19 A FRAMEWORK FOR DISTRIBUTED INFORMATION MANAGEMENT IN THE VIRTUAL ENTERPRISE: THE VEGA PROJECT

Alain Zarli*, Patrice Poyet
CSTB - Centre Scientifique et Technique du Bâtiment, France

Networking technology, object-oriented distributed systems, workflow mechanisms, Product Data Technology and the WEB are now the obvious foundations for complex information management and concurrent engineering in the distributed virtual enterprises to come. The outcome aims at promoting standard, flexible and extensible IT infrastructures, with solutions for potential scalability, robustness and universal platform neutrality in operating heterogeneous enterprise environments, and seamless access to multiple corporate data. This paper provides an overall picture of the objectives, technological research and achieved results of the ESPRIT project VEGA[i], supplying the architectural basements of open distributed frameworks to deal with new IT systems challenges.

INTRODUCTION: CONTEXT AND OBJECTIVES

In order to support today's Large Scale Engineering (LSE) projects, the VEGA project aims to establish a 3-Tier IT infrastructure which supports the technical and business operations of Virtual Enterprises (VEs), in compliance with product data technology standardization activities and current specifications coming from the OMG (*Object Management Group*) and the ISO STEP[ii], and extending their capabilities for engineering collaboration in a flexible distributed environment. These LSE projects involve companies in remote sites to group together on short term business relationships and location independence, for the realization of complex products, with the need to communicate effectively, and to quickly and accurately store, access, manage and transfer any kind of (complex) information.

* CSTB - Centre Scientifique et Technique du Bâtiment, 290, route des Lucioles, B.P. 209, 06904 Sophia Antipolis, France, tel :(33) 4 93 95 67 36, e-mail: zarli@cstb.fr

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: 10.1007/978-0-387-35577-1_37

L. M. Camarinha-Matos et al. (eds.), *Infrastructures for Virtual Enterprises*

Several topics have to be targeted in order to provide a reliable and relevant solution for future distributed IT-based information systems, some of them are:

- Different software applications within the VE must be able to integrate and inter-operate with project/product information.
- Information must remain consistent over the distributed environment, and concurrent access must be controlled in order to avoid data inconsistencies.
- Business is moving faster: enterprises must be able to promptly modify their information systems, and this is likely only if relying on standardized and flexible IT foundations, with solutions for potential scalability, robustness and platform neutrality in operating heterogeneous enterprise environments.

The VEGA platform is actually managing four different technologies:

- product-data modeling for the specification of useful project information models;
- middleware technology for the distribution of project information;
- workflow management for the control of the flow of information and work in the VE, relying on workflow technology as defined by the WfMC (*Workflow Management Coalition*) for design of process control;
- the WEB and its set of associated de facto standards (HTML, VRML, etc.), dealing with new paths to world-wide information communication and distribution, and WEB-oriented intuitive access to information by end users.

Moreover, VEGA delivers a solution for a tighter integration of STEP, CORBA and WEB technologies within a DIS[iii], thus providing both the support of distributed and interoperable client/server (C/S) information systems (through CORBA) and the support of WEB based access to information and services through an Internet based navigation, building upon CGI and Java technologies.

VEGA is a significant contribution to the emergence of appropriate technical solutions supporting the advent of distributed object architectures and value-added distributed information services as a means to implement VEs. The VEGA infrastructure is well adapted to many-to-many relationships across companies applications within the VE, with more efficient exchange and access to information for companies collaborating together, but at the same time enabling those companies to protect (at least parts of) their information systems through workflow-based control of the VE global process. This paper reports on the main components and related developments undertaken in VEGA, for data distribution, sharing, exchange and remote access to information relying on standardized PDT such as STEP, EXPRESS and APIs (e.g. SDAI, COAST), the global VE process mastery through workflow-based devices, and the proved complementary of WEB and middleware technology. It also elaborates on potential enhancements to VEGA for the full deployment of strategic applications, especially for support of distributed transactions, and concludes with the VEGA platform as being a potential basement for future integration and distribution of business components leading to improved support of enterprise business processes, as promoted for instance in another Esprit Project WONDA[iv]. WONDA has specified a scalable 3-Tier component-based architecture, for the deployment of concurrent business applications in Intra/Extranet environments, with secure and sophisticated access to versatile information systems and electronic content (for more information about WONDA, one can refer to (Buckley 1998), (Richaud 1998) and (Zarli 1998a)).

CONCEPTS, TECHNOLOGIES AND COMPONENTS WITHIN THE VEGA PLATFORM

VEGA relies on major concepts and underlying technologies, which have been the basis for the development of a set of components that constitute the VEGA platform.

Main concepts

Modern global information infrastructures now have no other choice than to integrate various major concepts which have fully emerged in the '90s:

- object oriented (OO) development, encapsulating structure, data and functions (methods) within a same boundary (an object), and inducing an easier way to deal with administration rules on information (appliance, invariants, etc.),
- distributed architectures, based on component middleware approaches, like CORBA, COM/DCOM and Java/RMI,
- A friendly dialog with the end user, through appealing unified "look-and-feel" and standardized interfaces and mechanisms, essentially thanks to the WEB,
- process management, especially related to workflow mechanisms.

