In a JXTA network, peers communicate through pipes, a protocol-independent abstraction. Pipes may use TCP/IP sockets, IP multicasting or HTTP to establish a connection between two endpoints on the network, no matter how far apart. A pipe is not necessarily a direct connection, but instead it is usually attained through a sequence of other peers. Special kinds of peers, such as routers and rendezvous peers, allow traffic to bypass firewalls.

Another powerful feature of JXTA is that peers may create and join *peer groups* where special-purpose services may be available. A service can be any functionality that implements behavior on the peer-to-peer network. JXTA services may be used for searching, instant messaging, or any other purpose that peers may find useful. Peer groups may be protected by membership rules that allow only peers with certain credentials to join the group. Inside a peer group, peers implement services that may or may not be available outside that group. Therefore, a peer group is as a protected service space.

In any peer-to-peer infrastructure there are two main kinds of services: *multicast services* and *unicast services*. For example, Gnutella (Clip2, 2001) specifies a distributed search protocol that broadcasts queries across a peer-to-peer network, and it specifies a file download protocol that allows a peer to transfer files from another peer; the distributed search protocol is a multicast service, while the file download protocol is a unicast service. With JXTA there are also multicast and unicast services: the DiscoveryService, for example, is a multicast service because it allows a peer to propagate queries to other peers, whereas the EndpointService is a unicast service because it allows a peer to send a message to a specific remote endpoint address. The PipeService can be used either as a multicast service or as a unicast service, since it allows one-to-many (using propagate pipes) and one-to-one interactions (using unicast pipes).

Both of these kinds of service are actually needed to support the interaction between enterprises. When an enterprise searches the market for a supplier, for example, it could make use of a multicast service in order to disseminate its purchase need. When the enterprise is negotiating with the supplier, however, then it should make use of a unicast service.

The peer-to-peer integration infrastructure has been built as a software layer on top of JXTA; it allows peers to create and make use of multicast and unicast services. A multicast service allows any given peer to propagate queries to other peers, while a unicast service allows two given peers to exchange messages with each other. Multicast services are implemented using propagate pipes, while unicast services are implemented using direct endpoint addressing.

The peer-to-peer integration infrastructure is accessible to each peer by means of a set of simple Java classes, which hide the intricate details of JXTA and make it easier to use multicast and unicast services (i.e., to exchange multicast and unicast messages). Both kinds of messages have the same XML structure, and they allow peers to exchange a set of properties (name-value pairs) and documents (files). Message handling is also greatly simplified by allowing properties and documents to be easily written to or read from an XML message. When a message is being transmitted as an XML stream, all property values are expressed as string values, and all file content is encoded with base64 (IETF, 1996).

4.3. Integrating the workflow kernel with the integration infrastructure

Multicast and unicast services are the integration mechanisms that allow enterprises to interact with each other. Now, if these multicast and unicast services could be invoked from within a workflow process, then it would be possible to model and execute business processes that extend across enterprises. For this purpose, six types of "actions" have been developed:

- *Run Multicast Service* (RMS) and *Run Unicast Service* (RUS) are actions that wait for incoming request messages;
- *Send Multicast Request* (SMR) and *Send Unicast Request* (SUR) are actions that send request messages and wait for reply messages;
- *Send Multiple Unicast Requests* (SMUR) is an action that sends a unicast message to multiple endpoint addresses;
- *Send Service Reply* (SSR) is an action that sends a reply message in response to a previous request message.

Using these workflow actions, enterprises can link their processes and automate their interactions. Figure 3 shows how an enterprise (the "buyer") can search for a business partner (a "supplier") by means of SMR and SSR actions. Internal workflow activities, which may require the use of other services and actions, are represented as darker places. The numbering in figure 3 illustrates a possible process execution sequence. The same approach can be applied to other kinds of interaction (such as negotiation) by means of unicast services.

