PONTIFÍCIA UNIVERSIDADE CATÓLICA
DO RIO DE JANEIRO

Enio Emanuel Ramos Russo

Um Metamodelo para Configuração de Espaços de Trabalho Virtuais Colaborativos: Aplicação no Gerenciamento de Desastres de Estruturas *Offshore* de Óleo e Gás

Tese de Doutorado

Tese apresentada como requisito parcial para obtenção do título de Doutor pelo Programa de Pós-Graduação em Informática da PUC-Rio.

Orientador: Prof. Marcelo Gattass
Co-orientador: Prof. Terrence Peter Fernando
Co-orientador: Prof. Alberto Barbosa Raposo

Rio de Janeiro, 27 de março de 2006

Enio Emanuel Ramos Russo

Um Metamodelo para Configuração de Espaços de Trabalho Virtuais Colaborativos: Aplicação no Gerenciamento de Desastres de Estruturas *Offshore* de Óleo e Gás

Tese apresentada como requisito parcial para obtenção do grau de Doutor pelo Programa de Pós-graduação em Informática do Departamento de Informática do Centro Técnico Científico da PUC-Rio. Aprovada pela Comissão Examinadora abaixo assinada.

Prof. Marcelo Gattass
Orientador
Departamento de Informática – PUC-Rio

Prof. Terrence Peter Fernando
School of Construction and Property Management - University of Salford

Prof. Alberto Barbosa Raposo
Departamento de Informática – PUC-Rio

Prof. Carlos José Pereira de Lucena
Departamento de Informática – PUC-Rio

Prof. Hugo Fuks
Departamento de Informática – PUC-Rio

Dr. Álvaro Maia da Costa
Centro de Pesquisas e Desenvolvimento Leopoldo Miguez de Mello – Petrobras

Prof[a.] Flávia Maria Santoro
Departamento de Informática Aplicada – UNIRIO

Prof[a.] Renata Mendes de Araujo
Departamento de Informática Aplicada – UNIRIO

Prof. José Eugenio Leal
Coordenador Setorial do Centro Técnico Científico

Rio de Janeiro, 27 de março de 2006

Enio Emanuel Ramos Russo

Mestre em Informática pela PUC-Rio em 1988 e Mestre em Matemática Aplicada pelo Instituto de Matemática Pura e Aplicada (IMPA) em 1988, graduou-se em Engenharia Eletrônica pelo Instituto Militar de Engenharia (IME) em 1983. Foi Gerente de Informação do CENPES da Petrobras e co-fundador do Tecgraf da PUC-Rio. Atualmente realiza pesquisa em Computação Gráfica e Realidade Virtual.

Ficha Catalográfica

Russo, Enio Emanuel Ramos

Um metamodelo para configuração de espaços de trabalho virtuais colaborativos : aplicação no gerenciamento de desastres de estruturas offshore de óleo e gás / Enio Emanuel Ramos Russo ; orientadores: Marcelo Gattass, Terrence Peter Fernando, Alberto Barbosa Raposo. Rio de Janeiro : PUC, Departamento de Informática, 2006.

198 f. : il. ; 30 cm

Tese (Doutorado) – Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática.

Inclui referências bibliográficas.

1. Informática – Teses. 2. Colaboração. 3. Metamodelo. 4. Espaços virtuais. 5. Óleo e gás. I. Gattass, Marcelo. II. Fernando, Terrence Peter. III. Raposo, Alberto Barbosa. IV. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. V. Título.

CDD: 004

À minha maravilhosa esposa Márcia.

Agradecimentos

A Deus, por ter me dado forças e saúde para superar os momentos mais difíceis.

À minha maravilhosa esposa Márcia, que sempre me incentivou e me apoiou, com muito amor e paciência ao longo desses quatro anos.

Aos meus pais, Enio e Dilce, e aos meus avós, Danilo (saudades) e Eddy, pelo carinho e exemplo de vida que me deram, estando sempre presentes ao meu lado, e às minhas irmãs, Patrícia e Cláudia, que sempre torceram por mim.

A toda a família da minha esposa, começando pelo meu querido e falecido sogro Jorge (que Deus o tenha), minha sogra Ana Maria e meus cunhados Marcelo e Marília, por todo o carinho e incentivo que me deram.

Aos meus orientadores Marcelo Gattass, Terrence Fernando e Alberto Raposo, pelas mãos firmes que me apontaram com segurança o melhor caminho a seguir. A eles, verdadeiros amigos, o meu muito obrigado!

Ao Gerente e amigo Álvaro Maia, inspirador e grande incentivador desta Tese!

Ao Prof. Lucena, meu professor de tão longa data, por todo o seu apoio, idéias e interesse em estreitar as relações com a Universidade de Salford.

Aos meus amigos Börje e Pozzer, que tanto me ajudaram na implementação.

Aos amigos Prof. Bruno Feijó, Flávio Szenberg e Rodrigo Toledo, pelas palavras animadoras para que eu prosseguisse no doutorado.

Aos muitos que contribuíram com idéias para esta Tese, Profs. Markus Endler, Renato Cerqueira e Hugo Fuks, Luiz Coelho, Antonio Nascimento, Ivan Menezes, Carlos Cassino, João Luiz, Ernesto Fleck e Eduardo Thadeu da PUC, Prof[as.] Renata Araujo e Flávia Santoro da UNIRIO, Isaias Masetti, Mauro Costa, Carlos Jordani, Luiz Levy e Heitor Araújo, do CENPES, e Roberto de Beauclair, do IMPA.

À Petrobras, por todo o apoio e suporte financeiro, em especial aos gerentes Anelise Lara, Fernando José, Bruno Zeeman, Roberto Murilo e Luís Antônio.

Ao Tecgraf, à PUC e à Universidade de Salford, por tornar disponíveis recursos e apoiar a elaboração da Tese.

Aos meus médicos Jorge André de Segadas, José Joaquim Seabra, Armando José Pimenta, Ricardo Alvariz, Antônia Márcia Cupello, Alexandre Gripp, Milton Arantes e Américo Cardoso, responsáveis pelo restabelecimento da minha saúde.

Aos Profs. Casanova, Paulo Cezar, Luiz Henrique e Waldemar, Felipe Carvalho, Romano, Sérgio, Manuel, Luciana, Ismael, Luciano, Gustavo, Thiago, Márcio, Eduardo, Pedro, Pablo, Felipe Lobo, Ricardo, Fábio, Maurício, Aurélio, Paulo, Haroldo, Rozo, Villarim, Delio, Otacilio, Flávio, Orlando e Norman, pela grande força que me deram como amigos de todas as horas.

À Deborah, Ruth, Manu, Alex, Sandra, Claudinei, Herivelto, Paulo, Elson, Sérgio e Michelle, por todo o apoio ao longo de todo o doutorado.

E a todas as outras pessoas que também me ajudaram a realizar este trabalho.

Resumo

Russo, Enio Emanuel Ramos. **Um Metamodelo para Configuração de Espaços de Trabalho Virtuais Colaborativos: Aplicação no Gerenciamento de Desastres de Estruturas *Offshore* de Óleo e Gás.** Rio de Janeiro, 2006. 198 p. Tese de Doutorado - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Várias companhias têm criado equipes virtuais para agregar trabalhadores de diversas especialidades que estão dispersos geograficamente, aumentando a demanda por aplicações CSCW (*Computer Supported Cooperative Work*). De modo a facilitar o desenvolvimento de uma ampla gama destas aplicações colaborativas, devemos prover uma arquitetura genérica que seja adaptável a diferentes situações, tarefas e configurações de um modo flexível.

Este trabalho investiga como um ambiente de trabalho distribuído pode apoiar o gerenciamento de desastres, envolvendo equipes técnicas colaborativas distribuídas. Primeiramente, identificamos os requisitos para o espaço de trabalho distribuído, a partir dos atores envolvidos em um desastre, e analisamos os sistemas de emergência comerciais disponíveis. Em seguida, elaboramos um metamodelo de multi-perspectiva para auxiliar a configurar este espaço de trabalho virtual colaborativo. Finalmente, derivamos, a partir do metamodelo, um protótipo para o gerenciamento de desastres de estruturas *offshore* de óleo e gás e desenvolvemos uma implementação aderente ao padrão HLA (*High Level Architecture*) para este protótipo, como prova de conceito deste metamodelo.

Palavras-chave

Trabalho Colaborativo Auxiliado por Computador; Metamodelo; Espaços de Trabalho Virtuais; Óleo e Gás.

Abstract

Russo, Enio Emanuel Ramos. **A Metamodel for Configuring Collaborative Virtual Workspaces: Application in Disaster Management of Oil & Gas Offshore Structures.** Rio de Janeiro, 2006. 198 p. D.Sc. Thesis - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Many companies have been creating virtual teams that bring together geographically dispersed workers with complementary skills, increasing the demand for CSCW (Computer Supported Cooperative Work) applications. In order to facilitate the development of a wide range of these collaborative applications, we should offer a general architecture that is adaptable to different situations, tasks, and settings in a flexible way.

This work investigates how a distributed workspace environment can support disaster management, involving distributed collaborative technical teams. We first identify the requirements for the distributed workspace, from the stakeholders involved in a disaster, and analyse the commercial emergency systems available. We then elaborate a multi-perspective metamodel to support configuring this collaborative virtual workspace. Finally a prototype for oil & gas offshore structures disaster management based on our multi-perspective metamodel is derived and an HLA (High Level Architecture) compliant implementation for this prototype is developed as a proof-of-concept of the metamodel.

Keywords

Computer-Supported Cooperative Work; Metamodel; Virtual Workspaces; Oil & Gas.

Sumário

Lista de figuras

Lista de tabelas

1
Introdução

Existem sérios riscos envolvidos na operação de unidades *offshore*, ocorrendo muitos desastres que não apenas podem causar fatalidades e danos ambientais sérios como têm forte impacto nos negócios. As empresas podem sofrer prejuízos da ordem de bilhões de dólares com a perda de uma unidade *offshore*, e muito maiores do que isso em conseqüência da interrupção da produção de petróleo. A Petrobras, por exemplo, perderia US$ 700 milhões com a perda da plataforma *offshore* P-40 e mais de US$ 6 bilhões por não produzir 150.000 barris de petróleo em três anos.

Entre os piores desastres houve o do navio-tanque Exxon Valdez, no Alasca, EUA, em 1989, que acarretou custos diretos com tratamento e limpeza de 4 a 8 bilhões de dólares e 10 anos de esforços para permitir que o ecossistema fizesse a área retornar a seu estado natural. Em 1988, o desastre da plataforma de petróleo Piper Alpha, no Mar do Norte, causou 167 mortes.

A Petrobras também enfrentou dois acidentes de grande porte no início desta década. Em 2001, a P-36, a maior plataforma semi-submersível do mundo, com 40 andares de altura e pesando 31.000 toneladas, afundou matando 11 empregados e interrompendo uma produção diária de 84.000 barris de petróleo e 1,3 milhão de metros cúbicos de gás natural. Em 2002, a unidade FPSO (*Floating Production, Storage and Offloading* – Produção, Armazenamento e Descarregamento Flutuante) P-34, com uma produção diária de 35.000 barris e capacidade de armazenamento de 58.000 m^3 de óleo, pesando 62.000 toneladas, sofreu um problema de estabilidade e quase afundou, interrompendo imediatamente as operações. Desta vez, a Petrobras conseguiu salvar a unidade sem perda de vidas.

Como resultado direto desses acidentes em grande escala, as empresas de óleo e gás costumam tomar medidas em dois sentidos principais: um com o objetivo de corrigir e aprimorar os procedimentos operacionais, e um segundo que visa elaborar uma série de projetos para melhorar o nível tecnológico da empresa,

visando minimizar o risco de acidentes futuros. O processo de aprendizagem organizacional das empresas também tem melhorado o nível de proteção ao meio-ambiente e garantido maiores padrões de segurança operacional para os empregados e as instalações (Costa, 2004).

Considerando o segundo aspecto mencionado acima e a importância de minimizar os impactos dos desastres, verificamos a necessidade de desenvolver uma arquitetura de sistema capaz de permitir que as pessoas trabalhem conjuntamente como uma equipe virtual, permitindo-lhes explorar vários planos de resgate e chegar rapidamente a um consenso.

Muitas empresas têm criado equipes virtuais que congregam empregados geograficamente dispersos com capacidades complementares, o que vem aumentando a demanda por aplicações CSCW (*Computer Supported Cooperative Work* – Trabalho Cooperativo Assistido por Computador). Para tornar mais efetivo o desenvolvimento de uma vasta gama dessas aplicações colaborativas, seria necessário oferecer uma arquitetura geral adaptável a diferentes situações, tarefas e configurações, de forma flexível. A motivação para este trabalho foi portanto a necessidade de se desenvolver um espaço de trabalho virtual colaborativo para a gestão de desastres de estruturas *offshore* de óleo e gás de uma empresa global (Russo et al., 2004) – veja o Apêndice B.

1.1. Motivação

A Petrobras, uma das empresas do ramo de óleo e gás que têm procurado empregar processos e tecnologias eficientes em resposta a incidentes recentes, vem realizando ações importantes para garantir a segurança. Por exemplo, em 2000, a Petrobras lançou o Programa de Excelência em Gestão Ambiental e Segurança Operacional (PEGASO), com um investimento de 1,7 bilhão de dólares e o desenvolvimento de 4.000 projetos em 4 anos (Petrobras, 2004). Como resultado direto do acidente com a P-36, a comissão de inquérito recomendou a implementação de outro Programa de Excelência Operacional (PEO), específico para unidades de produção *offshore* (Petrobras, 2001).

Entretanto, a implementação desses processos passa por reunir um grande número de grupos e recursos diversos e geograficamente distribuídos, de modo a

se tomar decisões apropriadas em um curto espaço de tempo. Tais grupos são compostos por muitos especialistas técnicos e responsáveis pelo processo decisório, como engenheiros navais, engenheiros estruturais, analistas de *risers* e oceanógrafos, além de gerentes e diretores. Tipicamente, um grupo decisório de nível alto trabalha desde a unidade operacional, e a equipe técnica desde uma base em terra próxima do desastre, da sede da empresa e/ou de vários centros de pesquisa. Esses grupos precisam estar em comunicação constante com os operadores que estão dentro da unidade, os mergulhadores e a equipe de segurança, além de, ocasionalmente, com especialistas em trânsito para executar o plano de contingência. A principal missão da equipe técnica nessas situações consiste em apresentar rapidamente uma solução para estabilizar a unidade, por meio da execução de diversos programas de simulação que levam em conta ondas, ventos, correntes e outras forças que agem sobre a unidade. Os membros da equipe técnica também podem estar distribuídos em diversos locais, conectados a vários centros de pesquisa e à sede.

Tendo em vista todos esses requisitos, há a necessidade de se desenvolver uma arquitetura de sistema capaz de permitir que as pessoas trabalhem conjuntamente como uma equipe virtual, permitindo-lhes explorar vários planos de resgate e chegar a um consenso. O objetivo desta tese é, portanto, explorar como as tecnologias de trabalho colaborativo podem dar assistência a esse processo de tomada de decisões e formação de consenso. Desse modo, elaboramos um metamodelo de multi-perspectiva para dar suporte à configuração desse espaço de trabalho virtual colaborativo.

1.2. Metas e Objetivos

A principal meta deste trabalho consiste em investigar como um ambiente de trabalho distribuído pode assistir na gestão de desastres congregando equipes técnicas geograficamente dispersas. Especificamente, esta pesquisa enfoca um espaço de trabalho distribuído para que grupos técnicos operem como uma equipe virtual colaborativa, explorem diversas opções de simulação e comuniquem seus resultados aos responsáveis por tomar as decisões.

Essa meta será alcançada por meio dos seguintes objetivos:

- elaboração de um levantamento:
  - identificação dos requisitos do espaço de trabalho distribuído, a partir dos participantes envolvidos em um cenário de desastre;
  - análise dos sistemas de emergência comerciais disponíveis;
- elaboração de um metamodelo para configurar espaços de trabalho virtuais colaborativos;
- condução de um estudo para analisar os sistemas distribuídos mais importantes;
- definição de um ambiente de trabalho distribuído baseado nesse metamodelo para que a equipe técnica reúna os esforços de contingência.

1.3. Estrutura da Tese

A seqüência de capítulos desta tese está organizada como se segue.

No Capítulo 2 são estipulados os requisitos necessários a partir de dois estudos de caso da Petrobras: o da P-36 e o da P-34. É apresentado ainda, de forma resumida, um levantamento dos principais sistemas de emergência comerciais disponíveis, identificando-se a necessidade de se desenvolver uma arquitetura de sistema capaz de congregar os recursos distribuídos presentes em um cenário de contingência, em especial simuladores distribuídos sendo executados em sistemas de visualização de alto desempenho. Tal arquitetura deve poder oferecer comunicação síncrona entre equipamentos diferentes, tendo a co-localização virtual como uma de suas características, constituindo assim um espaço de trabalho virtual colaborativo para a gestão de desastres.

No Capítulo 3 é elaborado um metamodelo de multi-perspectiva que ajudará a definir e configurar a arquitetura do espaço de trabalho virtual colaborativo. Demonstramos também como os modelos derivados desse metamodelo podem ser reconfigurados para se adequar a novas situações, o que é particularmente importante em cenários de contingências.

No Capítulo 4 é apresentado um estudo sobre as principais tecnologias empregadas no desenvolvimento de ambientes distribuídos, a partir das abordagens de *middleware* e ambientes virtuais distribuídos específicos.

Selecionamos duas dessas tecnologias para validar nosso metamodelo: HLA (*High Level Architecture*) e InfoGrid.

No Capítulo 5, derivamos um primeiro modelo para a gestão de desastres envolvendo estruturas *offshore* de óleo e gás com base em nosso metamodelo de multi-perspectiva. Desenvolvemos ainda um protótipo compatível com HLA como prova do conceito do metamodelo e discutimos como o protótipo pode ser implementado utilizando o InfoGrid. Ainda com relação à aplicação para gestão de desastres, apresentamos um segundo modelo e seu protótipo, mostrando que é possível derivar diferentes modelos de uma mesma aplicação. Finalmente, para validar a generalidade do metamodelo, esboçamos um modelo para outra aplicação: visualização CAD (*Computer-Aided Design* – Projeto com Auxílio de Computador) em ambientes virtuais.

No Capítulo 6 são apresentadas as conclusões e possíveis trabalhos futuros.

No Apêndice A, relatamos as entrevistas que foram realizadas ao longo da elaboração deste trabalho.

No Apêndice B, são reproduzidos quatro artigos já elaborados sobre este trabalho e o resumo da palestra apresentada durante uma oficina internacional.

Finalmente, no Apêndice C, apresentamos algumas telas do protótipo compatível com HLA que foi desenvolvido.

2
Levantamento de Requisitos

Os requisitos foram levantados a partir dos estudos de caso elaborados pela Petrobras sobre os incidentes com a P-36 e a P-34. Eles foram utilizados para identificar os papéis e atributos das pessoas envolvidas em uma operação típica de gestão de desastre. Também foram realizadas entrevistas para identificar procedimentos e expectativas dos usuários com relação ao espaço de trabalho colaborativo. Nesse tipo de ambiente, é importante modelar as relações entre os usuários e identificar as principais características da colaboração que os usuários gostariam de ter à sua disposição.

Uma vez completada a fase de levantamento dos requisitos dos usuários, o passo seguinte foi definir os requisitos técnicos em termos de modelos de colaboração, simulações, espaços de trabalho globais e personalizados, visões sincronizadas, reprodução de vídeos, etc. O espaço de trabalho colaborativo deve ter nós distribuídos com interfaces personalizadas que representem as várias perspectivas dos usuários.

Realizamos a seguir um estudo dos principais sistemas comerciais de gestão de emergências disponíveis, de modo a relacionar suas principais características e identificar os principais aspectos ainda desenvolvidos de maneira insuficiente.

2.1.
A Evolução da Gestão de Desastres na Petrobras

A Petrobras enfrentou recentemente dois grandes incidentes (P-36 e P-34). Esta seção resume esses incidentes no intuito de ilustrar a complexidade do problema em termos dos processos e grupos de pessoas envolvidos nessas situações de desastre. A partir desta discussão, veremos que a Petrobras tem melhorado continuamente seu programa de gestão de desastres.

Durante o acidente com a P-36, houve uma explosão mecânica e uma explosão química com fatalidades, o que dificultou a execução de ações rápidas no sentido de salvar a unidade. Já no desastre com a P-34 não houve nenhuma

explosão, o que permitiu às equipes reagirem rapidamente, ainda que a comunicação entre os membros pudesse ter sido melhor. Esta pesquisa visa dar o próximo passo na evolução desse processo, no sentido de empregar ICT (*Information and Communication Technology* – Tecnologia de Informação e Comunicação) para aperfeiçoar a colaboração entre aqueles envolvidos em ações de contingência.

No caso da P-36, a Petrobras identificou a necessidade de atualizar os procedimentos de emergência e executar as ações em um espaço de tempo mais curto, de modo a poder salvar a unidade. Esse caso levantou a necessidade de se investigar modelos colaborativos e de tomada de decisão capazes de ajudar equipes complexas a evitar maiores desastres.

No caso da P-34, já havia um modelo atualizado da unidade *offshore* e uma forma de trabalho distribuído que ajudou a equipe de resgate a agir com rapidez. Havia também um simulador estático que permitiu aos especialistas executar diferentes simulações. Ainda assim, a equipe não dispunha de um ambiente adequado para trabalhar na forma de uma equipe virtual, de modo a compartilhar conhecimentos, discutir conjuntamente possíveis planos de resgate e chegar rapidamente a um consenso.

Conseqüentemente, foi necessário que as pessoas se reunissem no mesmo local físico, o que provocou algum atraso no processo. Além disso, algumas das informações não estavam diretamente disponíveis para aqueles responsáveis por tomar as decisões. Esse incidente demonstrou a necessidade de se fortalecer a colaboração entre as equipes distribuídas, oferecendo melhor interação, simulação e discussão ao longo de toda a operação de salvamento.

2.2. Requisitos

A partir da análise dos desastres descritos, observamos a necessidade de ir além e desenvolver um paradigma de reação a situações de emergência que deve garantir, como característica essencial, a colaboração entre todos os participantes de cada cenário de contingência. Idealmente, cada nó participante desse ambiente distribuído deve poder compartilhar e discutir seus resultados com outros

participantes, além de ter acesso a todos os dados de que precisa para suas tarefas de simulação.

Discutiremos agora a natureza distribuída das equipes e dos recursos que precisam ser congregados no paradigma proposto para o espaço de trabalho colaborativo.

2.2.1. A Natureza Distribuída das Equipes

No caso particular da Petrobras, quando ocorre um acidente, a sede é imediatamente contatada e o Gerente Geral da unidade operacional fica responsável por gerenciar a crise. Todo o trabalho fica sob seu controle no espaço de trabalho decisório. O Departamento de Segurança, Meio Ambiente e Saúde inicia os procedimentos de emergência e, ao mesmo tempo, especialistas técnicos começam a agir. Esse espaço de trabalho técnico é constituído por engenheiros navais, engenheiros estruturais, analistas de *risers* e oceanógrafos. Quando eles trabalham juntos de forma colaborativa, em geral há os seguintes grupos distribuídos principais:

- uma equipe de alto nível com responsabilidade para tomar decisões na unidade operacional;
- um grupo de força-tarefa que coordena o processo de tomada de decisão:
  - na Unidade de Negócios (no Rio de Janeiro, para plataformas localizadas na Bacia de Campos) se a plataforma não foi seriamente prejudicada, com dois ou três operadores que permanecem dentro da unidade e realizam as operações necessárias; ou
  - em uma cidade que seja o local mais próximo do acidente em terra (Macaé, para as plataformas localizadas na Bacia de Campos), de onde os mergulhadores recebem orientações e realizam as únicas atividades possíveis quando a unidade está seriamente danificada ou tem problemas de segurança;
- uma equipe de suporte técnico na sede da empresa no Rio, na Unidade de Negócios e no centro de pesquisas;

- especialistas em trânsito, que podem estar em mar ou viajando e também precisam estar conectados.

Além desses grupos, e trabalhando em conjunto com eles, há equipes de segurança localizadas em unidades de resgate que se mobilizam rumo à região do acidente e oferecem assistência durante todo o tempo da crise.

2.2.2. A Natureza Distribuída dos Recursos

Não somente os especialistas, como também os recursos do sistema estão distribuídos nesse cenário:

- simuladores que requerem uso intenso de computadores precisam ser executados remotamente em um supercomputador ou em um aglomerado de computadores para obter resultados rápidos;
- o sistema computacional precisa ter acesso a bancos de dados remotos, que contêm modelos CAD e modelos de simulação da unidade;
- cada local envolvido na resolução da crise pode ter diferentes configurações, como um Centro de Realidade Virtual (RV), um computador *desktop* ligado à intranet e um *laptop* conectado à rede;
- os especialistas em trânsito podem se conectar por meio de tecnologias móveis;
- a conexão entre a unidade e as pessoas em terra pode variar. No melhor dos casos, a unidade é um dos nós da rede e, no pior, a comunicação entre os operadores somente pode ser feita por rádio ou telefone.

2.3. Sistemas Comerciais de Gestão de Emergências

Após determinar os requisitos do espaço de trabalho colaborativo para gestão de desastres, realizamos um estudo dos principais sistemas comerciais de gestão de emergências disponíveis, identificando suas características mais

importantes, as principais áreas abrangidas, os recursos de última geração e aqueles que ainda estão pouco desenvolvidos.

Durante a realização deste estudo, foram analisados os sistemas de gestão de emergências de alguns fabricantes além dos métodos de intervenção em caso de crise que estão sendo praticados em empresas como a Statoil, a Norsk Hydro, a Elf e a British Petroleum (BP). Os sistemas estudados estão resumidos nas próximas subseções.

2.3.1. Relatório de Comparação de Recursos do Departamento de Justiça dos EUA

O Departamento de Justiça dos Estados Unidos desenvolveu um Relatório de Comparação de Recursos (Hart, 2002) dos principais CIMS (*Crisis Information Management Software* – Software de Gestão de Informações sobre Crises) disponíveis comercialmente. Esse estudo produziu resultados importantes, dos quais listamos a seguir aqueles diretamente relacionados ao presente trabalho. O programa deve:

- permitir o acesso remoto de usuários autorizados localizados fora da LAN (*Local Area Network* – Rede de Área Local);
- estar de acordo com as normas e os padrões do ICS (*Incident Command System* – Sistema de Comando de Incidentes). O ICS é uma ferramenta para modelar comandos, controles e a coordenação de respostas, desenvolvida em torno de cinco atividades principais de gestão relativas a um incidente:
  - Comando;
  - Operações;
  - Planejamento;
  - Logística;
  - Finanças/administração;
- integrar outros sistemas, como de mapeamento, outros CIMS e sistemas de notificação telefônica;
- integrar o sistema de saúde pública à gestão de emergências;
- operar em uma variedade de configurações de redes;

- ter uma vasta gama de recursos consistentes com as quatro fases das operações de gestão de emergências: planejamento, mitigação, reação e recuperação.

O relatório também concluiu que não há um produto que seja o melhor ou que se adeqüe perfeitamente. O produto mais apropriado deve ser escolhido com base no orçamento, no ambiente de sistema, na dimensão da operação, na sofisticação da operação, na disciplina para sua implementação e em questões de ordem política.