It is also worth noticing that needs for persistent storage of information of course remain, that now can be realized through dedicated OO databases, even if relational databases are still largely integrated in corporate information systems. Eventually, the development and deployment of all the infrastructure components (including client browsers, application servers, middleware level, databases, and business logic components) are nowadays quite complex, but can be facilitated by IDEs (Integrated Development Environments), assisting the application developer with graphical handling and (at least partial) code generation. Except this last item, VEGA manages all the different concepts introduced below.

Technologies and VEGA components

The VEGA technologies and infrastructure have been already presented in previous papers: (Amar 1997), (Zarli 1997), (Zarli 1998b), (Stephens 1998) among others. So we shortly summarize in this section the main technologies tackled in VEGA, along with the main results achieved so far.

Technologies and related standards

VEGA first refers to some research efforts related to effective standardization of modeling methodologies and languages for PDM and data exchange, in order to enable software applications to inter-operate on the basis of an underlying common semantics for data. Data sharing can then be realized either through standardized format for data exchange, or standard functional API to access data, or a common communication protocol and framework between the applications. Within VEGA, the following standards have been put in practice:

- STEP [(ISO 1994a), (Fowler 1995)], an ISO Standard for the uniform representation and exchange of product data during the whole life-cycle of the product, especially through the EXPRESS language (ISO 1994b), a format for

STEP physical files (ISO 1994c) and an API for common access and sharing of product databases (ISO 1995) via data and application independent mechanisms.

- The IFC (*Industry Foundation Classes*), developed by the IAI (*International Alliance for Interoperability*) as a universal model for integration purposes and collaborative work in the AEC/FM industry.

Regarding distributed networked infrastructures, VEGA promotes a CORBA-based backbone. CORBA [(OMG 1998a), (Mowbray 1997), (Pope 1998)] is an OMG specification for application interoperability in C/S distributed architectures, allowing objects described in any language to be shared across different operating systems and platforms, in a heterogeneous network environment. All CORBA compliant applications are coupled to an ORB (*Object Request Broker*), being the middleware in charge of establishing the client-server relationships between objects, and seamlessly interconnecting multiple object systems.

Eventually, VEGA integrates a set of widespread WEB formats and technologies within a DIS: HTTP, for navigating between hyper-linked documents across the Internet, HTML , the WEB mark-up language used to create hypertext documents, VRML (*Virtual Reality Modeling Language*), a modeling language to specify interactive 3D objects and worlds and intended to be a universal interchange format for 3D visualization through the WEB, and of course Java, an OO architecture-neutral and platform-independent language, along with large classes libraries and APIs for interfacing with existing technologies (databases, middleware, etc.), and a full execution environment for deploying distributed applications.

The VEGA platform components

The COAST communication middleware

The VEGA platform fundamentally relies on the COAST (*COrba Access to STEP information storage*) integration system. Built on top of CORBA, it grants transparent access to distributed (product model) data specified in EXPRESS schemata. The COAST is designed as a general platform supporting the sharing of product model information in a distributed and heterogeneous environment, provided that any model be defined using an EXPRESS schema. Based on an ORB, it supplies a set of services that are partially standard OMG services and partially native COAST services. It also exposes a COAST API (Köthe 1998), a true OO access method supporting by default distributed heterogeneous environments, and hiding to COAST-compliant applications all details about distribution, heterogeneity and storage schema details. Moreover, COAST activities are to be transactional, supporting safe information sharing.

Based on the COAST, VEGA aims to provide a distributed model driven platform, with the promise that different applications can exchange specific types of objects without dependency on the user's knowledge of how the data is handled in the other applications. An important issue in the development of COAST has been its dissemination to the OMG. The COAST specification and implementation have strongly contributed to the OMG "PDM Enabler specification" industrial standard (OMG 1998b) on product management systems, as developed in the OMG Manufacturing Domain Taskforce.

Workflow components for process control in the VEGA platform

Workflow systems bridge the gap between processes and data modeling worlds (Gawlick 1994). As described in (Zarli 1997) and (Schulz 1998), VEGA develops:

- First, a workflow process meta-model to define workflows and to link product model data to the workflow definition, that meets the specific requirements of concurrent engineering in LSE VEs. Its core is based on a generic meta-model of workflow processes specified by the WfMC, which supports the Workflow Process Definition Language - WPDL – (WfMC 1994), (WfMC 1996)).
- On the other hand, a workflow management architecture to manage workflows across company boundaries, realized through a Distributed Workflow Service (DWS), along with a specification for the integration of invoked applications (Schulz 1998). Today, companies using workflow systems most of the time require to harmonize workflow processes with their customers and suppliers while still keeping their own internal business processes hidden. The DWS addresses these requirements by using a two-fold workflow approach, where a distinction is being made between those parts of a workflow that are public to the partners of the VE and those that are private workflows and must be kept secure.