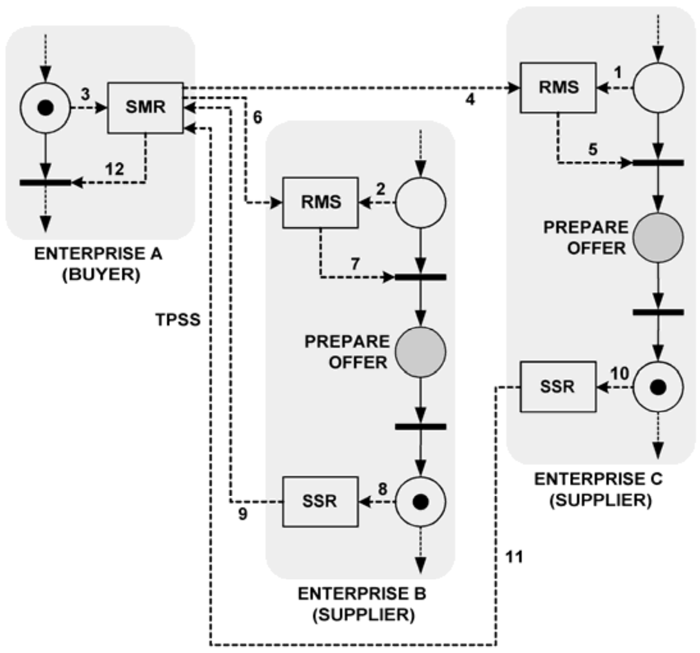


Figure 3. Automating the search for a business partner

Workflow actions have been implemented as a set of objects in a separate software module, which actually creates a bridge between two very different technologies. On one side, there is the workflow kernel and its set of COM interfaces; on the other side, there is the peer-to-peer integration infrastructure, whose Java classes have been exposed as a set of CORBA interfaces. An additional wrapper layer was required in order to translate these CORBA interfaces into a set of equivalent COM interfaces that can be invoked from within COM-based action objects. The TAO ORB (Schmidt et al., 2002) and the Active Template Library (ATL) (Rector and Sells, 1999) have been used for this purpose.

5. Application scenario: a semiconductor supply chain

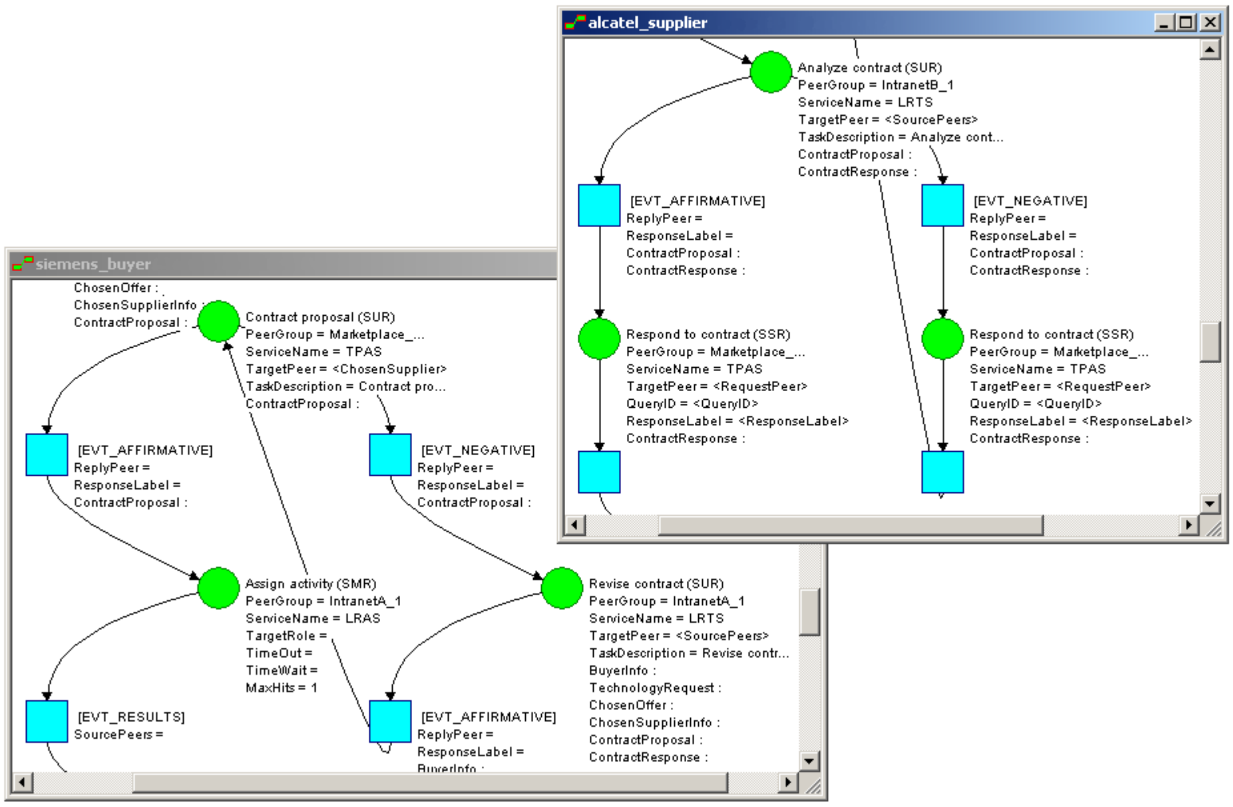
In this section, we will try to describe how the proposed solution could be used in a real-world business scenario. This scenario is based on the CO-OPERATE project (Azevedo et al., 2002) since it involves three industrial partners of that project: Siemens-VDO Automotive, Alcatel Microelectronics (later acquired by STMicroelectronics), and MEMC Electronic Materials. However, whereas the main goal of CO-OPERATE was to develop a supply chain management system focusing on order processing and production planning, here we will explore an alternative scenario where these partners would meet and develop business relationships with one another.

