A Service-Oriented Architecture for a Collaborative Engineering Environment in Petroleum Engineering

Ismael H. F. Santos 1, Alberto B. Raposo2, Marcelo Gattass2

(1) : CENPES, Petrobras Research Center, Ilha do Fundão, 21949-900, Rio de Janeiro, Brazil
+55 21 3865-4365/6272 /+55 21 3865-6793
E-mail : ismaelh@petrobras.com.br, ismael@tecgraf.puc-rio.br

(2) : TecGraf, Computer Graphics Technology Group, PUC-Rio
+55 21 2512-5984 r231 /+55 21 3527-1848
E-mail : {abraposo, mgattass}@tecgraf.puc-rio.br

Abstract: In this paper we discuss the scenario of Petroleum Engineering projects at Petrobras, a large Brazilian governmental oil & gas company. Based on this scenario, we propose a Service-Oriented Architecture (SOA) for a Collaborative Problem Solving Environment (CPSE), that we call Collaborative Engineering Environment (CEE), responsible for controlling and executing specialized engineering projects in the oil&gas industry. The environment is composed by the integration of three different technologies for distributed group work: Scientific Workflow Management System (ScWfMS), Multimedia Collaborative System (MMCS) and Collaborative Virtual Environments (CVE).

Key words: Collaborative Problem Solving Environment, SOA, Enterprise Service Bus, Scientific Workflows.

1- Introduction

The present work is motivated by the necessity of finding effective solutions for collaboration of team workers during the execution of large and complex Petroleum Engineering projects at Petrobras, a large Brazilian governmental oil&gas company. Presently the oil&gas industry faces increasing hydrocarbon finding and lifting costs, especially in more remote locations, ultra-deep (400 m or deeper) water reservoirs or in hostile environments. A bottom-line business driver is the need to improve recovery from existing fields. Such a business measure sets up the dilemma between improved productivity and best cost-performance. Visualization and Collaboration technologies help us to bridge the cost-productivity problem.

High-end visualization systems are commonplace in the oil&gas industry, especially in the Exploration & Production (E&P) segment, also called Upstream. In the nineties oil companies were among the first to make industrial use of the so-called virtual reality centres (VRCs), equipped with immersive projection systems with large display walls (e.g., cave, cave-like, curved-panel, and powerwall); videoconference tools (VC) and auditory display systems. Techniques such as three-dimensional geometric modeling, scientific visualization, immersive virtual environments (VEs), commonly used in VRCs, pushed the limits of teamwork activities especially in Geosciences, Reservoir, Petroleum and Offshore Engineering.

Figure 1: Engineers in a collaborative section
from Petrobras and PTC Division Reality [DR1].

The configuration of VRCs greatly improved visual communication and group collaboration in technical work sessions and decision-making meetings. The possibility of visualizing and manipulating virtual models in the VRCs has completely changed the way of working, notably for the geologists and engineers (Figures 1 and 2).

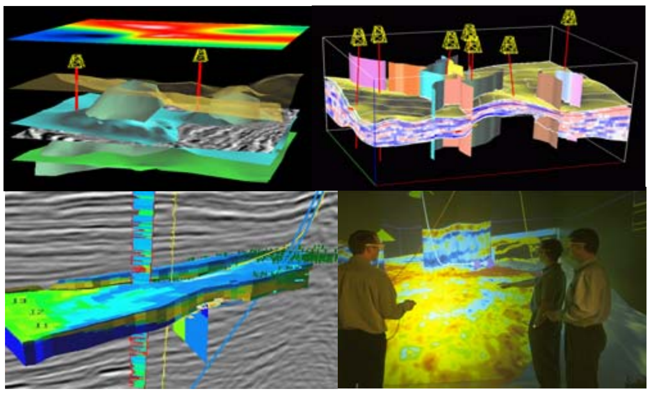


Figure 2: Geologists, Geophysicists in a collaborative section
from GoCAD [G1], LandMark [LG1] and Schlumberger [SI1].

In this paper we propose a Service-Oriented Architecture for a Collaborative Problem Solving Environment for Petroleum Engineering Applications (CEE), tailored for assisting the control and execution of engineering projects in the oil&gas industry. The proposed CEE is intended to create a useful workspace for *collaboration* through the composition of three different technologies for distributed group work:

1- *process-oriented collaboration tool* - a Scientific Workflow Management System (ScWfMS);
2- *synchronous communication tool* - a Multimedia Collaborative System (MMCS);
3- *collaborative shared workspace* - Collaborative Virtual Environment (CVE).

The CEE's intention is to control the execution of Petroleum Engineering projects involving many geographically distributed teams. It also allows an easy integration of different applications providing the team workers with means of information exchange, aiming to reduce the barriers imposed by applications with limited or no collaboration support. This environment needs to be *extensible*, *flexible* and *platform-independent*, allowing a transparent flow of information among different teams and their models.

The difficulties in building an effective CEE can be analyzed in four domains:

1- *collaborative work* - in this domain there is the necessity of providing effective human-to-human interaction and communication for solving conflicts and enhancing group productivity. Also there is the necessity of some support for coordinating the execution of tasks.;
2- *project management* - this domain points to the necessity of reducing costs and time-to-market of new products, which further requires a computerized solution capable of managing projects, controlling time scheduling and costs*;*
3- *distributed execution* - in this domain resides the necessity of involving specialists in different areas located in different places and using distributed resources, requiring the solution to have the ability to be easily and seamlessly distributed;
4- *system interoperability* - here there is a myriad of software that specialists are forced to use to accomplish their tasks in a reasonable time, what implies the necessity of interoperability among the components of the solution.

Our solution intends to tackle with all these requirements combining different technologies. For example, for the *collaborative work* domain we propose the combination of a ScWfMS, the Kepler system [K1], and a home-made Videoconferencing tool, CSVTool (Collaboration Supported by Video) described in [PR1]. For the *project management* domain we count on the resources of an Enterprise Content Management System (ECMS) [EX1] to control all the documents and data generated during project's life-cycle. For the last two domains, *distributed execution* and *system interoperability*, we use an Enterprise Service Bus (ESB) [HK1] and [O1], the next generation of integration middleware, which establishes an enterprise-class messaging bus that combines a messaging infrastructure with message transformation and content-based routing in a layer of integration logic between service consumers and providers (Figure 3). The use of an ESB allows a seamlessly integration of distributed applications modeled as SOA services. For every external Engineering Application that might be invoked by the scientific workflow during the execution of a user job, we had to build a service interface that allows the application be called from inside the workflow or from any other application connected to the ESB. An SOA is a very attractive architecture for allowing independence between service providers and consumers. In this way the CEE consists of a flexible and effective environment that improves the productivity of teams involved in large and complex engineering projects.

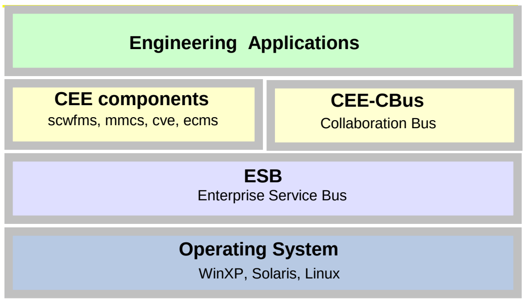


Figure 3: CEE Architecture

The integration of the ScWfMS with the other components is done in a seamless way through the Collaboration Bus (CEE-CBus) (Figure 4) in a way that the user always interacts with the same interface independent of the application he/she is currently using. This is a very important aspect of the solution to keep the user conscious of what he/she is doing and what should be the next steps of the current task being executed. The CBus represents the collaborative infrastructure provided by the CEE core functions to fulfill the requirements discussed in the text.

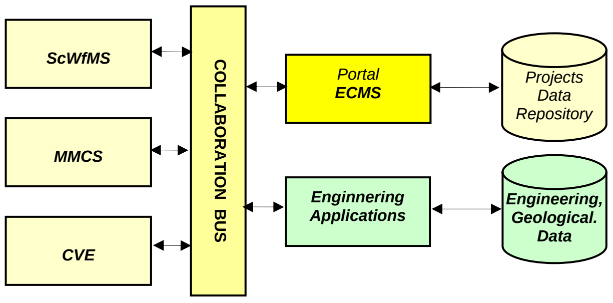


Figure 4: CEE Collaboration Bus.

Although Grid Computing is a hot topic for Scientific Workflows, as can be seen in the works of [FK1] and [YB1], in this work we concentrate on the collaboration aspects of the CEE. Of course, concerning the final implementation of the CEE a Grid-enabled Scientific Workflow Management System is a desirable achievement.

In the following sections we present some aspects of our solution. In section 2, we present the problem definition. In sections 3 to 7 we discuss some relevant characteristics of our collaborative solution. Related works are commented throughout the text. In section 8 the whole system architecture and some application scenarios are discussed. Conclusions in section 9 finish the paper.

2- Problem Definition

In this work we will focus our attention on Offshore Engineering projects as a case study. In this field research is being conducted to design oil production units, such as platforms, or to adapt old ships to work as Floating Production Storage and Offshore Loading (FPSO) units, for operating in ultra deep water. The project of a new production unit is a very lengthy and expensive process; it can last more than a year and consume a few hundred million dollars, depending on the complexity of the unit and the necessity to develop an adequate technology that makes the project technically and economically feasible.

2.1 – Offshore Engineering Projects

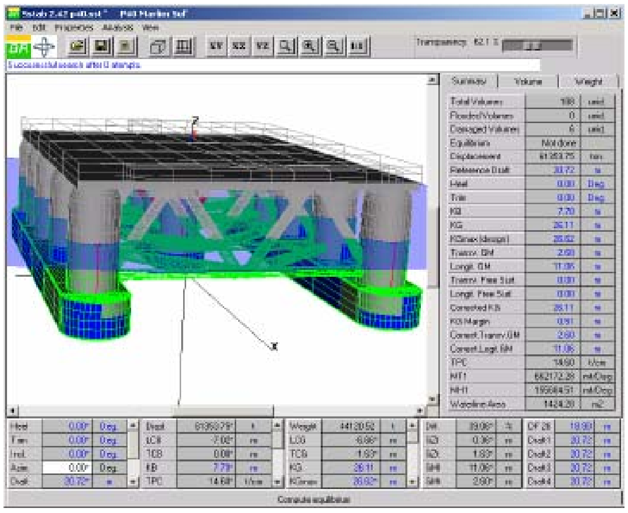
Usually Offshore Engineering projects involve not only geographically distributed teams but also teams of specialists in different areas using different software tools, both commercial and homemade. The interoperability of those tools is still an issue in the industry and is a mandatory requisite for any viable collaborative solution [SV1].
Due to their huge complexity, offshore engineering projects are divided into smaller interrelated subprojects where each one deals with an abstract representation of the others. During the conceptual design phase of the project the work is carried out basically, but not only, by the following teams:

1- *Naval engineers*: project the hull of the ship, defines the optimal positioning of the array of tanks, the mooring system, and study the dynamic stability of the unit based on meteo-oceanographic information about the wind, tide and water currents.
2- *Structural engineers:* defines the internal structure of the unit and its load capacity;
3- *Production and equipment engineers:* project the production system, encompassing risers, flowlines, and plan the installation of deep-water production equipments, such as manifolds and “christmas trees”;
4- *Chemical and process engineers:* projects the process plant based on the characteristics and expected volume of oil and gas that will be produced;
5- *Geotechnical engineers:* determine the position for anchoring the production unit based on studies of the behavior of the soil-structure interaction.