2.3.2. L-3 CRISIS *Command and Control System*

Uma das maiores fabricantes de sistemas de emergência é a Ship Analytics, com a qual a Petrobras já começou a estabelecer uma relação comercial. Um de seus principais produtos é o L-3 CRISIS *Command and Control System* (Sistema de Comando e Controle de Crise) (MPRI Ship Analytics, 2003), um dos sistemas computacionais comerciais considerados como padrão. Ele auxilia os coordenadores das ações de contingência, funcionando como a espinha dorsal durante a resposta a desastres e como ferramenta educativa em treinamentos para situações de emergência. Ele traz ainda simulações computacionais que permitem avaliar respostas alternativas, um sistema de planejamento para gestão de riscos e mitigação de danos ambientais, e um sistema de contabilidade para administrar ativos alternativos. Oferece também às equipes de gestão de incidentes um GIS (*Geographic Information System* – Sistema de Informações Geográficas) e SOP (*Standard Operating Procedures* – Procedimentos Padrão de Operação) para facilitar a reação a diferentes tipos de desastres, como enchentes, vendavais, vazamento de gases tóxicos, etc.

Do ponto de vista prático, os coordenadores de emergências consideram útil elaborar listas de verificação a partir do SOP para cada posição funcional relacionada a eventos específicos. As listas de verificação são desenvolvidas de modo a serem fáceis de ler e de implementar, e o L-3 CRISIS oferece um meio de automatizar a função de lista de verificação.

Quando necessário, são empregados *firewalls* para proteger informações confidenciais, permitindo aos usuários terem acesso às informações necessárias para realizarem suas tarefas. O sistema foi projetado para utilizar informações de entrada a partir de diversos equipamentos de detecção em tempo real, como sensores infravermelhos, imagens de satélite e bóias detectoras de hidrocarbonetos. O sistema é compatível com a Web, incorporando um navegador que permite aos usuários acessarem as informações mais atualizadas disponíveis na Internet, na Intranet e nas próprias bases de dados do sistema.

2.3.3. *Oil Spill Crisis Management Simulator*

Um dos simuladores baseados no L-3 CRISIS é o *Oil Spill Crisis Management Simulator* (Simulador de Gestão de Crises de Derramamento de Óleo). Ele auxilia em todas as etapas da gestão de incidentes, desde a prevenção/mitigação, passando pela preparação e até a reação e a remediação. O programa funciona como um centro de treinamento completo para Equipes de Gestão de Incidentes, Comandantes e Executores no Local, capacitando-os a resolver diversos incidentes por meio da simulação de exercícios, planos de reação, procedimentos operacionais e listas de verificação.

O SMS (*Spill Management Simulator* – Simulador de Gestão de Vazamentos) incorpora modelos padrão de derramamento de óleo e de mitigação. Os participantes do exercício interagem com o cenário utilizando os módulos de reação do L-3 CRISIS, o qual traz uma interface com bancos de dados relacionais complexos, um Sistema de Informações Geográficas (GIS) e modelos científicos de destino e trajetória de vazamentos.

O SMS oferece ainda um repositório central que contém informações cruciais para a administração eficiente da resposta a crises, como os locais e as especificações dos equipamentos de resposta disponíveis, o pessoal disponível e dados ambientais e econômicos essenciais. O componente de simulação do sistema, que modela o destino e a trajetória físicos do óleo derramado e rastreia as embarcações e as operações simuladas de reação, armazena os dados de saída no banco de dados central.

Para simular um exercício, são criados bancos de dados de especialistas e equipamentos específicos para cada cenário associando-se todos ou um subconjunto dos bancos de dados da Central de Recursos a um determinado cenário. Quando o exercício está em andamento, os instrutores/operadores podem modificar e/ou adicionar registros de recursos específicos daquele cenário sem afetar os bancos de dados da Central. Os dados de recursos são utilizados pela Equipe de Gestão de Incidentes ou pelos Participantes do Exercício para alocar e rastrear equipamentos e pessoal.

Os cenários de vazamentos são desenvolvidos ou modificados antes da realização de um exercício por meio de um recurso de definição de cenários.

2.3.4. *Systems Requirements Document* (SRD) do *Automated Resource Management System* (ARMS) dos EUA

Outro documento importante que também foi estudado é de autoria da Booz Allen Hamilton (2003), a qual foi encarregada de auxiliar a Divisão de Preparo da FEMA (*Federal Emergency Management Agency* – Agência Federal de Gestão de Emergências) do governo dos Estados Unidos na elaboração de um documento com requisitos para o ARMS (*Automated Resource Management System* – Sistema Automatizado de Gestão de Recursos). O SRD (*Systems Requirements Document* – Documento de Requisitos de Sistema) do ARMS traz uma especificação de alto nível dos requisitos desse sistema, identificando e definindo os dados correspondentes, as normas empresariais e os requisitos funcionais, operacionais e técnicos para um site da Web destinado a auxiliar os governos estaduais e municipais a melhorar sua capacidade de assistência mútua durante situações de emergência.

O ARMS é definido como um "sistema automatizado que ajuda coordenadores de gestão de emergências a localizar recursos para melhorar sua resposta a emergências." Entre seus recursos incluem-se pessoal, equipamentos e suprimentos. O ARMS é a faceta computadorizada da *National Mutual Aid and Resource Management Initiative* (Iniciativa Nacional de Ajuda Mútua e Gestão de Recursos) do governo dos EUA, e visa aperfeiçoar o processo de assistência mútua.

2.3.5.
Análises do *Crisis Intervention and Operability* (CRIOP)

Outro documento importante que apresenta um método por meio de cenários para intervenções em crises e análises de operabilidade é o CRIOP (*Crisis Intervention and Operability* – Intervenção e Operabilidade em Crises) (Johnsen, 2004), elaborado em conjunto pela Scandpower, SINTEF, Statoil e NTNU, com ajuda da Norsk Hydro, Saga, Elf, NORSOK, BP, Safetec, DNV e Aker.

O CRIOP consiste de uma metodologia empregada para verificar e validar a capacidade de um centro de controle de lidar com segurança e eficiência com todos os modos de operação, incluindo iniciação, operação normal, manutenção e revisão, alterações no processo, situações críticas de segurança e finalização. Tal metodologia pode ser aplicada a salas de controle central, cabines de perfuração, guinchos e outros tipos de cabine, além de salas de controle *onshore*, *offshore* e de emergência.

Os elementos centrais do CRIOP são listas de verificação que abrangem áreas relevantes no projeto de Centros de Controle (CC), Análise de Cenário de alguns cenários chave e uma área de aprendizagem onde empregados de operações, projetistas e administradores podem se reunir e avaliar o melhor modelo de CC. As análises do CRIOP são iniciadas com uma fase de preparação e organização para identificar os participantes, coletar a documentação necessária, estabelecer grupos de análise e decidir quando a análise deve ser executada.

O método se concentra na interação entre as pessoas, a tecnologia e as organizações. Um dos princípios mais importantes do método CRIOP consiste em garantir que o foco permaneça nos fatores humanos importantes, no que diz respeito à operação e ao tratamento de situações anormais em centros de controle *offshore*, e em validar soluções e resultados.

2.3.6.
Conclusões sobre os Sistemas de Emergência

Finalizando este levantamento sobre gestão de emergências, podemos observar que a maioria dos sistemas possui algumas características comuns, a saber:

- geralmente funcionam como ferramentas tanto de gestão de incidentes quanto de treinamento e planejamento;
- possuem grande capacidade de integração, não somente com bancos de dados e sistemas internos como também com sistemas públicos de gestão de emergências;
- geralmente têm uma arquitetura flexível em termos de integração com redes e software, integrando-se com um ou mais simuladores relacionados com o tipo de desastre que está sendo abordado;
- normalmente integram-se a um sistema gráfico, como um Sistema de Informações Geográficas (GIS), o qual é responsável por apresentar dados em tempo real do incidente em questão;
- são capazes de registrar e rastrear atividades e recursos, o que é importante não somente durante uma emergência real mas também como ferramenta para fins de treinamento;
- a maioria dos sistemas de emergência utiliza listas de verificação como um método eficiente e rápido para abordar os múltiplos requisitos simultâneos presentes em um cenário de emergência. A capacidade de oferecer meios de automatizar o máximo possível a função de lista de verificação pode ser um fator determinante na resolução rápida e segura de uma situação de emergência.

Apesar de todos os recursos listados acima, identificamos dois pontos negativos principais dos sistemas atuais de gestão de emergências:

1. Falta de integração adequada e completa dos simuladores com sistemas de visualização de alto desempenho;
2. Recursos de controle de acesso e segurança inadequados.

Este estudo demonstrou que, a despeito da integração que a maioria dos sistemas de gestão de emergências possui com simuladores, é preciso se desenvolver uma arquitetura de sistema capaz de suportar recursos distribuídos, particularmente simuladores distribuídos sendo executados em sistemas de visualização de *alto desempenho*. Tal arquitetura também deve fornecer comunicação síncrona entre diferentes equipamentos com co-localização virtual como uma de suas características.

A integração de simuladores com sistemas de visualização de alto desempenho em um ambiente distribuído sincronizado é o aspecto dos cenários de emergência no qual vamos nos concentrar. Como base para definir a arquitetura desse ambiente, foi elaborado um metamodelo. Desenvolvemos então um protótipo compatível com HLA como prova de conceito do metamodelo, tendo em conta que o HLA é um padrão para simulações de alto desempenho em tempo real.

3
Um Metamodelo para a Configuração de Espaços de Trabalho Virtuais Colaborativos

Neste capítulo é elaborado um metamodelo de multi-perspectiva para ajudar a definir e configurar a arquitetura do espaço de trabalho virtual colaborativo.

Os sistemas colaborativos são definidos como "uma combinação de tecnologia, pessoas e organizações que facilita a comunicação e a coordenação necessárias para que um grupo trabalhe efetivamente em conjunto em busca de um objetivo comum, obtendo ganhos para todos os membros." (Haynes et al., 2004) Empregando esse conceito, muitas empresas têm criado equipes virtuais de seus próprios funcionários, congregando profissionais geograficamente dispersos com capacidades complementares (Handel & Herbsleb, 2002; Bos et al., 2004), o que aumentou a demanda por aplicações CSCW. Para tornar mais fácil e efetivo o desenvolvimento de uma vasta gama dessas aplicações colaborativas, seria preciso oferecer uma arquitetura geral adaptável a diferentes situações, tarefas e configurações de uma forma flexível (Schuckmann et al., 1996).

Até o presente, a pesquisa em CSCW abordando as características da arquitetura mencionadas acima tem enfocado sobretudo as diferenças entre: *(i)* trabalho co-localizado e à distância (Bos et al., 2004; Dourish & Bly, 1992; Fussell et al., 2000; Herbsleb & Grinter, 1999; Herbsleb et al., 2000; Lauche, 2005); ou *(ii)* trabalho com pessoas da mesma cultura ou com bases comuns (afinidades, conhecimentos mútuos, crenças, objetivos, atitudes, etc) e trabalho com pessoas de diferentes culturas (trabalho inter-cultural) ou bases comuns (Fussell et al., 2000; Greenspan et al., 2000; Setlock et al., 2004). Essas perspectivas foram denominadas, respectivamente: *Centrada nos Lugares* e *Centrada nas Pessoas* (Jones et al., 2004).

Em vez de empregar uma dessas perspectivas para derivar um metamodelo para o projeto de aplicações CSCW, nossa proposta consiste em adotar uma visão diferente do problema, baseada nas atividades realizadas pelas equipes que

participam do trabalho colaborativo. Denominamos esta perspectiva *Centrada nas Atividades*. O termo *Atividade* aqui incorporado provém da teoria da atividade, sendo um conceito amplo e de múltiplos níveis.

Como detalharemos nas seções que se seguem, a perspectiva Centrada nas Atividades pode ser entendida como um conceito de multi-perspectiva (também de acordo com a forma como o termo foi incorporado a partir da teoria da atividade), visto que não somente engloba as perspectivas Centrada nos Lugares e Centrada nas Pessoas como também permite adotar cada uma delas, ou, tipicamente, ambas (de uma maneira híbrida) na medida desejada, além de admitir alterações passando de uma perspectiva para a outra.

É importante destacar que esta abordagem não se baseia na tecnologia, mas sim nos usuários ou nas pessoas (Ishii et al., 1994; Laurillau & Nigay, 2002; Palen, 1999; Sachs, 1995), enfocando um estudo qualitativo (Neale et al., 2004) das equipes envolvidas em vez de uma tecnologia específica.

A motivação para este trabalho foi a necessidade de configurar um espaço de trabalho virtual colaborativo para a gestão de desastres em estruturas de óleo e gás *offshore* de uma empresa global. Para identificarmos os requisitos do sistema, realizamos entrevistas semi-estruturadas com indivíduos chave, e não somente observamos sua prática profissional como efetivamente participamos de projetos e atividades conjuntas por mais de uma década.

3.1. Um Metamodelo Centrado nas Atividades

Iniciamos esta seção definindo *metamodelo*: trata-se de um modelo lógico – em contraposição a um modelo físico ou implementação – com ênfase na semântica declarativa em vez de nos detalhes de implementação do modelo. Espera-se que as implementações baseadas no metamodelo estejam de acordo com sua semântica (Athanasopoulos et al., 2003).

Diversos pesquisadores vêm desenvolvendo arquiteturas baseadas nesse conceito. A arquitetura colaborativa genérica de Dewan (1999) estrutura uma aplicação *groupware* em um número variável de camadas, desde o nível dependente do domínio até o nível de hardware, na qual uma camada é um componente de software que corresponde a um nível específico de abstração. De

forma semelhante, o metamodelo da arquitetura Clover (Laurillau & Nigay, 2002) também estrutura uma aplicação *groupware* em um número variável de camadas, decompondo cada camada em três sub-componentes funcionais dedicados à produção, comunicação e coordenação. No conjunto de ferramentas Prospero, Dourish propõe uma abordagem na qual uma arquitetura em metaníveis é usada pelos programadores para separar a representação e a funcionalidade de alto nível dos recursos de baixo nível, mapeando as primeiras nos segundos (Dourish, 1998).

O metamodelo que propomos adota uma abordagem semelhante de múltiplos níveis, de acordo com a versão da teoria da atividade de Leontjev (1978), na qual um esquema em três níveis descreve a estrutura hierárquica da atividade. Este esquema foi sintetizado por Tuikka (2002) da seguinte forma:

- O nível central é o das *ações*. As ações são orientadas visando *objetivos*. Em geral, os objetivos são subordinados a outros objetivos, os quais podem ser subordinados a outros, e assim por diante.
- Subindo na hierarquia de objetivos, finalmente chegamos a um objetivo no nível mais alto, que não está subordinado a nenhum outro. Este objetivo mais alto, que na teoria da atividade é denominado *motivo*, é o objeto de uma atividade completa.
- Descendo na hierarquia de ações, chegamos a processos automáticos que ajustam as ações às situações atuais. Segundo a teoria da atividade, tais processos são *operações*.
- As atividades, dirigidas pelos motivos, são realizadas por meio de ações que visam objetivos, os quais, por sua vez, são implementados por meio de operações.

De forma ortogonal a esta abordagem, semelhante ao metamodelo Clover, o metamodelo Centrado nas Atividades também permite desdobrar os componentes correspondes a um nível específico. Essas duas abordagens ortogonais aplicadas juntas contribuem para a generalidade de nosso metamodelo.

3.1.1.
Níveis de Abstração do Metamodelo

O nível mais alto de nosso metamodelo, o motivo, é representado por um *nó* complexo que engloba toda a atividade. O motivo pode ser muito variado, como a elaboração de um artigo (Figura 1) ou a gestão de um desastre em uma estrutura *offshore* de óleo e gás (Figura 2). O nível imediatamente abaixo do nível superior contém as principais ações que devem ser executadas para realizar toda a atividade. Essas ações resultam das interações dos grupos, com cada grupo sendo representado por um nó complexo e as interações entre eles sendo representadas por *arestas*, o que significa que cada nível de abstração pode ser representado por um grafo.

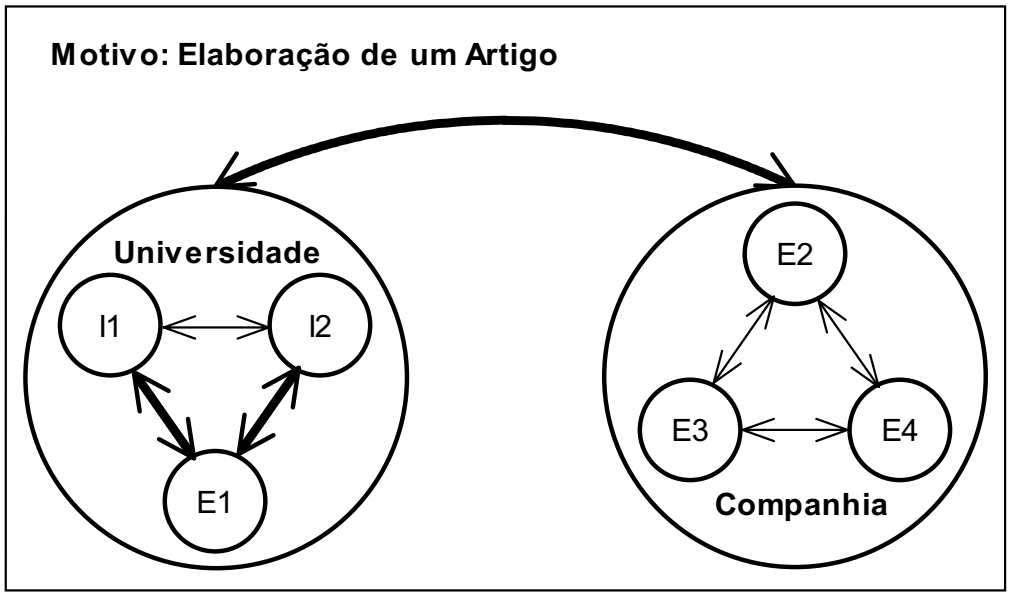

Figura 1 - Modelo colaborativo para a elaboração de um artigo: I1 e I2 são pesquisadores de Informática e E1, E2, E3 e E4 são Engenheiros

Continuamos descendo nos níveis, dividindo cada nó complexo do nível de abstração superior em nós mais elementares, até atingirmos um *nó folha*, que tipicamente será uma *pessoa*. A esses nós folhas associamos atributos de implementação e de hardware, tais como a aplicação a ser executada e o computador no qual ela deve rodar. Às vezes, o nó folha não é uma pessoa real mas um *agente de software*, responsável por um determinado conjunto de tarefas (Ellis et al., 1991).

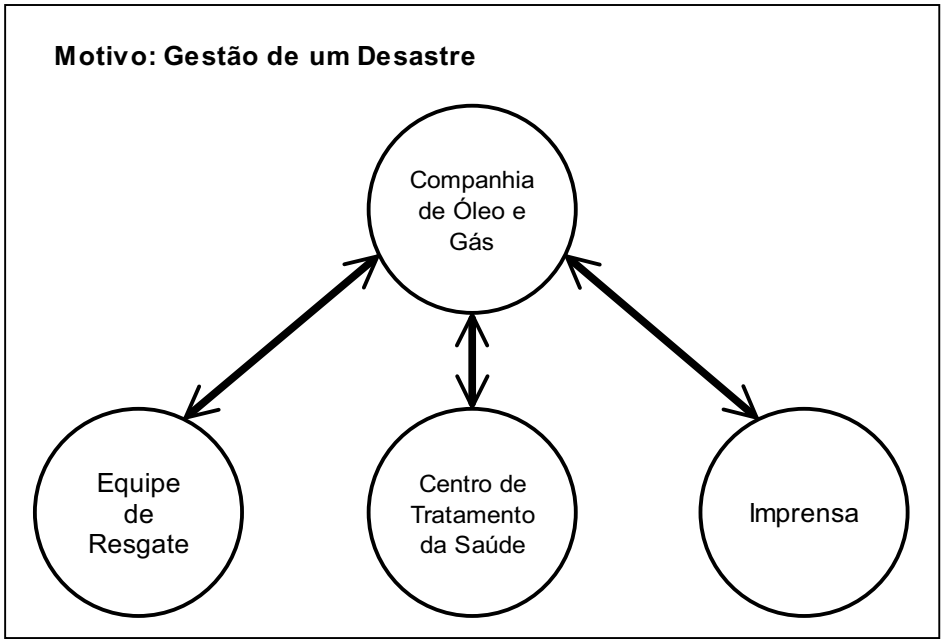

Figura 2 - Modelo colaborativo para a gestão de um desastre em uma estrutura *offshore* de óleo e gás

Na Figura 2, podemos ver que este metamodelo é suficientemente genérico para dar conta até mesmo de modelos que incluam grupos ou nós inter-organizacionais. Nesses casos, os mecanismos tradicionais de controle de acesso de *groupware* devem ser estendidos para lidar com aspectos novos, como privacidade, confidencialidade, interesses mútuos e contraditórios, culturas diferentes, etc (Cohen et al., 2000; Godefroid et al., 2000; Setlock et al., 2004; Stevens & Wulf, 2002).

3.1.2. Desdobrando os Componentes de um Nível de Abstração Específico

De forma ortogonal a esta abordagem, o metamodelo Centrado nas Atividades também permite desdobrar os componentes correspondentes a um nível específico. A idéia central desse mecanismo ficará mais clara se a ilustrarmos com um exemplo prático.

Consideremos um nível de abstração específico do exemplo do modelo de gestão de desastres: o primeiro nível abaixo do superior, relativo à companhia de óleo e gás (Figura 3). Observamos dois grupos principais: *Equipes Técnicas* e *Tomadores de Decisões.* Para facilitar a compressão do mecanismo de

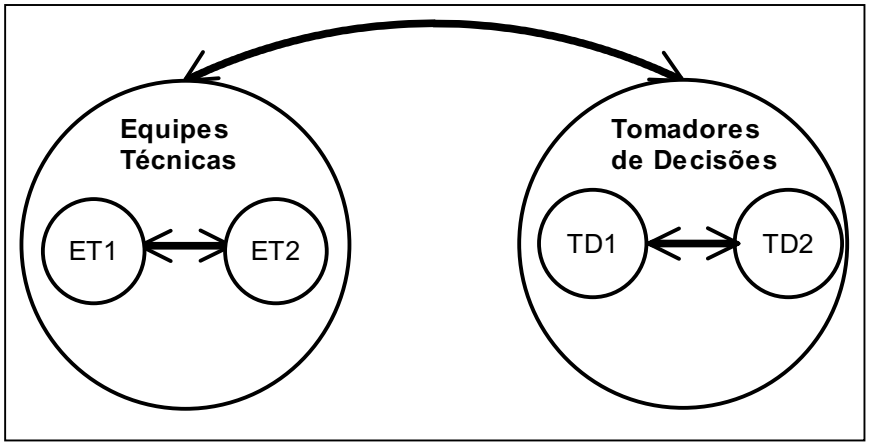

Figura 3 - Primeiro nível abaixo do superior, relativo à companhia de óleo e gás do modelo colaborativo de gestão de desastres: ET1 e ET2 representam a Equipe Técnica 1 e a Equipe Técnica 2, respectivamente, e TD1 e TD2 representam, respectivamente, o Tomador de Decisões 1 e o Tomador de Decisões 2

desdobramento, também representamos nessa figura o nível imediatamente abaixo do que estamos considerando: o grupo *Equipes Técnicas* é decomposto em dois subgrupos, *Equipe Técnica 1* e *Equipe Técnica 2*, e o grupo *Tomadores de Decisões* é decomposto em dois subgrupos (pessoas), *Tomador de Decisões 1* e *Tomador de Decisões 2*. Nessa configuração, ambos os *Tomadores de Decisões* estão sendo considerados como tendo os mesmos conhecimentos e o mesmo nível de interação com as *Equipes Técnicas.* Suponhamos agora que o *Tomador de Decisões 1* seja um gerente mais técnico e que o *Tomador de Decisões 2* seja um gerente de um nível organizacional mais alto dentro da empresa, como um diretor, e que não esteja tão envolvido tecnicamente com o problema. A forma mais simples de acomodar essa diferença é desdobrar o grupo *Tomadores de Decisões* em dois novos grupos nesse mesmo nível de abstração, *Tomador de Decisões 1* e *Tomador de Decisões 2*, cada um destes grupos com apenas um gerente. Devemos também substituir as arestas anteriores por novas, que representem as novas interações entre os novos grupos formados. A configuração resultante está mostrada na Figura 4. Agora está claro que o gerente no nível intermediário é o responsável pela comunicação direta com as *Equipes Técnicas* e pela comunicação com o diretor.

Este exemplo simples ilustra a utilidade deste mecanismo de desdobramento, que nos permite fazer experiências até chegar ao modelo mais apropriado para uma situação específica.

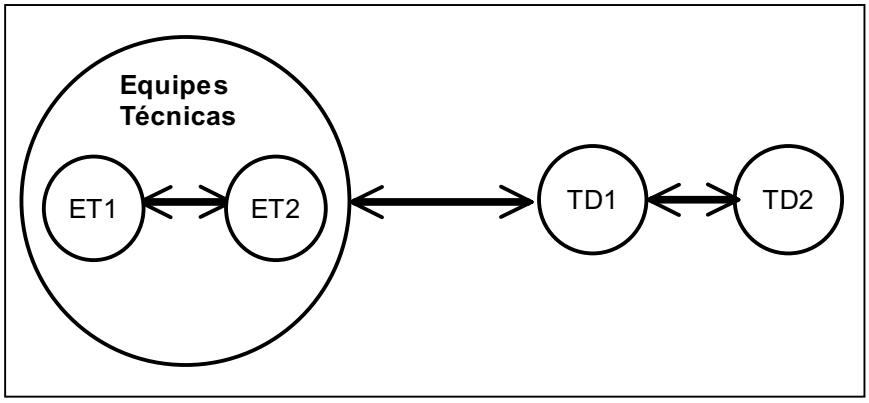

Figura 4 - Primeiro nível abaixo do superior, relativo à companhia de óleo e gás no modelo colaborativo de gestão de desastres, agora com o Tomador de Decisões 1 entre o nó das Equipes Técnicas e o Tomador de Decisões 2

3.1.3. Componentes do Metamodelo

Agora descreveremos os componentes principais de nosso metamodelo.

3.1.3.1. Nós

Os *nós* são componentes essenciais do metamodelo. Eles vão do nível mais alto, que representa toda a atividade (motivo), passando pelos muitos nós de diferentes níveis que representam grupos e subgrupos, até os *nós folhas*, que representam uma *pessoa* ou um *agente de software*. Os nós possuem um conjunto de *atributos*, como preferências de interface com o usuário e a língua utilizada, que são aplicados por meio de um conceito hierárquico de classe: o valor de um atributo de um nó vale para todos os sub-nós abaixo do nó considerado, a menos que seja explicitamente redefinido, sendo essa redefinição também válida para todos os níveis abaixo do nível considerado.

Os nós também possuem um atributo denominado *artefatos*, definido como "todos os objetos nos quais os usuários podem operar" (Gross & Prinz, 2004). Exemplos de artefatos são desenhos, rascunhos, modelos físicos, protótipos e descrições de produtos, documentos e arquivos, canetas e quadros negros, relatórios, artigos e comentários de revisão, planilhas e gráficos, todos em versão eletrônica, ou mais complexos, como calendários eletrônicos (Palen, 1999), pastas e armários compartilhados (Pankoke-Babatz & Syri, 1997), CAD e mapas

computadorizados (Pettersson et al., 2004), protótipos virtuais compartilhados (Tuikka, 2002), superfícies interativas compartilhadas (Brignull et al., 2004), monitores *tabletop* compartilhados (Morris et. al, 2004a), *tabletops* compartilhados entre múltiplos usuários (Morris et al., 2004b) e telas compartilhadas interativas (Paek et al., 2004). De acordo com o conceito de classe, um artefato associado ao nó de um grupo é compartilhado por todos os membros do grupo, a menos que seja explicitado de outra forma. Nesse caso, um mecanismo – tal como uma lista de controle de acesso – determinará quem possui acesso compartilhado ao artefato.