The VEGA DWS realizes both WfMS interoperability and global workflow monitoring across companies, through a CORBA-based integration architecture. The DWS implementation is based on the workflow management software tool PrM (*Process Manager*) as a workflow backbone responsible to handle the global VE workflow, thus acting as a complementary technology for the VE partner's individual workflows which can dynamically evolve at runtime. PrM is used to launch activities that are individual applications or activities for another workflow system in the VE to carry out: for instance, an application for Energy Calculations has been integrated in the LinkWorks[v] workflow system, in turn linked with PrM. A global workflow monitoring is achieved thanks to a (logically) centralised database collecting all runtime-information about workflows. An additional global workflow monitor can access the database to show and analyze the information.

The VEGA component for schema interoperability

Besides applications interoperability, interoperability between the various models involved in a product design process is needed. VEGA developed a Schema Interoperability Service (SIS - (Rangnes 1997)), especially for the interoperability of STEP product models, and in the future for standardized models coming from other initiatives as well. Supported by the EPM EXPRESS Data Manager (EDM), the SIS provides STEP physical file I/O, a conformance checking service and a schema mapping service based on the STEP EXPRESS-X language. The SIS has already been used by the project to develop various models conversions, e.g. a set of mapping rules that convert IFC1.5 data to AP225 (see next section).

The VEGA Distributed information Service

In VEGA, the DIS is a first approach towards an end-user oriented service managing various information representations with standardized front-end services. The WEB offers standards and technologies for public or private networks, with unified user interfaces and a universal way to manage presentations of multimedia information. As one of the DIS objectives is to access distributed information through COAST middleware and present it at end-user desktops (Zarli 1998b), the DIS accommodates the WEB (presentation layer) and CORBA (OO communication layer). Actually, three DIS applications (that are examples of applications connected to the VEGA framework) have been developed:

- A Web-oriented application for compliance checking of EXPRESS based product data, using IFC1.5. Rules have been developed that (partially) check a building design against rules ensuring proper access to disabled persons.
- A Web server for VRML-based visualization of geometrical (AP225) STEP data.
- A Web server for the visualization of the structure of IFC-based product data.

To assist these applications, two more developments have been undertaken in the VEGA DIS. The first is a late "Java Binding" on top of the SDAI/C binding of the SIS (and the augmented EDM interface). This library can be imported and called directly from any Java program to access the EDM database. The other is a "Java Generator", written in Java on top of the previous Java Binding, that generates for any EXPRESS schema (available as meta-data in the SIS) a set of Java interfaces and classes directly used within Java applications. The next section reveals the VEGA distributed infrastructure with a full insight in one of the DIS applications.

THE DISTRIBUTED VEGA ARCHITECTURE

A 3-Tier distributed architecture

The VEGA platform relies on a 3-Tier (sometimes called tierless) infrastructure, expanding the classical C/S (2-Tier) model, where only the client user interface is separated from the combined application and data server. As shown in Figure 1, a 3-Tier model suggests as well a clear separation between the application server itself, and the persistent storage of the corporate raw data. Extended with WEB technologies, one can summarize as follows the main benefits of what we can call the WEB-oriented application server model:

- "Wedding" of Inter/Intranet and C/S technologies, thus summing the well-known advantages of C/S systems (application- and server-centric architectures), with the openness of the WEB (with WEB browsers and servers: unified information access, dialog control, user-friendly GUI, etc.)
- Linking HTTP world and back-end enterprise services (that are to be secure, transactional, and so on). When dealing with CORBA, the intermediate layer offers object-level heterogeneous application interoperability, integration of legacy systems, and standardized high level services.
- Presentation on the (thin) client side in a WEB standardized way, and potential code shipping (i.e. the ability to move code from the server towards the client).
- Application logic and the data behind enterprise corporate firewalls.

Application servers are the promise to provide in a near future the industry with the appropriate architecture design to do scalable client/server computing with Web protocols. Thus, the combination of (CORBA) middleware and WEB technologies has a potential for offering extended and generic solutions for a future dominant computing model for enterprises, promoting a new generation of such systems associating WEB services, object-based functionality (CORBA services), and distributed N-tier enterprise application services and transactions. From an "LSE project" point of view, such a 3-Tier model offers quite attractive characteristics:

- The corporate data are residing (and protected if required) in a database first tier, and can be accessed by any application remotely, thus independently of the database location. Moreover, considering CORBA communication mechanism

between application and database servers, data can be distributed over several heterogeneous databases.

- Different companies in the VE might have their own application server, getting raw data from various distributed databases, and then processing them in order to serve value-added information to any remote client desktop. This middle-tier is targeted to play a major role, ensuring data protection (through selective data access) and more process reliability (through transactional mechanisms), and providing the final end user with information that has been adapted to his needs.