It can be seen by each team’s attribution that the necessity for collaboration is crucial because decisions are interdependent. Each team activity or new decision can affect others activities. For example changing the position of a large and heavy equipment in the unit can compromise the stability of the ship. In some cases there is also an intrinsic coupling among the solutions of the different subprojects which requires a lot of interactions and discussions among the teams involved. This is the case of the mooring system and of the production risers subprojects. On one hand if the mooring system allows great fluctuations of the ship, it can simply damage the production risers; on the other hand the presence of the risers itself helps to weaken the movements of the ship which contributes positively to the equilibrium of the system.
Another situation where collaboration among the teams is very important is the case of static and dynamic stability analysis of oil platforms and offshore structures as discussed next.

2.1.1 – Stability analysis of Oil platforms

In the oil&gas industry, the upkeep of complex structures where thousands of barrels of oil are produced daily in the open sea also requires calculating the action of sea currents, waves and winds on semi-submersible platforms and FPSOs. Additionally, these production units may be afloat in regions two thousand meters deep or more, therefore requiring the deployment of complex mooring systems. To deal with such problems, Petrobras developed software such as Dynasim [CK1], created to compute the supervening forces and consequential movements on anchored structures; and MG [CG1] designed to generate mesh representations of surfaces and to allow the visualization of stresses and strains calculated by numerical simulators.


Sstab 2.42 p40.sst - P40 Marlim Sul
File Edit Properties Analysis View
Transparency 62.1 %
Summary
Volume
Weight
Compute equilibrium

Total Volumes	198	unid
Flooded Volumes	0	unid
Damaged Volumes	6	unid
Equilibrium	Not done	
Displacement	61353.75	ton
Reference Draft	30.72	m
Heel	0.00	Deg
Trim	0.00	Deg
KB	7.70	m
KG	26.11	m
KGmax (design)	28.82	m
Transv. GM	2.80	m
Longit. GM	11.06	m
Transv. Free Surf	0.00	m
Longit. Free Surf	0.00	m
Corrected KG	26.11	m
KG Margin	0.91	m
Corrected Transv. GM	2.80	m
Corrected Longit. GM	11.06	m
TPC	14.60	t/cm
MT1	66217.28	mt/Deg
MH1	19568.51	mt/Deg
Waterline Area	1424.20	m2

Heel	0.00°	Deg	Displ	61353.79°	t	Weight	44120.52	t	Dif	39.06°	%	DF 28	18.90	m
Trim	0.00°	Deg	LCB	-7.00°	m	LCG	-6.86°	m	...	0.36°	m	Draft1	30.72	m
Incl.	0.00°	Deg	TCB	0.00°	m	TCG	1.83°	m	...	1.83°	m	Draft2	30.72	m
Azim.	0.00°	Deg	KB	7.79°	m	KG	26.11	m	GML	11.06°	m	Draft3	30.72	m
Draft	30.72°	m	TPC	14.60°	t/cm	KGmax	28.82°	m	GMT	2.90°	m	Draft4	30.72	m

Figure 5: Sstab’s GUI.

Sstab, The Floating Units Stability System, is used to design and analyze static conditions of floating units such as semi-submersible oil platforms [CJ1]. This system has been successfully used not only by certifying companies but also in emergency scenarios under extreme conditions. It also works integrated with the Dynasim system (Figures 5 and 6).

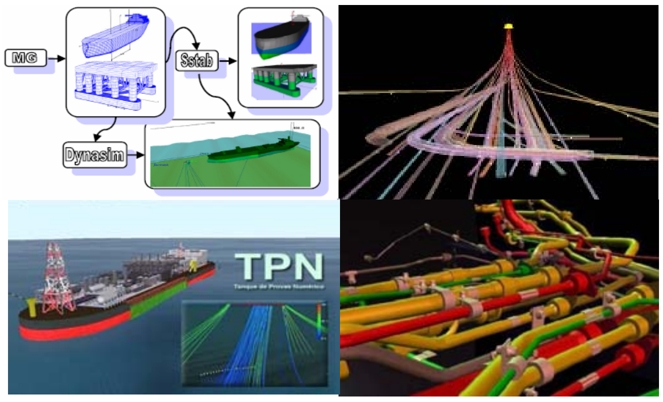


Figure 6: Sstab, Dynasim and NOT integration for the design of an FPSO’s mooring system and production risers.

Another important simulator developed by Petrobras that make use of visualization systems is The Numerical Offshore Tank (NOT). The NOT is a computer-intensive applied hydrodynamics lab whose main goal is to simulate and analyze floating production and storage oil&gas systems. The strong points in the NOT are the hydrodynamic numerical and empirical models that considers the interaction between waves and currents, the line dynamics and the damping, a strong graphical interface and full 3D visualization of the results of simulations. NOT has incorporated several innovations, such as massive parallel computing through a high-performance parallel PC computers cluster and an immersive visualization system that presents the simulation results in true 3D stereoscopic graphic.

2.1.2 – Difficulties in data management

Another difficulty presented in Offshore Engineering projects is that although the specialists deal with the same artifacts (platforms, mooring systems, etc.) they usually have different data representations for those objects according to the needs of each application requiring some support for *multi-resolution* representation of the data. For example, in structural and naval engineering the models usually have dense polygonal meshes, with a few objects representing the outline of the artifacts, suitable for static and dynamic stability studies with numerical methods. In CAD/CAE the models usually have objects with coarse grid meshes suitable for good visual representation, but the problem is that all objects that comprise the artifact should be represented yielding huge models. A typical CAD/CAE platform model usually has dozens of millions of polygons while the mooring system itself has a few millions of polygons. For real time visualization those models are a tough problem and therefore represent a great challenge in computer graphics [RC1].

2.1.3 – Potential applications

There are other important activities that will benefit with the implementation of the CEE as defined in this paper, such as training and security simulations; design, planning and optimization of marine installations and sub-sea layout arrangement of production equipments; remote teleoperation and interventions on submarine equipments; preparing maintenance and inspections plans in production units; planning oil pipeline installation and monitoring [SC1] and emergency scenario applications [RR1].

3 – Collaborative Applications

3.1 – The 3C Model

Collaboration may be seen as the combination of communication, coordination and cooperation. *Communication* is related to the exchange of messages and information among people. *Coordination* is related to the management of people, their activities and interdependencies and of the used resources. *Cooperation* is the production of common artifacts taking place on a shared space through the operations available to the group. This model, called 3C model, was originally proposed by Ellis et al. [EG1].
Collaborative applications, according to the 3C model, are composed of tools providing one or more of the three functionalities described above (Figure 7). Another central aspect of the 3C model is the notion of user awareness, which is defined as the way users perceive other participants of the collaboration and what they are doing, without direct communication between them [DB2]. Awareness elements are essential for the collaboration flow, because they enable the user to build his/her own work context and to coordinate his/her activities with those of the others. Therefore, user awareness may be considered the fourth element of the 3C model, which is deeply related to *communication*, *coordination* and *cooperation*.

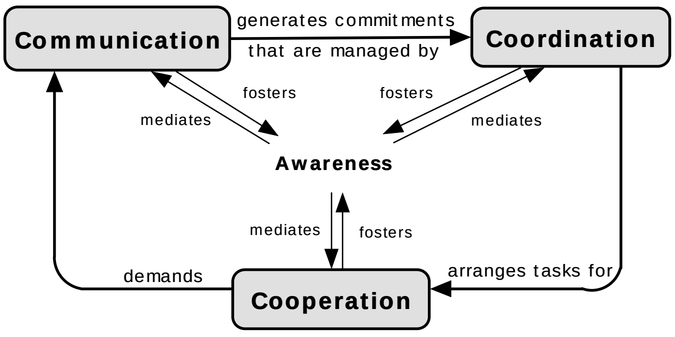


Figure 7: The 3C collaboration model [FR1].

The relative importance of each of components of the 3C Model depends on the objectives pursued by specific types of collaborative systems. For instance, Mediaspaces [M1] emphasize informal communication to enhance team awareness, usually supporting no cooperation and restricted coordination functionalities for controlling of the simultaneous use of communication channels. Workflow Systems [E1], on the other hand, emphasize coordinated communication allowing groups of people execute, monitor, and coordinate the flow of work cases within a distributed office environment.

3.2 – Collaboration Applications

The applications available for Computer-Supported Collaborative Work (CSCW) can be classified depending on how the support for collaboration is related to the application implementation. They can be seen as *collaboration-aware* or *collaboration-unaware* applications [RS1].

3.2.1 – Collaboration-unaware Applications

Collaboration-unaware applications are originally developed to be single user applications, but may be used collaboratively by means of an external support system. This external support system may be an application-sharing system, such as Microsoft Windows NetMeeting^{TM}, or a GUI event multiplexing system. In the application-sharing approach only one instance of the application is running on one user's machine, the application server, under the control of the application-sharing package. This package is responsible for broadcasting application's output (windows' contents) to all connected users, gathering all users' inputs, serializing these into one single input stream which is then passed to the application that operates as if a single user was controlling it. In the GUI event multiplexing system, all users are running an instance of the application, with a special interface layer between the application's GUI and its event handler. This layer broadcasts all GUI events to connected users which interprets all received remote events as if they

had been generated by the local user. In both cases the applications do not explicitly support collaboration; they are implemented as single user applications [T1].
The main common advantage of all *collaboration-unaware* schemes is that no specifically developed version of the application for collaboration is necessary; standard applications can be immediately used collaboratively.

3.2.2 – Collaboration-aware Applications

Collaboration-aware applications, on the other hand, are specially developed or adapted to support collaboration. They typically constitute distributed systems, with centralized or replicated data sharing, where each user has access to a locally executed application instance. All running applications are connected to a server process, in a client/server architecture, or interconnected, peer-server or peer-peer, and exchange information over designated communication channels. All the peers are aware of the communication channels shared with its peer applications; which information is exchanged among them; the number of connected peers and their role in the collaboration; and the coordination policies adopted by the group.

3.2.3 – Collaborative Applications Degrees

The methodology used for the development of the CEE is based on 3C collaboration model. This approach is denominated Groupware Engineering, which is based on Software Engineering, enhanced by concepts originated from the field of CSCW and related areas. The Groupware Engineering cycle is based on the spiral software development model, which combines the classical sequential model and the iterative behavior of incremental prototyping. The phases of groupware development used are Domain Analysis, Requirement Analysis, Design, Implementation and Testing [FR1]. In order to better define our solution and its requirements, it is necessary to define the collaboration needs of the Offshore Engineering projects scenario and which kind of collaborative application should be used.