Um artefato possui duas *propriedades* básicas (Dourish & Bellotti, 1992): *sincronia*, que pode assumir os valores *síncrono* (há trabalho sendo realizado sobre ele neste momento) ou *assíncrono*; e *persistência*, que pode assumir os valores *persistente* (perdura após o trabalho sobre ele ter sido realizado) ou *volátil*.

Os nós folhas possuem atributos de nível baixo, que serão mapeados para uma implementação específica, tal como a aplicação a ser executada (Gross & Prinz, 2004).

3.1.3.2. Arestas

As *arestas* de nosso metamodelo representam os *caminhos de interação* entre os nós. Elas podem ser unidirecionadas ou bidirecionadas, representando os sentidos de interação possíveis. Quando uma aresta é representada por uma seta fina, isso significa que os nós em suas extremidades estão co-localizados. Quando a seta é grossa, ela indica que um nó está um local remoto em relação ao outro.

Uma aresta possui um ou mais *canais*, cada um representando o canal mediado eletronicamente que permite a comunicação entre dois nós (Greenspan et al., 2000).

Segundo Nardi et al. (2000), a comunicação é principalmente sobre o intercâmbio de informações. As informações trocadas podem ser de muitos tipos, como texto, gráfico, voz e vídeo. Fuks et al. (2005) listam alguns elementos que os canais de comunicação devem considerar para que o intercâmbio de informações ocorra: mídia (textual, falada, pictórica ou gestual), modo de transmissão (em blocos ou continuamente; síncrono ou assíncrono), políticas de

restrições, meta-informações (sobre a mensagem, como o assunto, data, prioridade e categoria), estrutura conversacional (linear, hierárquica ou em forma de rede) e caminhos conversacionais.

Um canal possui quatro propriedades básicas:

- *Sincronia* (Dourish & Bellotti, 1992; Greenspan et al., 2000): a distinção síncrono-assíncrono depende de se o conteúdo é comunicado ou escolhido. Entre os exemplos de canais síncronos ou em tempo real, incluem-se: mensagens instantâneas, *chat* e vídeo-conferência. Uma característica importante dessas aplicações síncronas é que elas transmitem informações de percepção de presença (*presence awareness*) (Dourish & Bellotti, 1992; Godefroid et al., 2000; Pankoke-Babatz & Syri, 1997) sobre os membros que participam da sessão de colaboração. Alguns exemplos de canais assíncronos são: correio eletrônico, conferência por computador estilo fórum e páginas da Web. Por um lado, os canais síncronos podem oferecer retorno em tempo real; por outro, os assíncronos oferecem maior controle sobre a criação de conteúdo e facilitam o arquivamento e o encaminhamento de mensagens. Além desses dois modos tradicionais, Dourish and Bellotti (1992) apresentam o termo *semi-síncrono*, que abrange os modos síncrono e assíncrono e é associado a espaços de trabalho compartilhados: os sistemas semi-síncronos apresentam informações sobre colaboradores co-presentes sincronamente, ao mesmo tempo em que representam atividades anteriores realizadas por outros colaboradores que não estejam sincronamente presentes.
- *Persistência* (Dourish & Bellotti, 1992; Pankoke-Babatz & Syri, 1997): o canal é denominado *volátil* se só pode oferecer informações no momento em que o evento ocorre, como no caso de vídeo-conferência. Por outro lado, é chamado *persistente* se pode oferecer informações após algum tempo de ausência, para informar sobre progressos intermediários, ou quando solicitado a dar mais detalhes, ou para informar sobre a história de um objeto. Um exemplo de canal persistente é um *chat* que tenha a capacidade de armazenar um

histórico da discussão (Handel & Herbsleb, 2002; Ribak et al., 2002) ou uma conferência por computador estilo fórum.

- *Simetria* (Greenspan et al., 2000): um canal é considerado *simétrico* se o receptor de uma mensagem puder responder com o mesmo tipo de mensagem. Um exemplo é uma ferramenta de e-mail que permita ao mesmo tempo ler, compor e enviar mensagens. A vídeo-telefonia é um canal totalmente *simétrico* que permite ao canal audiovisual transmitir nos dois sentidos, enquanto as transmissões por televisão e rádio são totalmente *assimétricas.* As aplicações de ensino à distância trazem a possibilidade de formas híbridas de comunicação, nas quais ambos os lados podem se ouvir mas só o apresentador é visível para os estudantes.
- *Mídia*: esta propriedade está relacionada à riqueza de mídia (Greenspan et al., 2000) do canal de comunicação, envolvendo, por exemplo, aspectos de som e imagem. A mídia da comunicação presencial não mediada por computador é considerada mais "rica" do que a audiovisual, a qual por sua vez é mais "rica" do que a comunicação somente por som ou somente por imagem.

3.2. Instanciação do Metamodelo Centrado nas Atividades: Modelos Centrados nas Atividades

Nesta seção, enfocaremos a característica mais importante de nosso metamodelo Centrado nas Atividades: a capacidade de acomodar múltiplas perspectivas, correspondentes a diferentes modelos, cada uma como uma instância do metamodelo.

Para facilitar a compreensão das múltiplas perspectivas, derivaremos modelos a partir do modelo colaborativo de elaboração de um artigo apresentado na Subseção 3.1.1, com a única diferença de que agora haverá pesquisadores em duas universidades diferentes.

3.2.1.
Um Modelo Centrado nos Lugares

Suponhamos que os aspectos mais importantes de nossa aplicação colaborativa estejam relacionados com o local onde as pessoas estão efetivamente trabalhando. Essa prioridade pode ter sido motivada, por exemplo, pela necessidade de empregar uma comunicação com mídia "rica", como a presencial (Greenspan et al., 2000), ou porque os diferentes lugares possuem instalações diferentes, como computadores *desktop* e salas de visualização, com comportamentos profissionais diferentes. Um possível modelo usando essa perspectiva Centrada nos Lugares para a aplicação colaborativa da elaboração do artigo é o mostrado na Figura 5.

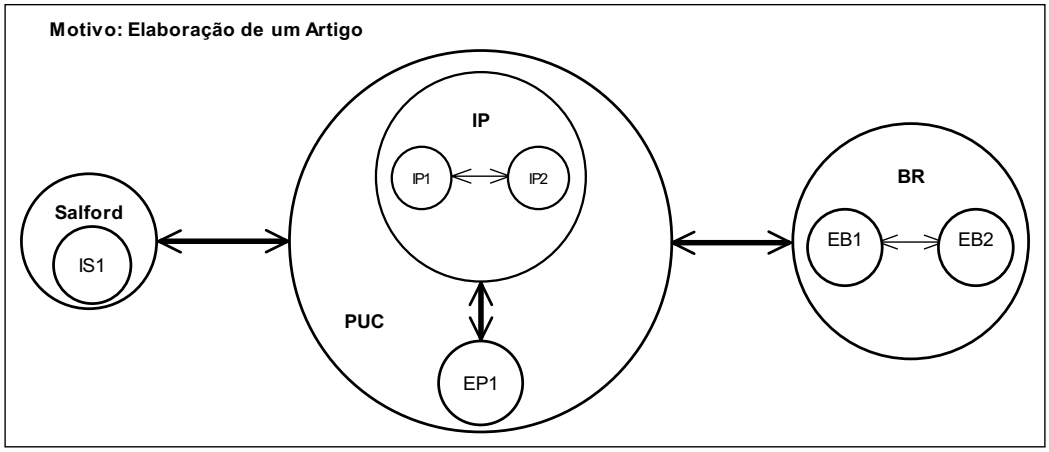

Figura 5 - Modelo colaborativo da elaboração de um artigo: perspectiva Centrada nos Lugares

Nessa figura, podemos observar que os grupos são formados a partir de uma perspectiva Centrada nos Lugares:

- Há três nós principais: a Universidade PUC, a Universidade de Salford e a Petrobras (BR). Visto que em nosso exemplo o artigo trata de Informática, o nó central, neste caso exercendo a função principal na preparação do artigo, é a PUC, que se comunica remotamente com a Universidade de Salford e com a Petrobras (neste modelo, não há vínculo direto entre Salford e a Petrobras).
- Dentro da PUC, há dois subgrupos: um é o Departamento de Informática (IP, I de Informática e P de PUC), que possui dois pesquisadores de Informática co-localizados e trabalhando em

conjunto, IP1 e IP2; e o outro é o Departamento de Engenharia, que possui somente um engenheiro (EP1). Os dois departamentos ficam em edifícios diferentes e também se comunicam, só que remotamente. Para não poluir visualmente a imagem, quando um nó contém somente um membro, esse nó será designado pelo nome do nó folha. É claro que é preciso ter cuidado com essa notação, pois pode ser interessante dispor de propriedades associadas a diferentes níveis, mesmo quando há somente um membro contido no nó pai, como no caso do nó EP1 do exemplo em questão. Esse pode ser o caso da hierarquia Universidade → Departamento → Pesquisador, com atributos específicos relativos a cada nível, mesmo com apenas um nó folha representando todos os grupos.

- Dentro da Universidade de Salford há apenas um pesquisador de Informática, chamado IS1 (I de Informática e S de Salford). Nesse caso, apenas para deixar clara a hierarquia de Salford, pusemos o nó folha IS1 dentro do nó pai Salford (conforme a notação acima, poderíamos apenas escrever IS1 dentro deste nó).
- Finalmente, dentro do nó Petrobras (BR), há dois engenheiros co-localizados trabalhando em conjunto: EB1 e EB2 (E de Engenheiro e B de BR).

Assim, nos modelos Centrados nos Lugares agrupamos os vários nós de modo a privilegiar os aspectos relacionados aos ambientes nos quais o trabalho está sendo realizado, como salas especiais (de visualização, por exemplo) ou dispositivos interativos especiais (como monitores interativos compartilhados), a necessidade de se ter comunicação presencial, etc.

3.2.2.
Um Modelo Centrado nas Pessoas

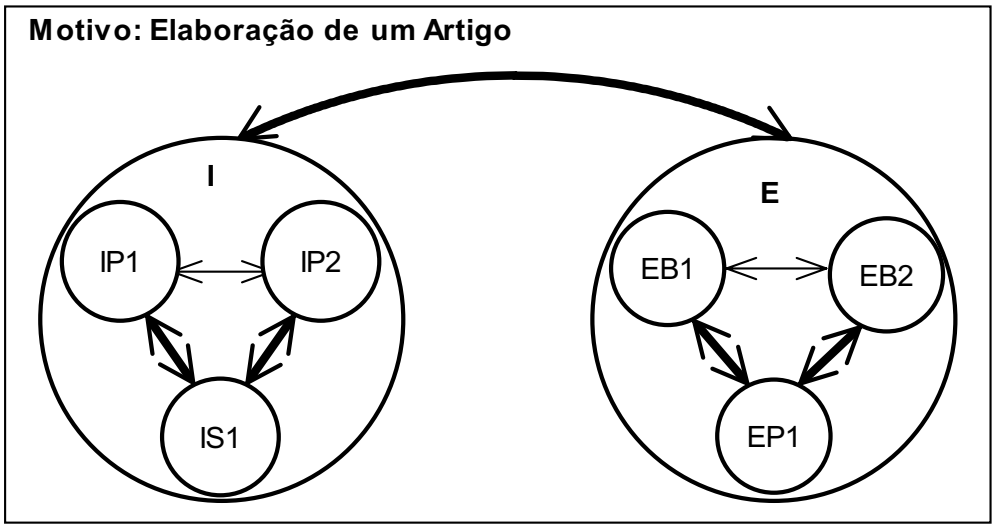

Figura 6 - Modelo colaborativo da elaboração de um artigo: perspectiva Centrada nas Pessoas

Consideremos agora que as questões principais de nossa aplicação colaborativa estão relacionadas às barreiras de cultura e bases comuns. Nesse caso, devemos derivar um modelo com uma perspectiva Centrada nas Pessoas, como o ilustrado na Figura 6:

- Agora há somente dois nós principais: o grupo de pesquisadores de Informática (I) e o grupo de Engenheiros (E), que se comunicam remotamente.
- Dentro do grupo I há três pesquisadores de Informática que trabalham juntos: IP1, IP2 e IS1. IP1 e IP2 estão co-localizados e IS1 fica localizado remotamente a ambos, possivelmente co-localizado em termos virtuais. É importante notar que, apesar de IS1 ser de outra universidade, quando comparado a IP1 e IP2, as bases comuns são tão intensas que ele pertence ao mesmo subgrupo que IP1 e IP2, apesar de estar localizado remotamente a eles. Se essa relação não for tão intensa, ou se houver uma barreira cultural, I deveria ser dividido em dois subgrupos, um com dois pesquisadores co-localizados, IP1 e IP2, que se comunicassem remotamente com IS1, único membro de outro subgrupo de Informática.

- O mesmo raciocínio pode ser aplicado ao grupo E. Há três Engenheiros, EP1, EB1 e EB2. EB1 e EB2 estão co-localizados e EP1 fica localizado remotamente a ambos, possivelmente co-localizado virtualmente. De forma análoga à análise sobre IS1, se houver algum tipo de barreira cultural entre EP1 e o grupo formado por EB1 e EB2, devemos dividir o grupo E em dois subgrupos, um com os Engenheiros EB1 e EB2 co-localizados comunicando-se remotamente com EP1, único membro de outro subgrupo de Engenharia.

Logo, nos modelos Centrados nas Pessoas, os principais aspectos levados em conta na definição dos grupos estão relacionados às características dos participantes: sua cultura ou bases comuns, suas especialidades, suas habilidades, comportamentos, etc.

3.2.3. Combinação de Perspectivas: um Modelo Centrado nas Atividades

Agora combinaremos as duas perspectivas anteriores na perspectiva que denominamos Centrada nas Atividades. No caso da aplicação colaborativa que estamos examinando, parece ser mais adequado enfocar a atividade completa que está sendo realizada – a elaboração de um artigo – e então derivar os grupos que serão formados. Para elaborar o artigo, os autores IP1, IP2, IS1 e EP1 procuram derivar um novo modelo teórico baseado nos requisitos identificados por meio de estudo de campo, trabalhando em conjunto com os Engenheiros EB1 e EB2. Assim, juntamos essas pessoas em dois grupos principais, formados com base em sua atividade principal: o grupo de Teoria e o grupo de Campo. Os grupos então são organizados de acordo com a Figura 7:

- Os dois grupos principais, Teoria e Campo, se comunicam remotamente entre si.
- Dentro do grupo Teoria há dois subgrupos: um com três pesquisadores de Informática (I), IP1 e IP2, trabalhando em conjunto e co-localizados, com IS1 trabalhando em posição remota, possivelmente co-localizado virtualmente; e outro com um só

Engenheiro, EP1, trabalhando em posição remota com relação ao grupo I.

- O grupo Campo é equivalente ao grupo BR na Figura 5, com dois Engenheiros, EB1 e EB2, co-localizados trabalhando em conjunto.

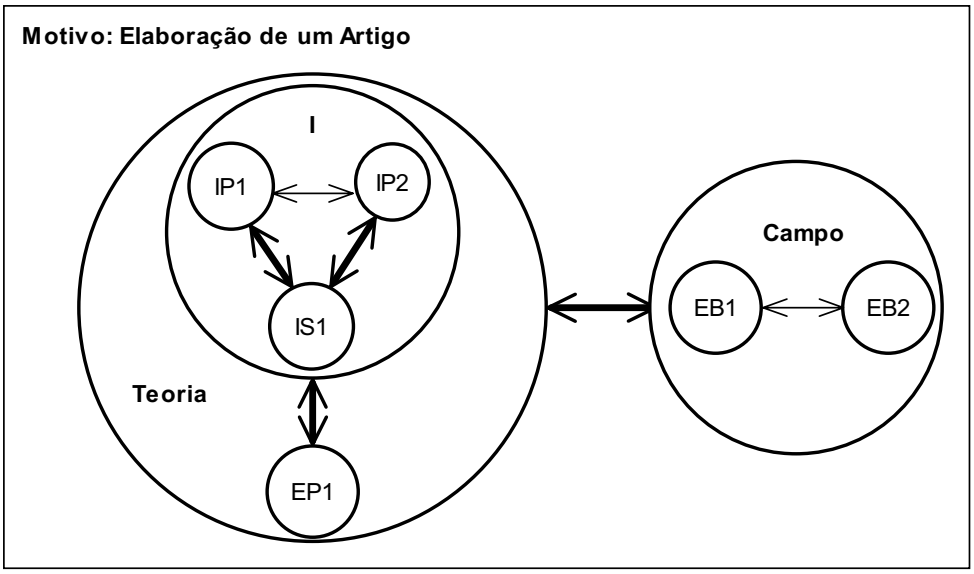

Figura 7 - Modelo colaborativo da elaboração de um artigo: perspectiva Centrada nas Atividades

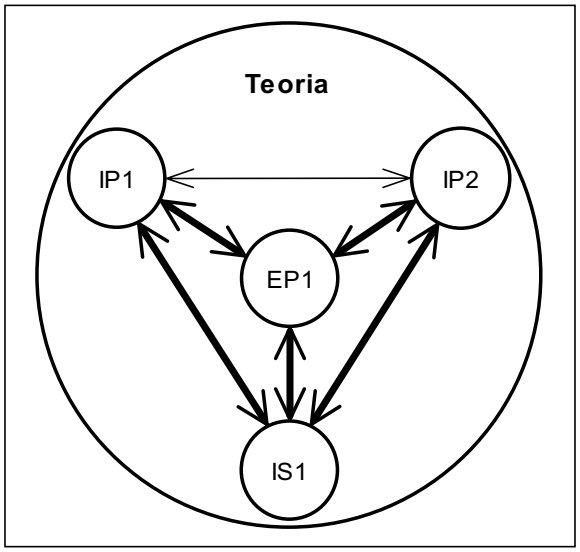

Figura 8 - Nova configuração do grupo Teoria

É importante reparar no papel desempenhado por EP1. Claramente, EP1 está sendo considerado de uma forma um pouco diferente dos outros membros do grupo Teoria, provavelmente agindo mais como um consultor para questões de

engenharia. Contudo, se considerássemos que EP1 tem as mesmas bases e cultura dos outros membros do grupo Teoria, capaz de ter com eles o mesmo nível de discussão, provavelmente seria melhor integrá-lo no mesmo grupo, ficando o novo grupo Teoria como o ilustrado na Figura 8.

3.2.4. Metamodelo Centrado nas Atividades: Algumas Conclusões

A partir dos exemplos acima, observamos a utilidade deste metamodelo Centrado nas Atividades, capaz de acomodar diferentes perspectivas e permitir ao projetista testar modelos e escolher aquele que melhor atende a seus requisitos. Observamos também que as necessidades pessoais e os recursos do espaço físico, tal como um espaço de trabalho compartilhado, orientam a configuração do modelo.

Outro aspecto importante que norteia a topologia do modelo é a informação que deve ficar disponível em cada nó definido para o modelo. Ao elaborar um modelo, é importante considerar não somente os aspectos topológicos, como também aqueles que envolvem a colaboração em si, ou seja, como os membros do grupo podem cooperar melhor por meio da produção, manipulação e organização de informações, além da construção e do aprimoramento de objetos de cooperação.

Também é importante saber reconfigurar o modelo, pois parece que algumas configurações funcionam melhor do que outras. Sem um reconfigurador automático, deve ser interessante ao menos derivar algumas configurações predefinidas adequadas, que pudessem ser selecionadas ao se iniciar uma aplicação colaborativa.

Deve-se destacar ainda que este metamodelo de multi-perspectiva não constitui, intrinsecamente, um metamodelo diferente quando comparado a muitos outros já desenvolvidos, de forma que podemos utilizar muitas linguagens de especificação disponíveis para descrevê-lo, como veremos mais adiante neste capítulo. O que parece ser a maior contribuição deste metamodelo é sua capacidade expressiva, que ajuda a esclarecer, por meio de uma representação gráfica simples, as relações entre os grupos envolvidos na atividade principal que está sendo realizada. Observamos com usuários reais que essa representação

simples lhes trouxe uma melhor compreensão do problema, além de estimulá-los a discutir e propor modificações na configuração dos modelos. As setas finas e grossas das arestas também transmitem rapidamente a noção de quem está localizado local ou remotamente em relação aos demais, além de indicar a necessidade de requisitos menos ou mais sofisticados em termos da rede de comunicação.

Por meio de uma análise detalhada, identificamos os três componentes do modelo 3C em nosso metamodelo Centrado nas Atividades, a saber: comunicação, coordenação e cooperação (Ellis et al., 1991; Fuks et al., 2005). O componente de cooperação às vezes recebe denominações diferentes, mas que representam os mesmos aspectos: colaboração (Ellis et al., 1991), produção (Laurillau & Nigay, 2002) e acoplamento de trabalho (Neale et al., 2004).

O componente de comunicação é o relacionado ao intercâmbio de mensagens e informações entre as pessoas (Fuks et al., 2005). Em nosso metamodelo, ele é representado pelas arestas que ligam os nós.

O componente de cooperação é a operação conjunta que ocorre durante uma sessão em um espaço compartilhado. Os membros de um grupo cooperam produzindo, manipulando e organizando informações, além de desenvolvendo e refinando artefatos. Podemos concluir que, em nosso metamodelo, o componente de cooperação é formado pela associação dos muitos níveis de abstração diferentes, que representam a atividade completa que está sendo realizada durante a sessão colaborativa, incluindo todas as aplicações que são executadas nos nós folhas e os artefatos que são produzidos ou manipulados – isto é, o espaço de trabalho que está sendo modelado. Todos esses elementos estão sendo armazenados durante a sessão colaborativa e podem ser utilizados como uma base de conhecimentos, seja para uma auditoria ou para uma reconfiguração do modelo em sessões futuras.

Finalmente, o componente de coordenação está relacionado ao gerenciamento de pessoas, suas atividades e recursos (Raposo & Fuks, 2002). Em uma atividade em grupo fracamente acoplada, na qual a seqüência de processamento é resultado das ações realizadas pelos participantes, a coordenação é feita implicitamente. Por outro lado, se o trabalho envolve atividades fortemente acopladas, há a necessidade de prescrever a ordem das ações, como ocorre em sistemas com fluxo de trabalho sistemático (Pankoke-Babatz & Syri, 1997).

Nosso metamodelo, com os elementos que foram considerados, é capaz de representar trabalho em grupo pouco acoplado. Para que o metamodelo também dê conta de atividades muito acopladas, identificamos a necessidade de introduzir novos elementos, que corresponderão à componente de coordenação do modelo 3C e serão descritos na próxima seção: *elementos de especialização das arestas, regras de papéis* e *tabela de atributos de mensagens*.

3.3. Metamodelo Centrado nas Atividades: Componentes de Coordenação

Conforme mencionado, identificamos a necessidade de introduzir novos elementos no metamodelo, correspondentes à componente de coordenação do modelo 3C, que descrevemos nas subseções que se seguem: *elementos de especialização das arestas, regras de papéis* e *tabela de atributos de mensagens*.

3.3.1. Elementos de Especialização das Arestas

Analisando uma mensagem através do caminho de interação desde um nó remetente até um nó receptor, identificamos alguns requisitos adicionais que nosso metamodelo deveria incluir. Suponhamos, por exemplo, uma situação em que há um grupo de engenheiros que utilizam computadores *desktop* e enviam um vídeo para outro grupo de engenheiros, localizados em uma sala de visualização, com uma tela de exibição maior. Seria interessante se o segundo grupo pudesse ajustar automaticamente algumas propriedades de vídeo para suas condições tecnológicas antes de exibi-lo na tela da sala de visualização. Isso poderia ser feito por meio do que denominamos *módulo de processamento pós-comunicação*, executado pelo nó receptor imediatamente após receber a mensagem através do canal de comunicação e imediatamente antes de encaminhá-la aos membros do grupo.

Outro exemplo é o uso de uma ferramenta de captura de vídeo durante uma sessão de simulação colaborativa. Dentro do grupo técnico, todos os membros devem receber todos os quadros de um simulador que é executado em um nó folha. Por outro lado, para os gerentes que fazem parte do grupo de tomadores de decisões, seria suficiente receber apenas um de cada dez quadros. Aqui, a solução

seria adicionar um *módulo de processamento pré-comunicação* a ser executado pelo nó remetente, como se fosse um filtro, imediatamente antes de enviar a mensagem (um quadro) através do canal de comunicação.

Um terceiro e mais abrangente exemplo seria o envio de uma mensagem de um grupo de brasileiros para um grupo de chineses, sendo o inglês a língua comum. No caso deste exemplo, consideremos que não há um tradutor português-chinês-português disponível. Neste caso, seria preciso empregar processamento pré e pós-comunicação. O primeiro seria executado pelo nó remetente, que traduziria a mensagem do português para o inglês imediatamente antes de enviá-la através do canal de comunicação. A mensagem trafegaria então pelo canal de comunicação e chegaria ao nó de destino. Lá, um módulo de processamento pós-comunicação traduziria a mensagem do inglês para o chinês, enviando-a aos receptores.

A partir dos exemplos acima, concluímos que seria interessante dividir esses módulos de processamento pré e pós-comunicação em duas classes diferentes:

- A primeira classe é constituída pelos módulos de pré e pós-processamento associados aos nós folhas. Eles representam os módulos de processamento a serem executados particularmente sobre uma mensagem específica enviada entre dois nós, os quais ficam armazenados em uma tabela especial com os identificadores (*id_mensagem*, *receptor*), como será descrito na Subseção 3.3.3.
- A segunda classe é a constituída pelos módulos de pré e pós-processamento associados a nós grupos em diferentes níveis da hierarquia do metamodelo, representando as políticas desses grupos, respectivamente, ao enviar (políticas de envio) e receber (políticas de recebimento) mensagens. Visto que não há políticas particulares relacionadas a uma mensagem específica, mas sim políticas mais gerais associadas a grupos em diferentes níveis do metamodelo, elas podem ser armazenadas no próprio metamodelo.

Para ilustrar melhor esta idéia, apresentamos na Figura 9 os diferentes módulos de processamento pré e pós-comunicação que podem ser executados quando uma mensagem é enviada de um Pesquisador de Informática PI1, do

Departamento de Informática DI1 da Universidade U1, para um Pesquisador de Informática PI2, do Departamento de Informática DI2 da Universidade U2.