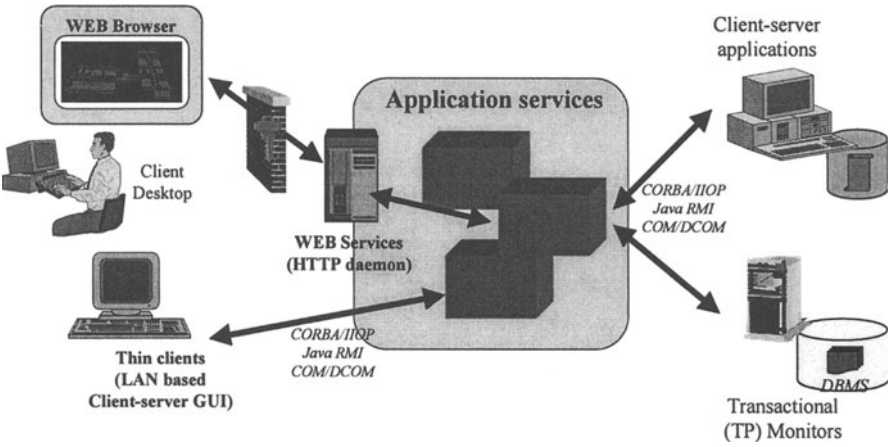

Figure 1 – The 3-Tier application server model.

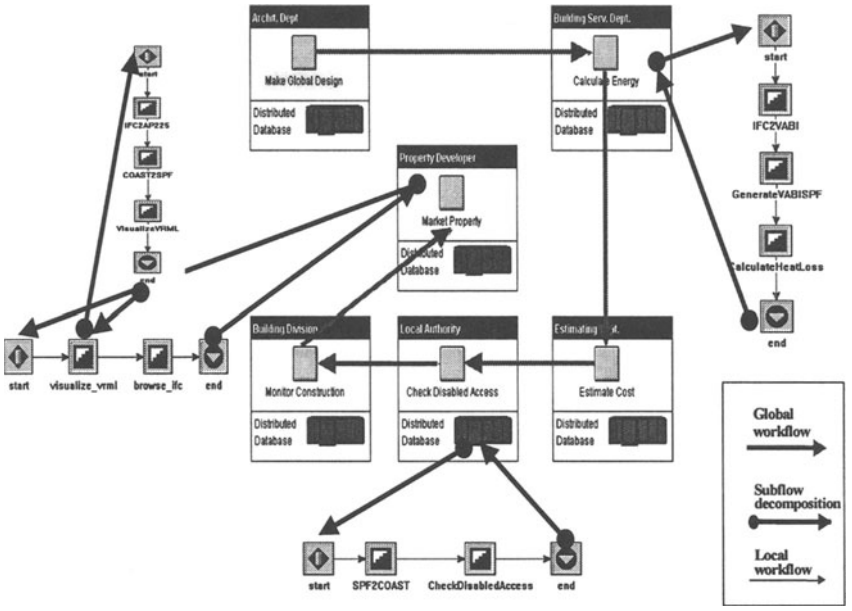

Figure 2 – Global and local workflows in the VEGA demonstration.

Integration of a DIS application in the VEGA platform

In the framework of the VEGA platform and demonstration development, several applications have been connected to the common services of VEGA. A business scenario was defined that led to the workflow definition as illustrated in Figure 2. During the runtime of a workflow instance, users, applications, global VE services, work item manager, and workflow management systems collaborate with each other.

We hereunder give an insight on one of the DIS applications (detailed in (Zarli 1998b)), in the Figure 3. This application checks the conformity of a building design against the regulation to ensure access to disabled persons. In VEGA, the building has been stored into a COAST database. At the level of the application server, the code checking the rules is written in C++ and uses the COAST C client API to access data. This application is used as a CGI script (WEB server-side application).

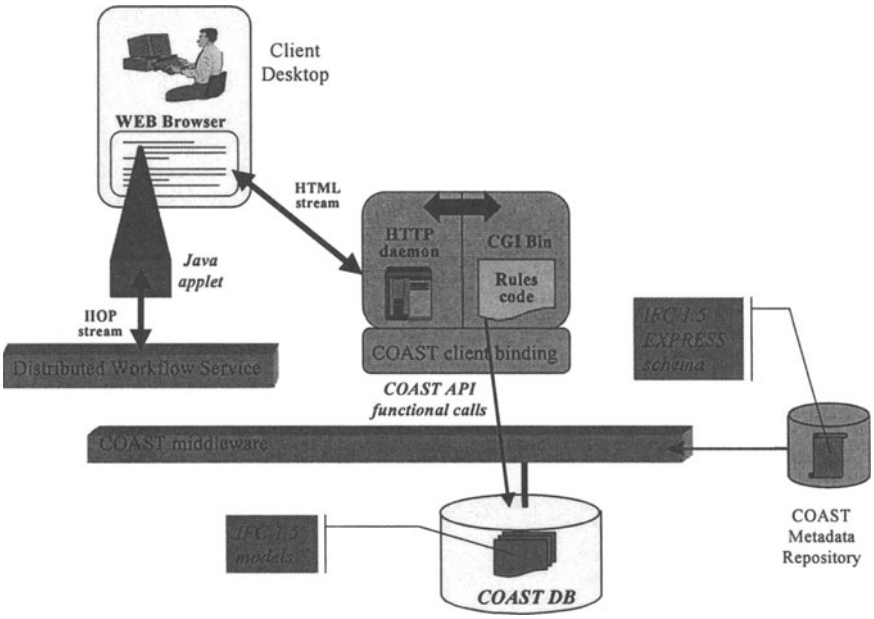

Figure 3 – The "Check Disable Access" VEGA DIS application.