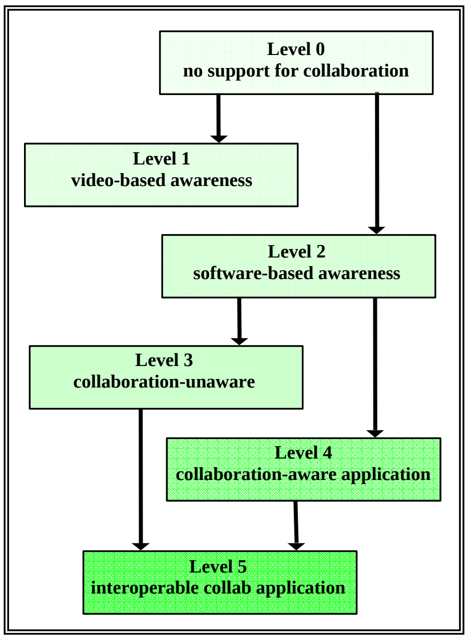


Figure 8: Hierarchy of collaborative applications.

The model presented in [SR1], defines hierarchical levels for collaboration resources (Figure 8) that serves as a guideline to for the incremental development of collaborative applications. At each level, different collaboration degrees are supposed.
At level 0, no support for collaboration is defined. At level 1, called *video-based awareness*, a higher degree of communication is achieved with integrated audio and videoconferencing system to the solution. At this level, the collaboration scenario is not complete, since the peer users are not able to interact with each other's application. At level 2, a degree of cooperation and coordination is possible, given the user capabilities to interact with the remote workspace. Level 3 gives collaboration support to applications that were originally developed to be single user, and do not provide explicit support for that *(collaboration unaware)*. Applications at level 4 are similar to level 3, but the support for collaboration is provided by distributed applications especially developed for that purpose (*collaboration aware*). At level 5, a framework for interoperability among different applications should be provided.
It is important to mention that the lower levels, 1 and 2, though having poorer collaborative resources, are easier to implement and, in some cases, are the only feasible solutions due to the available infrastructure and budget constraints. Moreover, in some cases where the most important tools used in the environment are commercial software with non-extensible functionalities it is not possible to reach higher collaboration levels, which require intrusive interventions in the software. Based on the description of the problem we can say that our CEE should constitute an application between levels 4 and 5 depending on the degree of interoperability that will be supplied to the users by the middleware component.

4- Collaborative Problem Solving Environments

A Problem Solving Environment (PSE) is a specialized software system that provides all the computational facilities needed to solve a target class of problems. These features include advanced solution methods, automatic and semiautomatic selection of solution methods, and ways to easily incorporate novel solution methods. Moreover, PSEs use the language of the target class of problems, so users can run them without specialized knowledge of the underlying computer hardware and software technology [HG1].
PSEs have been built for a number of scientific domains. For example, Parker et al. [PM1] describe SCIRun, a PSE that allows users to interactively compose, execute, and control a large-scale computer simulation by visually "steering" a dataflow network model. SCIRun supports parallel computing and output visualization, but originally has no mechanisms for experiment managing and archiving, optimization, real-time collaboration, or modifying the simulation models themselves.
Klie et al. [KB1] proposes a PSE composed of high performance numerical methods, tools and grid-enabled middleware system for scalable and data-driven computations for multiphysics simulation and decision-making processes in integrated multiphase flow applications for oil reservoir management.

Collaborative Problem Solving Environments (CPSE) focus on the development and integration of scientific tools and technologies coupled with collaborative environments to support the modeling and simulation of complex scientific and engineering problems. These capabilities enable engineers to easily setup computations in an integrated environment that supports the storage, retrieval, and analysis of the rapidly growing volumes of data produced by computational studies.

Experience in dealing with large-scale engineering design and analysis problems has indicated the critical need for CPSEs with four distinguishing characteristics:

1- interoperability facilities to integrate different applications;
2- support for human collaboration;
3- transparency for the use of distributed resources;
4- advisory support to the user.

In principle, CPSEs can solve simple or complex problems, support both rapid prototyping and detailed analysis, and can be used both in introductory education and at the frontiers of science and engineering [DB1].

The integration of effective graphical user interfaces, Scientific Visualization, and Virtual Reality techniques, Analysis and Modeling Tools aid in the automation of modeling analysis and data management. To enhance engineers ability to share information and resources with colleagues at remote locations, collaborative and real-time technologies integrated into a CPSE provide a unified approach to the scientific and engineering discovery and analysis process.

As we have mentioned our CEE is a specialized CPSE tailored for assisting the control and execution of engineering projects in the oil&gas industry. Based on our previous works in the related area [SV1], and on an analysis of the domain of *Offshore Engineering* as our prototype scenario, we define a set of requirements for a CEE that follows.

4.1 – CEE Requirements

The requirements for a CEE, in our case designed as a *collaborative-aware application*, can be defined according to each aspect of the 3C Model [EG1] mentioned earlier.

1- ***Communication support*** – is a fundamental requirement in our scenario. The CEE should provide different communication support possibilities: synchronous or asynchronous, enabled in various media types (audio, video and text based communication). This should be provided in a seamless way, so that our users can start a communication of one or of another type while they are interacting with the CEE. They also should be able to plan a certain time for a specific communication interaction. The communication support should be integrated to the other tools in the CEE and provide means of recording conversation and retrieving old ones. This requirement helps user solve their project's problems in critical situation, with fast interaction and negotiation, and it allows the recovery of useful pieces of communication used to solve similar problems in the past.
2- ***Coordination support*** – at the project management level, multiple and different visions of the on-going project must be provided by the CEE. Users have different background (e.g. managers, engineers) and need different types of information to execute their duties. Project management should also be feasible.
3- ***Cooperation and flexibility support*** – there should exist flexible process modeling support, like dynamic change of process instances during run-time to support dynamically evolving processes, possibility of executing rollback of processes (reset, redo, undo, recover, ignore, etc), reuse of process fragments and component libraries. The cooperation support must provide different levels of data access: local and distributed, shared, public and private access, versioning control of engineering models and related data, concurrency control and synchronization. It is also necessary to provide support for different types of data interchange, concurrent work on shared copies, change propagation, and physically shared data access. Different types of model visualization should also be available at the CEE, as well as some data management infrastructure related to these models, like real-time simulation and visualization of 3D models, possibilities of walkthroughs in the models, object interaction and manipulation, edition and planning and also access to organizational work history.
4- ***Awareness*** – there are different types of awareness support that can be foreseen in a CEE. In our scenario, the most important ones are:
   a. ***event monitoring*** – observes what is going on in all separate parts and provide active notification to the right person, at the right time and the right sub-system;
   b. ***workspace awareness in the virtual environment*** – provides control of collaborative interaction and changing of the user location;
   c. ***mutual awareness*** – allows users see each other's identity and observe each other's actions;
   d. ***group awareness*** – facilitates the perception of groups of interest connecting people who need to collaborate more intensely.
5- ***Integration Management Infrastructure*** – at this level, several smaller services should be available in order to guarantee the data and modeling persistency, and the different levels of access control to different user roles in our scenario. Here we include the shared workspace and results service, access control service, user management service, data synchronization service, security service and software mediators.
6- ***CVE specific requirements*** – *high performance* and *scalability* are important aspects of *VE* architectures intended to support execution of large shared virtual worlds over long periods of time, especially when the virtual worlds vary widely in size and number of participants; a *persistence mechanism* to save and restore world state between activations; *version-safe* updating mechanisms, because large and long-lived virtual worlds tend to incorporate different versions of the same components; *composability*, so that one

may easily and effectively combine worlds and world components developed by different organizations; *dynamic extensibility*, i.e., to as large an extent as possible, the architecture must permit the seamless run time extension and replacement of any part of its hosted application.

In the following sections we present the CEE components that help us to fulfill the previous mentioned requirements.

5- Business and Scientific Workflows

5.1 – Business Workflows (BWfMS)

In [E1] Ellis presents Workflow Management Systems (WfMS) as a tool to assist in the specification, modeling, and enactment of structured work processes within organizations. These systems are a special type of collaboration technology which can be described as "*organizationally aware groupware*" [EN1]. According to the Workflow Management Coalition (WfMC), a WfMS is "the computerized facilitation or automation of a business process, in whole or in part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules" [W1], [W2].

A WfMS contains two basic components:

1- ***Workflow modeling component***, which enables administrators and analysts to define processes (or procedures) and activities, analyze and simulate them, and assign them to people, agents or processes. This component is sometimes called "*specification module*" or "*build time system*".

2- ***Workflow execution component*** (or *enactment*), sometimes also called the "*run-time system*". It consists of the execution interface seen by end-users and the "***workflow engine***", an execution environment which assists in coordinating and performing the processes and activities. It enables the units of work to flow from one user's workstation to another as the steps of a procedure are completed. Some of these steps may be executed in parallel; some executed automatically by the computer.

Typically, a workflow system is implemented as a server machine which has and interprets a representation of the steps of the procedures and their precedences; along with client workstations, one per end-user, which assists the user in performing process steps. This is typically combined with a network and messaging system (or communication mechanism) to allow the server to control and/or to interact with end-user workstations. Also included is a database that stores the process representation, attributes of end-users, and other pertinent workflow information. Many of the workflow products are combined with imaging and/or Document Management Systems (DMS).

There are different types of workflows, which suit different organizational problems. They are:

1- ***Production workflow*** – the key goal is to manage large numbers of similar tasks, and to optimize productivity.

2- ***Administrative workflow*** – its most important feature is the ease to define the process. Flexibility is more important than productivity, and these systems handle one or two orders of magnitude lower numbers of instances per hour than Production Workflow Systems.

3- ***Collaborative Workflow*** – focuses on teams working together towards common goals. Groups can vary from small, project-oriented teams, to widely dispersed people with interests in common. Effective use of collaborative workflow to support team working is now considered a vital element in the success of enterprises of all kinds. Throughput is not an important consideration, and Process Definitions are not rigid and can be amended frequently.

4- ***Ad-hoc Workflow*** – allows users to create and amend Process Definitions very quickly and easily to meet circumstances as they arise. So it is possible to have almost as many Process Definitions as there are instances of the definitions. It maximizes flexibility in areas where throughput and security are not major concerns. Whereas in Production Workflow, clearly the organization owns the process, Ad-Hoc Workflow users own their own processes.

The type of workflow used in this work is called ***Adaptive Workflow*** and follows the definition of ***Collaborative*** and ***Ad-hoc Workflow*** plus some additional characteristics. This kind of workflow enables the coordination of different types of exceptions, dynamic change problems and possibilities of late modeling and local adaptation of particular workflow instances. The support for managing partials workflows present in an ***Adaptive Workflow*** is very attractive for our purposes because processes in engineering domains have a very dynamic nature which means that they cannot be planned completely in advance and are under change during execution. Furthermore, in contrast to well-structured business processes, they are characterized by more cooperative forms of work whose concrete process steps cannot be strictly prescribed.

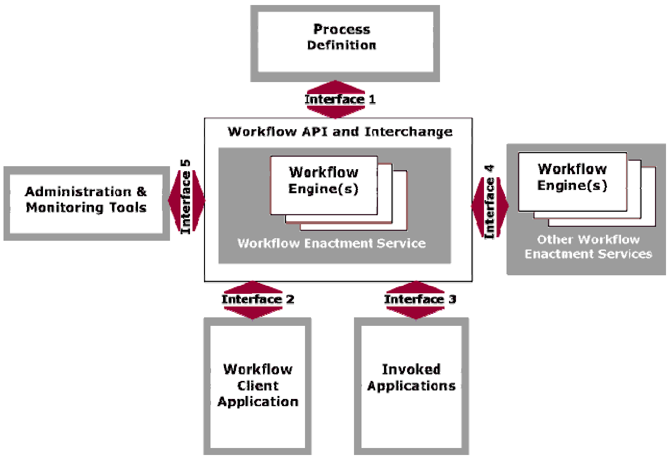

Process Definition
Interface 1
Workflow API and Interchange
Administration & Monitoring Tools
Interface 5
Workflow Engine(s)
Workflow Enactment Service
Interface 4
Workflow Engine(s)
Other Workflow Enactment Services
Interface 2
Interface 3
Workflow Client Application
Invoked Applications

Figure 9: WfMC reference model
from Workflow Management Coalition [W2].