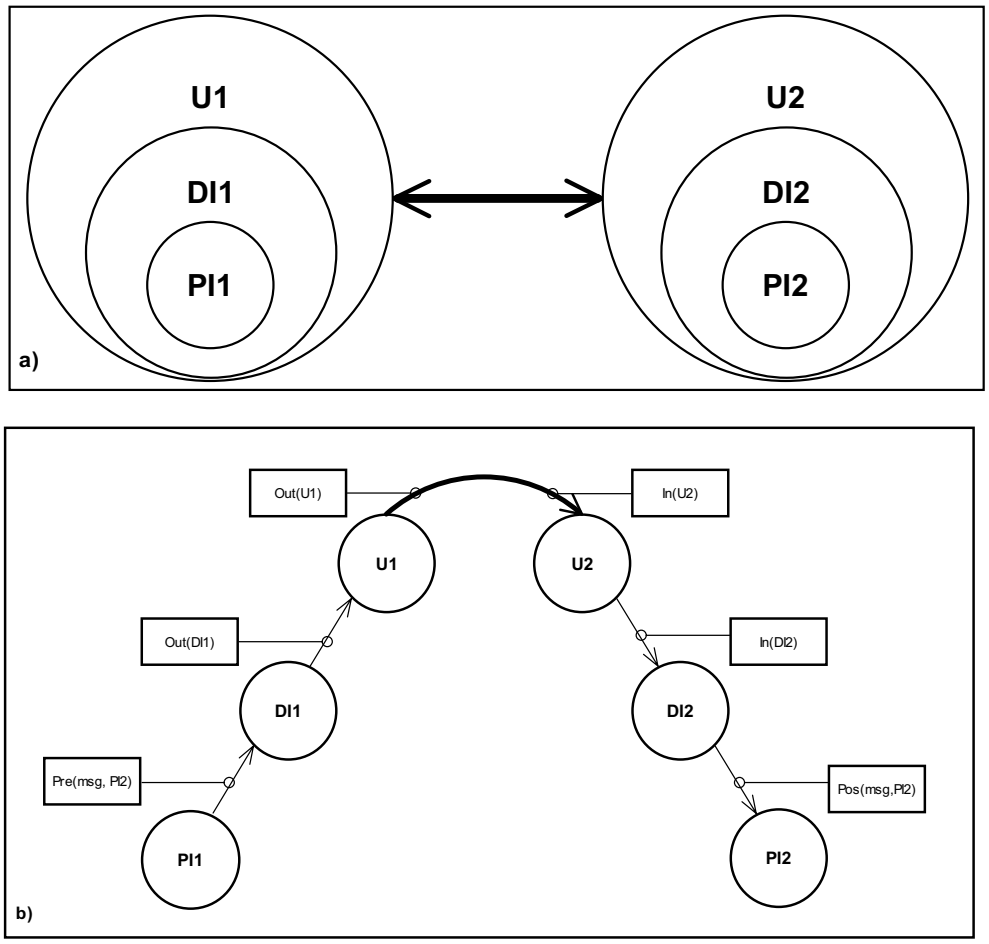

Figura 9 - Metamodelo Centrado nas Atividades, processamento pré e pós-comunicação: a) ponto-de-vista do metamodelo; b) ponto-de-vista do pré e pós-processamento

Na Figura 9 é possível acompanhar a seqüência dos módulos de processamento executados:

- O primeiro módulo de pré-processamento a ser executado é o associado com a mensagem em particular que está sendo enviada de PI1 para PI2.
- Então, as políticas de envio associadas ao departamento DI1 são executadas.
- Finalmente, as políticas de envio associadas à universidade U1 são executadas.
- Então a mensagem é enviada através da aresta de comunicação específica (que liga os nós superiores de cada lado).

- Agora, do lado receptor da aresta, o primeiro módulo de pós-processamento a ser executado são as políticas de recebimento associadas à universidade U2.
- Então, as políticas de recebimento associadas ao departamento DI2 são executadas.
- Finalmente, é executado o terceiro módulo de pós-processamento, associado com a mensagem em particular que está sendo enviada de PI1 para PI2.

Analisando este exemplo, surgem duas questões importantes. A primeira diz respeito a quem executará esses módulos de processamento pré e pós-comunicação. Em relação à aresta, do lado do remetente, o candidato natural para executar os módulos de pré-processamento é o nó folha que está enviando a mensagem. Do lado do receptor, isso poderia ser feito adicionando-se um atributo ao primeiro nó para o qual a aresta aponta (no exemplo, a universidade U2), que corresponde ao nó folha desse grupo que executará os módulos de pós-processamento.

A segunda questão envolve os algoritmos a serem executados quando uma mensagem é enviada, tanto do lado do remetente quanto do receptor. Apresentaremos esses algoritmos mais adiante, após definir os outros dois componentes de coordenação de nosso metamodelo, nas próximas subseções: as regras de papéis e a tabela de atributos de mensagens.

Os elementos de especialização das arestas aumentam a flexibilidade do metamodelo em dois aspectos. Um é a concentração de valores de atributos nos módulos que podem ser modificados dinamicamente em tempo de execução. O segundo aspecto enfatiza a adequação do modelo aos protocolos sociais (Morris, 2004b) envolvendo aplicações colaborativas. Como exemplo desses protocolos sociais, em nosso metamodelo o processamento pós-comunicação transfere para uma folha de um nó grupo receptor a responsabilidade de decidir a quais membros deste grupo a mensagem recebida será encaminhada. Isso parece razoável, visto que ela conhece bem o grupo e foi por ele designado para assumir esse papel. Outro exemplo: retomando o exemplo prático já visto, sobre a ferramenta de captura de vídeo, o grupo remetente poderia considerar mais educado e conveniente enviar todos os quadros do vídeo para o grupo de tomadores de

decisões e deixá-lo decidir se deseja cortar alguns quadros ou não (o filtro seria incluído no processamento de pós-comunicação). É claro que essa segunda abordagem aumentaria a carga da rede, mas esse seria o preço de um protocolo social mais "diplomático".

Outros exemplos práticos de uso de pré e pós-processamento:

- Conversão de formatos de arquivos (ex.: CAD) de um sistema de um lado da aresta para outro sistema do outro lado. Isto poderia ser feito em qualquer um ou em ambos os lados se, por exemplo, um arquivo neutro fosse transmitido através do canal de comunicação.
- Transformação de coordenadas UTM para XYZ (em qualquer um dos lados).
- Procedimentos de detecção de vírus e políticas de *firewall* (em qualquer um dos lados).

Para finalizar esta subseção, é interessante destacar que esses conceitos de pré e pós-processamento são geralmente levados em conta em estudos sobre aplicações colaborativas, embora nem sempre seguindo a mesma abordagem. Por exemplo, empregando a mesma abordagem, Cortés e Mishra (1996) utilizam funções de *pré* e *pós-execução* em seus programas de coordenação DCWPL. David e Borges (2001) também propõem o uso de filtros de entrada e saída pela aplicação e pelo usuário para assegurar a privacidade das informações, selecionar os eventos, informações de percepção (*awareness*) e/ou para modificar o processamento desses eventos. Dourish (1998) usa esses mesmos conceitos no conjunto de ferramentas Prospero, denominando-os *métodos anteriores* e *métodos posteriores*, os quais são executados pelo conjunto de ferramentas. Finalmente, Stevens e Wulf (2002), empregando uma abordagem diferente, utilizam os termos *ex-ante*, *uno-tempore* e *ex-post*, associados ao momento em que as permissões de controle de acesso são definidas.

3.3.2. Regras de Papéis

Os estudos em CSCW têm empregado *regras de papéis* para a estrutura de coordenação por mais de uma década. Por exemplo, Furuta e Stotts (1994)

apresentam uma evolução do modelo Trellis na qual os protocolos de interação entre os grupos são representados de forma separada dos processos de interface que utilizam esses protocolos para a coordenação. Os protocolos são interpretados, de modo que as interações entre os grupos podem ser modificadas à medida que a tarefa em colaboração progride. As alterações podem ser feitas ou por uma pessoa, editando as especificações do protocolo em tempo real, ou por um processo de "observação silenciosa". O Trellis combina navegação em hipermídia com suporte à colaboração – um *hiperprograma* – com esse modelo baseado sobretudo em uma rede de Petri temporizada por transição, executada sincronamente, que é a estrutura do hiperprograma. A notação de rede utilizada é denominada CTN (*colored timed nets* – redes coloridas com indicação de tempo). Um hiperprograma Trellis é completado por meio da inserção de anotações sobre os componentes da CTN: a CTN é a descrição da tarefa e as diferentes anotações são as informações necessárias para realizar a tarefa. Uma categoria de anotação importante é o *conteúdo*. Fragmentos de informação (textos, gráficos, vídeos, áudio, códigos executáveis, outros hiperprogramas) são associados a lugares na CTN. Outra categoria de anotação são os *eventos*, que são mapeados nas transições da CTN. Uma terceira categoria de anotação é o par *atributo/valor (A/V)*. Uma lista de pares A/V é mantida em cada local, cada transição, cada arco e para a CTN como um todo.

Edwards (1996) apresenta o Intermezzo, sistema que permite aos usuários controlar a colaboração por meio de *políticas* que funcionam como regras gerais para restringir e definir o comportamento do sistema, em reação ao estado do mundo. As políticas são descritas em termos dos direitos de controle de acesso aos objetos de dados, e são associadas a grupos de usuários denominados *papéis.* Os papéis representam não somente conjuntos de usuários definidos estaticamente, como também descrições dinâmicas de usuários que são avaliadas enquanto as aplicações são executadas. Esse aspecto em tempo de execução dos papéis permite que eles reajam de forma flexível ao dinamismo inerente à colaboração. O Intermezzo oferece mecanismos para criar papéis estáticos e dinâmicos, uma implementação de políticas sobre o sistema de controle de acesso "bruto" e uma linguagem de especificação declarativa para os papéis e as políticas que pode ser utilizada para controlar e configurar o subsistema de políticas. Enquanto os papéis estáticos são implementados por meio de listas simples de controle de acesso, os

papéis dinâmicos são implementados associando-se uma função de predicado a um conjunto de direitos de controle de acesso.

Para Cortés e Mishra (1996), um programa colaborativo deve ser dividido em dois componentes principais: um programa computacional que modele os artefatos compartilháveis e um programa de coordenação que especifique a forma como esses artefatos devem ser compartilhados. Os programas de coordenação podem ser facilmente modificados, muitas vezes sem nenhuma alteração no programa computacional. Os autores desenvolveram uma linguagem de coordenação – DCWPL (*Describing Collaborative Work Programming Language* – Linguagem de Programação para a Descrição de Trabalho Colaborativo – pronunciada "*decouple*") – e seu intérprete em tempo de execução para que os programadores criem programas de coordenação, especificando mecanismos de controle separados dos programas computacionais, que são escritos em uma linguagem de programação tradicional. Um programa de coordenação em DCWPL é composto de uma ou mais primitivas de coordenação: artefatos, papéis, armazenamento, funções de coordenação, políticas e gerenciamento de sessão.

Li e Muntz (1998) propõem a COCA (*Collaborative Object Coordination Architecture* – Arquitetura de Coordenação de Objetos Colaborativos) para ser um *framework* genérico para o desenvolvimento de sistemas colaborativos evolutivos e a modelagem de políticas de coordenação. Assim como nos três trabalhos já mencionados, eles também são a favor de separar coordenação e computação. COCA oferece aos desenvolvedores uma linguagem de especificação baseada em lógica para modelar as políticas de coordenação. Nessa arquitetura, os participantes de uma colaboração são divididos por papéis e são definidas regras ativas para cada papel, de modo a resguardar suas interações com os outros colaboradores. Essas políticas são interpretadas em tempo de execução pela máquina virtual de COCA, da qual uma cópia é executada na máquina de cada participante. Essa abordagem torna conveniente fazer alterações tanto durante o desenvolvimento como em tempo de execução. As políticas de COCA incluem não somente controle de acesso como também controle de sessão, controle de concorrência, tomada de vez, etc.

Roussev et al. (2000) adotam uma abordagem baseada em componentes para separar dados e controle. O estado compartilhado de um componente de dados é modelado como propriedades do componente, e as alterações sobre seu

estado são modeladas como eventos de alteração das propriedades. O projeto componível baseia-se em padrões de programação que eliminam a ligação implícita entre a estrutura lógica de um objeto compartilhado e abstrações particulares definidas pelo sistema, o que aumenta a gama de objetos suportados e permite a extensibilidade.

Laurillau e Nigay (2002) apresentam o modelo arquitetônico Clover, resultante da combinação da abordagem em camadas da arquitetura genérica de Dewan (1999) com a decomposição funcional do modelo Clover. O modelo Clover define três classes de serviços que uma aplicação *groupware* pode suportar: serviços de produção, comunicação e coordenação. Essas três classes de serviços podem ser encontradas em cada camada funcional do modelo. A partição funcional de Clover estabelece um mapeamento direto entre os conceitos de projeto e a modelagem da arquitetura de software. Essa partição funciona como um guia na organização das funcionalidades identificadas durante a fase de projeto. Além disso, ela complementa a partição tradicional das funcionalidades em componentes, na qual cada um corresponde a um nível de abstração – por exemplo, na arquitetura de Dewan. De fato, para um componente correspondente a um nível de abstração, o metamodelo Clover defende três sub-componentes dedicados à produção, comunicação e coordenação. Essa implementação modular está de acordo com as propriedades de modificação, reutilização e extensibilidade. No metamodelo Clover, as funcionalidades de coordenação, produção e comunicação são todas tratadas da mesma forma, diferentemente de outros estudos, como a arquitetura de Dewan, na qual as funcionalidades de coordenação são tratadas de forma especial, mas não as de produção e comunicação. A arquitetura conceitual é mapeada para uma arquitetura de implementação designando processos para os componentes; os processos, por sua vez, são atribuídos aos computadores. Podem ser aplicadas diferentes abordagens de distribuição a uma arquitetura conceitual. A distribuição e a decomposição de camadas (ou decomposição de componentes de software) são dois mecanismos ortogonais. Em particular, uma camada compartilhada no nível conceitual não implica um ponto central no nível da implementação.

Então, Yang e Li (2004) retomam a abordagem baseada em componentes empregada previamente por Roussev et al. (2000), também separando dados e controle, mas agora suportando protocolos adaptáveis de consistência em sistemas

colaborativos. Ao separar claramente dados e controle, eles permitem que os protocolos de consistência sejam vinculados dinamicamente aos dados compartilhados no nível do objeto. Os protocolos podem ser comutados em tempo de execução sem modificar o código fonte.

Em consonância com todos esses estudos, exceto de algum modo pelo modelo Clover, decidimos adotar a estratégia de separar a estrutura de coordenação e o programa computacional. Em nosso metamodelo, utilizamos uma linguagem de especificação baseada em lógica para especificar as políticas de coordenação. Forneceremos mais detalhes sobre essa linguagem de especificação no Capítulo 5. Por ora, destacamos os seguintes pontos:

- Declaramos uma barra de colaboração, que é usada para conectar todos os participantes desta colaboração; a barra de colaboração deve ter pelo menos uma declaração de canal. Permitimos a execução de muitas colaborações diferentes ao mesmo tempo, cada uma com sua barra de colaboração correspondente.
- Os papéis são definidos especificando-se comportamentos e restrições individuais em diferentes conjuntos de regras lógicas.
- A comunicação entre os participantes ocorre através de um ou mais canais de mensagens associados a uma barra de colaboração.
- As tarefas básicas de recebimento e envio de mensagens são realizadas por:
    - Para receber mensagens, utilizamos uma regra ativa chamada *on-arrive*, com os argumentos *canal*, *receptor*, *id_mensagem* (e *remetente*).
    - Para enviar mensagens, usamos uma fórmula *send* com os argumentos *canal*, *remetente, id_mensagem* (e *receptor*).

Explicamos por que os últimos argumentos dessas tarefas estão entre parênteses:

- Para a tarefa de receber mensagens, *remetente* é opcional. Para a tarefa de enviar mensagens, se por exemplo for adotado *multicast* de IP (*Internet Protocol* – Protocolo de Internet) como o modelo de comunicação do grupo, a mensagem é enviada a todos os receptores,

tornando o argumento *receptor* desnecessário (no caso de um serviço *unicast*, ele seria necessário).

- No entanto, o principal motivo para pôr esses argumentos entre parênteses é que, para aumentar a flexibilidade, decidimos construir uma *tabela de atributos de mensagens* que inclui o *remetente*, o *receptor* e outros atributos de cada mensagem, como veremos na próxima subseção. Nesse caso, a alteração do receptor de uma mensagem seria feita apenas alterando-se uma linha da tabela, sem modificar o programa de coordenação. Fizemos também algumas modificações na regra e na fórmula usada na COCA (Li & Muntz, 1998), as quais se transformaram respectivamente em:
  - *on-arrive*(canal,tab_mensagem(*dummy*,id_mensagem,receptor));
  - *send*(canal,tab_mensagem(remetente,id_mensagem,*dummy*)).

Um típico programa de coordenação de nosso metamodelo teria um formato semelhante ao apresentado na Tabela 1. Nele, *self* é um *functor* que denota o participante atual, *source* associa a identificação do participante com a mensagem e *display* é empregado para exibir o conteúdo da mensagem na tela.

Acompanhando esse programa de coordenação linha a linha, temos:

- A primeira linha definindo a colaboração, com o nome *simulacao_integrada*.
- A segunda linha definindo uma barra de colaboração com apenas um canal *remoto*.
- A terceira linha abrindo um conjunto de regras correspondentes ao papel *tecnico_0* (que pode ser instanciado pelos participantes durante a execução da colaboração).
- E então, definimos as três regras que constituem o papel *tecnico_0*:
  - A primeira regra, *on-init*, é disparada quando a colaboração *simulacao_integrada* é iniciada. Quando isso ocorre, uma mensagem com *id_mensagem* 1 é enviada ao receptor determinado pela linha na tabela de atributos de mensagens com chaves *remetente*=*source*(*self*) (i.e., o participante que está instanciando o papel de *tecnico_0*) e *id_mensagem*=1.

- A segunda regra, *on-arrive*, é ativada quando uma mensagem com *id_mensagem* 2 é recebida por *source*(*self*). Quando isso ocorre, a mensagem é exibida no console do participante.
- Finalmente, a terceira regra, outra *on-arrive*, é ativada quando uma mensagem com *id_mensagem* 3 é recebida por *source*(*self*). Quando isso ocorre, a mensagem é exibida no console do participante.

```
collaboration simulacao_integrada
{ collaboration_bus { channel(remoto). }
  role tecnico_0 //técnico responsável pela coordenação das simulações
  {
        on-init(simulacao_integrada) :-
              send(remoto, tab_mensagem(source(self), 1, dummy)).
        on-arrive(remoto, tab_mensagem(dummy, 2, source(self))) :-
              display(tab_mensagem(dummy, 2, source(self))).
        on-arrive(remoto, tab_mensagem(dummy, 3, source(self))) :-
              display(tab_mensagem(dummy, 3, source(self))).
  }
}
```

Tabela 1 - Metamodelo Centrado nas Atividades: programa de coordenação típico

De modo geral, as regras de papéis definem políticas de coordenação para cada papel presente em uma colaboração. Quando essas regras também determinam a ordem na qual os eventos ocorrem, elas também definem as regras do fluxo de trabalho.

3.3.3. Tabela de Atributos de Mensagens

Foram apresentados os módulos de processamento pré e pós-comunicação associados a cada mensagem enviada, então a escolha natural é montar uma tabela na qual possamos definir quais módulos de processamento devem ser executados a cada momento. Isso também facilita a alteração do nome de um módulo de pré ou pós-processamento em particular, mesmo em tempo de execução (além da possibilidade de alterar também o código em tempo de execução antes de ele ser invocado).

Assim, elaboramos uma *tabela de atributos de mensagens* em nosso metamodelo visando aumentar a flexibilidade do programa de coordenação, separando as regras de coordenação dos dados especificamente relacionados com cada mensagem, em consonância com o conceito geral do metamodelo de separar dados e controle, com este acesso indireto permitindo a reconfiguração dinâmica. A única exceção a esse conceito é que decidimos incluir o receptor (relativo ao controle, não aos dados) como uma coluna da tabela. Isso foi motivado pelo fato de que, em fluxos de trabalho fortemente acoplados, parece interessante ao menos alterar os receptores de uma dada mensagem em tempo de execução, sem a necessidade de modificar o programa de coordenação. Por exemplo, uma pessoa do grupo remetente pode não saber exatamente para quem a mensagem deve ser enviada, do lado dos receptores. Então, ela pode enviar a mensagem para o nó abstrato do grupo receptor e permitir que este determine, por meio do módulo de pós-processamento, a quais receptores a mensagem deve ser encaminhada. Outro exemplo seria o de um observador (p.ex., um gerente não responsável por tomar as decisões) que entrasse depois na sessão colaborativa e desejasse receber todas as mensagens a partir daquele momento. Seria uma questão de editar a tabela, incluindo novas linhas que associassem as mensagens futuras ao novo participante. Repare que as arestas que conectam esse novo participante aos demais devem estar previstas e incluídas no modelo antes do início da colaboração, e que o novo participante deve assumir um papel previamente definido que não interfira no processo principal.

A Tabela 2 apresenta as primeiras linhas de uma tabela de atributos de mensagens típica associada a uma aplicação colaborativa fortemente acoplada.

remetente	id_mensagem	receptor	aresta	pré-processamento	pós-processamento
T0	1	T1	a1	Pre_1_T1	Pos_1_T1
T1	2	T0	a1	Pre_2_T0	Pos_2_T0
T1	2	G1	a2	Pre_2_G1	Pos_2_G1
T1	3	T0	a1	Pre_3_T0	Pos_3_T0
T1	3	G1	a2	Pre_3_G1	Pos_3_G1

Tabela 2 - Colunas e primeiras linhas de uma tabela de atributos de mensagens típica

As chaves desta tabela são *remetente*, *id_mensagem* e *receptor*. Conforme já visto na Subseção 3.3.2, entramos na tabela utilizando o par (*remetente*, *id_mensagem*) ao enviar uma mensagem e utilizamos o par (*id_mensagem*, *receptor*) para receber uma mensagem.

3.3.4. Algoritmos de Remetente e Receptor

Apresentamos agora os algoritmos executados quando uma mensagem é enviada, tanto do lado do remetente quanto do receptor.

Observamos primeiro que cada mensagem tem um remetente inicial, que é necessariamente um nó folha, e um ou mais receptores finais, que podem ser folhas ou grupos. Já tendo apresentado as regras de papéis e a tabela de atributos de mensagens, descrevemos agora os algoritmos a serem executados de cada um dos lados, como mostrado na Tabela 3.

Consideremos a tabela de atributos de mensagens apresentada na subseção anterior (Tabela 2) para analisar os algoritmos e estabelecer em que momentos os módulos de pré e pós-processamento são executados para cada mensagem. É importante enfatizar duas questões:

- G1 (Grupo 1) não é um nó folha, mas um nó grupo. De acordo com os algoritmos, o módulo de pós-processamento *Pos_2_G1* é responsável por determinar a quais nós folhas do grupo G1 a mensagem (2, G1) deve ser encaminhada.
- Se por um lado enviar a mesma mensagem para cada receptor confere flexibilidade a nosso metamodelo, com a possibilidade de executar módulos de processamento pré e pós-comunicação particulares para cada receptor, além de várias vantagens, como as mencionadas nos exemplos fornecidos na Subseção 3.3.1, por outro é preciso considerar o impacto dessa estratégia na performance. Conforme relatam Li e Muntz (1998), os primeiros modelos eram fortemente baseados no agrupamento de múltiplas comunicações ponto-a-ponto. Nesse caso, um pacote com *n* receptores pretendidos deve ser enviado pelo remetente *n* vezes, independentemente da localização física das partes envolvidas. Essa abordagem produz custo extra tanto para os remetentes da mensagem quanto para os canais de comunicação, e a latência aumenta com o tamanho da população de receptores. Portanto, é preciso equilibrar nossas necessidades ao escolher a estratégia a ser usada na aplicação

colaborativa. Por exemplo, nosso algoritmo parece ser adequado se houver poucos nós e muita diversidade entre os nós. Por outro lado, poderia ser mais eficiente escolher um modelo de comunicação com *multicast* de IP se houver muitos nós com não muita diversidade entre eles.

sender (*remetente*, *receptor*, *flag*)
- *until* o *receptor* ser encontrado *repeat*
  - no nível atual, procura a sub-árvore que contém o *receptor*
  - *if* o *receptor* for encontrado (e todo o caminho do *remetente* até o *receptor* for determinado)
    - *if flag = na_tabela*
      - executa o módulo de pré-processamento associado com o par (*mensagem*, *receptor*)
    - *else*
      - cria uma nova linha na tabela de atributos de mensagens com o par (*mensagem*, *receptor*), indicando o módulo de pós-processamento a ser executado
    - executa todas as políticas de envio associadas aos grupos nos níveis do caminho que começa no *remetente* até que seja atingida a aresta de comunicação
    - envia a mensagem com o *receptor* para o nó folha designado no atributo de pós-processamento do nó grupo *receptor*, ou para o próprio *receptor*
  - *else*
    - vai para o nível superior

Lado do remetente da aresta de comunicação (executado pelo nó folha *remetente_inicial*):
- *for* cada *receptor_final* associado com a mensagem
  - *sender* (*remetente_inicial*, *receptor_final*, *na_tabela*)

Lado do receptor da aresta de comunicação:
- recebe a mensagem
- executa todas as políticas de recebimento associadas aos grupos nos níveis do caminho que começa no nó atual até que seja atingido o nó *receptor_final*
- *if receptor_final* é um nó folha
  - *if receptor_final* é igual ao nó de execução de pós-processamento
    - executa o módulo de pós-processamento associado com o par (*mensagem*, *receptor_final*)
  - *else*
    - envia a mensagem para *receptor_final*
- *else*
  - executa o módulo de pós-processamento associado com o par (*mensagem*, *receptor_final*), que neste caso deve determinar o(s) nó(s) folha ou grupo(s) que receberá(ão) a mensagem
  - *for* cada nó determinado acima (*no_corrente*)
    - *sender* (*receptor_final*, *no_corrente*, *nao_na_tabela*)

Tabela 3 - Metamodelo Centrado nas Atividades: algoritmos de remetente e receptor

Descrevemos agora os dois algoritmos, do lado do remetente e do lado do receptor, apresentados na Tabela 3. Primeiro detalharemos um método comum, chamado *sender*, que é utilizado dos dois lados. O objetivo básico deste método é encontrar, no componente de rede de nosso metamodelo, o nó receptor que receberá a mensagem, determinando o caminho completo do remetente até o receptor. O método tem três argumentos: *remetente*, *receptor* e uma *flag* que determina se a mensagem atual está ou não na tabela de atributos de mensagens. Ele possui um laço principal *until...repeat* para encontrar o *receptor*, que é executado começando pela busca do *receptor* na sub-árvore definida pelo nível imediatamente acima do nível do *remetente* (um nó folha), indo para cima

executando o mesmo procedimento até que se encontre o *receptor*. Quando isso ocorre, há duas situações possíveis:

- A mensagem já está na tabela de atributos de mensagens:
  neste caso, o módulo de pré-processamento associado com o par (*id_mensagem*, *receptor*) é executado.
- A mensagem não está na tabela de atributos de mensagens:
  isso significa que se trata de uma nova mensagem que está sendo criada em tempo de execução por um nó grupo do lado do receptor, definindo uma nova linha na tabela que inclui o módulo de pós-processamento.

Finalizando o método *sender*, há outros dois passos:

- O primeiro executa todas as políticas de envio associadas aos grupos nos níveis do caminho que começa no *remetente* até que seja atingida a aresta de comunicação.
- O segundo finalmente envia a mensagem com o *receptor* para o nó folha designado no atributo de pós-processamento do nó grupo *receptor*, ou para o próprio *receptor* (se for um nó folha).

Descrevemos agora o algoritmo do lado do remetente. Com o método *sender* definido acima, este algoritmo, executado pelo nó folha *remetente_inicial*, é simplesmente um laço *for* que repete o passo a seguir para cada *receptor_final* (determinado nas linhas da tabela de atributos de mensagens) associado com a mensagem atual que está sendo enviada:

- *sender* (*remetente_inicial*, *receptor_final*, *na_tabela*).