Siemens-VDO is a manufacturer of electronic subsystems for the automotive industry. Most of these components require application-specific integrated circuits (ASICs), which Siemens-VDO orders from Alcatel Microelectronics. In order to produce these and other customized integrated circuits, Alcatel Microelectronics needs silicon wafers, which are supplied by MEMC. Silicon wafers are standard products, which MEMC can supply from its own stock.

For the purpose of our scenario, let us first assume that none of these companies know each other *a priori*, so that they will have to create their business network (which is, in this case, the above described supply chain). Furthermore, let us assume that they will create this business network according to a five-phase life cycle. During the *search phase*, Siemens-VDO will search for a supplier of ASICs. Then, having identified a list of potential suppliers, Siemens-VDO will engage in conversations in order to select the best supplier, which eventually will be Alcatel; this will be called the *selection phase*. In the *contracting phase*, Siemens-VDO and Alcatel will sign a contract or Trading Partner Agreement (TPA), which specifies how the purchase will take place, from initial ordering to final payment. In the *operation phase*, Alcatel will produce the ASICs and send them to Siemens-VDO so that Siemens-VDO can proceed with its own manufacturing process. Finally, in the *evaluation phase*, Siemens-VDO will measure the performance of the supplier so that this information can be taken into account in future partner selections.

This business-to-business (B2B) trading life cycle is basically equivalent to that of (Piccinelli et al., 2001). While Siemens-VDO develops its relationship with Alcatel, Alcatel will develop a relationship with MEMC according to the same kind of procedure. The timing of this relationship is assumed to be such that when Siemens-VDO searches for a supplier of ASICs, Alcatel searches for a supplier of wafers; when Siemens-VDO signs a contract with Alcatel, Alcatel signs a contract with MEMC; and so on.

Figure 4 illustrates how the interactions between these business partners can be modeled and executed with the workflow kernel. In particular, figure 4 shows how the contracting phase can be modeled at Siemens-VDO (bottom left) and at Alcatel (upper right). By means of an SUR action, Siemens-VDO sends a contract proposal to Alcatel, which will receive the contract (via an RUS action not seen in the figure) and then analyze it. The contract proposal may or may not satisfy Alcatel, which will produce an affirmative or negative reply, respectively, to be sent with an SSR action. If the reply is negative, then Alcatel will go back to a state where it waits for another proposal, whereas Siemens-VDO will revise the contract and then re-submit it. If the reply is affirmative, then both processes will proceed to the operation phase.



Analyze contract (SUR)
PeerGroup = IntranetB_1
ServiceName = LRTS
TargetPeer = <SourcePeers>
TaskDescription = Analyze cont...
ContractProposal :
ContractResponse :

[EVT_AFFIRMATIVE]
ReplyPeer =
ResponseLabel =
ContractProposal :
ContractResponse :

[EVT_NEGATIVE]
ReplyPeer =
ResponseLabel =
ContractProposal :
ContractResponse :

Respond to contract (SSR)
PeerGroup = Marketplace_...
ServiceName = TPAS
TargetPeer = <RequestPeer>
QueryID = <QueryID>
ResponseLabel = <ResponseLabel>
ContractResponse :

Respond to contract (SSR)
PeerGroup = Marketplace_...
ServiceName = TPAS
TargetPeer = <RequestPeer>
QueryID = <QueryID>
ResponseLabel = <ResponseLabel>
ContractResponse :

ChosenOffer :
ChosenSupplierInfo :
ContractProposal :