5.1.1 – Workflow Components

To achieve workflow interoperability, the Workflow Management Coalition (WfMC) created The Workflow Reference Model that describes FIVE Interface definitions

(Figure 9).

1- ***Interface 1 (Process Definition)*** - deals with passing Process Definitions from external tools to the workflow engine where they are enacted. This is the link between the so-called "Process Definition Tools" and the "Enactment Service".
2- ***Workflow APIs (Interfaces 2 & 3)*** - these interfaces have been combined and cover the WfAPIs (Workflow API's). The support of these interfaces in workflow management products allows the implementation of front-end applications that need to access workflow management engine functions (workflow services). Such implementations might be written by workflow management exploiters or workflow systems integrators (WfSI). Integration between workflow and other desktop tasks (calendar, mail, reminders, etc) is often a common target and the workflow APIs allow workflow task integration into a common desktop.
3- ***Inter-Engine Workflow (Interface 4)*** - defines the mechanisms that workflow product vendors are required to implement in order that one workflow engine may make requests of another workflow engine to effect the selection, instantiation, and enactment of known process definitions by that other engine. The requesting workflow engine is also able to pass context data (workflow relevant or application data) and receive back status information and the results of the enactment of the process definition. As far as possible, this is done in a way that is "transparent" to the user. This interface is intended for the use of WfSIs, and not users. As a side effect of facilitating communication between workflow engines, there is a requirement for audit data to be produced.
4- ***Audit and Monitoring (Interface 5)*** - the support of this specification in workflow products allows analysis of consistent audit data across heterogeneous workflow products. During the initialization and execution of a process instance, multiple events occur which are of interest to a business, including WfAPI events, internal workflow management engine operations and other system and application functions. With this information, a business can determine what has occurred in the business operations managed by workflow.

5.1.2 – Process Definition Language

The WfMC defines a *Process Definition* as "the representation of a business process in a form which supports automated manipulation, such as modeling, or enactment by a workflow management system. The *Process Definition* consists of a network of activities and their relationships, criteria to indicate the start and termination of the process, and information about the individual activities, such as participants, associated IT applications and data, etc." [W2]. This reveals the necessity for a *Process Definition* interchange mechanism. First, within the context of a single workflow management system there has to be a connection between the design tool and the execution/run-time environment. Second, there may be the desire to use another design tool. Third, for analysis purposes it may be desirable to link the design tool to analysis software such as simulation and verification tools. Fourth, the use of repositories with workflow processes requires a standardized language. Fifth, there may be the need to transfer a definition interchange from one engine to another.

```
<WorkflowProcess Id="Sequence">
    <ProcessHeader DurationUnit="Y"/>
    <Activities>
        <Activity Id="A">
            ...
        </Activity>
        <Activity Id="B">
            ...
        </Activity>
    </Activities>
    <Transitions>
        <Transition Id="AB" From="A" To="B"/>
    </Transitions>
</WorkflowProcess>
```

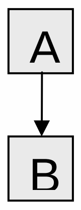


Figure 10: Workflow pattern Sequence in XPDL.

The *XML Process Definition Language* (XPDL) is a format standardized by the WfMC to interchange *Business Process* definitions between different workflow products like modeling tools and workflow engines. XPDL defines a XML schema for specifying the declarative part of workflow. This language is a low level language and it can be used to model higher level business languages.

A workflow pattern is a specialized form of a design pattern as defined in the area of software engineering. Workflow patterns refer specifically to recurrent problems and proven solutions related to the development of workflow applications in particular, and more broadly, process-oriented applications. Figure 10 presents an example of Sequence pattern [AH1].

5.2 – Scientific Workflows (ScWfMS)

Although the above definitions make reference to "*Business Process*", WfMS is not only employed by business applications. In recent years, several industries have improved their operations through WfMS (improvement of data management and better coordination of activities through specific Business and Scientific and Engineering Process). However, there are remarkable differences between Business (BWfMS) and Scientific Workflows (ScWfMS). In [MW1] the authors identified that in a scientific environment scientists will typically specify their workflows themselves, while in a business environment, a system administrator is commonly responsible for this task. Another characteristic of ScWfMS pointed in their work is the need to trace workflow executions. An engineer may need to reuse a workflow in order to reproduce results. The operations a user performs on a given data must be recorded in order to provide engineers with the benefits of successful and unsuccessful workflows.

Scientific Workflows (ScWfMS) describe series of structured activities and computations that arise in scientific problem-solving. In many science and engineering areas, the use of computation is not only heavy, but also complex and structured with intricate dependencies. Graph-based notations, e.g., generalized activity networks (GAN), are a natural way of representing numerical and human processing. These structured activities are often termed *studies or experiments*. However, they bear the following similarities to what the databases research community calls workflows:

1- Scientific problem-solving usually involves the invocation of a number and variety of analysis tools. However, these are typically invoked in a routine manner. For example, the computations involve much detail (e.g., sequences of format translations that ensure that the tools can process each other's outputs), and often routine verification and validation of the data and the outputs. As data sets are consumed and generated by the pre and post processors and simulation programs, the intermediate results are checked for consistency and validated to ensure that the computation as a whole remains on track.
2- Semantic mismatches among the databases and the analysis tools must be handled. Some of the tools are designed for performing simulations under different circumstances or assumptions, which must be accommodated to prevent spurious results. Heterogeneous databases are extensively accessed; they also provide repositories for intermediate results. When the computation runs into trouble, semantic rollforward must be attempted; just as for business workflows, rollback is often not an option.
3- Many large-scale scientific computations of interest are long-term, easily lasting weeks if not months. They can also involve much human intervention. This is especially so during the early stages of process (workflow) design. However, as they are debugged, the exceptions that arise are handled automatically. Thus, in the end, the production runs frequently require no more than semiskilled human support. The roles of the participating humans involved must be explicitly represented to enable effective intervention by the right person.
4- The computing environments are heterogeneous. They include supercomputers as well as networks of workstations. This puts additional stress on the run-time support and management. Also, users typically want some kind of a predictability of the time it would take for a given computation to complete. Making estimates of this kind is extremely complex and requires performance modeling of both computational units and interconnecting networks.

Consequently, it is appropriate to view these coarse-granularity, long-lived, complex, heterogeneous, scientific computations as workflows. By describing these activities as workflows, we bring to bear on them the advanced techniques that have been developed in workflows research. These include sophisticated notions of workflow specification and of toolkits and environments for describing and managing workflows. In this way, Scientific Workflows will be to Problem-Solving Environments (PSE) what Business Workflows are to Enterprise Integration (EI).

Scientific workflows often begin as *research* workflows and end up as production workflows. Early in the lifecycle, they require considerable human intervention and collaboration; later they begin to be executed increasingly automatically. Thus in the production mode, there is typically less room for collaboration at the scientific level and the computations are more long-lived. During the research phase, scientific workflows need to be enacted and animated (fake enactment) far more intensively than business workflows. In this phase, which is more extensive than the corresponding phase for business workflows, the emphasis is on execution with a view to design, and thus naturally includes iterative execution. The corresponding activity can be viewed as a “Business Process Engineering” (BPE). For this reason, the approaches for constructing, managing, and coordinating process models will be useful also in scientific settings.

The characteristics, requirements and differences between business and scientific workflows are still being discussed. In two recent events, the Scientific Data Framework Workshop [SD1] and the e-Science Workflow Services [EW1], scientific workflows issues like workflow representation, parallelism, service composition, service description, scripting language support to manage runtime execution, mapping to resources, business workflow languages, among others were debated. As pointed out in these events, ScWfMS are more *data-flow oriented* while BWfMS are more *control-flow oriented***.** BWfMS require the coordination of a number of small messages and document exchanges. In ScWfMS usually no documents undergo modifications. Instead, often a dataset is obtained via analysis and transformation of another dataset. BWfMS need complex control flow, but they are not data-intensive pipelines. On the other hand, ScWfMS must deal with the heterogeneity, complexity, volume, and physical distribution of scientific data. In addition to these data problems, ScWfMS often deal with legacy or third-party programs, which can also be heterogeneous, and possible with no source code available.

5.3 – Workflow Integration with other technologies

In the literature there are a lot of proposals concerning integration of a WfMS and other technologies. [J1] proposes the combination with a Document Management System. He suggests the creation of a new data-oriented perspective for the WfMS, centered on the documents and data produced during the execution of tasks, in order to improve the coordination and cooperation support for engineering processes.

[WV1] proposes the junction with a Geographic Information System to combine a data-oriented view with a process-oriented view aiming to support the complex cycle of process and data modeling in environmental-related geoprocessing applications. This integration is very suitable for our solution because many activities in Offshore Engineering require the use of geo-referenced data.

6- Multimedia Collaborative Virtual Environment

6.1 – Multimedia Collaborative Systems

Audio and video communications are fundamental components of collaborative systems [IT1]. Audio is an essential channel for supporting synchronous work, and video is important to provide a sense of co-presence facilitating negotiation tasks.

MMCS such as Videoconferencing Systems (VCS), contain no knowledge of the work processes, and therefore are not "*organizationally aware*". These systems are best suited for unstructured group activities once that audiovisual connectivity and shared documents enable flexible group processes. The drawback is that all coordination tasks are left to the conference participants [RS1]. The combination of VCS and WfMS can support problems which cannot be well supported by each one of them isolated. Embedding synchronous teamwork as part of the workflow produces a complementary way of conducting project activities. Such integration would enable a continuous stream of tasks and activities in which fast, informal, ad hoc, and direct actions can be taken through conferences within the usual formal workflow.

The integration of MMCS systems into a WfMS is not new; [WP1] proposed the integration of a VC tool into a WfMS in order to furnish a synchronous collaboration work. To allow the coordination of the conference by the WfMS he suggests the creation of new entity in the workflow model, called "*conference activity*". Another important aspect is the time dimension. Conferences that are already planned at the time of the creation of the workflow are called pre-scheduled, while an ad-hoc conference is the one that was not foreseeable at the time when the workflow model is specified. This implies that in the former case some of the steps can be formally prescribed in the WfMS providing a tighter control of the results and documents generated during the conference section by the workflow engine, while in the later the results of the section should be updated by the users in the system.

6.2 – Collaborative Virtual Environments

The terms Virtual Environment (VE) and Virtual Reality (VR) are often used synonymously to describe a computer-generated, artificial environment or reality that is presented to a user. A VE tries to evoke a strong sense of reality in the user. This is achieved by the generation of artificial input to the user's visual, acoustic and haptic senses.

By interfacing some of the user's articulations in the real world back into the VE, the user can consciously interact with the environment. Typically, interfaces to direct-manipulation devices are used, but nowadays more advanced interaction techniques like speech and gesture recognition have become a major research interest.