Podemos ver que o método é invocado com *flag=na_tabela*, o que significa que essas mensagens já estão predefinidas na tabela de atributos de mensagens.

Finalmente, descrevemos agora o algoritmo do lado do receptor:

- O primeiro passo, executado pelo nó designado no atributo de pós-processamento do nó grupo *receptor* ou pelo próprio *receptor* (se for um nó folha), consiste exatamente em receber a mensagem.

- O segundo passo executa todas as políticas de recebimento associadas aos grupos nos níveis do caminho que começa no nó atual até que seja atingido o nó *receptor_final*.
- O terceiro e último passo principal é executado de forma diferente dependendo de se o *receptor_final* é ou não é um nó folha:
  - Se o *receptor_final* é um nó folha:
    - O *receptor_final* pode ser o próprio nó de execução do pós-processamento, quando simplesmente executa o módulo de pós-processamento associado com o par (*id_mensagem*, *receptor_final*), ou pode não ser, caso em que o nó de execução do pós-processamento ainda precisa enviar a mensagem ao nó folha *receptor_final*.
  - Se o *receptor_final* é um nó grupo, são executados dois passos:
    - O módulo de pós-processamento associado com o par (*id_mensagem*, *receptor_final*) é executado, caso em que deve determinar o(s) nó(s) folha(s) ou grupo(s) para receber a mensagem.
    - Um laço *for* que repete o seguinte passo para cada nó determinado no passo acima (o *no_corrente*) é executado:
      - *sender* (*receptor_final*,*no_corrente*,*nao_na_tabela*).

      Podemos ver que, neste caso, o método *sender* é invocado com *flag=nao_na_tabela*, o que indica que se tratam de mensagens novas que estão sendo criadas em tempo de execução pelo nó grupo.

3.4. Metamodelo Centrado nas Atividades: Linguagem de Especificação para o Componente de Rede

Como foi visto na Seção 3.2, o componente principal de nosso metamodelo Centrado nas Atividades é a rede constituída por nós e arestas que definem, respectivamente, os grupos que realizam a atividade e os caminhos da interação entre eles.

É preciso utilizar uma linguagem de especificação para descrever a rede, escolhendo dentre muitas linguagens de especificação disponíveis, tais como *Process Algebra* (Milner, 1989), *Petri Nets* (Murata, 1989), *Unified Modeling Language* (UML) *Diagrams* (UML, 2006) e *Use Case Maps* (UCM) (Buhr & Casselman, 1996). Escolhemos a linguagem de especificação proposta (Russo, 1988; Santos, 1986) para o projeto EITIS (*Environment of Integrated Tools for Interactive Systems* – Ambiente de Ferramentas Integradas para Sistemas Interativos), da PUC-Rio (Melo, 1987), principalmente por duas razões:

- Sua simplicidade, inspirada em uma notação BNF (Backus-Naur-Form) modificada, que nos permite manter o foco no trabalho que está sendo desenvolvido.
- O fato de que já a havíamos utilizado em um projeto anterior do Departamento de Informática da PUC-Rio.

A seguir, descrevemos brevemente a notação utilizada em nossos blocos de especificação:

- Em caixa alta, representamos as entidades e os relacionamentos, assim como os atributos que estão sendo descritos e os tipos e funções das entidades.
- Em caixa baixa, representamos os atributos que não estão sendo descritos e a sintaxe associada aos atributos que estão sendo descritos.
- O sinal + indica uma possível ocorrência de mais de uma instância de um objeto específico.
- O sinal | representa um "OU" lógico.
- Os relacionamentos ou atributos entre colchetes podem ou não ocorrer em uma instância específica da entidade que está sendo considerada.

Os blocos de especificação que definem as entidades e os relacionamentos principais do componente de rede de nosso metamodelo são mostrados na Tabela 4.

NÓ: id-nó
DESCRIÇÃO: texto
NOME: texto
TIPO: FOLHA | GRUPO
[PAI: id-nó]
[NÓ DE EXECUÇÃO DE PÓS-PROCESSAMENTO: id-nó]
[COMPUTADOR: id-computador]
[ATRIBUTOS: {nome-atrib tipo-atrib valor-atrib}+] //preferências interface c/usuário, linguagem
[COMPARTILHA: id-artefato+]
[POLÍTICA DE RECEBIMENTO: id-política]
[POLÍTICA DE ENVIO: id-política]

ARESTA: id-aresta
DESCRIÇÃO: texto
DISTÂNCIA: LOCAL | REMOTA
DIREÇÃO: UNIDIRECIONADA | BIDIRECIONADA
REMETENTE: id-nó
RECEPTOR: id-nó
POSSUI: id-canal+

ARTEFATO: id-artefato
DESCRIÇÃO: texto
NOME: texto
TIPO: vários
SINCRONIA: SÍNCRONO | ASSÍNCRONO
PERSISTÊNCIA: PERSISTENTE | VOLÁTIL
[LCA: id-lca] //Lista de Controle de Acesso

CANAL: id-canal
DESCRIÇÃO: texto
NOME: texto
TIPO: vários
SINCRONIA: SÍNCRONO | ASSÍNCRONO
PERSISTÊNCIA: PERSISTENTE | VOLÁTIL

Tabela 4 - Componente de rede: blocos de especificação que definem entidades e relacionamentos

Então, esses blocos de especificação devem ser armazenados em uma estrutura de dados com uma DDL (*Data Description Language* – Linguagem de Descrição de Dados) semelhante à mostrada na Tabela 5 (onde estão descritas somente as entidades principais do componente de rede, i.e., nós e arestas).

```
RECORD no
ITEM id-no NUM 2 KEY
ITEM descricao CHAR 35
ITEM nome CHAR 30
ITEM tipo CHAR 5
ITEM pai NUM 2
ITEM pos NUM 2
ITEM computador NUM 12
.
.

RECORD aresta
ITEM id-aresta NUM 4 KEY
ITEM descricao CHAR 35
ITEM distancia CHAR 6
ITEM direcao CHAR 11
ITEM remetente NUM 2
ITEM receptor NUM 2
.
.
```

Tabela 5 - Componente de rede: Linguagem de Descrição de Dados (DDL)

Finalmente, apresentamos na Tabela 6 os registros de carga para os nós e arestas correspondentes à Linguagem de Descrição de Dados da Tabela 5.

Registros de Carga

Nó

tipo-reg	id-no	descricao	nome	tipo	pai	pos	computador	…

Aresta

tipo-reg	id-aresta	descricao	distancia	direcao	remetente	receptor	…

Tabela 6 - Componente de rede: registros de carga para nós e arestas

3.5. Modelo Centrado nas Atividades: Um Exemplo Simples Completo

Consideremos agora um exemplo simples, porém completo (Figura 10) para ilustrar como descrevemos os três componentes principais de nosso metamodelo: a rede, as regras de papéis e a tabela de atributos de mensagens.

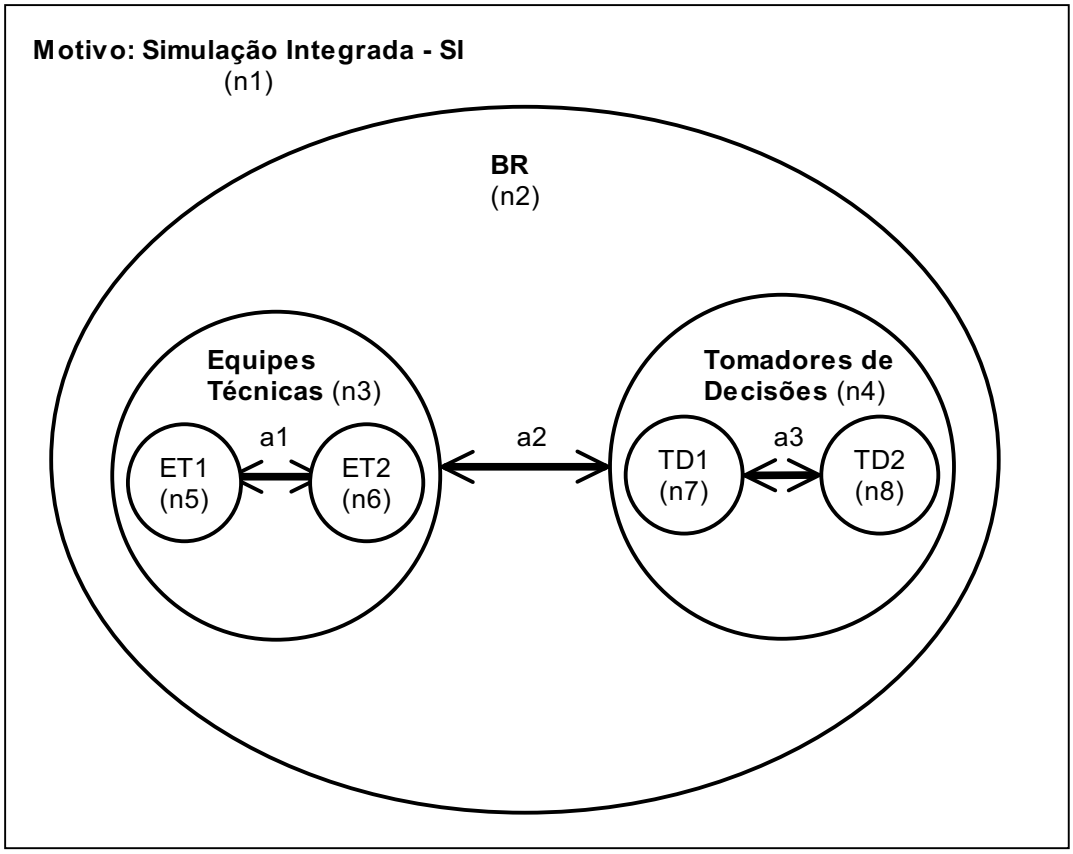

Figura 10 - Um exemplo simples completo de um modelo centrado nas atividades: Simulação Integrada

O nó do nível superior deste exemplo, o motivo, é a Simulação Integrada (n1). Ele tem um nó principal, a companhia de óleo e gás BR (n2), constituída por dois grupos: Equipes Técnicas (n3) e Tomadores de Decisões (n4).

Equipes Técnicas é formado por dois nós folhas, o técnico ET1 (n5) e o técnico ET2 (n6); Tomadores de Decisões é formado por dois outros nós folhas, o tomador de decisões TD1 (n7) e o tomador de decisões TD2 (n8).

Também representadas no componente de rede estão as arestas a1, a2 e a3, que mostram os caminhos da interação entre os nós, todas com setas grossas (remotas).

Os registros de carga desses nós e arestas estão apresentados na Tabela 7.

1 | 1 | Simulação Integrada | SI | GRUPO | 0 | 5 | …
1 | 2 | Companhia de Óleo e Gás | BR | GRUPO | 1 | 5 | …
1 | 3 | Equipes Técnicas | ET | GRUPO | 2 | 5 | …
1 | 4 | Tomadores de Decisões | TD | GRUPO | 2 | 7 | …
1 | 5 | Técnico 1 | ET1 | FOLHA | 3 | 0 | …
1 | 6 | Técnico 2 | ET2 | FOLHA | 3 | 0 | …
1 | 7 | Tomador de Decisões 1 | TD1 | FOLHA | 4 | 0 | …
1 | 8 | Tomador de Decisões 2 | TD2 | FOLHA | 4 | 0 | …
2 | 1 | canal de técnicos | REMOTA | BIDIRECIONADA | 5 | 6 | …
2 | 2 | canal técnico-gerencial | REMOTA | BIDIRECIONADA | 3 | 4 | …
2 | 3 | canal de gerentes | REMOTA | BIDIRECIONADA | 7 | 8 | …

Tabela 7 - Modelo Centrado nas Atividades: registros de carga do componente de rede

Nesta tabela, podemos identificar os oito nós e as três arestas que constituem o componente de rede do modelo de simulação integrada. Em termos dos nós, vemos que os registros definem quais nós são grupos e quais são folhas, quais são os pais dos nós (0, quando o pai é o motivo) e quais são os nós de execução de pós-processamento dos grupos (0, quando os nós são folhas). Por exemplo, a terceira linha (1 | 3 | Equipes Técnicas | ET | GRUPO | 2 | 5 | …) começa definindo que ela se refere a um nó (registro tipo 1), seguido pela sua identificação (3), descrição (Equipes Técnicas) e nome (ET). Então, o tipo de nó é definido como GRUPO e seu pai é definido como o nó com *id*=2 (definido na linha anterior – BR). Finalmente, o último atributo apresentado define o nó que executa o módulo de pós-processamento associado a este grupo (neste caso, o nó com *id*=5 definido duas linhas abaixo – ET1).

Quanto às arestas, identificamos quais são as remotas (neste exemplo, todas) e as locais, quais são as bidirecionadas (novamente, todas as deste exemplo) e as unidirecionadas, e os nós que definem as arestas. Por exemplo, a última linha (2 | 3 | canal de gerentes | REMOTA | BIDIRECIONADA | 7 | 8 | …) começa definindo que ela se refere a uma aresta (registro tipo 2), seguida de sua *id* (3) e descrição (canal de gerentes). Então, a aresta é definida como REMOTA e BIDIRECIONADA. Finalmente, os dois últimos atributos identificam os nós conectados pela aresta (neste caso, os nós com *id* 7 e 8 – TD1 e TD2, respectivamente).

A Tabela 8 apresenta as regras de papéis definidas para *tecnico_1* instanciadas por ET1 do grupo Equipes Técnicas.

```
collaboration simulacao_integrada
{ collaboration_bus { channel(remoto). }
  role tecnico_1 //Técnico 1 de Equipes Técnicas, responsável pela coordenação das simulações
  {
  on-init(simulacao_integrada) :-
    send(remoto, tab_mensagem(source(self), 1, dummy)). //envia "Técnico 2, por favor inicie a simulação."
  on-arrive(remoto, tab_mensagem(dummy, 2, source(self))) :-
    display(tab_mensagem(dummy, 2, source(self))). //exibe "A simulação foi iniciada."
  on-arrive(remoto, tab_mensagem(dummy, 3, source(self))) :-
    display(tab_mensagem(dummy, 3, source(self))), //exibe "Fim da simulação."
    send(remoto, tab_mensagem(source(self), 4, dummy)). //envia "Tomador de Decisões, por favor valide a
                                                        simulação."
  on-arrive(remoto, tab_mensagem(dummy, 5, source(self))) :-
    display(tab_mensagem(dummy, 5, source(self))). //exibe "Simulação validada."
  }
}
```

Tabela 8 - Modelo Centrado nas Atividades: regras de papéis para *tecnico_1*

Acompanhando esse programa de coordenação linha a linha, temos:

- A primeira linha definindo a colaboração, com o nome *simulacao_integrada*.
- A segunda linha definindo uma barra de colaboração com apenas um canal *remoto*.
- A terceira linha abrindo um conjunto de regras correspondentes ao papel *tecnico_1* (que, neste exemplo, é instanciado pelo participante ET1).
- Então, definimos as quatro regras que constituem o papel *tecnico_1*:
  - A primeira regra, *on-init*, é disparada quando a colaboração *simulacao_integrada* é iniciada. Quando isso ocorre, uma mensagem com *id_mensagem* 1 e conteúdo "Técnico 2, por favor inicie a simulação." é enviada ao receptor determinado pela linha na tabela de atributos de mensagens com a chave *sender=source*(*self*) (i.e., ET1) e *id_mensagem*=1. Na Tabela 9, essa linha é a primeira e indica que o *receptor* é ET2.
  - A segunda regra, *on-arrive*, é ativada quando uma mensagem com *id_mensagem* 2 é recebida por *source*(*self*) (i.e., ET1).

Quando isso ocorre, a mensagem é exibida no console do participante: “A simulação foi iniciada.”

- A terceira regra, outra *on-arrive*, é ativada quando uma mensagem com *id_mensagem* 3 é recebida por *source*(*self*) (i.e., ET1). Quando isso ocorre, a mensagem 3 é exibida no console do participante: “Fim da simulação.” Além disso, uma mensagem com *id_mensagem* 4 e o conteúdo “Tomador de Decisões, por favor valide a simulação.” é enviada ao receptor determinado pela linha na tabela de atributos de mensagens com a chave *sender*=*source*(*self*) (i.e., ET1) e *id_mensagem*=4. Na Tabela 9, esta linha é a sexta e indica que o *receptor* é TD.
- Finalmente, a quarta regra, mais uma *on-arrive*, é ativada quando uma mensagem com *id_mensagem* 5 é recebida por *source*(*self*) (i.e., ET1). Quando isso ocorre, a mensagem é exibida no console do participante: “Simulação validada.”

De forma semelhante, poderíamos definir regras de papéis para outros participantes da aplicação colaborativa. Vale notar que, uma vez que neste exemplo as regras de papéis definem a ordem dos eventos, elas também podem ser consideradas regras para fluxo de trabalho.

Finalizamos nosso exemplo simples apresentando a tabela de atributos de mensagens (Tabela 9) com todas as mensagens enviadas pelos participantes, a qual, juntamente com as regras de papéis, define a estrutura de coordenação de nosso metamodelo.

remetente	id_mensagem	receptor	aresta	pré-processamento	pós-processamento
ET1	1	ET2	a1	Pre_1_ET2	Pos_1_ET2
ET2	2	ET1	a1	Pre_2_ET1	Pos_2_ET1
ET2	2	TD	a2	Pre_2_TD	Pos_2_TD
ET2	3	ET1	a1	Pre_3_ET1	Pos_3_ET1
ET2	3	TD	a2	Pre_3_TD	Pos_3_TD
ET1	4	TD	a2	Pre_4_TD	Pos_4_TD
TD1	5	ET1	a2	Pre_5_ET1	Pos_5_ET1

Tabela 9 - Metamodelo Centrado nas Atividades: tabela de atributos de mensagens para o modelo BR

Nesta tabela, vemos também os diferentes módulos de pré e pós-processamento a serem executados ao enviar as mensagens, em particular as mensagens 2 e 3, que têm diferentes módulos de processamento associados a diferentes receptores. Também é importante observar que a mensagem 2 enviada por ET2, a mensagem 3 enviada por ET2 e a mensagem 4 enviada por ET1 destinam-se ao grupo TD. Isso significa que os módulos de pós-processamento *Pos_2_TD*, *Pos_3_TD* e *Pos_4_TD*, respectivamente, todos executados pelo nó de execução de pós-processamento do grupo TD, TD1 (veja a Tabela 7, na qual ele está com *id* 7), devem determinar quais nós folhas do grupo TD devem receber as mensagens. Neste exemplo, definimos que as mensagens 2 e 3 são re-encaminhadas para TD1 e TD2 (visto que são apenas mensagens de acompanhamento), e que a mensagem 4 é encaminhada a TD1, já que ele é o responsável por tomar a decisão final sobre a simulação integrada.

Finalizamos esta seção observando que é preciso garantir a consistência entre as três partes do metamodelo. Isso significa, por exemplo, que os tipos de canais presentes nas regras de papéis, local ou remoto, devem ser os mesmos representados no componente de rede (setas finas ou grossas). Por sua vez, as mensagens presentes na tabela de atributos de mensagens somente devem ser definidas se as arestas correspondentes, ligando os remetentes e os receptores no componente de rede, existirem. Os algoritmos do remetente e do receptor devem verificar essa consistência em tempo de execução. Finalmente, observamos que, nas fórmulas *send* utilizadas nas regras de papéis, visto que os receptores são determinados apenas indiretamente por meio da tabela de atributos de mensagens, indicamos apenas o canal usado para a comunicação com o receptor principal da mensagem. Os canais utilizados na comunicação com os outros receptores são determinados ao se identificar os receptores e as arestas empregadas (e seus canais) na tabela de atributos de mensagens.

4
Abordagens Tecnológicas para o Desenvolvimento de Ambientes Virtuais Distribuídos

Depois de elaborarmos um metamodelo para auxiliar a configurar a arquitetura do espaço de trabalho virtual colaborativo, neste capítulo nos concentramos em rever e analisar os DVEs (*Distributed Virtual Environments* – Ambientes Virtuais Distribuídos) existentes de modo a selecionar quais deles possuem as características mais apropriadas para servir como base para apoiar a implementação que será desenvolvida para validar nosso metamodelo.

Tratamos de duas das abordagens tecnológicas atuais para criar e construir sistemas distribuídos, isto é, *Middleware* e Ambientes Virtuais Distribuídos Puros, e a seguir descrevemos brevemente os sistemas que foram avaliados.

4.1.
Considerações sobre o Desenvolvimento de Ambientes Colaborativos

Para ser eficiente, um CVE (*Collaborative Virtual Environment* – Ambiente Virtual Colaborativo), que é outro nome para DVE, deve garantir suporte à distribuição, uma rede eficiente com grande largura de banda, comunicação adequada e modelos de armazenamento de dados. Além disso, esse ambiente distribuído deve propiciar interfaces colaborativas, operações e metáforas apropriadas para permitir que os usuários trabalhem efetivamente.

Para um ambiente virtual ser colaborativo, ele precisa ser distribuído entre os participantes que desejem compartilhá-lo. A escolha da arquitetura de comunicação é parametrizada pelo grau em que as estruturas de dados que representam o ambiente virtual são replicadas ou postas em cache nos nós computacionais e na tecnologia base de transporte (West & Hubbold, 2001).

Entretanto, qualquer que seja a tecnologia, as latências de comunicação são em última análise insuperáveis. A interação compartilhada em tempo real é particularmente sensível ao atraso, especialmente no caso de interfaces imersivas.

A sensibilidade da aplicação a tal atraso depende do grau desejado de interação genuinamente compartilhada. Por exemplo, em um passeio passivo através de um modelo, pequenas discrepâncias temporais entre as experiências do mundo pelos participantes podem passar despercebidas. No extremo oposto, duas pessoas operando uma vareta através dos limites de um equipamento de petróleo irão constatar que a coordenação é difícil ou impossível com qualquer atraso apreciável.

Se não é possível atingir a sincronia adequada, uma solução é ao menos focar os recursos naquelas atividades que são mais sensíveis ao atraso, i.e., naquelas que produzem as descontinuidades mais pronunciadas de experiência perceptual quando o atraso está presente. Isto envolve determinar adequadamente uma métrica de significância e urgência de priorização dentro da infra-estrutura de comunicação (West & Hubbold, 2001). Em última análise, como os componentes de um sistema CVE interagem de maneiras complexas, o projetista precisa considerar a aplicação como um sistema unificado. Invariavelmente, tentar otimizar um elemento de um CVE pode ter impacto adverso no comportamento de outros componentes. De fato, o desenvolvimento de um CVE é um ato de difícil balanceamento entre requisitos conflitantes de engenharia (Singhal & Zyda, 1999).

É impossível prever os requisitos de rede dos CVEs de modo isolado; preferivelmente, necessitamos de um modelo de operação de CVE que englobe a aplicação, o usuário e assuntos de software e hardware. Greenhalgh (2001) propõe um modelo que tem seis camadas:

1. Requisitos de tarefa/aplicação/colaboração: o que as pessoas querem ou tentam fazer.
2. Comportamento do usuário: que ações particulares as pessoas executam e quando. É importante considerar o comportamento do usuário quando se está tentando entender os requisitos de rede dos CVEs porque quase todos esses requisitos derivam do que os usuários escolhem fazer e quando escolhem fazê-lo. Por exemplo, se os usuários falam apenas raramente, e nunca ao mesmo tempo, então o requisito de rede para áudio pode ser muito limitado. Por outro lado, para alguns cenários de uso, é necessário haver uma largura de

banda suficiente para cada usuário poder falar ao mesmo tempo. Este poderia ser o caso, por exemplo, de sistemas de emergência.

3. Comportamento do processo: como a aplicação reage. Todo sistema CVE tem seu próprio conjunto de capacidades e suas próprias maneiras de representar usuários e mundos virtuais. Por exemplo, cada sistema suporta um diferente subconjunto de ações possíveis do usuário. Novamente, o cenário da emergência pode ser um bom exemplo: enquanto as pessoas responsáveis por toda a operação podem executar qualquer comando, outros especialistas podem executar apenas as tarefas para as quais foram designados.
4. Arquitetura de distribuição: o que se comunica com o quê. A escolha da arquitetura de distribuição determina que informação precisa ser comunicada a quais partes do sistema. A arquitetura de distribuição irá portanto determinar como os mundos virtuais são organizados e divididos, quais usuários podem se comunicar com quais outros e se existem ou não processos ou servidores de coordenação central. As arquiteturas de distribuição podem basicamente ser divididas em três modelos: cliente/servidor, *peer-to-peer* e híbrido.
5. Protocolos de comunicação: como a informação é trocada. Os protocolos podem ser *unicast* ou *multicast*. O desenvolvedor tem que escolher a melhor combinação arquitetura de distribuição/protocolo de comunicação que se adeqüe à aplicação particular que está sendo elaborada.
6. Comunicação na rede: o que realmente acontece na rede. Existem alguns requisitos de rede que dependem do lugar exato na rede em que a aplicação está sendo executada e como a rede está conectada (p.ex. LAN – *Local Area Network* ou WAN – *Wide Area Network*), já que a largura de banda real em diferentes partes da rede irá variar. Isto também inclui sistemas móveis.

A escolha de uma arquitetura de distribuição é o principal fator determinante de grande parte do "sentimento" de multi-usuário de um CVE. Por exemplo, a arquitetura de distribuição determinará tipicamente:

- A estrutura possível de um mundo virtual, isto é se ele lida bem com grandes espaços externos ou com interiores de construções complexos. Na indústria de óleo e gás, os usuários têm que lidar com ambas as situações, por exemplo ao construirem uma estrutura *offshore* complexa (interior complexo) ou monitorarem oleodutos dispostos sobre uma montanha (grande espaço externo). O desafio se torna ainda mais complexo em alguns cenários de emergência, em que os especialistas possivelmente têm que examinar vários detalhes da estrutura usando um ponto-de-vista egocêntrico (espaço interno), observando ao mesmo tempo o efeito da simulação de possíveis operações de emergência usando um ponto-de-vista exocêntrico (espaço externo). Nesses casos, o sistema tem que tratar simultaneamente dos requisitos dos espaços interno e externo.
- Quantos usuários podem compartilhar um único mundo virtual. Hoje em dia, as aplicações de óleo e gás típicas são executadas em não mais do que meia dúzia de salas de visualização simultaneamente, com não mais do que 20 especialistas em cada uma. Em alguns casos, especialistas espalhados pelo espaço externo podem ter a necessidade de entrar no mundo virtual usando um sistema móvel.
- A dinâmica do mundo virtual, incluindo as maneiras como os comportamentos podem ser executados e que coisas podem ou não ser mudadas. Por exemplo, é extremamente difícil mudar dinamicamente geometrias básicas em alguns sistemas, enquanto isto é relativamente fácil em outros. Esta é uma questão muito importante das aplicações de óleo e gás, tendo em vista a complexidade dos dados envolvidos. O que usualmente é feito é replicar o modelo de dados nos locais antes de a sessão colaborativa começar. Para evitar sobrecarga na rede, normalmente durante uma sessão é permitido aos usuários fazer apenas atualizações incrementais, transmitindo apenas pequenos dados e/ou mensagens de ações.

Até este momento, é razoável dizer que não existe uma escolha universal de arquitetura de distribuição ou comunicação, mas mais propriamente uma faixa de contrabalanços em questões de performance e organização.