Contract proposal (SUR)
PeerGroup = Marketplace_...
ServiceName = TPAS
TargetPeer = <ChosenSupplier>
TaskDescription = Contract pro...
ContractProposal :

[EVT_AFFIRMATIVE]
ReplyPeer =
ResponseLabel =
ContractProposal :

[EVT_NEGATIVE]
ReplyPeer =
ResponseLabel =
ContractProposal :

Assign activity (SMR)
PeerGroup = IntranetA_1
ServiceName = LRAS
TargetRole =
TimeOut =
TimeWait =
MaxHits = 1

Revise contract (SUR)
PeerGroup = IntranetA_1
ServiceName = LRTS
TargetPeer = <SourcePeers>
TaskDescription = Revise contr...
BuyerInfo :
TechnologyRequest :
ChosenOffer :
ChosenSupplierInfo :
ContractProposal :
ContractResponse :

[EVT_RESULTS]
SourcePeers =

[EVT_AFFIRMATIVE]
ReplyPeer =
ResponseLabel =
BuyerInfo :

Figure 4. Automating the contracting phase

At the level of the integration infrastructure, Siemens-VDO, Alcatel and others are peers in a peer-to-peer network that provides unicast and multicast services, allowing them to communicate with each other. These services are invoked by actions such as SUR, RUS and SSR in the previous example. On the other hand, it should be noted that some activities – namely "analyze contract" at Alcatel and "revise contract" at Siemens-VDO – are performed locally within a given enterprise. The interaction with local resources is achieved by means of the same actions, but with another set of multicast and unicast services. These services can be deployed on a separate, private JXTA network, where each enterprise resource is a separate peer. These will be called "local services", as opposed to the previous ones, called "remote services", which are used between enterprises.

Local services are especially useful during the operation phase, when several resources must be invoked in order to carry out a manufacturing process. In fact, each company – Siemens-VDO, Alcatel or MEMC – has its own manufacturing process, defined as sub-process within the operation phase of the five-phase life cycle, as shown in figure 5. Throughout their manufacturing processes, Siemens-VDO, Alcatel and MEMC can synchronize with each other by means of the same actions and remote services as before – it is just the local activities performed at each enterprise that differ.

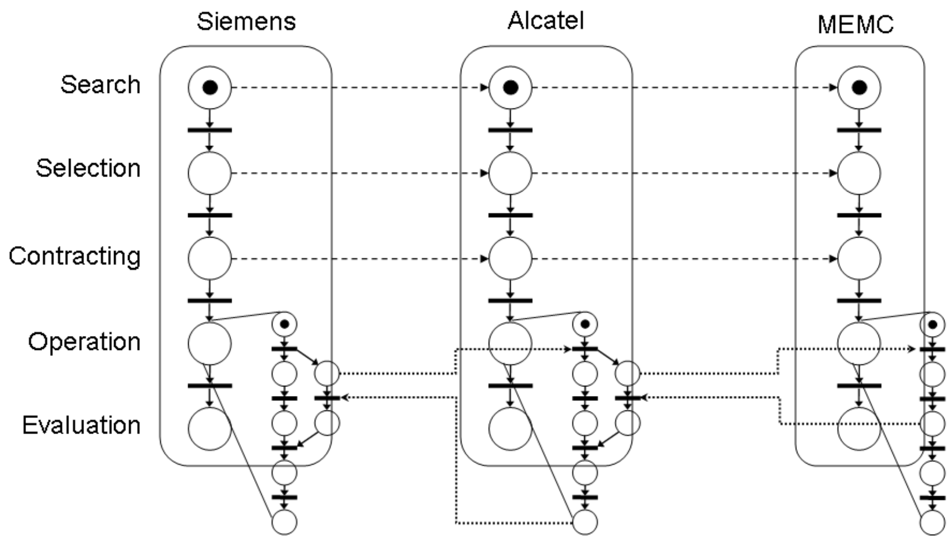


Figure 5. Proposed B2B trading life cycle

6. Conclusion

Business networking brings, among other challenges, the problem of integrating business processes between autonomous enterprises, and this problem requires new integration architectures. Perspectives from Enterprise Integration allow us to identify the key elements of any integration architecture: (1) an integration infrastructure, (2) modeling and execution services, and (3) the life-cycle of each entity. But, in addition, the nature of business networking is such that each enterprise must have the freedom to implement its own business processes in any desired way and to establish business relationships with any available business partner and according to any business structure.