The generation of high-quality visual feedback from the virtual environment is often considered the most important aspect in generating a high degree of immersion. The desire to increase the degree of immersion led to the development of sophisticated image generators and display devices. Beginning with low-resolution monoscopic CRT displays used in early flight simulators and image generators that where capable of rendering only a few hundred polygons per second, the development progressed toward today's high-resolution stereoscopic display systems like the CAVE [CS1] and readily available graphic cards that render hundreds of millions of polygons per second.

Parallel to the development of new display devices, image generators and input devices, various toolkits and application frameworks are developed. They provide a basic software infrastructure for the development of VE applications. The main goal of these efforts is the maximization of software reuse in order to minimize the necessary development resources for application development. Designed for different application domains, the only common nominator of most toolkits and frameworks is a scene-graph based object model. The provided API, the supported hardware and operating systems and the set of supported display and input devices vary greatly.

Collaborative Virtual Environments (CVEs) are a special case of Virtual Reality Environments [T2], where the emphasis is to provide distributed teams with a common virtual space where they can meet as if face-to-face, co-exist and collaborate while sharing and manipulating, in real-time, the virtual artifacts of interest [GL1]. They can be seen as the result of a convergence of research interests within the Virtual Reality and Computer Supported Cooperative Work (CSCW) communities. CVEs are becoming increasingly used due to a significant increase in cost-effective computer power, advances in networking technology and protocols, as well as database, computer graphics and display technologies. They have been used mainly by automotive and aircraft manufactures aiming to improve the overall product's quality and also aiming to reduce project's life cycle, cutting down costs and reducing the time-to-market of new products. Examples of applications are Visualization of real-time simulation of 3D Complex Phenomena, Collaborative Virtual Design and Product Development, Training and Edutainment, Telepresence and Telerobotics, Business meetings among others.

Studies of a cooperative work in real-world environments have highlighted the important role of physical space as a resource for negotiating social interaction, promoting peripheral awareness and sharing artifacts [BH1]. The shared virtual spaces provided by CVEs may establish an equivalent resource for telecommunication. In teleimmersive environments (TE), a VCS is integrated with a CVE to provide collaborators at remote sites with a greater sense of presence in the shared space [LJ1]. TEs may enable participants to discuss and manipulate shared 3D models and visualizations in such a way that each user can adopt their own viewpoint and can naturally indicate the others where they look and point. Scientific visualization has also been used in many application areas and has proven to be a powerful tool in understanding complex data [FB1]. Those characteristics of TEs are very important for Virtual Prototyping as in projects of oil production units explained in section 2.

The development of CVE technology has been driven mainly by the challenge of overcoming technological problems such as photo realistic rendering and supporting multiple users in CVEs. Once those users are geographically distributed over large networks like the Internet, and the number of users has been increasing continuously, scalability turns to be a key aspect to consider for real-time interactions [LM1].

Other important aspects are *composability* and *extensibility* or *dynamic reconfigurability* for assembling applications and improving adaptability of system at runtime with component-based system design, plug-ins functionality and service discovery mechanisms. In order to support the execution of *CVEs* with large-scale virtual worlds over long periods of time, they must be based on technologies that allow them to adapt, scale and evolve continuously. *VE* applications offer an almost limitless number of opportunities for the inclusion of plug-in technology. Graphical plug-ins may generate 3D models on the fly; network plug-ins may provide support for new protocols and filtering schemes; plug-ins for physical

simulation may introduce previously unknown forces that improves the reality of the simulation. *Persistence* and *portability* aspects have also to be considered in order to guarantee the ability of building reusable large virtual worlds commonly needed in engineering projects.
The following VE projects provided us very important insights that we adopted in our solution. Next follows a brief review of the most important aspects of them.

6.2.1 – ATLAS

ATLAS [LL1] studied scalability in networked VEs in terms of four scalability issues: *communication architecture*, *interest management*, *data replication* and *concurrency control*. It adopts a peer-server model with multicast support. A client-server model is also supported for flexibility reasons, although it may suffer from problems like server flooding. The interest management mechanism is based on user interests and spatial distance. Users with the same interests dynamically form a multicast group when they get close. Each user in the group multicasts update messages to the rest of the group whenever he/she moves or interact with the world. The concurrency control is based on an entity-centric prediction-based concurrency control scheme where only the users surrounding a target entity multicast the ownership request by using the multicast group address assigned to the entity. For data replication, ATLAS uses partial replication with on-demand transmission plus user-based caching and prefetching exploiting the object's access priority, which is defined based on spatial distance and individual user's interest in objects on the world.

6.2.2 – DIS, NPSNET-V

Distributed Interactive Simulation (DIS) and its predecessor SIMNET are standards for distributed interactive simulations [L1]. DIS retains a replicated database and uses dead reckoning to keep track of others. Dead reckoning is a strategy for dealing with network latency problems. It involves packaging an entity's location, a time stamp and a velocity vector. The host application receiving the data can then predict where the object should be. Each entity runs a local simulation together with the dead reckoning model, and when the two diverge by more than a certain tolerance a message is sent out to all participants to reconcile the discrepancy. DIS brings together several interesting ideas for distributed virtual reality:

1- Standard message formats.
2- No central server.
3- Dead reckoning.
4- Area of Interest Management (AOIM).
5- Potential use of multicasting.

Multicast messages can be addressed to a specified group of machines. Multicasting is popular in distributed VR systems, particularly with systems that support large numbers of peers, where it can considerably reduce network traffic [MZ1].
A number of applications implement the DIS protocols, such as PARADISE [S1] and NPS Networked Vehicle Simulator (NPSNET). The authors in NPSNET-V [KM1], developed by the Naval Postgraduate School, propose the use of a JAVA^{TM} component-based dynamic extensibility framework to allow one to construct applications as components hierarchies rooted at an invariant micro-kernel. Applications could be implemented as a federation of dynamically loadable modules, loosely coupled by a minimal set of well defined relationships. NPSNET-V defines object entities which contain information about artifacts in the virtual environment (e.g., tanks, people). The entity defines state information, such as location and velocity. From a list of defined protocols, the entity can be imbued with a set of behaviors.

6.2.3 – HLA

The High Level Architecture (HLA) effort [H1], [DW1] aims to facilitate the interoperability and composability of the broadest range of component-based simulations. The HLA did not originate as an open standard, but was later recognized and adopted by both the Object Management Group (OMG) and the Institute of Electrical and Electronics Engineers (IEEE). HLA development occurred in much the same way as did the DIS standard – namely, the initial design was produced within the U.S. Department of Defense and then delivered to an external body (IEEE, OMG) for standardization [S2].
The HLA architecture can be thought as an object-oriented net-VE design. Each simulator, known as a federate, is a component which represents a collection of objects, each having a set of attributes and capable of initiating and receiving events. The federate registers each of its objects with a piece of middleware called the Run-Time Infrastructure (RTI). The RTI collaborates with RTI instances on other hosts to learn about remote peers (objects) and delivers information about those participants to the local federate. The local federate, in turn, typically instantiates local objects representing those remote participants. Attribute updates and events are also exchanged through the RTI, which is responsible for handling area of interest management, time synchronization, and other low-level net-VE services on behalf of the application. The collection of federates, along with their associated RTI instances, is termed a federation. Simulators in the federation send and receive state information via calls to and from the RTI [S2].

6.2.4 – AVANGO

Avango [T2] is a Distributed Virtual Environment (DVE) developed by the German National Research Center for Information Technology (GMD). It is ostensibly an object oriented framework built atop of Performer for constructing networked virtual environments (Figure 11).
The major subsystems (interaction, graphics, networking) in a VR application are integrated into one. It provides a replicated scene graph across the network. The objects can be instantiated as either local or distributed (public or private). It defines two categories of object:

1- ***Nodes*** – define scene graph elements for rendering;
2- ***Sensors*** – import external device data into the application.

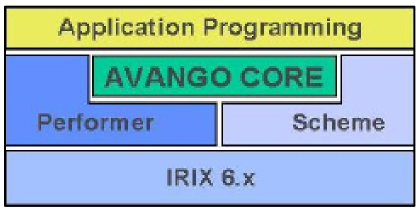


Figure 11: AVANGO model.

Avango is built on a dataflow model using fields within the objects to support a generic streaming interface. The dataflow graph defines the behavior of the objects in the world. Avango has a C++ API and a binding to an interpreted language, Scheme. Typically complex and performance critical functionality is implemented in C++. From then on, the application is implemented using Scheme scripts. Avango uses a process group model. Each group member is guaranteed to receive the delivery of network messages in exactly the same order. In addition, when a new member joins the group, all communication is suspended until the new member's state is updated. This guarantees state consistency.

6.2.5 – DIVE

DIVE (Distributed Interactive Virtual Environment) [D1] is developed by the Swedish Institute of Computer Science (Figure 12). The development effort concentrated primarily on the distribution component of the virtual environment. Certainly, in its early days DIVE emerged as an important example of how to develop distributed VR system. The distribution element is conceptually quite simple. Shared memory is provided over a network and the system controls the sending of signals to processes. DIVE implements a distributed fully replicated, dynamic database. It has the capability to add new objects and modify the existing databases in a reliable and consistent fashion. Reliable multicast protocols and concurrency control are employed through a distributed locking mechanism to facilitate database updates.

Figure 12: DIVE
from Dive website [D1].

6.2.6 – MASSIVE

MASSIVE [GP1] is an object oriented system developed at the University of Nottingham to support multi-user collaborative virtual environments. MASSIVE is based on a ‘spatial model’ of interaction [BB1]. It is mainly developed as an academic research tool, for researchers investigating online collaboration in virtual environments. The work is focused on promoting awareness of others in the virtual environment.
The system supports interaction with the virtual environments via text, 2D and 3D graphical interfaces, and has support for real-time audio, allowing users to communicate with one another using speech. The emphasis is on user-to-user interaction, rather than a user's interaction with objects in the environment. MASSIVE uses a combination of peer-to-peer and client/server processes. It employs multicasting as a collaboration baseline.

6.2.7 – MOVE

MOVE [GM1] is a CVE constructed on top of a component groupware framework, where users (avatars) can interact with each other or with shared artifacts. The proposal of the authors is to provide an extensible infrastructure offering a set of collaborative services in a seamless way. At the conceptual level, it provides essential collaborative services: shared sessions, support for synchronous and asynchronous components, security, coordination and a server-sided awareness infrastructure. At the architectural level, the framework is constructed on top of a middleware integration platform and uses high performance publish/subscribe notification services.

6.3 – Collaborative Engineering Environments

6.3.1 – SEVY

Sevy et al. [SZ1] proposes the creation of an integrated system called Collaborative Design Studio to enhance the design engineering process through the integration of a Computer-Aided Design and engineering tools (CAD/CAE), a MMCS, and archiving functions.

6.3.2 – DDRIVE

DDRIVE [DH1] is a system developed at HRL Laboratories at General Motors Research & Development Center, for Distributed Design Review in VEs. At the center of the software architecture is the Human Integrating Virtual Environment (HIVE), a collaboration infrastructure and toolset to support research and development of multi-user, geographically distributed, 2D/3D shared applications.