The connection to the DWS is done from inside the user's WEB browser's interface thanks to a Java applet. This Java applet is a DWS interface that allows the user to connect himself to the DWS, retrieve tasks and acknowledge the DWS of the performed task results. As shown in Figure 3, this applet uses IIOP protocol to contact the DWS over LANs or WANs: as the DWS relies on a CORBA-based backbone, the applet integrates a CORBA client runtime environment, enabling it to deal with CORBA/IIOP communication for accessing the DWS services through the network. The main steps of the execution scenario of this application are:

- The user launches his WEB browser and opens the "check access" HTML entry page from the WEB server. On reception of this page, the user then has to fill the Java applet form to identify himself to the DWS in order to ask for a possible task to be done.

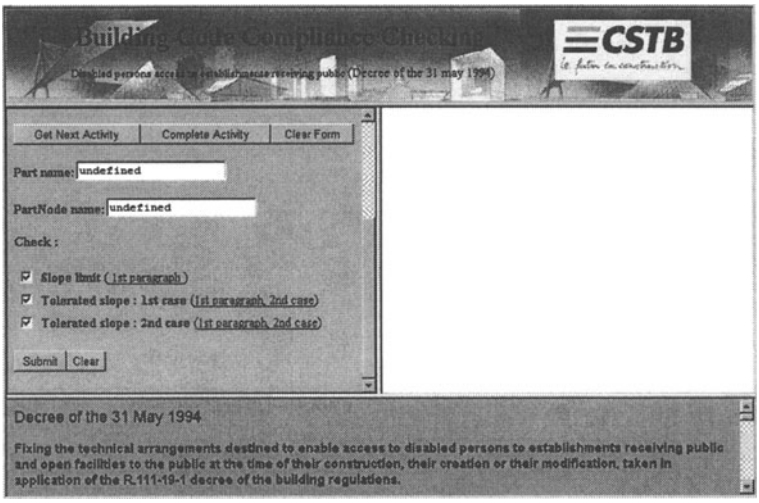

Figure 4 – Web interface

- When the applet receives a task to be executed, the HTML form, the interface to the CGI script gets updated with the corresponding COAST part and partnode names which contain the building model to check.
- Then the user chooses the rules he wants to be checked on the building. By pressing the "Submit" button, the user sends an CGI request to the WEB server. The WEB server then launches the application with the appropriate parameters.
- The application now connects to the COAST and triggers the rules selected on the corresponding entities. If the building contains errors according to the regulations, anomalies and special information on the element that caused the failure of the rule are stored in an HTML report then passed back to the user.
- The user can then inform the DWS that his task has been performed and can ask for the previous Workflow steps to be redone in case anomalies are detected or can permit the workflow's normal continuation.

DISTRIBUTED TRANSACTIONS

Need for reliable transactional mechanisms

The VEGA platform appears to be a sound candidate for the implementation of future distributed standardized information management systems and application servers, with an underlying CORBA based backbone, combined to WEB compliant technologies and standards that will assess the Internet and Intranet. But in order to manage in the right way all the numerous operations between the manifold applications and databases, such a complex system requires to manipulate the data so that they remain at each time consistent and safe, and that actions on those data can be controlled at any time. There is consequently an obvious need for the integration of distributed transactions mechanisms within VEGA-like middleware platforms. Those mechanisms will have to ensure that any operation is safely modifying data, and if not that they can be rolled-back so that the whole system keeps a consistent state, and of course in the context of a distributed environment

where the various data involved in a single transaction can originate from multiple databases.

Practically, and considering for instance some design process within the VE, the major hurdle to overcome with concurrent engineering is what happens when separate engineers are performing a number of editions/modifications on the same data (or set of data) at the same time. Any system, and especially if distributed, must provide some reliable way to control the execution of such concurrent processes, possibly warning the users that some conflict has arisen, and providing some resolution mechanism. In fact, the transaction paradigm is the computer equivalent of some contract between all the actors involved in a given process. If nothing goes wrong, the contract only implies an additional overhead, whereas if something unexpected happens, it describes how to roll back to the state prior to the contract execution, in order not to deal with inconsistencies and wrong data later on. The next section is a general presentation about the major concepts and mechanisms related to (distributed) transactions.