To meet these requirements, we have devised a decentralized, workflow-based solution that requires minimum coupling between enterprises: by means of a common set of peer-to-peer services, enterprises can discover and exchange messages with each other without having to expose any application programming interface; and by means of appropriate workflow mechanisms, enterprises are able to link and synchronize their business processes, while keeping their internal processes private. The resulting architecture allows enterprises to automate the full life cycle of their business relationships, and hence to develop arbitrary business networks.

Acknowledgment

The authors highly appreciate the support of the Foundation for Science and Technology (http://www.fct.mct.pt), which has funded this research. Special thanks to César Toscano at INESC Porto for having provided valuable insights into the CO-OPERATE project.

References

Aalst, W. van der (1998), "Interorganizational Workflows", in Jacucci, G. (Ed.), Proceedings of the Tenth International IFIP WG 5.2/5.3 Conference (PROLAMAT 98), Trento, Italy

Aalst, W. van der (1998), “The Application of Petri Nets to Workflow Management”, Journal of Circuits Systems and Computers, vol. 8, no. 1, pp. 21-66

AMICE ESPRIT Consortium (1993), “CIMOSA: Open System Architecture for CIM”, 2nd Ed., Springer-Verlag

Azevedo, A. , Toscano, C., Bastos, J., "An Intelligent Agent-Based Order Planning for Dynamic Networked Enterprises", Enterprise Information Systems III, Kluwer Academic Publishers, pp.124-131, 2002.

Banavar, G., Chandra, T., Strom, R., Sturman, D. (1999), “A Case for Message Oriented Middleware”, Lecture Notes in Computer Science, 1693, Springer

Cerami, E. (2002), “Web Service Essentials”, O’Reilly & Associates

Clip2 Distributed Search Services (2001), “The Gnutella Protocol Specification v0.4” (available online at http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf)

David, R., Alla, H. (1992), “Petri Nets and Grafcet: Tools for modelling discrete event systems”, Prentice-Hall

Y. Huang, J.-Y. Chung, “A Web services-based framework for business integration solutions”, Electronic Commerce Research and Applications, vol. 2, no. 1, 2003

Huhns, M., Singh, M. (Eds.) (1998), “Readings in Agents”, Morgan Kaufmann

IETF (1996), “Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies”, RFC 2045, http://www.ietf.org/rfc/rfc2045.txt

IFIP-IFAC Task Force (1999), “GERAM: Generalised Enterprise Reference Architecture and Methodology”, Version 1.6.3, http://www.cit.gu.edu.au/~bernus/taskforce/geram/versions/geram1-6-3/v1.6.3.html

Microsoft, DEC (1995), “The Component Object Model Specification”, Version 0.9, http://www.microsoft.com/com/resources/comdocs.asp

Oram, A. (Ed.) (2001), “Peer-to-Peer: Harnessing the Power of Disruptive Technologies”, O’Reilly & Associates

Österle, H., Fleisch, E., Alt, R. (2001), “Business Networking: Shaping Collaboration Between Enterprises”, Springer

Piccinelli, G., Stefanelli, C., Morciniec, M., Casassa-Mont, M. (2001), “Policy-based Management for E-Service Delivery”, HP OpenView University Association (HP-OVUA) 8th Annual Workshop, Berlin, June 24-27

Rector, B., Sells, C. (1999), “ATL Internals”, Addison-Wesley

Schmidt, D., Natarajan, B., Gokhale, A., Wang, N., Gill, C. (2002), "TAO: A Pattern-Oriented Object Request Broker for Distributed Real-time and Embedded Systems", IEEE Distributed Systems Online, vol. 3, no. 2

Shim, S., Pendyala, V., Sundaram, M., Gao, J. (2000), “Business-to-Business E-Commerce Frameworks”, IEEE Computer, vol.33, no.10

Smith, A. (1776), “An Inquiry into the Nature and Causes of the Wealth of Nations” (1998 edition by Oxford World Classics)

Sun Microsystems Inc. (2001), "Project JXTA", http://www.jxta.org/

Vernadat, F. (1996), “Enterprise Modelling and Integration: Principles and Applications”, Chapman & Hall

W3C (2004), “Web Services Architecture”, Web Services Architecture Working Group, Note 11 (available online at http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/)

Wodtke, D. (1997), “Modellbildung und Architektur von verteilten Workflow-Management-Systemen”, Dissertationen zu Datenbanken und Informationssystemen, vol.31, Infix/AKA