6.3.3 – SAVE

SAVE [HH1], Safety Virtual Environment, is a CVE based safety training system for dangerous and hazardous facilities. It is a multi purpose VR software system that is mainly intended for employee training. It provides a framework and software system for a variety of training scenarios using VR technology. Each virtual training scenario comprises a scene in which the trainee can move freely and interact with objects like pumps, valves, and other control devices. The SAVE comprises four major parts: visual simulation; the motion simulator; instructor control and supervision; and the scenario builder.

6.4 – CAD Visualization Tools for Oil&Gas

Due to the high cost of operating an offshore platform, it is imperative that the maintenance system does not fail. The availability of oil&gas production units is largely dependant upon the reliability and maintainability of equipment, which in turn is influenced by human operational and maintenance activities.
Defining the best maintenance choices is a constant concern of platform operators in order to uphold the highest level of safety and controlled maintenance costs. Decisions must be based on functional needs which are identified by considering the importance of preventive maintenance which represents an average of two thirds of the global maintenance costs.

6.4.1 – Walkinside

Walkinside™ [WI1], is a visualization tool for viewing complex 3D CAD models in real time. Walkinside™ is useful for project reviews, simulation facilities (VR centers), facility maintenance programs, safety and security training exercises. By using in-house the VR model that perfectly mimics the environment, the technicians can walk inside everywhere they want to and access all critical parts before joining the offshore platform (Figure 13).

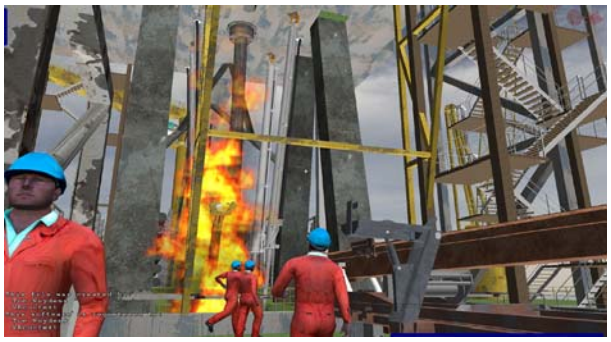
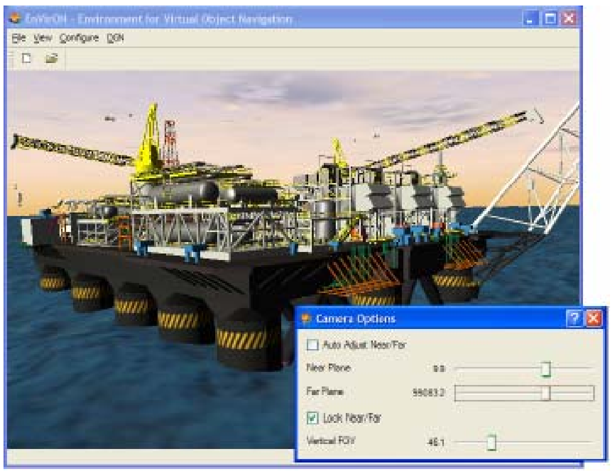


Figure 13: Safety and reliability simulation from Walkinside website [W3].

6.4.2 – ENVIRON

ENVIRON (ENvironment for VIRtual Objects Navigation) [RC1], is a tool developed to facilitate the use of CAD models in VR applications. It is a system composed of a 3D environment for real time model visualization, and exportation plugins, which translate model data from other applications into a format that can be understood by ENVIRON (Figure 14). This allows ENVIRON to view and interact with different kinds of 3D data, as long as they are in a format which ENVIRON may import or exported. Currently there are DGN [B1] and a 3ds Max [A1] exporter developed, respectively, as MicroStation and 3ds Max plugins. The goal of MicroStation plugin is not only to convert DGN files into a graphic format that enables the real time interaction and navigation in the CAD model, but also to recover and export the semantic information associated to the CAD objects. In parallel, the 3dsmax plugin enables the use of more photo-realistic models, not necessarily generated by a CAD tool.



Auto Adjust Near/Far
Near Plane 9.9
Far Plane 99083.2
Lock Near/Far
Vertical FOV 46.1

Figure 14: ENVIRON screenshot.

To enable the use of ENVIRON in VR environments with different kinds of devices, we developed an independent tool, called ViRAL (Virtual Reality Abstraction Layer), which may also be used with other VR applications. ViRAL is a tool designed to facilitate the development of VR applications. Applications developed with ViRAL are device independent, because ViRAL abstracts the context in which the application will be executed. The application doesn't have to know, for example, in how many windows, with how many users, or with which devices it is going to function. All these unknowns are configurations defined by the application operator, making it possible to execute the same application in different VR systems without the necessity to modify the application. ViRAL has a graphical user interface used to configure all devices, displays, projections and scenes.
An important feature for a VR system, when working with CAD generated models, is the ability to move, rotate and scale objects during the visualization, and also to measure distances. This is interesting for various purposes, like joining different models in a scene, testing the placement of equipment on a plant, or permitting the visualization of hidden portions of the model (Figure 15).

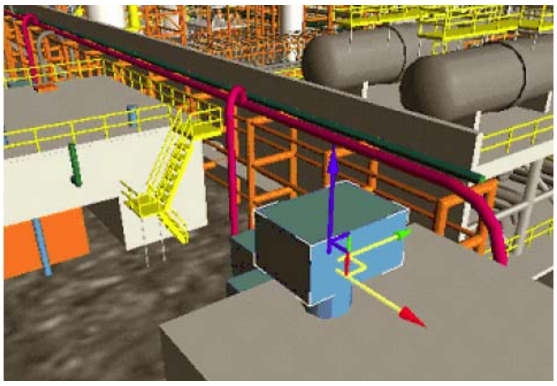


Figure 15: Moving an object in ENVIRON using a gizmo object.

7- Service-Oriented Architecture

Service-Oriented Architecture (SOA) is a style of architecting software systems by packaging functionalities as services that can be invoked by any service requester [HK1], [O1]. An SOA typically implies a *loose coupling* between modules. Wrapping a well-defined service invocation interface around a functional module hides the details of the module implementation from other service requesters. This enables *software reuse* and also means that changes to a module's implementation are localized and do not affect other modules as long as the service interface is unchanged. Once services in SOA are *loosely coupled*, applications that use these services tend to scale easily because there are few dependencies between the requesting application and the services it uses. The adoption of an SOA will produce a dramatic *reduction of technology development costs* by leveraging functions already built into legacy systems, by reusing services developed for other process, and by simplifying maintenance and support through elimination of redundant, siloed applications. Indeed SOA architectures are becoming a popular and useful means of leveraging Internet technologies to improve business processes in the oil&gas industry nowadays [GF1], [SB1].

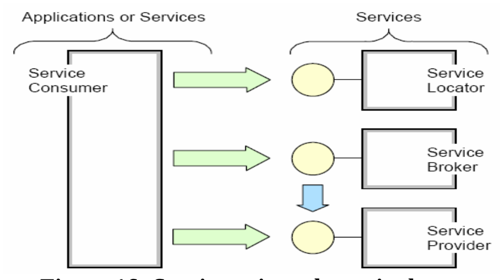


Figure 16: Service-oriented terminology.
from IBM RedBooks [EA1]

In service-oriented design a service is generally implemented as a course-grained, discoverable software entity that exists as a single instance and interacts with applications and other services through a loosely-coupled, message-based communication model. The following definitions comprise important service-oriented terminology (Figure 16):

1- ***Services***: logical entities, with contracts defined by one or more published interfaces.
2- ***Service provider:*** network-addressable software entity that implements a service specification. Accepts and executes requests from consumers. It publishes its services and interface contract to the service registry so that service consumer can discover and access.
3- ***Service consumer (or requestor):*** an application, a software module or another service that requires a service from a service provider. It initiates the enquiry of the service in the registry, binds to the service over a transport, and executes the service function. The service consumer executes the service according to the interface contract.
4- ***Service locator:*** a specific kind of service provider that acts as a registry and allows for the lookup of service provider interfaces and service locations.
5- ***Service broker:*** a specific kind of service provider that can pass on service requests to one or more additional service providers.
6- ***Service registry:*** the enabler for service discovery. It contains a repository of available services and allows for the lookup of service provider interfaces to interested service consumers.

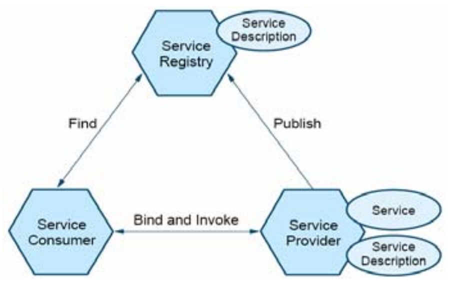


Figure 17: Collaborations in SOA.
from IBM RedBooks [EA1]

SOA constitutes an approach for building distributed systems that deliver application functionality as services to either end-user applications or other services. The collaborations in SOA follow the ***"find, bind and invoke"*** paradigm (Figure 17) where a *service consumer* performs dynamic *service location* by querying the *registry* for a *service* that matches its criteria. If the service exists, the *registry* provides the consumer with the interface contract and the *endpoint* address for the *service.*

The ***"find, bind and invoke"*** paradigm presents some drawbacks. First, the point-to-point nature of interaction between services means that service consumers often need to be modified whenever the service provider interface changes. This is often not a problem on a small scale, but in large enterprises it could mean changes to many client applications. It can also become increasingly difficult to make such changes to legacy clients. Second, it can lead to a fragile and inflexible architecture when a large number of service consumers and providers communicate using point-to-point "spaghetti" style connections. Last, every new deployed service requires that each *service consumer* has a suitable protocol adapter for that new *service provider*. Having to deploy multiple protocol adapters across many client applications adds to cost and maintainability issues.

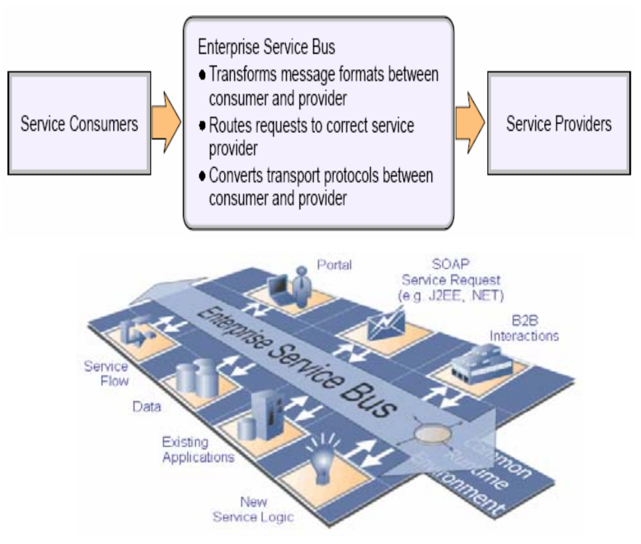

Service Consumers
Enterprise Service Bus
• Transforms message formats between consumer and provider
• Routes requests to correct service provider
• Converts transport protocols between consumer and provider
Service Providers
Portal
SOAP Service Request (e.g. J2EE, .NET)
B2B Interactions
Enterprise Service Bus
Service Flow
Data
Existing Applications
New Service Logic
Common Runtime Environment

Figure 18: ESB Conceptual model.
from IBM RedBooks [EA1]

7.2 – Enterprise Service Bus

An Enterprise Service Bus (ESB) is a pattern of middleware that unifies and connects services, applications and resources within a business (Figure 18). ESB is a platform built on the principles of SOA and other open standards to help applications integrate seamlessly. Put another way, it is the framework within which the capabilities of a business' application are made available for reuse by other applications throughout the organization and beyond. The ESB is not a new software product, it's just a new way of looking at how to integrate applications, coordinate distributed resources and manipulate information. Unlike previous approaches for connecting distributed applications, such as RPC or distributed objects, the ESB pattern enables the connection of software running in parallel on different platforms, written in different languages and using different programming models.