Distributed transactions: main concepts and usefulness

As already mentioned, the concept of transaction is now widely accepted as a key paradigm to constructing reliable compound operational applications, especially those that require concurrent access to shared data. Initially defined in the universe of mainframes, then used for client/server applications, this concept has been extended to the broader context of distributed computation. In an environment with distributed objects, a transaction can be defined as a (indivisible) working unit composed of a set of operations executed on objects that are potentially distributed across distant database servers. It is worth noticing that one of the main current wave of efforts at the moment, in the field of application servers, is the integration of ORBs with TPMs (Transaction Processing Monitors):

- The ORB (as in CORBA) is a software bus for the integration of OO distributed and heterogeneous applications, making totally transparent where the data are located and the communication between distant objects and applications.
- A TPM is a software that realizes the management of large volume of transactions through multiple databases, ensuring data integrity and security of transactions, load balancing, fault tolerance, executing client requests in parallel when possible and performing automatically multithreading management, etc.. TPMs have been largely developed and used in C/S architectures.

The result of such an integration is commonly called an Object Transaction Manager (OTM), achieving the union of data distribution through software bus (CORBA, DCOM) and data consistency and integrity (along with a set of services, particularly applications uploading) guaranteed by TPMs. The future VEs, and the EDI and electronic commerce as well, will more and more generate a need for object middleware able to deal with transactions in a secure way on Internet or in Intra/Extranets. Among the main advantages of OTMs or similar systems are:

- They enable to benefit within distributed systems of the capabilities of TP monitors, with adequate communication modes (synchronous or asynchronous).
- They promote a simplified API (through a unique global interface for various demands) for the client applications that enable them to access to grouped set of services without the necessary use of a lot of APIs and gateways.

- They enrich the intermediate (middleware) level to let it sustain huge loads in terms of transactions, specially for the e-commerce applications to come.

When dealing with transactions within CORBA-based frameworks, a natural choice for implementing distributed transactions is the OTS (*Object Transaction Service*). The OTS[vi], promoted by the OMG, is a specification for the implementation of transactional mechanisms in (CORBA based) OO distributed systems. Generally speaking, the OTS can be seen as a redefinition of the X/Open DTP (*Distributed Transaction Processing*) with some few additional features. In order to ensure openness to heterogeneous (DB) systems, the OTS specifies interfaces between an OTS implementation (managing transactional objects) and both relational servers (through X/Open standard) and object servers (through ODMG standardization).

Need for Concurrency control

Transactions ensure that processes are accomplished in a safe way regarding data status. This is however not enough in a full concurrent environment. For instance, considering an engineer having selected some information from the database, the system must know what the user has selected and retrieved, and there must be an option to have the information selection logged. This informs other parties that they can't access (or at least modify) the information, and also prevents information to be modified by various parties at the same time. Additionally, this leads to means for a user to be informed if the information that was selected is modified at a later date.

Still in CORBA-based frameworks, the Concurrency Control Service[vii] (CCS) defined by the OMG enables multiple clients to co-ordinate their access to shared resources. Coordinating access to a resource means that when multiple, concurrent clients access a single resource, any conflicting client actions are reconciled so that the resource remains in a consistent state. The following items are worth noticing:

- The CCS does not define what a resource is. It is up to the clients of the CCS to define resources and properly identify potentially conflicting uses of resources. In a typical use, an object would be a resource, and its implementation would use the CCS to co-ordinate concurrent access to the object by multiple clients.
- The CCS does not define what a transaction is. This is left to the OTS, and the CCS should be used in conjunction with the OTS. Hence, the CCS co-ordinates concurrent use of a resource using locks. A lock represents the ability of a specific client to access a specific resource in a particular way. Each lock is associated with a single resource and a single client. Co-ordination is achieved by preventing multiple clients from simultaneously possessing locks for the same resource if the activities of those clients might conflict. To achieve co-ordination, a client must obtain an appropriate lock before accessing a shared resource.

The VEGA approach

Designed to be a framework for safe access to distributed (STEP) information, the VEGA platform and its COAST backbone aim at offering transactional support. Indeed, transactions are requested even in non distributed environments, to ensure data integrity (especially through Atomicity and Coherency of the ACID[viii] properties), e.g. any STEP compliant platform that fully complies with the

ISO/STEP SDAI specification should be designed so as to support transactions, as suggested by the standard. Regarding the COAST, the API (Köthe 1998) introduces as well a set of functions for transactions handling, that any COAST implementation must fulfil according to various ways of implementation. Moreover, concurrency[ix], most of the time coupled with transaction issues, is an obvious requirement in a multi-user environment. With respect to transaction concerns, the VEGA project has identified several COAST conformance levels for COAST implementations (level 0 is a level of reference, and corresponds to the SDAI specification, which was the state-of-the-art for STEP implementations before the COAST specification):

- COAST Level 1: remote access to STEP information (one COAST server only).
- COAST Level 2: level 1 + COAST server transactions management.
- COAST Level 3: level 1 or 2 + heterogeneous COAST servers.
- COAST Level 4: level 3 + distributed transactions management.
- COAST Level 5: level 4 + all CORBA 2.2 services (including POA[x] and inter CORBA/COAST platforms communication).