Outro requisito importante em ambientes distribuídos é a necessidade de se ter um modelo de armazenamento de dados para suportar as atividades colaborativas. De acordo com Macedonia e Zyda (1997), os modelos de armazenamento de dados podem ser classificados em: *(i)* bancos de dados centralizados e compartilhados; *(ii)* bancos de dados replicados de mundos homogêneos; *(iii)* bancos de dados distribuídos e compartilhados com modificações *peer-to-peer*; e *(iv)* bancos de dados cliente/servidor distribuídos e compartilhados.

Os bancos de dados centralizados fornecem um modelo de programação fácil. O desenvolvedor pode acessar informações em qualquer tempo, ignorando a existência da rede. A única indicação de se um dado está em cache ou está remoto será no tempo de acesso. Adicionalmente, a sincronização está assegurada visto que (ignorando a operação de colocação em memória cache) existe apenas uma representação de cada objeto no banco de dados. A principal desvantagem dos bancos de dados centralizados é uma perda de performance em potencial, associada com o acesso dos dados através da rede.

Os bancos de dados replicados são populares em muitos sistemas de realidade virtual. As representações locais dos dados significam que as atualizações locais são rápidas.

Devido ao alto valor dos seus dados, as aplicações de óleo e gás possuem requisitos rigorosos de consistência e segurança de bancos de dados, sugerindo um modelo de bancos de dados centralizado e compartilhado. O conceito de grade computacional parece atender a esses requisitos, já que ele é conceitualmente centralizado com dados reais espalhados em vários lugares de modo transparente para a aplicação. Todavia, devido a requisitos de performance e como a grade computacional ainda é um conceito novo, pode ser observado que quase todas as aplicações práticas no momento usam o modelo replicado, com algumas permitindo aos usuários fazer o que chamamos de atualizações incrementais.

4.2. Abordagens de Arquiteturas

Depois de discutirmos sobre considerações gerais acerca do desenvolvimento de ambientes colaborativos, iremos agora nos concentrar em duas abordagens tecnológicas atuais para criar e construir sistemas distribuídos, isto é, *Middleware* e Ambientes Virtuais Distribuídos Puros, descrevendo brevemente os sistemas que foram avaliados.

4.2.1. *Middleware*

Os sistemas baseados na abordagem *Middleware* têm o objetivo de criar uma interface que estabeleça um padrão prático e portável para a comunicação entre processos. Nas subseções seguintes, descrevemos brevemente cinco desses sistemas.

4.2.1.1. *Message Passing Interface* (MPI)

O objetivo da MPI (*Message Passing Interface* – Interface de Passagem de Mensagens) dito de forma simples é desenvolver um padrão amplamente utilizado para escrever programas de passagem de mensagens (Dongarra et al., 1993). As principais vantagens de se estabelecer um padrão de passagem de mensagens é a portabilidade e a facilidade de uso. Em um ambiente de comunicação de memória distribuída no qual as rotinas e/ou abstrações de nível mais alto são construídas sobre rotinas de passagem de mensagens de nível mais baixo, os benefícios da padronização ficam particularmente evidentes.

A passagem de mensagens é um paradigma amplamente utilizado em certas classes de máquinas paralelas, especialmente aquelas com memória distribuída. Embora existam diferentes variações, o conceito básico de comunicação entre processos por meio de mensagens é bem entendido. Nos últimos dez anos, foi feito um progresso substancial na modelagem de aplicações importantes por meio deste paradigma. Cada fabricante implementou sua própria variação.

A MPI-1 (Dongarra et al., 1993) foi completada no início de 1993 e enfocava principalmente a comunicação ponto-a-ponto. Ela não incluía nenhuma rotina de comunicação coletiva e não era segura com relação a *threads*.

A MPI-2 (2006) adicionou algumas extensões ao padrão básico. A MPI/RT é a última versão. Ela incorpora inúmeras facilidades, tais como uma API (*Application Programmer's Interface*), suporte para ambientes heterogêneos, *bindings* C e Fortran, canais ponto-a-ponto e coletivos, uma interface de comunicação confiável e segurança com relação a *threads*.

4.2.1.2. CORBA e TAO

CORBA – *Common Object Request Broker Architecture* (OMG, 1995) é um padrão aberto para comunicação entre processos locais e remotos. Sobre o pacote de transmissão, CORBA oferece diversos serviços padrão que automatizam várias tarefas comuns de programação de rede, como registro, localização e ativação de objetos; operação de demultiplex de solicitação e despacho de operação. No nível base, a comunicação é definida por RPCs (*Remote Procedure Calls* – Chamadas de Procedimento Remoto) entre processos. O desenvolvedor define a interface para os processos usando a IDL (*Interface Description Language* – Linguagem de Descrição de Interface). A IDL é compilada em uma interface que existe entre o cliente e o servidor. Depois disso, os detalhes da comunicação são manipulados por CORBA. Os serviços mencionados acima são então dispostos em camadas sobre este *framework* de comunicação.

As implementações iniciais de CORBA não lidavam com questões de processamento em tempo real abertamente (embora isto não tenha impedido o desenvolvimento de um pequeno número de aplicações de realidade virtual baseadas em CORBA, p.ex., COVRA-CAD (Junghyun et al., 1998)).

Doug Schmidt e colegas da Universidade do Estado de Washington desenvolveram *The ACE ORB* (TAO) (Schmidt, 2006), um *framework middleware* compatível com CORBA que trata de alguns desafios de tempo real do processamento distribuído. Schmidt e colegas identificam um conjunto de padrões e componentes de *framework* que podem ser aplicados sistematicamente

para eliminar vários aspectos tediosos, sujeitos a erros e não portáveis do desenvolvimento e da manutenção de aplicações distribuídas.

4.2.1.3. Grade Computacional e Globus

A grade computacional pode ser definida pela seguinte lista de verificação imaginária:

"(Uma Grade Computacional…)

- Coordena recursos que não estão sujeitos a um controle centralizado – (Uma Grade Computacional integra e coordena recursos e usuários que experimentam diferentes domínios de controle – por exemplo, o *desktop* do usuário vs. computação central; diferentes unidades administrativas da mesma companhia; ou diferentes companhias – e trata das questões de segurança, política, pagamento, associação e assim por diante, que surgem nessas configurações.)
- Utilizando protocolos e interfaces padrões, abertos e de propósito geral – (Uma Grade Computacional é construída a partir de protocolos e interfaces multipropósito que tratam de questões fundamentais tais como autenticação, autorização, descoberta de recursos e acesso a recursos...)
- Para oferecer qualidades não triviais de serviço – (Uma Grade Computacional permite que seus recursos constituintes sejam usados de uma maneira coordenada para oferecer várias qualidades de serviço, relacionadas por exemplo com o tempo de resposta, *throughput*, disponibilidade e segurança, e/ou co-alocação de múltiplos tipos de recursos para atender a demandas complexas do usuário, de modo que a utilidade do sistema combinado seja significativamente maior que a soma de suas partes.)" (Foster & Kesselman, 1998).

Existem várias infra-estruturas de software para Grade Computacional (p.ex. Globus (Foster & Kesselman, 1997), Legion (Grimshaw & Wulf, 1997) e SNIPE

(Fagg et al., 1997)). Serviços como autenticação, acionamento de programas e mecanismos de transferência de dados estão incluídos nas infra-estruturas.

Globus é indiscutivelmente a mais popular dessas infra-estruturas e foi implementada como o mecanismo de distribuição em vários *frameworks* de realidade virtual (incluindo CAVERNSoft (Park et al., 2000)).

Globus é projetado para oferecer facilidades tais como acesso uniforme a recursos distribuídos com diversos mecanismos de agendamento, serviço de informação para publicação, descoberta e seleção de recursos, e performance melhorada por meio de múltiplos protocolos de comunicação (Foster & Kesselman, 1997).

4.2.1.4. *Common Component Architecture* (CCA)

A CCA – *Common Component Architecture* (Common Component Architecture Forum, 2006) define um *framework* de comunicação baseado em componentes que conceitualmente se situa sobre o *middleware*, tal como CORBA ou Globus, ou IP de nível mais baixo.

Existem inúmeras vantagens em se usar programação por componentes ao se implementar ferramentas de visualização de alto desempenho. Isto é verificado quando se considera que o desenvolvimento de aplicações de realidade virtual é cada vez mais um empreendimento multidisciplinar. Ao prover estruturas de núcleo flexíveis, o esforço de desenvolvimento mais amplo pode incorporar equipes de pesquisa com uma gama significativa de habilidades. A programação por componentes suporta as diferentes abordagens e requisitos inevitáveis em um esforço de desenvolvimento multidisciplinar. Entretanto, possivelmente a faceta mais interessante da programação por componentes com relação ao presente trabalho é que os componentes podem ser configurados para serem executados localmente ou em lugares remotos.

CCA apresenta um meio transparente de definir e gerenciar componentes de software. Dessa forma, ela trata da interoperabilidade no nível do componente, isto é, da conexão flexível a uma arquitetura por um componente compatível por meio de uma interface bem definida. Existem diversos *frameworks* compatíveis com CCA em uso atualmente, incluindo CCAFFEINE (Armstrong et al., 1999) e

XCAT (2006). O modelo de fluxo de dados no SCIRUN (Parker, 1999) também incorpora um *framework* compatível com CCA.

4.2.1.5. InfoGrid

InfoGrid, previamente CSGrid (Lima et. al, 2005), é um sistema cliente/servidor para ambientes de grade computacional que, além do suporte ao uso e gerenciamento de recursos computacionais distribuídos, oferece facilidades para integrar aplicações e gerenciar dados e usuários (Figura 11). O InfoGrid apresenta a seus usuários, por meio de um navegador Web, um espaço de trabalho com todas as aplicações disponíveis e com os arquivos de dados de cada usuário organizados por projeto. Um usuário pode estender o sistema, adicionando novas aplicações. O InfoGrid também oferece a seus usuários algumas facilidades de trabalho colaborativo.

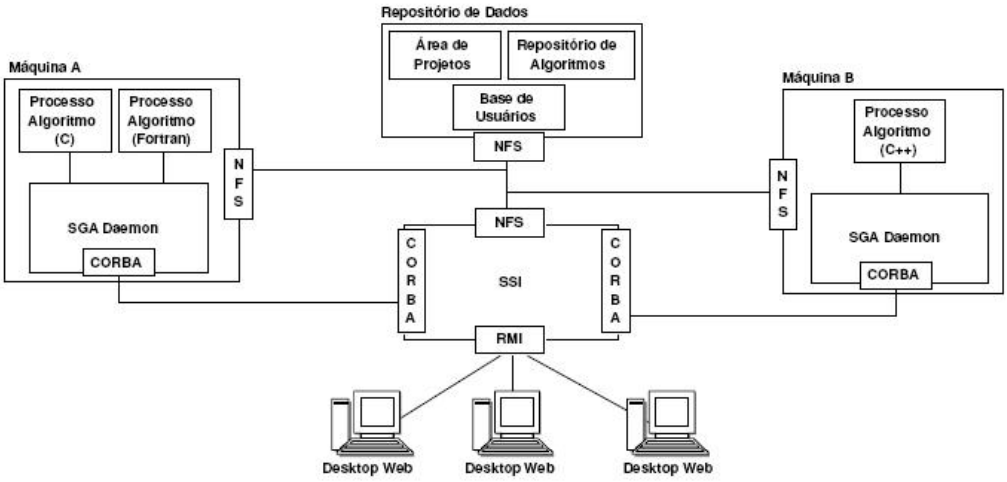

Figura 11 - Arquitetura do InfoGrid

As aplicações que são executadas no cliente utilizam os serviços disponíveis no servidor para ter acesso e gerenciar os recursos computacionais distribuídos. Um desses serviços é o de execução remota de algoritmos em máquinas que estão ligadas ao InfoGrid. Algoritmos, no contexto do InfoGrid, são programas executáveis implementados em qualquer linguagem que admitem parâmetros de entrada, geram uma saída e não possuem nenhum tipo de interação com o usuário durante sua execução. Diversas máquinas podem ser incorporadas ao ambiente de grade computacional para servir como plataforma de execução de algoritmos.

Novos algoritmos são facilmente tornados disponíveis no ambiente e o processo para executá-los passa a ser uma tarefa transparente para o usuário.

Uma característica importante na arquitetura do InfoGrid é a interoperabilidade entre o servidor principal do sistema e os servidores que são executados nas diversas máquinas de execução de algoritmos. Independente da linguagem de implementação dos algoritmos e das máquinas onde estes estão sendo executados, o InfoGrid oferece uma interface padrão para a execução remota e a monitoração das máquinas do ambiente.

O InfoGrid possui um *desktop* Web onde os usuários utilizam os recursos computacionais distribuídos como se estivessem trabalhando localmente. Por meio deste cliente, os usuários podem criar suas próprias áreas de trabalho, monitorar as máquinas da grade computacional, instalar algoritmos, executar aplicações instaladas no *desktop*, comandar a execução remota de algoritmos e monitorar os processos que estão executando esses algoritmos nas máquinas do ambiente.

O InfoGrid foi implementado como uma extensão do CSBase (2006), um *framework* para o gerenciamento de recursos e execução de algoritmos em um ambiente computacional distribuído e heterogêneo. O CSBase é o resultado de diferentes projetos desenvolvidos pela parceria entre o Tecgraf/PUC-Rio e a Petrobras. O primeiro desses projetos, WebSintesi, implementou e tornou disponível um ambiente integrado para análise e síntese de dados geofísicos. No WebSintesi, máquinas multiprocessamento e aglomerados de computadores são ligados a um servidor principal que realiza, de modo transparente para os usuários, a execução remota de programas que necessitam de processamento de alto desempenho e manipulam dados sísmicos de centenas de gigabytes.

Após o primeiro ano de desenvolvimento do WebSintesi, o projeto InfoPAE Dinâmico foi iniciado para atender a usuários da Petrobras que utilizam aplicações de gestão de emergências, tais como programas de análise de estabilidade de unidades *offshore*. Para permitir o re-uso de serviços já implementados para o WebSintesi e para facilitar o desenvolvimento de novos serviços e aplicações particulares para os projetos WebSintesi e InfoPAE Dinâmico é que foi criado o *framework* CSBase.

Em função de todos esses desenvolvimentos, decidimos tomar o InfoGrid como um exemplo desta primeira abordagem (*Middleware*) para delinear um protótipo (descrito no Capítulo 5) que será uma das provas do nosso metamodelo.

4.2.2. Ambientes Virtuais Distribuídos Puros

Nesta subseção, investigamos a segunda abordagem para o desenvolvimento de ambientes virtuais distribuídos, isto é, Ambientes Virtuais Distribuídos Puros, na qual ambientes distribuídos específicos são desenvolvidos para atender aos requisitos. Esta abordagem oferece um nível de abstração mais alto quando comparada com *Middleware*, no sentido de que seus protocolos, arquiteturas e sistemas oferecem aos desenvolvedores APIs que são mais próximas do nível da aplicação do que do nível da implementação.

4.2.2.1. SIMNET, DIS e HLA

Distributed Interactive Simulation (DIS) e seu predecessor SIMNET são padrões para simulações interativas distribuídas (Locke, 1993; Youngman, 2006). DIS retém um banco de dados replicado e usa *dead reckoning* para manter o acompanhamento de outros. *Dead reckoning* é uma estratégia para lidar com problemas de latência de rede que envolve o empacotamento do local de uma entidade, de uma marca de tempo e de um vetor de velocidade. O computador central da aplicação que recebe os dados pode então predizer onde o objeto deve estar. Cada entidade executa uma simulação local junto com o modelo de *dead reckoning*, e quando os dois divergem por mais do que uma certa tolerância uma mensagem é enviada a todos os participantes para reconciliar a discrepância.

DIS traz em conjunto várias idéias interessantes para a realidade virtual distribuída:

- Formatos padrões de mensagem.
- Não uso de servidor central.
- *Dead reckoning*.
- AOIM (*Area of Interest Management* – Gestão de Área de Interesse).
- Uso potencial de *multicasting*.

Mensagens *multicast* podem ser enviadas para um grupo específico de máquinas. *Multicasting* é popular em sistemas de realidade virtual distribuída, particularmente os que suportam um grande número de participantes. Macedonia et al. (1995) afirmam que foram mostradas algumas simulações que reduziram o tráfego de rede de 90% ao empregarem *multicasting*.

Algumas aplicações implementam os protocolos DIS, tais como *VR-Link* (Morrison, 1995), *Close Combat Tactical Trainer* (CCTT) (Mastaglio & Callahan, 1995), PARADISE (Singhal, 1996) e NPS *Networked Vehicle Simulator* (NPSNET) (Macedonia et al., 1995). NPSNET-V (Capps et al., 2000) é desenvolvida pela *Naval Postgraduate School*. Ela é implementada em Java e incorpora o conceito de uma entidade para representar um objeto do mundo virtual, definindo entidades de objetos que contêm informações sobre artefatos no ambiente virtual (p.ex., tanques ou pessoas). As entidades definem informações de estado, tais como local e velocidade. A partir de uma lista de protocolos definidos, a cada entidade pode ser atribuído um conjunto de comportamentos.

O desenvolvimento da *High Level Architecture* (HLA) (Defense, 2006; Dahmann et al., 1999) tem por objetivo facilitar a interoperabilidade e a composição da maior gama possível de simulações baseadas em componentes. A HLA não se originou como um padrão aberto, mas foi mais tarde reconhecida e adotada pelo *Object Management Group* (OMG, 2006) e pelo *Institute of Electrical and Electronics Engineers* (IEEE, 2006). O desenvolvimento da HLA ocorreu de forma semelhante ao padrão DIS – isto é, o projeto inicial foi produzido dentro do Departamento de Defesa dos EUA e então entregue a um organismo externo (IEEE e/ou OMG) para padronização (Singhal & Zyda, 1999).

A arquitetura HLA pode ser entendida como um projeto de ambiente virtual de rede orientado a objetos. Cada simulador, conhecido como um *federado*, é um componente que representa uma coleção de objetos, cada um destes tendo um conjunto de atributos e sendo capaz de iniciar e receber eventos. O federado registra cada um de seus objetos com um pedaço de *middleware* chamado *Run-Time Infrastructure* (RTI). A RTI colabora com instâncias RTI em outros computadores para aprender sobre participantes (objetos) e fornecer informações sobre esses participantes para o federado local. O federado local, por sua vez, tipicamente instancia objetos locais representando esses participantes remotos. Atualizações de atributos e eventos são também trocados através da RTI, que é

responsável por manipular a gestão de área de interesse, a sincronização do tempo e outros serviços de ambiente virtual de rede de nível baixo, segundo o interesse da aplicação. A coleção de federados, juntamente com as suas instâncias associadas de RTI, é chamada de *federação*. Os simuladores na federação enviam e recebem informações de estado por meio de chamadas para a e da RTI (Singhal & Zyda, 1999). A HLA como um padrão IEEE será melhor discutida mais adiante neste capítulo.

4.2.2.2. DIVE

DIVE (*Distributed Interactive Virtual Environment*) (Swedish, 2006) é desenvolvido pelo Instituto Sueco de Ciência da Computação. O esforço de desenvolvimento se concentrou principalmente no componente de distribuição do ambiente virtual. Certamente, em seus primeiros dias DIVE emergiu como um exemplo importante de como desenvolver realidade virtual distribuída.

O elemento de distribuição é conceitualmente muito simples. A memória compartilhada é tornada disponível na rede e o sistema controla o envio de sinais para os processos. DIVE implementa um banco de dados distribuído, completamente replicado e dinâmico. Ele tem a capacidade de adicionar novos objetos e modificar bancos de dados existentes de um modo confiável e consistente. Protocolos *multicast* confiáveis e controle de concorrência são empregados por meio de um mecanismo de bloqueio distribuído para facilitar a atualização desses bancos de dados.

4.2.2.3. MASSIVE

MASSIVE (Greenhalgh et al., 2000) é um sistema orientado a objetos desenvolvido na Universidade de Nottingham para apoiar ambientes virtuais colaborativos multi-usuário, baseado no “modelo espacial” de interação (Benford et al., 1993). Ele é desenvolvido principalmente como uma ferramenta de pesquisa acadêmica, para pesquisadores que investigam a colaboração on-line em ambientes virtuais. O foco do trabalho é promover a percepção (*awareness*) de outros no ambiente virtual.

O sistema suporta interação com os ambientes virtuais via texto, interfaces gráficas 2D e 3D, e possui suporte para áudio em tempo real, permitindo que os usuários se comuniquem entre si usando a fala. A ênfase é mais na interação usuário-a-usuário do que na interação de um usuário com objetos no ambiente. MASSIVE usa uma combinação de processos *peer-to-peer* e cliente/servidor, e emprega *multicasting* como a linha básica de colaboração.

4.2.2.4. Avango

Avango é desenvolvido pelo Centro de Pesquisas Nacional da Alemanha para a Tecnologia de Informação (GMD) (Bierbaum & Just, 1998; Tramberend, 1999). Ele é ostensivamente um *framework* orientado a objetos desenvolvido sobre o Performer para a construção de ambientes virtuais em rede.

Os principais subsistemas (interação, gráfico, operação em rede) em uma aplicação de realidade virtual são integrados em um. Ele provê um grafo de cena replicado através da rede. Os objetos podem ser instanciados ou como locais ou como distribuídos (públicos ou privados). São definidas duas categorias de objetos:

1. Nós – definem elementos de grafo de cena para *rendering*.
2. Sensores – importam dados de dispositivos externos para a aplicação.

Avango é desenvolvido sobre um modelo de fluxo de dados usando campos dentro dos objetos para suportar uma interface genérica de fluxo. O grafo de fluxo de dados define o comportamento dos objetos no mundo.

Ele possui uma API C++ e um *binding* para uma linguagem interpretada, Scheme (Dybvig, 1996). Tipicamente, funcionalidades complexas e críticas em termos de performance são implementadas em C++. A partir deste ponto, a aplicação é implementada usando *scripts* da Scheme.

Avango usa um modelo de grupos de processos. Assegura-se que cada membro do grupo recebe mensagens enviadas através da rede exatamente na mesma ordem. Além disso, quando um novo membro se junta ao grupo, toda a

comunicação é suspensa até que o estado do novo membro seja atualizado, o que garante consistência de estado.

4.2.2.5. DEVA/MAVERIK

DEVA (Pettifer et al., 2000) é um núcleo de realidade virtual distribuída desenvolvido pelo *Advanced Interfaces Group*, da Universidade de Manchester.

DEVA suporta bancos de dados que incluem objetos virtuais, e separa a representação de um objeto virtual em objeto do lado do cliente e objeto do lado do servidor. A parte do lado do servidor representa o objeto virtual e define o seu comportamento, uma parcela do qual pode ser genérica e outra pode ser específica de uma aplicação. A parte do lado do cliente contém interpretações do seu comportamento e só responde a instruções vindas do servidor. As mensagens do servidor para o cliente são semelhantes a um vocabulário de alto nível usado para descrever o efeito do comportamento. DEVA estabelece um local de controle que gerencia estados em um lugar. Isto significa que o estado de uma entidade pode ser gerenciado localmente em vez de no servidor, se isto for apropriado.

Em vez de usar *dead reckoning* para combater o atraso e a tremulação, DEVA emprega "*twines*" (Marsh et al., 1999), que suaviza atualizações irregulares.

DEVA está fortemente acoplado a um núcleo de realidade virtual (MAVERIK (Hubbold et al., 2001)), que provê capacidades de *rendering*, entrada, saída e gerenciamento espacial. A principal vantagem é que dados da aplicação não precisam ser duplicados e armazenados em uma estrutura de dados específica como o grafo de cena. E, em lugar disso, MAVERIK promove o uso das estruturas de dados próprias da aplicação, o que evita a necessidade de se ter que estar de acordo com um estrutura de dados rígida e potencialmente inapropriada.

4.2.2.6. Outros DVEs

O DIVERSE *Toolkit* (DTK), da Virginia Tech (Arsenault et al., 2001), é projetado para auxiliar a implementação de aplicações de realidade virtual distribuída e está disponível tanto sob a *Gnu Public License* (GPL) quanto sob a

LGPL. Este conjunto de ferramentas é implementado como uma API C++ para um servidor e clientes, e oferece uma biblioteca extensiva mas de relativo baixo nível para distribuição, em vez de um *framework* de realidade virtual.

COVISE (*Collaborative Visualisation and Simulation Environment*) (Rantzau et al., 1998) explora infra-estruturas de rede de alta velocidade em computação distribuída e engenharia colaborativa. Ele se concentra predominantemente em visualizar problemas de supercomputação *high end*. COVER (2006), baseado no IRIS Performer, provê suporte à visualização. COVISE media uma sessão distribuída por meio de um sistema de tomada de vez. O banco de dados é completamente replicado através dos nós ao se fazer a conexão. Toda a entrada do usuário, saída do sistema e administração do sistema é manipulada em um único ponto, reduzindo a complexidade de se manter todos os usuários sincronizados, mas com o custo de atrasos de ida-e-volta e o risco potencial de o ponto central de controle tornar-se um gargalo à medida que mais usuários são adicionados.

CAVERNsoft (Park et al., 2000) combina uma biblioteca de operação de rede e um banco de dados, facilitando o desenvolvimento de ambientes virtuais interativos colaborativos. CAVERNsoft é baseado em um modelo cliente-servidor. As aplicações usam o *Information Request Broker* (IRB) para mediar a comunicação entre as instâncias de rede. Os usuários podem solicitar o tipo de conexão de rede que desejam usar. CAVERNsoft fornece a opção de TCP/IP, UDP ou IP/*Multicast*. O usuário define a largura de banda desejada e atributos de rede aceitáveis (latência, tremulação), e o IRB remoto tenta satisfazê-los. Desse modo, o usuário especifica a qualidade desejada de serviço logo ao entrar.

Existem também *frameworks* de realidade virtual mais genéricos que oferecem suporte para implementar DVEs. Por exemplo, VR Juggler (Bierbaum et al., 2001) traz suporte a operações de rede fornecido por meio de um gerente abstrato de rede. Outro exemplo é Bamboo (Watsen & Zyda, 1998), desenvolvido sobre o ADAPTIVE *Communication Environment* (ACE) (Schmidt, 1994), que provê o sistema com operações de rede, concorrência (*threading*) e funções de sincronização.

4.2.2.7. HLA

Simulação distribuída é uma aplicação da tecnologia de sistemas distribuídos que habilita modelos a serem ligados através de redes como a Internet, de modo a trabalharem de forma conjunta (ou interoperarem) durante a execução de uma simulação. A HLA (*High Level Architecture*) é um padrão que define a tecnologia de sistema distribuído a tornar possível esta interoperabilidade. Mais do que um protocolo de rede (padrão para fio), como o DIS, a HLA define uma arquitetura com um conjunto de padrões de API. As aplicações de simulação (conhecidas como federados na HLA) se comunicam fazendo chamadas às APIs HLA. Um pedaço de software conhecido como RTI (*Run-time Infrastructure*) implementa a API HLA e é responsável por transportar dados de um federado para o outro. Assim como no DIS, padrões HLA são propriedade do IEEE. Existem três documentos que compreendem o padrão HLA, todos disponíveis a partir do IEEE (2006):

- IEEE 1516-2000 – IEEE *Standard for Modeling and Simulation* (M&S) *High Level Architecture* (HLA) – *Framework and Rules*: provê as regras e definições para implementar e usar HLA. Seu código de produto IEEE é SH94882;
- IEEE 1516.1-2000 – IEEE *Standard for Modeling and Simulation* (M&S) *High Level Architecture* (HLA) – *Federate Interface Specification*: define os vários serviços fornecidos por uma RTI de HLA (veja Figura 12) e contém as APIs. Seu código de produto IEEE é SH94883;
- IEEE 1516.2-2000 – IEEE *Standard for Modeling and Simulation* (M&S) *High Level Architecture* (HLA) – *Object Model Template* (OMT) *Specification*: define o formato usado para descrever modelos de objetos em HLA. Um modelo de objeto dita que tipos de dados um conjunto particular de federados HLA trocará. Seu código de produto IEEE é SH94884.