A basic ESB provides a messaging infrastructure along with basic transformations and routing. It mainly uses open standards like web services enabling application to talk. ESB is a centralized, scalable, fault-tolerant, service-messaging framework that:

1- Provides a transparent means for communicating with heterogeneous services over a diverse set of message protocols.
2- Provides a shared messaging layer by which enterprise engineering applications, services, and components can connect and communicate.
3- Can transmit messages synchronously or asynchronously to service endpoints and intelligently transform and secure the message content to meet the requirements of each service endpoint.
4- Provides sophisticated error recovery, allowing for failed message delivery, scalability problems, duplicate messages, network failure, etc.

The main aim of the Enterprise Service Bus is to provide virtualization of the enterprise resources, allowing the business logic of the enterprise to be developed and managed independently of the infrastructure, network, and provision of those business services. Resources in the ESB are modeled as services that offer one or more business operations. Implementing an Enterprise Service Bus requires an integrated set of middleware services that support the following architecture styles:

1- ***Service-oriented architectures***, where distributed applications are composed of granular re-usable services with well-defined, published and standards-compliant interfaces.
2- ***Message-driven architectures***, where applications send messages through the ESB to receiving applications.
3- ***Event-driven architectures***, where applications generate and consume messages independently of one another

Concerning the difficulties in building an effective CEE, mentioned in the Introduction, the use of an Enterprise Service Bus (ESB) will furnish the entire system infrastructure to enable *distributed execution of applications* as well as *system interoperability*.

8- CEE Architecture and Application Scenarios

Our proposed CEE has component-based architecture in order to facilitate the reuse of elements. The architecture of the CEE uses a ScWfMS as its kernel while the MMCS, CVE and the other components are seamlessly accessed through the ESB according to the collaborative necessities of the teamworkers.

When the service-oriented approach is adopted for designing the CEE, every component, regardless of its functionality, resource requirements, language of implementation, etc., provides a well-defined *service interface* that can be used by any other component in the environment. The service abstraction provides a uniform way to mask a variety of underlying data sources (real-time production data, historical data, model parameters, reports, etc.) and functionalities (simulators, optimizers, sensors, actuators, etc.). A Workflow, actually, in our context, a Scientific Workflow, is composed by coupling service interfaces in the desired order. These workflows specifications are created through a graphical or textual front end and the actual service calls are generated automatically and have their execution controlled by the workflow engine (Figure 19).

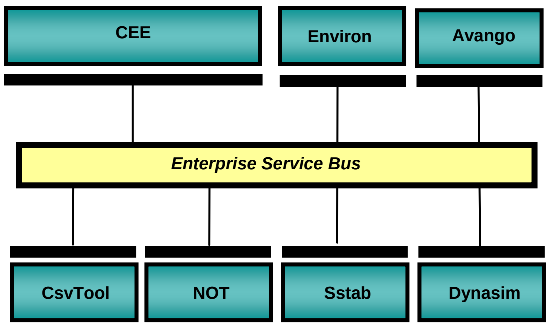


Figure 19: CEE architecture.

The integration of the WfMS and MMCS follows the same approach suggested by [WP1]. We have developed a multiplatform videoconferencing tool, CSVTool [PR1] (Collaboration Supported by Video). The use of a custom tool allows a tight integration of this service into the collaborative system, with no duplication of session-management functionalities, and the direct control of audio and video streams according to the coordination policies defined. CSVTool is implemented with JMF (Java Media Framework^{TM} from Sun), which employs a high-level source code and abstracts codecs and transmission protocol details. It was designed with the goal of providing integrated multimedia communication to collaborative applications, but can also be used as an independent videoconferencing tool.

Besides the transmission of audio and video on multi-participant, multi-platform sessions, CSVTool provides some interesting features:

1- the video stream sent by each participant can be switched from the image captured by the camera to the captured screen, to allow the use of video for remote display of the interface operation or for the presentation of other contents on the screen and for consistency checks;
2- a textual chat tool, which is providential in some situations (for instance, when somebody is having problems with capture devices);
3- snapshots, useful for documenting the work session.

All the consistency, adequacy and compatibility of the shared data among its users should be done by the kernel of the CEE, in order to avoid, or at least to diminish, non useful iterations during the project's life cycle. The ability of reusing partial workflows, which were previously stored in the system with some guidelines, provide an optimized usage of the available computational resources and also a better control of the costs and time scheduling.

Next as a proof of concept we present a scenario for the use of the proposed CEE to implement a collaborative visualization of CAD models.

8.1 – Collaborative CAD Visualization

The implementation of a Collaborative CAD Visualization system is straightforward in CEE, since the group communication mechanism, session management and concurrency control can be easily implemented as service providers for the ESB. Those services compose the Collaboration Bus service previously mentioned in the Introduction. The transparency of data location can also be achieved provided that a data management service is supplied. The ScWfMS contains the “Engineering Process” definition for the collaborative visualization and has all the information about the data and the users that will participate in the collaborative session. Bellow we show a scheme of the solution implemented with the CEE (Figure 20). If necessary the users can start a videoconference to help them solve possible conflicts.

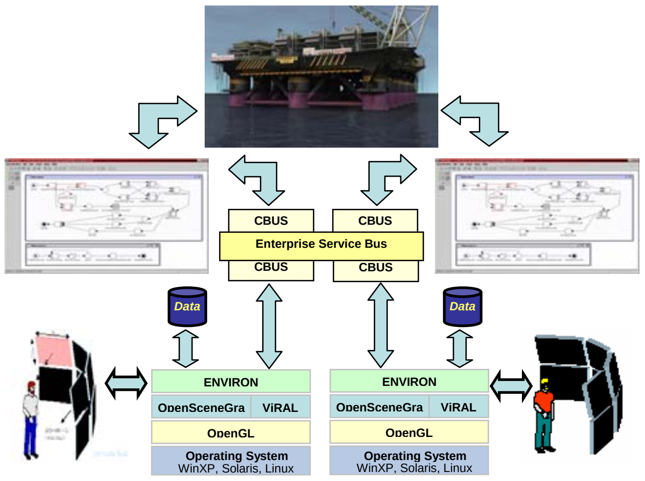


Figure 20: Collaborative CAD Visualization CEE session.

9- Conclusions

This paper presented an SOA of the CEE that we are currently developing. As a proof of concept we are developing a prototype that will be used by the Offshore Engineering group at Petrobras.
Through the use of the CEE we have build an effective collaborative environment that will allow users to easily mitigate their problems that usually happen during the execution of large and complex engineering projects. We also intend to improve the effectiveness of the use of VR technology once that it will be easily integrated into the workflow of the team workers. We conclude that our CEE constitutes an effective Collaborative Problem Solving tool for the engineers in our company.
Although this work is focused on a solution for Offshore Engineering projects, we believe that the proposed CEE could also be used in other areas as well.

Acknowledgments

The authors would like to thank Petrobras and Tecgraf/PUC-Rio for the expressive support. Ismael H. F. Santos works at CENPES, Petrobras‘s Research Center.

10- References

[A1] AUTODESK. 2005. 3ds Max. http://www.autodesk.com.

[AH1] W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, and A.P. Barros. Workflow Patterns. Distributed and Parallel Databases, 14(1), pp 5–51, 2003.

[B1] Bentley, 1995, MicroStation Ref. Guide, ch 18. Bentley Systems Incorporated: Intergraph Standard File Formats.

[BB1] S. Benford, A. Bullock, N. Cook, P. Harvey, R. Ingram, O.K. Lee. A spatial model of cooperation for virtual worlds. In: Informatique'93: Interface to Real & Virtual Worlds, pp 445-456, University of Nottingham, 1993.

[BH1] R. Bentley, J.A. Hughes, D. Randall, T. Rodden, P. Sawyer, D. Shapiro, I. Sommerville. Ethnographically-informed systems design for air traffic control. In The Proc. of the ACM Conf. on CSCW’92, Ontario, CA, 1992.

[CG1] L.C.G. Coelho, M. Gattass, L.H. Figueiredo. Intersecting and trimming parametric meshes on finite element shells. Intern Journal Num. Methods Eng,47,pp 777-800, 2000.

[CJ1] L.C.G. Coelho, C.G. Jordani, M.C. Oliveira and I. Q. Masetti, Equilibrium, Ballast Control and Free-Surface Effect Computations Using The Sstab System. 8 th Int. Conf. Stability of Ships and Ocean Vehicles - Stab, pp 377-388, 2003.

[CK1] L.C.G. Coelho, K. Nishimoto, and I.Q. Masetti, Dynamic Simulation of Anchoring Systems Using Computer Graphics. OMAE Conference, 2001.

[CS1] C. Cruz-Neira, D. J. Sandin, T.A. DeFanti, R.V. Kenyon, and J.C. Hart. The CAVE: Audio visual experience automatic virtual environment. Comm of ACM, 35-6, 1992.

[D1] Swedish Institute of Computer Science (SICS): DIVE – A Toolkit for Distributed VR Applications http://www.sics.se/dce/dive.

[DB1] J. Dunbar, W. Brooks, NASA Problem Solving Environments:Distributed Collaborative VirtualWindTunnel, http://www.nas.nasa.gov/Research/Tasks/probsolvingtasks.html

[DB2] P. Dourish and V. Bellotti. Awareness and coordination in shared workspaces. In Proceedings of the ACM Conference on Computer-Supported Cooperative Work, CSCW'92, ACM Press, NY, pp 107-114, 1992.

[DH1] M. Daily, M. Howard, J. Jerald, C. Lee, K. Martin, D. McInnes, P. Tinker. Distributed Design Review in Virtual Environments. In the Proceedings of the Third ACM International Conf. CVE’00, San Francisco, CA, USA, 2000.

[DR1] PTC Division Reality. http://www.ptc.com/solutions/ industry/index.htm

[DW1] J. Dahmann, R. Weatherly, F. Kuhl. Creating Computer Simulation Systems: An Introduction to the High Level Architecture. Upper Saddle River, Prentice-Hall, 1999.

[E1] C.A. Ellis, Workflow technology. In Computer Supported Co-operative Work, M. Beaudouin-Lafon, Ed., vol. 7 of Trends in Soft Series, John Wiley&Sons, pp 29-54, Chichester, 1999.

[EA1] M. Endrei, J. Ang, A. Arsanjani, et al. Patterns: Service-Oriented Architecture and Web Services, IBM RedBooks, April 2004, http://www.ibm.com/redbooks.