In a distributed environment such as the VEGA/COAST platform, transactional functionality might be implemented, in a first stage, on top of the transactional features provided with the underlying DBMS[xi]. This improvement helps to reach the conformance class level 2 in the COAST, with COAST transactions relying on database-level transaction mechanisms. When tackling conformance class level 4, transactions have to manage access to objects in multiple databases, and must rely on middleware-level transaction mechanisms, based on the integration of the COAST backbone with some TP monitors or on some OTS implementation.

CONCLUSION: MOVING TOWARDS A GLOBAL DISTRIBUTED ARCHITECTURE FOR BUSINESS-ORIENTED IT SYSTEMS

New needs have emerged in today working practices: ability to be easily linked and work together, location independence and short-term relationships, organizing the enterprise around projects[xii], need for information handling, management and maintenance, and support for more complex business processes, etc. all requiring new IT infrastructures such as VEGA which :

- promotes an architecture for more efficient exchange and access to standardized (STEP) information between applications within a company.
- offers a solution for Extranets and VEs as well, where companies have to collaborate together but also have to protect their systems, e.g. through firewalls. When connecting their applications and servers to a VEGA-like platform, firms still use their firewalls as usual thanks to a control of the information access and flow based on a DWS. Moreover, the collaborative network is controlled by the set of companies in the VE (and not a specific company), linking trading partners together in inter-organizational networks with seamless communication and cross-application information circulation. Thus, the VEGA framework provides a first level of security, though it needs extensions with respect to authentication and access control, or information integrity (e.g. audit, non repudiation, etc.).
- is well adapted to many-to-many relationships within the VE, and promotes push technology, where the result of an application is pushed to the next phase of the overall process, via the use of (CORBA-based) workflow technology.

- deals with the adjunction of a transactional layer in order to provide key functionality for the reliability and safety of (distributed) strategic applications, attesting consistency of shared data states through concurrent client accesses.

VEGA is suitable to support future extensions (components, vertical services, etc.), so as to deal with improved business processes and easier control in product development, for the industrial deployment of information and communication infrastructures. The introduction of specific models for the design of enterprise business domain and activities, as encouraged in WONDA through the concept of business objects[xiii] (BOs), should lead to the delivery of standard general service components on top of the VEGA platform. This corresponds to a move from a distributed (low-level) object technology, as managed in VEGA, to a distributed business component technology. Introduced so as to be the "glue" between client applications and enterprise data, BOs are expected to give high-level views on product data both throughout the lifecycle of products (time consideration) and for the various actors involved in the design, development and use of products (domain consideration). For the designer of the global information system, they are expected to be standard, flexible, reusable and interoperable components, to be assembled into frameworks for the development of industrial software component-based applications. Components are on the way to cause an explosion in the vertical (sectorial) markets because they could govern higher-level business functions, but in any case, they will require functions such as those offered by the VEGA platform.

Acknowledgments

The authors thank all the partners for their collaboration within VEGA, especially Jeff Stephens (Taylor Woodrow), Manfred Köthe and Karsten Schulz (Digital/Compaq), Robert Los, Michel Böhms and Helga van de Belt (TNO), Jorulv Rangnes and Hans Karsten Dahl (EPM), Rasso Steinmann (Nemetscheck), Richard Junge (CAB), and their CSTB colleagues Philippe Debras, Jean-Luc Monceyron and Mathieu Marache. A deep acknowledgement is given as well to Dr. Filos, the VEGA EC project officer, and the reviewers, Dr. Brian Hall and Prof. Rainer Anderl.

REFERENCES

1. (Amar 1997) V. Amar, M. Koethe, K. Schultz, A. Zarli, "An open STEP-based distributed infrastructure: the COAST Platform", Proceedings of the 1st International Conference on Concurrent Engineering in Construction '97, July 3-4 1997, London (UK), p. 227-240.
2. (Buckley 1998) E. Buckley, A. Zarli, C. Reynolds, O. Richaud, "Business Objects in Construct IT", Proceedings of the 2nd European Conference on Product and Process Modelling (ECPPM) 1998, october 19-21, Watford (UK), p. 117-130.
3. (Fowler 1995) J. Fowler, "STEP for Data Management, Exchange and Sharing", Technology Appraisals 1995.
4. (Gawlick 1994) D. Gawlick, M. Hsu, R. Obermarck, "Strategic Issues in Workflow Systems", 1994, Digital Equipment Corporation: Palo Alto, California, USA.
5. (ISO 1994a) Industrial automation systems and integration - Product data representation and exchange Part 1. Overview and fundamental principles. 1994. N° ISO/IS 10303-1
6. (ISO 1994b) Industrial automation systems and integration - Product data representation and exchange Part 11. Description methods: the EXPRESS language reference manual. 1994. N° ISO/IS 10303-11
7. (ISO 1994c) Industrial automation systems and integration - Product data representation and exchange Part 21. Implementation methods: Clear text encoding of the exchange structure. 1994. N° ISO/IS 10303-21