Existe um quarto documento que não é tecnicamente parte da definição da HLA, mas que define algumas das práticas recomendadas para usar HLA. Ele é

chamado de IEEE 1516.3-2003 – IEEE *Recommended Practice for High Level Architecture* (HLA) *Federation Development and Execution Process* (FEDEP), e seu código de produto IEEE é SH95088.

Enquanto a série IEEE 1516 de padrões representa a versão "corrente" de HLA, várias simulações HLA ainda estão usando uma versão mais antiga, conhecida como HLA 1.3. Esta versão era mantida pelo DMSO (*Defense Modeling and Simulation Office*) e pelo Departamento de Defesa dos EUA antes da padronização do IEEE.

A HLA foi inicialmente desenvolvida pelo Departamento de Defesa dos EUA para a simulação de cenários de campos de batalha. O padrão emergente HLA do IEEE tem as seguintes características:

- Suporta simulações compostas de diferentes componentes de simulação.
- Suporta ambiente de tempo real e problemas de simulação grandes e complexos.
- Suporta re-usabilidade: modelos de simulação por componentes podem ser re-usados em diferentes cenários e aplicações de simulação.
- Suporta interoperabilidade: simulações por componentes re-usáveis podem ser combinadas com outros componentes sem a necessidade de recodificação.

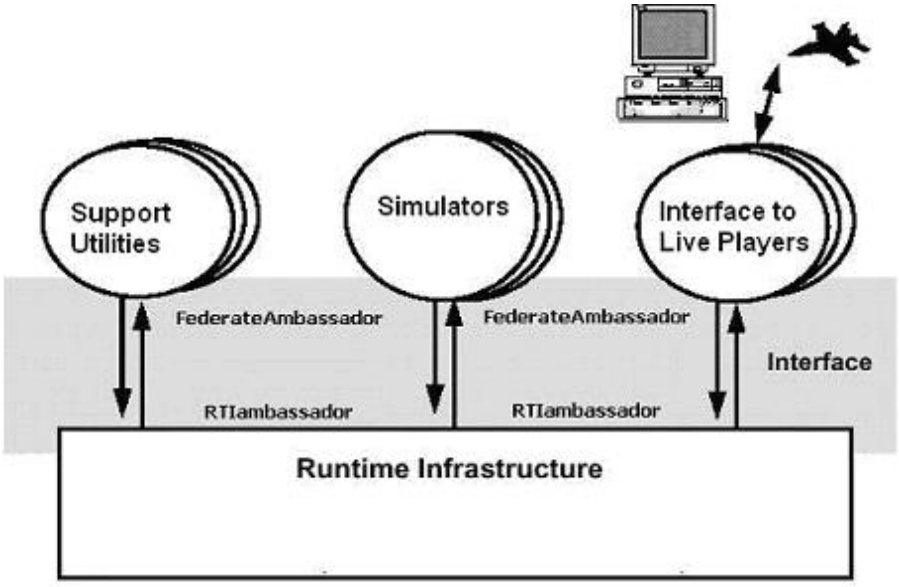

Figura 12 - HLA e RTI (Dahmann et al., 1999)

Podemos então concluir que a HLA preenche nosso requisito de suporte à colaboração em tempo real, assim como parece ser uma arquitetura baseada em componentes flexível, de acordo com todos os princípios que estivemos buscando até agora. A HLA também parece satisfazer vários aspectos de um espaço de trabalho colaborativo para a gestão de desastres na área de óleo e gás, exceto pelo fato de que ela é desenvolvida principalmente para ter vários nós com atualização precisa em tempo real, enquanto que na área de óleo e gás existem menos nós mas com maiores demandas em termos de visualização.

O uso do padrão HLA não é restrito a ambientes virtuais distribuídos, o que significa que, à medida que a demanda pela integração de processos, sistemas e bancos de dados dentro das companhias continua a crescer, esta é uma das abordagens que devem ser consideradas ao se definir soluções de integração.

Os conceitos fundamentais na HLA são:

- Federado: quando uma simulação é implementada como parte de uma simulação compatível com HLA, ela é denominada *federado*.
- Federação: é uma coleção de federados trabalhando juntos para resolver um problema específico.

Note que as federações podem incluir mais do que simulações. Elas também podem incluir interfaces para operadores humanos/jogadores, para hardware real e para software geral executando funções tais como coleta de dados, análise de dados e exibição de dados.

Os três principais componentes na HLA são:

- Regras HLA:
  - Asseguram interação apropriada de federados em uma federação.
  - Descrevem as responsabilidades dos federados e federações.
- Especificação da Interface:
  - Define os serviços e interfaces da RTI (*Run-time Infrastructure*).
  - Identifica funções "*callback*" que cada federado precisa prover.
- *Object Model Template* (OMT):
  - Prescreve o formato e a sintaxe para a gravação de informações.
  - Estabelece o formato de modelos chave:
    - *Federation Object Model* (FOM).

- *Simulation Object Model* (SOM).
- *Management Object Model* (MOM).

A Figura 13 mostra uma visão lógica de alto nível de uma Federação HLA em execução. Todos os componentes mostrados na figura são parte de uma única federação com exceção do RtiExec. Uma breve descrição de cada um desses componentes é fornecida abaixo:

- Federado: programa de simulação por componentes compatível com HLA, mais um SOM.
- Federação: simulação composta de um conjunto de federados interagindo via os serviços da RTI, mais um FOM.
- FedExec: gerencia a federação. Ele permite que os federados se associem e se desliguem da federação, e facilita a troca de dados entre federados participantes.
- Arquivo FDD: arquivo *FOM Document Data*, contém informações derivadas do FOM e usadas pela RTI em tempo de execução.
- RtiExec: processo global que gerencia a criação e a destruição de FedExec's.
- Arquivo RID: *RTI Initialization Data*. Informação específica do fabricante da RTI necessária para se executar uma RTI.

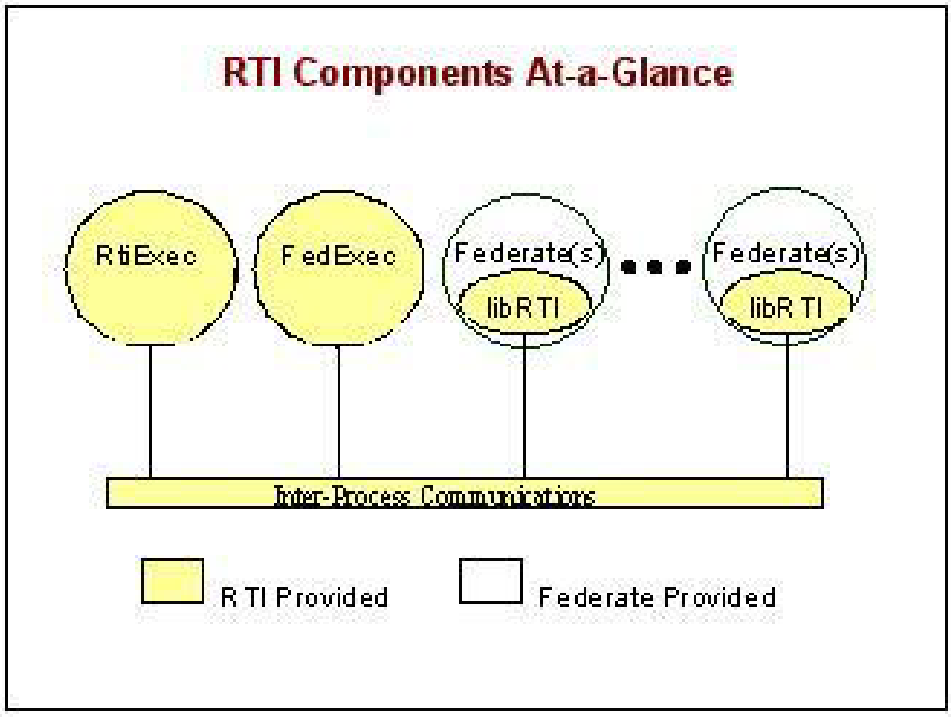

Figura 13 - Visão Lógica de uma Federação HLA (McLeod, 2006)

Consideremos agora um sistema usando HLA com múltiplas federações em execução. O sistema está executando duas federações: Federação 1 e Federação 2. Embora um sistema possa executar duas federações, as Regras da HLA especificam que elas são independentes uma da outra e não podem trocar nenhuma informação. Este sistema está ilustrado na Figura 14 e possui os seguintes componentes:

- Existe um único RtiExec e um único arquivo RTI.RID, compartilhado por ambas as federações.
- Cada federação contém seu próprio FedExec e arquivo FDD. FedExec1 controla a execução da Federação 1 e FedExec2 controla a execução da Federação 2.
- A Federação 1 contém dois federados: o Federado Branco (*White Federate*) e o Federado Verde (*Green Federate*).
- A Federação 2 contém três federados: os Federados Púrpura (*Purple*), Laranja (*Orange*) e Azul (*Blue*).

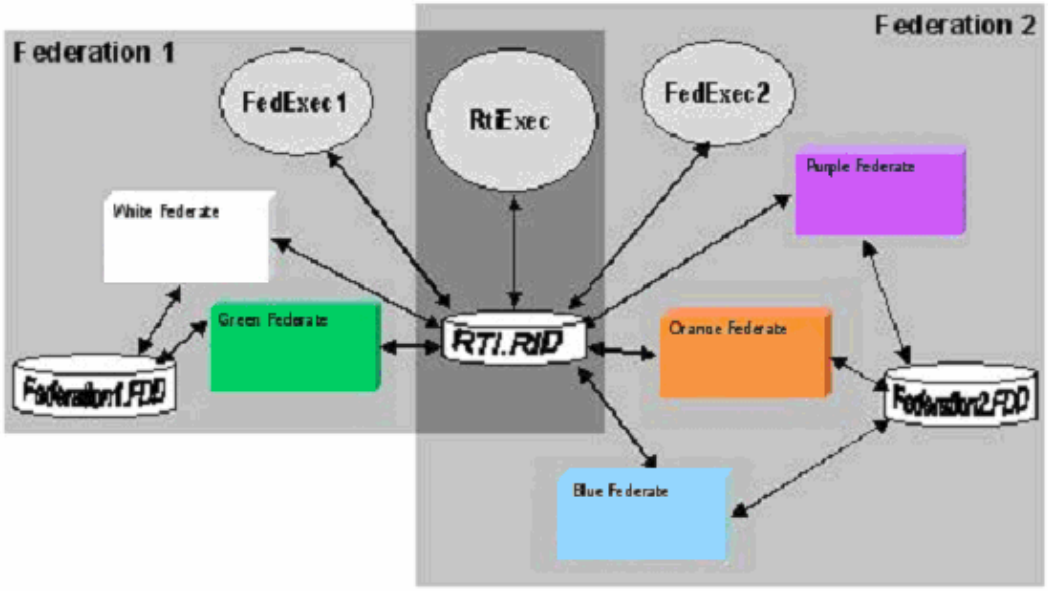

Figura 14 - O Esquema Geral – Federações em Execução (McLeod, 2006)

Na Figura 15, mostramos os passos no processo de iniciar uma execução de federação:

1. Quando uma federação é executada, o RtiExec é disparado primeiro.

2. Então um federado, agindo como um gerente, cria uma execução de federação invocando o método RTI *createFederationExecution*.
3. Então um nome é reservado com o RtiExec, um processo FedExec é gerado, e este FedExec registra seu endereço de comunicação com o RtiExec. A execução da federação está em andamento.
4. Uma vez que uma execução de federação existe, outros federados podem se ligar a ela. O RtiExec é consultado para obter o endereço do FedExec, e *joinFederationExecution()* é invocado no FedExec. Federados adicionais podem se associar por meio do mesmo processo.

Em face de todas essas características, escolhemos HLA como um exemplo dessa segunda abordagem (DVEs Puros) para delinear um protótipo (Capítulo 5) que será uma das provas do nosso metamodelo, do mesmo modo que o InfoGrid foi escolhido como uma plataforma da primeira abordagem (*Middleware*).

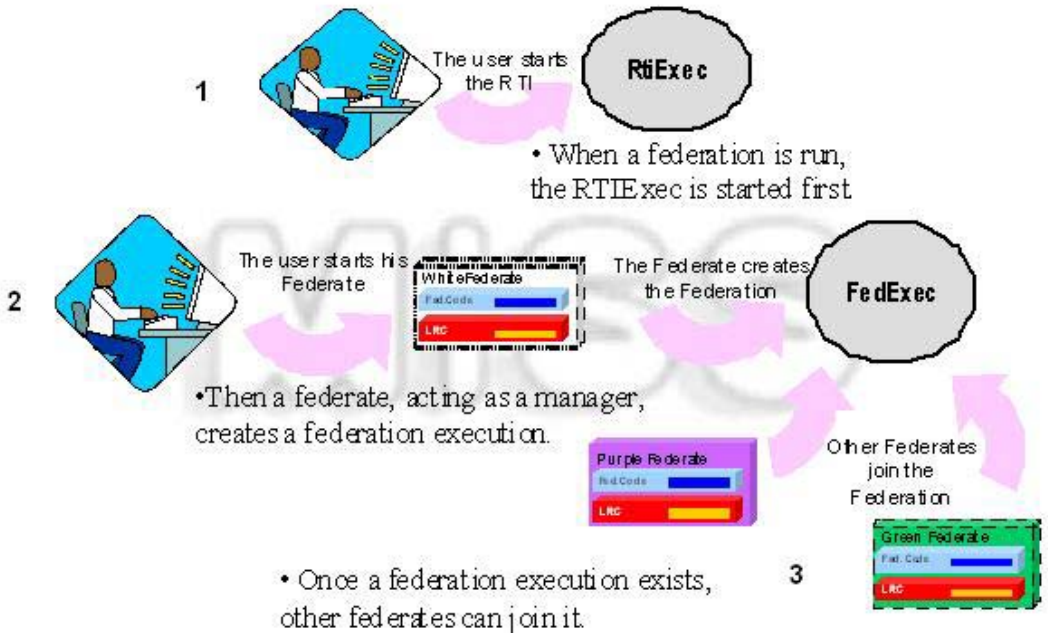

Figura 15 - Passos no Processo de Execução de uma Federação (McLeod, 2006)

Ainda tínhamos que escolher a RTI (*Run-time Infrastructure*) da HLA a ser usada com o nosso protótipo. Nossos requisitos eram:

- Ter código fonte aberto.

- Ser distribuível gratuitamente.

Com base nesses requisitos e no fato de que ela estava bem documentada em uma dissertação de mestrado, escolhemos a XRTI – *The Extensible Run-Time Infrastructure* – de Kapolka (2003). Suas características básicas são:

- Tem plena compatibilidade com o padrão HLA.
- Especifica um protocolo de mensagens padronizável.
- Suporta extensão e composição de modelos de objetos dinâmicos.
- Está escrita em Java e usa modelos de objetos XML.
- Usa uma topologia cliente-servidor pura na qual os federados só se comunicam com outros através do XRTI Executive, uma aplicação de servidor.
- Os federados mantêm dois canais com o Executive: um canal TCP (*Transmission Control Protocol*) para comunicação confiável e um canal UDP (*User Datagram Protocol*) para mensagens não confiáveis.

5
Metamodelo Centrado nas Atividades: Derivando Modelos e Protótipos

Neste capítulo, derivamos um primeiro modelo para a gestão de desastres de estruturas *offshore* de óleo e gás baseado no nosso metamodelo de multi-perspectiva. Também desenvolvemos um protótipo HLA como prova de conceito do metamodelo e discutimos como o protótipo poderia ser implementado usando o InfoGrid. Ainda para a aplicação de gestão de desastres, apresentamos um segundo modelo e seu protótipo, mostrando que podemos derivar diferentes modelos para a mesma aplicação. Finalmente, para validar a generalidade do metamodelo, também delineamos um modelo para outra aplicação, isto é, a visualização CAD em ambientes virtuais.

5.1.
A Aplicação de Gestão de Desastres de Estruturas *Offshore* de Óleo e Gás

A aplicação de gestão de desastres de estruturas *offshore* de óleo e gás foi o que motivou a criação do nosso metamodelo. Investigando os requisitos da aplicação, seguimos uma recomendação acadêmica de consenso geral, mencionada por Lauche (2005), que afirma que apenas um entendimento profundo das atividades que estão sendo projetadas permitirá que os sistemas ICT se tornem ferramentas significativas, adequadas à tarefa e ao contexto de uso. É portanto importante não apenas entrevistar, mas também observar os usuários em prática e compreender as implicações dessas descobertas antes de consolidá-las na forma de requisitos. Conduzimos então entrevistas semi-estruturadas com as pessoas chave (veja o Apêndice A) e não apenas observamos sua prática de trabalho, mas de fato participamos com elas de atividades e projetos conjuntos por mais de uma década.

A gestão de desastres de uma estrutura *offshore* de óleo e gás – que pode ser um navio ou uma plataforma semi-submersível – é uma operação complexa que

envolve vários grupos, tais como a companhia de óleo e gás (no nosso caso particular, a companhia brasileira Petrobras), a equipe de resgate, o centro de tratamento da saúde, a imprensa, entre outros. Podemos então ver que esta é de fato uma atividade complexa inter-organizacional, com todas as questões e características já mencionadas no Capítulo 3.

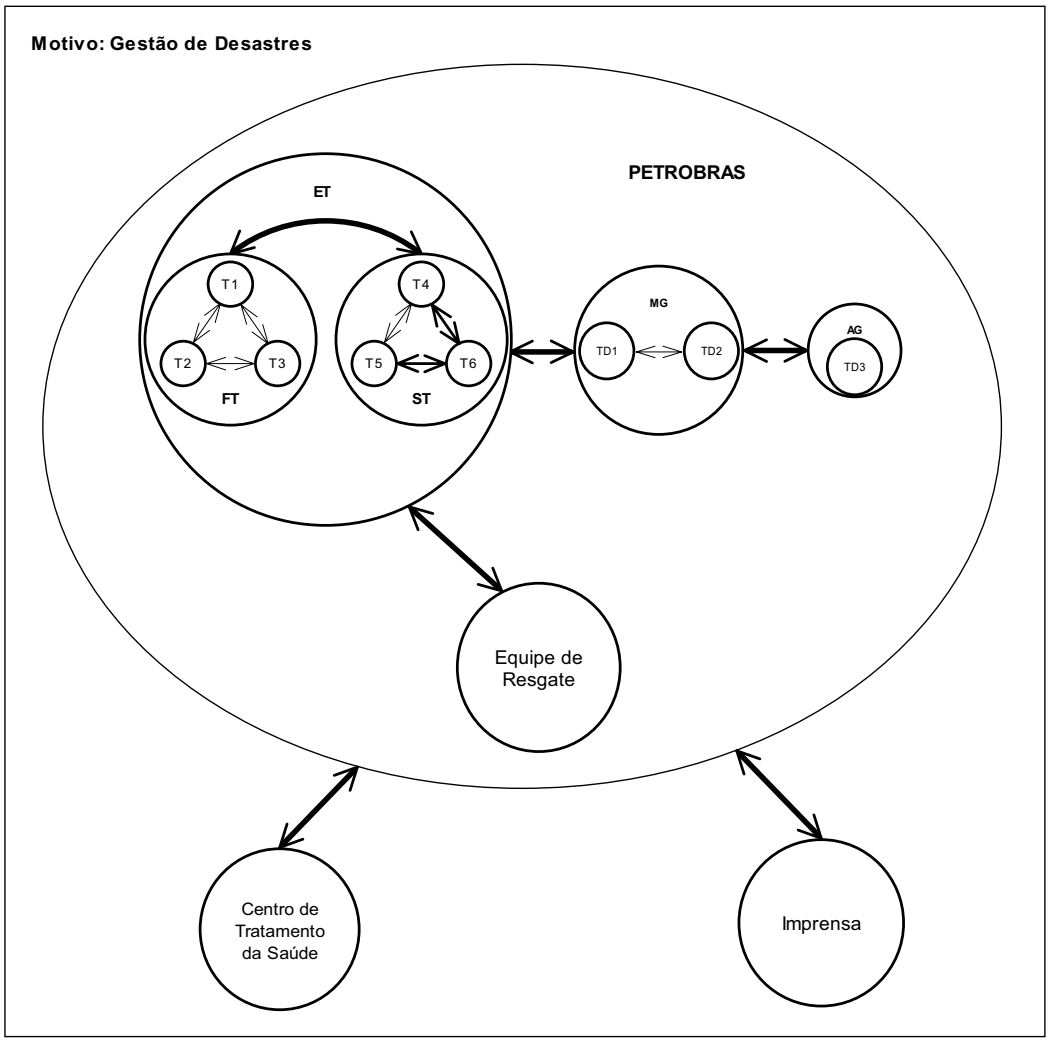

Figura 16 - O modelo colaborativo de gestão de desastres: figura global

Agora descrevemos os principais nós presentes na figura global do modelo de gestão de desastres (Figura 16):

- Representando quatro organizações, existem quatro nós – Petrobras, Equipe de Resgate, Centro de Tratamento da Saúde e Imprensa –, cada um localizado remotamente em relação ao outro. Podemos observar que a Petrobras está no centro de toda a atividade e portanto iremos detalhar o nó Petrobras e nele concentrar nosso foco. Antes

de fazer isso, é interessante situar onde grande parte das estruturas *offshore* da Petrobras está localizada: em uma região chamada Bacia de Campos, próxima ao Rio de Janeiro.

- Dentro do nó Petrobras, identificamos três grupos principais: o grupo das Equipes Técnicas ET, o grupo da Média Gerência MG e o grupo da Alta Gerência AG, cada um localizado remotamente em relação ao outro.
- O grupo ET é formado por dois outros subgrupos técnicos: a equipe de Força Tarefa FT e a equipe de Suporte Técnico ST. Estes dois subgrupos estão também localizados remotamente um em relação ao outro.
- A equipe FT executa o principal papel de toda a atividade de gestão de desastres, liderando o processo de tomada de decisão. Ela é constituída por especialistas tais como engenheiros navais, engenheiros estruturais, analistas de *risers* e oceanógrafos, neste exemplo representados por três técnicos co-localizados, isto é, T1, T2 e T3, sem perda de generalidade. Eles são trazidos para trabalharem juntos em um ambiente que permite a comunicação face-a-face e que tem várias facilidades, tais como acesso a bancos de dados remotos que mantêm modelos CAD e modelos de simulação da unidade. Este ambiente de trabalho pode ser organizado próximo ou distante do desastre – o que é importante é ter comunicação com a área do desastre. Lá a equipe FT executa de modo integrado diferentes simuladores para derivar a melhor solução para salvar a unidade *offshore*, permanentemente se comunicando com a equipe ST. Ela também mantém contato com o grupo MG, informando sobre a evolução de seus trabalhos e solicitando aprovação para a solução concebida. Uma vez que sua solução é aprovada, eles passam para o operador da unidade ou para a equipe de resgate a seqüência de comandos a ser executada. Pode ser observado que não representamos o operador na nossa figura global já que, em termos da aplicação colaborativa, ele não está diretamente envolvido com a simulação integrada, apenas recebendo seu resultado final. Também em alguns casos ele não está diretamente

conectado aos outros participantes em termos de ICT. De fato, poderiam ocorrer duas possíveis situações: *(i)* se a unidade não foi seriamente danificada, dois ou três operadores podem permanecer dentro dela e receber orientação da equipe FT (se algum tipo de comunicação permaneceu disponível, poderíamos incluir o operador na aplicação colaborativa); ou *(ii)* se a unidade foi seriamente danificada e possui problemas de segurança, apenas mergulhadores seriam capazes de trabalhar no local do acidente.

- A equipe ST, representada em nosso exemplo pelos técnicos T4, T5 e T6 sem perda de generalidade, pode ser invocada pela equipe FT para executar simulações especializadas, focando em algumas questões particulares que não poderiam ser realizadas no ambiente de trabalho de FT, ou para obter outra opinião ou visão sobre o problema. T4, T5 e T6 são técnicos trabalhando nas mesmas áreas que os técnicos de FT. No nosso exemplo particular, T4 e T5 estão trabalhando co-localizados, possivelmente em um Centro de Simulação Numérica, com T6 trabalhando remotamente a eles (de algum outro local da companhia, ou até mesmo interagindo via telefone celular).
- MG é constituído por gerentes de nível intermediário, no nosso exemplo TD1 e TD2 (usando TD para Tomador de Decisões) trabalhando co-localizados em um escritório da companhia, com um deles usualmente sendo responsável por tomar a decisão final. Eles possuem um bom conhecimento geral sobre as questões técnicas e trabalham interagindo constantemente com o grupo ET. Eles também se comunicam com o grupo AG, informando sobre a evolução dos trabalhos e eventualmente quando eles precisam tomar uma decisão mais crítica e entendem que seria importante receber a aprovação ou um conselho deste grupo.
- O grupo AG, no nosso exemplo um único gerente, TD3, que poderia ser um diretor trabalhando na sede da companhia, recebe periodicamente do grupo MG informações sobre a evolução dos trabalhos e às vezes é solicitado por eles a dar uma aprovação ou um conselho acerca de uma questão crítica em particular.

Depois de investigar as atividades envolvidas neste cenário de desastre, identificando os requisitos em termos de ICT, decidimos nos concentrar no grupo Equipes Técnicas para desenvolver um primeiro protótipo de aplicação colaborativa implementando um modelo particular do nosso metamodelo Centrado nas Atividades.

O primeiro protótipo é mais particularmente relacionado ao trabalho realizado pelo grupo Força Tarefa, incluindo os simuladores que eles executam, a comunicação entre eles e a interação deles com o grupo Média Gerência. Em termos deste último grupo, apenas para simplificar o protótipo, consideraremos apenas um gerente de nível intermediário integrado à nossa aplicação colaborativa.

Primeiramente investigamos como o grupo Força Tarefa executa os diferentes simuladores e quais são as relações entre eles. Durante a situação de crise, a Petrobras usa tipicamente três simuladores, que descreveremos brevemente a seguir.

O primeiro simulador a ser executado é o SSTAB (Coelho et al., 2003), o sistema de Estabilidade de Unidades Flutuantes, usado para analisar as condições estáticas da unidade flutuante (Figura 17). O SSTAB usa, como entradas, o modelo da unidade obtido de um sistema centralizado denominado GIEN e dados atualizados sobre a unidade obtidos por meio de um sistema de monitoração chamado ECOS. Ele fornece como saídas cinco arquivos, incluindo a matriz de inércia. Disponível em cada unidade, além de ser usado durante situações de emergência, o SSTAB também pode ser usado para planejar operações de manutenção e para projetar novas unidades.

O segundo simulador é denominado WAMIT (2006) e usa como entradas os arquivos de saída gerados pelo SSTAB. Ele trabalha no domínio da freqüência, derivando as forças de excitação da unidade e as forças de reação da água ao deslocamento lateral. O WAMIT é ativado por um programa de interface com o usuário chamado WMG.