[EG1] C.A. Ellis, S.J. Gibbs, and G.L. Rein. Groupware - Some Issues and Experiences. Communications of the ACM, Vol. 34, No. 1, pp. 38-58, 1991.

[EN1] C.A. Ellis, G. J. Nutt. Workflow: The Process Spectrum. In NSF Workshop on Workflow and Process Autom. in Info. Systems, Athens, GE, 1996.

[EW1] e-Science Workflow Services Workshop, Edinburgh, Scotland, 2003,http://www.nesc.ac.uk/events/303/index.html.

[EX1] eXo Plataform. Portal and Enterprise Content Management Integrated System http://docs.exoplatform.org/ exo-documents/exo.site/index.html

[FB1] B. Fröhlich, S. Barrass, B. Zehner, J. Plate, M.Goebel. Exploring GeoScience Data in Virtual Environments. In Proc. IEEE Visualization, 1999.

[FK1] I. Foster, C. Kesselman, J. Nick, and S. Tuecke: "The physiology of the Grid: An Open Grid Services Architecture for distributed systems integration", Technical report, Open Grid Services Architecture WG, Global Grid Forum, 2002.

[FR1] H. Fuks, A.B. Raposo, M.A. Gerosa, and C.J.P. Lucena,. Applying the 3C Model to Groupware Development. International Journal of Cooperative Information Systems (IJCIS), v.14, n.2-3, Jun-Sep 2005, World Scientific, ISSN 0218-8430, pp. 299-328, 2005.

[G1]The gOcad consortium. GoCad site. http://www.gocad.org/ www/consortium/index.xhtml.

[GF1] R. Gregovic, R. Foreman, D. Forrester, and J. Carroll. A common approach to accessing real-time operations data - Introducing SOA to E&P, SPE ATCE 2005.

[GL1] G. Goebbels, V. Lalioti, M. Göbel. Design and Evaluation of Team Work in Distrib. Collaborative Virtual Environments. In The Proc. of the ACM Symposium on VRST'03, Osaka, Japan, 2003.

[GM1] P. García, O. Montal, C. Pairot, R. Rallo, G.A. Skarmeta. MOVE: Component Groupware Foundations for CVE. In The Proc. 4th ACM Intern. Conf on CVE,GE, 2002.

[GP1] C. Greenhalgh, J. Pubrick, D.Snowdon. Inside MASSIVE-3: Flexible Support for Data Consistency and World Structuring. In: ACM Conf Collaborative VEs, CVE2000, San Francisco, USA, p. 119-127, 2000.

[H1] Defense Modeling and Simulation Office High Level Arquitecture. https://www.dmso.mil/public/transition/hla/

[HG1] E.N. Houstis, E. Gallopoulos, R. Bramley and J.R. Rice. Problem-Solving Environments for Computational Science. IEEE Computational Science and Engineering, 4(3): 18-21, 1997.

[HH1] M. Haller, R. Holm, J. Volkert, R. Wagner. A VR based safety training in a petroleum refinery. In The 20th Annual Conf. of the European Association of CG, Eurographics'99, Milano, Italy, 1999.

[HK1] R. High, S. Kindler, S. Graham. IBM SOA Foundation: An architectural introduction and overview. IBM developer Works.http://www-128.ibm.com/developerworks/ webservices/library/ws-soa-whitepaper/

[IT1] E.A. Isaacs and J.C. Tang, What Video Can and Cannot Do for Collaboration: A Case Study, Multimedia Systems, Vol. 2, 63-73, 1994.

[J1] G. Joeris, Cooperative and Integrated Workflow and Document Management for Engineering Applications. In 8th Intern. Workshop on Database and Expert Systems Applications, DEXA, 1997.

[K1] Kepler Project. Open-source Scientifc Workflow System. http://kepler-project.org/

[KB1] H. Klie, W. Bangerth, X. Gai, M.F. Wheeler, P.L. Stoffa, M. Sen, M. Parashar, U. Catalyurek, J. Saltz and T. Kurc. Models, Methods and Middleware for Grid-enabled Multiphysics Oil Reservoir Management. Engineering with Computers, Springer Verlag, 2006.

[KM1] A. Kapolka, D. McGregor, and M. Capps. A Unified Component Framework for Dynamically Extensible VEs. In The Proc. 4th ACM Intern. Conf. on CVE, Germany, 2002.

[L1] J. Locke. An Introduction to the Internet Networking Environment and SIMNET/DIS. Computer Science Department, Naval Postgraduate School, USA, 1993.

[LA1] B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger-Frank, M. Jones, E. Lee, J. Tao, Y. Zhao. Scientific Workflow Management and the Kepler System, Concurrency and Computation: Practice & Experience, Special Issue on Scientific Workflows, 17(1):1– 26, 2005.

[LG1] LandMark Geological and Geophysical Technologies. http://www.lgc.com/landmark/integrated+solutions/index.htm.

[LJ1] J. Leigh, A. Johnson, M. Brown, D. Sandin, T. DeFanti. Visualization in teleimmersive environments. IEEE Computer, vol 32(12), 1999.

[LL1] D. Lee, M. Lim and S. Han. ATLAS - A Scalable Network Framework for Distributed Virtual Environments. In The The Proc. 4th ACM Intern. Conf. on CVE, GE, 2002.

[LM1] D. Lee, M. Lim M. and S. Han. ATLAS - A Scalable Network Framework for Distributed Virtual Environments. In The Proc. 4th ACM Intern. Conf. on CVE, GE, 2002.

[M1] W.E. Mackay. Media Spaces: environments for informal multimedia interaction. In Computer Supported Co-operative Work, M. Beaudouin-Lafon, Ed., vol. 7 of Trends in Software. John Wiley & Sons, 55-82, 1999.

[MV1] C.B. Medeiros. G. Vossen, M. Weske, WASA: A Workflow-Based Architecture to Support Scientific Database Applications. DEXA, pp 574-583, 1995.

[MZ1] M.R. Macedonia, M.J. Zyda, D.R. Pratt, D.P. Brutzman, P.T Barham. Exploiting Reality with Multicast groups: A Network Architecture for Large-Scale Virtual Environments. IEEE Computer Graphics and Applications, 15(5), p. 38-45, 1995.

[O1] E. Ort, SOA Architecture and Web Services: Concepts, Technologies, and Tools. Sun Developer Network, http://java.sun.com/developer/technicalArticles/ WebServices/ soa2/

[PM1] S.G. Parker, M. Miller, C.D. Hansen, and C.R. Johnson. An integrated problem solving environment: The SCIRun computational steering system. In 31st Hawaii International Conference on System Sciences (HICSS-31), p 147-156, 1998.

[PR1] C.T. Pozzer, A.B. Raposo, I.H.F. Santos, J.L.E. Campos, L.P. REIS. CSVTool - A Tool for Video-Based Collaboration. IX Simpósio Brasileiro de Sistemas Multimídia e Web - WebMidia 2003, p.353-367. Salvador, Brazil. 2003.

[RC1] A.B. Raposo, E.T.L. Corseuil, G.N. Wagner, I.H.F. Santos,, M. Gattass. Towards the Use of CAD Models in VR Applications. ACM SIGGRAPH International Conference on Virtual-Reality Continuum and its Applications in Industry, VRCIA 2006, June 2006, CUHK, Hong Kong.

[RR1] E.E.R. Russo, A.B. Raposo, T. Fernando, M. Gattass. Workspace Challenges for the Oil & Gas Exploration & Production Industry. In: 4TH Conference of Construction Applications of Virtual Reality - CONVR 2004, p. 145-150, Lisboa, Portugal. 2004.

[RS1] W. Reinhard, J. Schweitzer, G. Völksen. CSCW Tools: Concepts and Architectures. In The IEEE Computer, vol 27(5), pp 28-36, 1994.

[S1] S.K. Singhal. Effective Remote Modeling in Large-Scale Distributed Simulation and Visualization Environments. PhD Thesis, Dep of Computer Science, Stanford University, 1996.

[SB1] R. Soma, A. Bakshi, A. Orangi, V.K. Prasanna and W.D. Sie. Service-Oriented Data Composition Architecture for Integrated Asset Management, SPE ATCE 2006.

[SC1] C.L.N. Santos, G.G. Cunha, L. Landau. Use of VRML in Collaborative Simulations for the Petroleum Industry. In Procs of 34th Annual Simulation Symposium, SS01, 2001.

[SD1] Scientific Data Management Framework Workshop, Argonne National Lab, USA. August 2003 http://sdm.lbl.gov/~arie/sdm/SDM.Framework.wshp.html

[SG1] I.H.F. Santos, M. Goebel, A.B. Raposo, M. Gattass. A Multimedia Workflow-Based Collaborative Engineering Environment for Oil & Gas Industry. ACM SIGGRAPH International Conference on Virtual-Reality Continuum and its Applications in Industry, VRCAI 2004, p 112-119, June 2004, Singapore, SI.

[SI1]Schlumberger Inside Reality website. http://www.slb.com/content/services/software/virtual/

[SK1] S.K. Singhal, M. Zyda. Networked Virtual Environments: Design and Implementation. New York: ACM Press, 1999.

[SR1] I.H.F. Santos, A.B. Raposo, and M. Gattass, Finding Solutions for Effective Collaboration in a Heterogeneous Industrial Scenario. In The 7th Inter. Conf. Comp. Supp. Coop. Work in Design, CSWID02, 2002.

[SV1] I.H.F. Santos, C. Valle, A.B. Raposo,, M. Gattass. A Multimedia Workflow-based Collaborative Engineering Environment: Integrating an Adaptive Workflow System with a Multimedia Collaboration System and a Collaborative Virtual Environment for Petroleum Engineering. 6th International Conference on Enterprise Information Systems. ICEIS 2004, Vol. 5, p.259-262. March 2004. Porto, Pt.

[SZ1] J. Sevy, V. Zaychik, T.T. Hewett, W.C. Regli. Evaluating Collabartive Engineering Studios. In The 9th Proc. of WET ICE'2000, USA, 2000.

[T1] D. Tietze. A Framework for Developing Component-based Cooperative Applications, Ph.D. Dissertation, Technischen Universitat Darmstadt, 2001.

[T2] H. Tramberend. Avocado: A Distributed Virtual Reality Framework. In The Proc of the IEEE Virtual Reality, 1999, http://www.avango.org

[W1] Workflow Reference Model, 1995. In WfMC - Workflow Management Coalitition. Document No. TC00-1003, Issue 1.1, 1995.

[W2] Workflow Management Coalition, http://www.wfmc.org Workflow Management Coalition Terminology and Glossary (WFMCTC-1011). Technical report, Workflow Management Coalition, Brussels, 1996.

[WI1] Walkinside. A visualization tool for viewing complex 3D CAD models in real time. http://www.walkinside.com/index.htm

[WP1] M. Weber, G. Partsch,, J. H. Schweitzer, Integrating Synchronous Multimedia Collaboration into WfMS, ACM SIGGROUP, Phoenix, USA,1997.

[WV1] M. Weske, G. Vossen, C.B. Medeiros, F. Pires. Workflow Management in Geoprocessing Apps. In Tech. Report No.04-98-I, Univ.Muenster, GE, 1998.

[YB1] J.Yu, R. Buyya. A Taxonomy of Scientific Workflow Systems for Grid Computing, SIGMOD, Vol.34, No.3, 2005.