8. (ISO 1995) Industrial automation systems and integration - Product data representation and exchange Part 22. Standard Data Access Interface. 1995. N° ISO/DIS 10303-22
9. (Köthe 1998) M. Köthe, K. Schulz, A. Bernotat, "COAST Architecture - The CORBA Access to STEP Information Storage Architecture and Specification", ESPRIT 20408 – VEGA D301, 1997 – Issued Revision 1.8.10: October 1998.
10. (Mowbray 1997) T. J. Mowbray, W. A. Ruh, "Inside CORBA: Distributed Object Standards and Applications", Addison Wesley, 1997.
11. (OMG 1998a) The Common Object Request Broker Architecture (CORBA) specification, Revision 2.2, March 1998, URL http://www.omg.org/techprocess/meetings/schedule/Technology_Adoptions.html
12. (OMG 1998b) The Product Data Management (PDM) Enabler specification, July 1998, URL http://www.omg.org/techprocess/meetings/schedule/Technology_Adoptions.html
13. (Pope 1998) A. Pope, "The CORBA Reference Guide: Understanding the Common Object Request Broker Architecture", Addison Wesley, 1998.
14. (Rangnes 1997) H.K. Dahl, S. Korsveien, J. Rangnes, "Interoperability between product models in the Virtual Enterprise: the VEGA platform", Proceedings of the 4th International Conference on Concurrent Enterprising (ICE'97), october 8-10 1997, Nottingham (UK), p. 81-88.
15. (Richaud 1998) O. Richaud, A. Zarli, "WONDA: An Architecture For Business Objects in the Virtual Enterprise", Position paper - Proceedings of the OOPSLA 98 - Interdisciplinary workshop on Objects, Components and the Virtual Enterprise, october 18-22, Vancouver (Canada), 6p.
16. (Schulz 1998) K. Schulz, M. Köthe, A. Zarli, "Implementation of the Distributed Workflow Service", ESPRIT 20408 – VEGA D303, 1998.
17. (Stephens 1998) J. Stephens, &al, "Virtual Enterprise using Groupware tools and distributed Architectures", Proceedings of the 2nd European Conference on Product and Process Modelling (ECPPM) 1998, october 19-21, Watford (UK), p. 459-468.
18. (WfMC 1994) WfMC, The Workflow Reference Model, 1994, Workflow Management Coalition: Brussels, Belgium.
19. (WfMC 1996) WfMC, Terminology & Glossary, 1996, Workflow Management Coalition: Brussels, Belgium.
20. (Zarli 1997) A. Zarli, et al., "Integrating emerging IT paradigms for the Virtual Enterprise: the VEGA platform", Proceedings of the 4th International Conference on Concurrent Enterprising (ICE'97), october 8-10 1997, Nottingham (UK), p. 347-359.
21. (Zarli 1998a) A. Zarli, O. Richaud, E. Buckley, "Requirements and Trends in Advanced Technologies for the Large Scale Engineering Take-up", Proceedings of the CIB WG78-98: The Life-cycle of Construction IT Innovations, june 3-5 1998, Stockholm (Sweden), p. 445-456
22. (Zarli 1998b) A. Zarli, P. Debras, "Integration of CORBA and WEB technologies in the VEGA DIS", Proceedings of the European Conference on Integration in Manufacturing (IiM) 1998, october 6-8, Gothenburg (Sweden), p. 184-197.

---

[i] This research has been partially funded by the Commission of the European Communities under the ESPRIT IV Programme EP 20.408 VEGA: Virtual Enterprises using Groupware tools and distributed Architectures, and with the following partners: CSTB (France), DEC (Germany), EPM (Norway), Nemetschek (Germany), Taylor Woodrow (UK) and TNO (The Netherlands).

[ii] Standard for the Exchange of Product model data, ISO/TC184/SC4.

[iii] Distributed Information Service (VEGA WorkPackage 4).

[iv] EP 25741 WONDA: WOrld wide eNterprise Data interoperAbility - http://www.bild.ie/wonda.

[v] LinkWorks is a solution framework for mission-critical business processes, using workflow technology to support the collaboration of people in an industrial environment.

[vi] OMG - Transaction Service Specification, Revised Edition, March 1995, Updated July 1997 – http://www.omg.org

[vii] OMG - Concurrency Control Service, Revised Edition, March 1995, Updated July 1997 – http://www.omg.org

[viii] Atomicity, Coherency, Isolation, Durability.

[ix] Unlike transactions, the SDAI has not been designed with concurrency in mind.

[x] Portable Object Adapter, a new specification from the OMG about generic adapters for CORBA servers.

[xi] Such a DBMS, interfaced with the COAST server methods and plugged through a COAST object adapter, becomes what is called a COAST server.

[xii] It is worth noticing that this is already most of the time the case in the AEC industry.

[xiii] This concept is under standardization by the OMG, along with a framework for business applications.