Finalmente, o terceiro simulador a ser executado é o DYNASIM (Coelho et al., 2001), para Estabilidade Dinâmica (Figura 18). Ele usa como entradas os resultados obtidos do WAMIT, como também os parâmetros H e P, representando respectivamente a altura (em inglês, *height*) e o período (em inglês, *period*) da onda no momento do desastre. O DYNASIM trabalha no domínio do tempo e de

fato possui outros dois módulos além do módulo central: um pré-processador chamado Pre-Dyna e um pós-processador chamado Pos-Dyna. O DYNASIM calcula as forças que agem nas linhas de ancoragem e nos *risers*, e o momento tombante. Quando essas forças são consideradas extremas, um processo de retro-alimentação é iniciado, executando todas as simulações novamente, começando com o SSTAB, para encontrar uma outra condição de estabilidade da unidade.

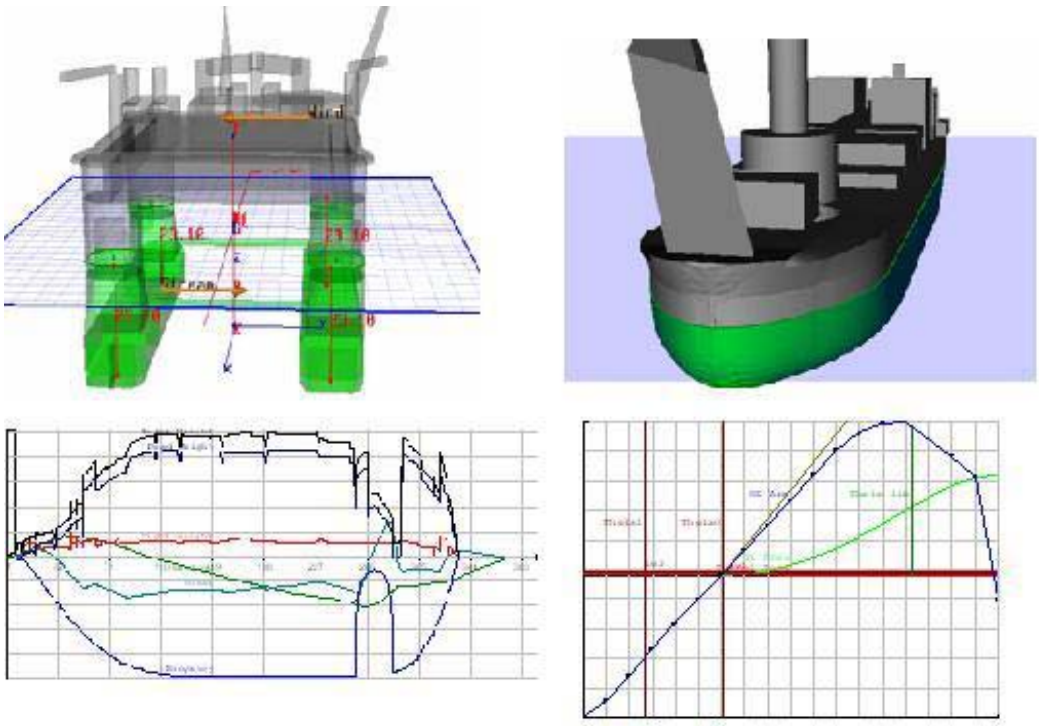

Figura 17 - SSTAB: sistema de Estabilidade de Unidades Flutuantes

De modo a ilustrar melhor como os simuladores se comunicam, tomamos o exemplo da unidade FPSO (*Floating Production, Storage and Offloading* – Produção, Armazenamento e Descarregamento Flutuante) P-34 e descrevemos abaixo os arquivos e parâmetros que eles usam:

- No começo da simulação do SSTAB, o operador do SSTAB abre o arquivo *p34.sst*, que contém o modelo geométrico da unidade.
- No final da simulação do SSTAB, o operador do SSTAB exporta um arquivo de dados geométricos do WAMIT de nome *p34.gdf*. De fato, quatro outros arquivos são exportados para o WAMIT neste momento: *fnames.wam*, que contém a lista de nomes de arquivos; *p34.pot*, que é o Arquivo de Controle de Potenciais; *p34.frc*, que é o

Arquivo de Controle de Forças; e *p34.cfg*, que é o arquivo de configurações do WAMIT (2006).

- O WAMIT então lê os cinco arquivos exportados pelo SSTAB e executa sua simulação.
- No final da simulação, o WAMIT gera um arquivo de saída chamado *p34.out*, contendo as forças de excitação que agem sobre a unidade.
- O operador do DYNASIM então abre o arquivo de projeto do DYNASIM *p34.prd*.
- Ele então abre o arquivo de saída do WAMIT *p34.out*, convertendo-o em um arquivo neutro do WAMIT *p34.wnf*.
- Ele finalmente entra com os parâmetros ambientais H e P recebidos, iniciando a simulação do DYNASIM.

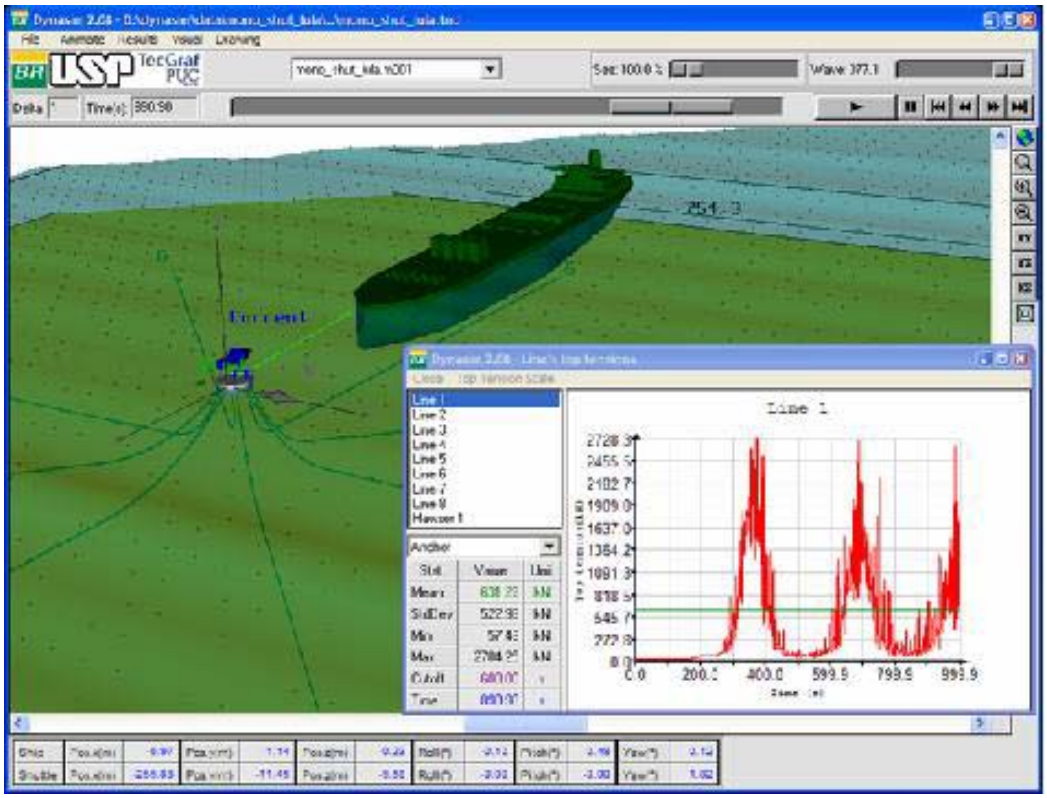

Figura 18 - DYNASIM: sistema de Estabilidade Dinâmica

O método utilizado para salvar a unidade *offshore* tem por objetivo definir uma seqüência de comandos a ser passada para os operadores da unidade ou para a equipe de resgate, de maneira que eles possam mover a unidade passo-a-passo,

desde sua condição instável inicial até que alcance de volta seu estado de equilíbrio normal. Este método é baseado no seguinte fluxo de trabalho:

- Primeiramente, usamos esses três simuladores (possivelmente usando retro-alimentação) para derivar as condições iniciais da unidade *offshore*.
- Definimos então uma configuração de tanques do próximo passo (por exemplo, movendo água de um tanque de lastro de um lado para um tanque de lastro do lado oposto, operando as válvulas dos tanques) e simulamos a unidade nesta nova condição usando novamente os três simuladores. Se não ficarmos satisfeitos com os resultados obtidos, definimos outra configuração de tanques e continuamos este processo, experimentando iterativamente configurações, até ficarmos satisfeitos com uma delas. Neste caso, dizemos que alcançamos a configuração de tanques do passo atual.
- A partir da configuração do passo anterior, tentamos agora derivar uma nova configuração de tanques, usando um processo análogo ao que acabou de ser descrito.
- Repetimos esse processo de derivar uma configuração de tanques para cada passo até alcançarmos de volta um estado de equilíbrio normal.

No final de todo esse processo, temos uma seqüência de comandos em termos de operações de válvulas de tanques, correspondentes a cada uma das configurações de passo descritas acima, de um modo passo-a-passo, o que era exatamente o nosso objetivo. No desastre da P-34, sem nenhum operador de unidade dentro dela, as ações eram tomadas de fora da plataforma, usando mangueiras para encher os tanques.

O que obtemos como resultado do emprego do método acima é uma seqüência de comandos que constitui uma única estratégia. Poderíamos, é claro, aplicar este método diversas vezes, derivando várias estratégias. Poderíamos também usar retorno do campo após a aplicação de cada passo sobre a unidade real, refinando passos futuros com esta nova informação. Finalmente, poderíamos coletar mudanças nas condições da unidade ou na área do desastre e introduzi-las

nas simulações. Na P-34, um navio de monitoração chamado Salgueiro e o grupo oceanográfico forneciam dados ambientais 24 horas por dia.

É importante notar que as execuções dos simuladores SSTAB e DYNASIM são processos de visualização altamente interativos. Em uma situação de crise, quando necessitamos testar rapidamente várias alternativas para responder ao desastre, esta característica dos simuladores é explorada intensamente. Também devemos considerar que, em situações de emergência, é muito importante ser tão rápido quanto possível, porque cada minuto perdido pode ser crucial no processo de salvamento da unidade. Então, procurando por pontos em que poderíamos poupar tempo, descobrimos que, se o WAMIT receber os resultados do SSTAB, ele pode ser automaticamente ativado assim que terminar a simulação do SSTAB.

5.1.1. Um Primeiro Modelo para a Aplicação Colaborativa de Gestão de Desastres

Nesta subseção, derivamos um primeiro modelo Centrado nas Atividades para a aplicação colaborativa de gestão de desastres. Concentramo-nos na atividade do grupo Equipes Técnicas, que é realizada principalmente pelos membros do grupo Força Tarefa executando seus simuladores (Figura 19).

O nível mais alto da parte técnica deste cenário de crise, o motivo, é a Simulação Integrada (SI), que possui dois nós localizados remotamente um em relação ao outro: BR, a companhia de óleo e gás, e Imprensa.

Dentro do BR, criamos dois grupos remotos, um em relação ao outro, Equipes Técnicas (ET) e Tomadores de Decisão (TD):

- ET é constituído pela equipe de Força Tarefa (FT), que possui os membros T0, T1 e T3, e o agente de software S2, todos eles co-localizados na sala da Força Tarefa e executando seus simuladores. Uma vez que em nosso modelo FT é a única equipe dentro de ET, não a estamos representando explicitamente (se ela possuir políticas de entrada e saída particulares, ela também deveria ser um nó do nosso modelo).
- TD neste modelo é constituído por um único gerente TD1 (Tomador de Decisões 1). Sem perda de generalidade, TD1 pode ser considerado um único representante de todos os participantes não

diretamente envolvidos com a parte técnica da atividade de simulação, tais como operadores e outros gerentes, que apenas recebem do grupo ET mensagens de acompanhamento, comandos a serem executados (no caso de operadores) ou solicitações de aprovação (no caso de gerentes), fornecendo respostas simples a eles.

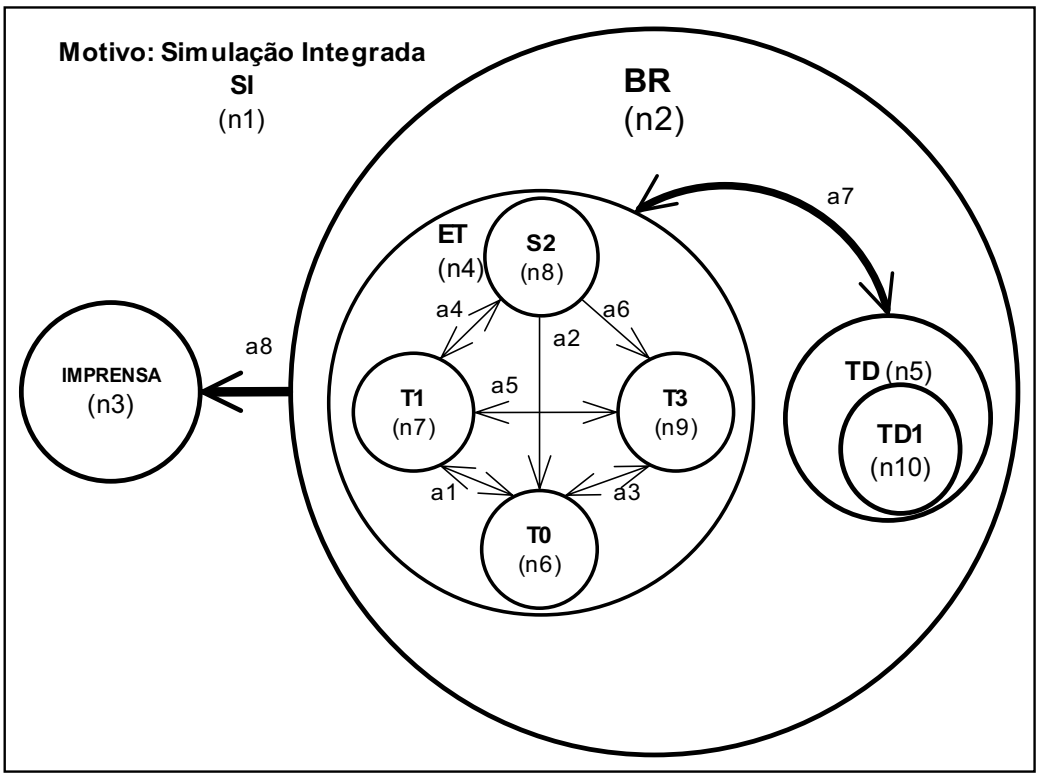

Figura 19 - Um primeiro modelo para a aplicação colaborativa de gestão de desastres, focando na simulação integrada

A Imprensa está representada por um único nó, que apenas recebe relatórios periódicos da companhia de óleo e gás informando sobre a evolução do desastre.

Existem alguns casos em que temos um tomador de decisões técnico, que participa diretamente da atividade de simulação. Por exemplo, no desastre da P-34 o tomador de decisões estava na sala de guerra também participando da execução dos simuladores. Neste caso, seria mais apropriado alocar o nó TD dentro do grupo ET.

Os registros de carga para os nós e arestas correspondentes ao componente de rede deste primeiro modelo são mostrados na Tabela 10.

1 | 1 | Simulação Integrada | SI | GRUPO | 0 | 6 | …
1 | 2 | Companhia de Óleo e Gás | BR | GRUPO | 1 | 6 | …
1 | 3 | Imprensa | IMPRENSA | FOLHA | 1 | 0 | …
1 | 4 | Equipes Técnicas | ET | GRUPO | 2 | 6 | …
1 | 5 | Tomadores de Decisões | TD | GRUPO | 2 | 10 | …
1 | 6 | Piloto da Emergência | T0 | FOLHA | 4 | 0 | …
1 | 7 | Operador do SSTAB | T1 | FOLHA | 4 | 0 | …
1 | 8 | WAMIT | S2 | FOLHA | 4 | 0 | …
1 | 9 | Operador do DYNASIM | T3 | FOLHA | 4 | 0 | …
1 | 10 | Tomador de Decisões 1 | TD1 | FOLHA | 5 | 0 | …
2 | 1 | Canal Piloto-SSTAB | LOCAL | BIDIRECIONADO | 6 | 7 | …
2 | 2 | Canal WAMIT-Piloto | LOCAL | UNIDIRECIONADO | 8 | 6 | …
2 | 3 | Canal Piloto-DYNASIM | LOCAL | BIDIRECIONADO | 6 | 9 | …
2 | 4 | Canal SSTAB-WAMIT | LOCAL | BIDIRECIONADO | 7 | 8 | …
2 | 5 | Canal SSTAB-DYNASIM | LOCAL | BIDIRECIONADO | 7 | 9 | …
2 | 6 | Canal WAMIT-DYNASIM | LOCAL | UNIDIRECIONADO | 8 | 9 | …
2 | 7 | Canal técnico-gerencial | REMOTO | BIDIRECIONADO| 4 | 5 | …
2 | 8 | Canal técnico-imprensa | REMOTO | UNIDIRECIONADO | 2 | 3 | …

Tabela 10 - Um primeiro modelo para a aplicação de gestão de desastres: registros de carga

Além do componente de rede de interações do modelo recém-descrito, também definimos regras de papéis e a tabela de atributos de mensagens, de modo a representar os seguintes papéis diferentes desempenhados pelos membros participantes deste cenário de desastre:

- O Piloto da Emergência T0 desempenha o papel principal nesta aplicação de desastres, coordenando a sessão colaborativa e liderando o processo de tomada de decisão. Ele solicita que o operador do SSTAB (T1) inicie sua simulação. Após receber uma mensagem do agente S2 indicando o fim da simulação, ele solicita que o operador do DYNASIM (T3) inicie sua simulação, informando os valores correntes de H e P das ondas. Ao receber de T3 uma mensagem de conclusão de simulação, ele toma uma decisão baseada nos valores das forças que estão agindo sobre as linhas de ancoragem e *risers*. Se ele entende que essas forças são extremas, ele

solicita que T1 comece todo o processo novamente, de modo a encontrar uma nova condição de equilíbrio da unidade, e este laço continua até que ele esteja satisfeito com os valores de forças obtidos. Neste caso, a simulação para a condição inicial da unidade é considerada encerrada. T0 então repete este processo para cada passo de configuração do método já descrito, até que ele determina o fim do processo de simulação ao alcançar a seqüência de comandos desejada. Neste caso, ele faz contato com o Tomador de Decisões (TD1), solicitando sua aprovação para a solução encontrada. Se TD1 a aprova, T0 então passa a seqüência de comandos a ser executada para o operador da unidade (não representado aqui), para salvar a unidade.

- T1, o operador do SSTAB, inicia a simulação do SSTAB toda vez que ele recebe uma ordem de T0. Ele estuda interativamente várias configurações de tanques tentando derivar uma configuração de tanques para o passo atual, de acordo com o método descrito acima, e informa quando termina a simulação deste passo.
- S2, Simulador 2, é de fato um agente reativo em lugar de uma pessoa real (usado aqui para poupar tempo). Ao receber uma mensagem de T1 indicando o fim de sua simulação, S2 automaticamente executa o simulador WAMIT, usando a informação passada por T1 (os resultados do SSTAB). S2 informa aos membros participantes quando ele inicia e termina a sua simulação.
- T3, o operador do DYNASIM, inicia uma simulação do DYNASIM toda vez que ele recebe uma ordem de T0. Ele usa como entradas os resultados dos outros dois simuladores, H e P, e estuda interativamente a estabilidade dinâmica da unidade baseada nessas condições. Quando termina a sua simulação, ele informa aos participantes seu resultado, que pode ser *vermelho* (forças extremas estão atuando), *amarelo* (forças moderadas agindo) ou *verde* (forças leves agindo).
- Finalmente TD1, o Tomador de Decisões, recebe de T0 a seqüência de comandos que define uma estratégia e responde a ele aprovando

ou não a estratégia definida. Caso não aprove a estratégia, ele pode solicitar a T0 que reinicie todo o processo novamente ou pode discutir mais o problema com o grupo Alta Gerência, não representado aqui.

Devemos observar que todos os participantes devem ser informados sobre o trabalho dos outros durante a sessão colaborativa.

Embora o fluxo de trabalho completo seja o que acabou de ser descrito, para facilitar o entendimento iremos considerar no momento em nosso modelo apenas um passo de configuração com possível retro-alimentação. Com esta simplificação, as regras de papéis correspondentes ao nosso fluxo de trabalho são as mostradas nas Tabelas 11, 12, 13 e 14.

Além das regras *on-arrive* e *on-init*, e das fórmulas *send* e *display* já introduzidas no Capítulo 3, identificamos a necessidade de definir regras e fórmulas adicionais, seguindo a terminologia Prolog (SWI-Prolog, 2006), de modo a descrever completamente as regras de papéis do nosso modelo:

- regra *when* com argumentos *term* e *term_value*: a regra é disparada quando *term* no banco de conhecimentos tem valor *term_value*;
- fórmula *read*: é usada para ler uma entrada do console;
- fórmula *write*: é usada para escrever conteúdos das mensagens na tabela de atributos de mensagens;
- predicado *assert* com argumento *term*: é usado para adicionar um fato ou cláusula no banco de dados.
- predicado *abolish* com argumento *term*: é usado para remover uma cláusula do banco de dados (não sendo usada no presente modelo).

Podemos agora descrever brevemente os conteúdos das Tabelas 11, 12, 13 e 14. Na Tabela 11, apresentamos a definição da barra de colaboração e as regras de papel para o Piloto da Emergência. A barra do nosso exemplo tem um canal local e um canal remoto. As regras que definem o papel do Piloto da Emergência são semelhantes às do exemplo já mostrado no Capítulo 3, com algumas novas declarações. Quando chega a mensagem 5, os valores H e P têm que ser lidos do console (usando a fórmula *read*) e escritos na linha correspondente à mensagem 6

na tabela de atributos de mensagens (usando a fórmula *write*) antes de ela ser enviada aos receptores. Também, quando a mensagem 8 chega, o termo *resultado_simulacoes* tem seu valor lido do console (usando a fórmula *read*) e é adicionado ao banco de conhecimentos (usando o predicado *assert*). Então a seqüência de comandos a ser executada é determinada por duas regras *when* subseqüentes, com base no valor de *resultado_simulacoes*. Finalmente, a última seqüência de comandos a ser executada é determinada pelas duas últimas regras *when*, com base no valor de *aprovacao_simulacoes*. É interessante notar que, em ambas as seqüências de comandos disparadas pelas duas regras *when*, aguardamos o usuário pressionar um botão (usando a fórmula *read*), o primeiro para parar a simulação e o segundo para enviar o relatório de notícias para a Imprensa.

Na Tabela 12, apresentamos as regras de papéis para o operador do SSTAB e para o simulador WAMIT. As regras e fórmulas básicas usadas são semelhantes às apresentadas na Tabela 11, com pequenas diferenças. Na seqüência de comandos da primeira regra *on_arrive* do papel do operador do SSTAB, lemos *botao_inicio_SSTAB* e *botao_fim_SSTAB* (usando a fórmula *read*) para indicar, respectivamente, o início e o fim da simulação do SSTAB. Nas regras de papel do simulador WAMIT, como este simulador é acionado automaticamente quando chega a mensagem 3, definimos um método chamado *wamit()* para ativar o simulador WAMIT. Também utilizamos um termo chamado *fim_wamit* para indicar o fim deste simulador (usado como um argumento da última regra *when*).

Na Tabela 13, apresentamos as regras de papel para o operador do DYNASIM, também com as mesmas regras e fórmulas básicas, como as apresentadas nas Tabelas 11 e 12. Um aspecto interessante que podemos destacar aqui é a seqüência de três regras *when* usadas para determinar a seqüência de comandos a ser executada depois do fim da simulação do DYNASIM. A seleção de comandos é feita baseada no valor do termo *resultado_DYNASIM*, que foi previamente lido do console e que pode assumir um dentre três valores possíveis: *vermelho*, indicando que existem forças extremas atuando; *amarelo*, indicando a ação de forças moderadas; ou *verde*, indicando a ação de forças leves.

Na Tabela 14, apresentamos as regras de papéis para o Tomador de Decisões, a Imprensa e para o grupo Equipes Técnicas. Novamente, as regras e fórmulas básicas são semelhantes às já mostradas nas Tabelas 11, 12 e 13. Nas

```
collaboration simulacao_integrada
{ collaboration_bus {
            channel(local).
            channel(remoto).}
 role piloto_emergencia //técnico responsável pela coordenação das simulações
 {
  on-init(simulacao_integrada) :-
   send(local, tab_mensagem(source(self), 1, dummy)). //envia "Técnico 1, por favor inicie a simulação do SSTAB."

  on-arrive(local, tab_mensagem(dummy, 2, source(self))) :-
   display(tab_mensagem(dummy, 2, source(self))). //mostra "Simulação do SSTAB começou."

  on-arrive(local, tab_mensagem(dummy, 3, source(self))) :-
   display(tab_mensagem(dummy, 3, source(self))). //mostra "Fim da simulação do SSTAB."

  on-arrive(local, tab_mensagem(dummy, 4, source(self))) :-
   display(tab_mensagem(dummy, 4, source(self))). //mostra "Simulação do WAMIT começou."

  on-arrive(local, tab_mensagem(dummy, 5, source(self))) :-
   display(tab_mensagem(dummy, 5, source(self))), //mostra "Fim da simulação do WAMIT."
   read H e P, //lê H e P do console
   write mensagem 6, //escreve mensagem 6 com valores H e P na tabela de atributos de mensagens
   send(local, tab_mensagem(source(self), 6, dummy)). //envia "Técnico 3, por favor inicie o DYNASIM com h e p."

  on-arrive(local, tab_mensagem(dummy, 7, source(self))) :-
   display(tab_mensagem(dummy, 7, source(self))). //mostra "Simulação do DYNASIM começou."

  on-arrive(local, tab_mensagem(dummy, 8, source(self))) :-
   display(tab_mensagem(dummy, 8, source(self))), //mostra o resultado do DYNASIM
   read resultado_simulacoes, //lê resultado_simulacoes (OK ou nao_OK) do console
   assert resultado_simulacoes. //grava resultado_simulacoes no banco de conhecimentos

  when(resultado_simulacoes, nao_OK) :-
   send(local, tab_mensagem(source(self), 9, dummy)), //envia "Um novo ciclo de simulações vai começar."
   send(local, tab_mensagem(source(self), 1, dummy)). //envia "Técnico 1, por favor inicie a simulação do SSTAB."

  when(resultado_simulacoes, OK) :-
   send(remoto, tab_mensagem(source(self), 10, dummy))//envia "Tomador de Decisões, por favor aprove as simulações."

  on-arrive(remoto, tab_mensagem(dummy, 11, source(self))) :-
   display(tab_mensagem(dummy, 11, source(self))). //mostra a decisão tomada

  when(aprovacao_simulacoes, nao_OK) :-
     read botao_parada_simulacoes. //lê botao_parada_simulacoes do console

  when(aprovacao_simulacoes, OK) :-
   read botao_envia_para_imprensa, //lê botao_envia_para_imprensa do console
   send(remoto, tab_mensagem(source(self), 12, dummy)). //envia "Novo relatório de notícias postado."
}
```

Tabela 11 - Definição da barra de colaboração e das regras para o Piloto da Emergência

regras do Tomador de Decisões, temos uma seqüência semelhante à apresentada no papel do Piloto da Emergência: quando a mensagem 10 chega, o termo *aprovacao_simulacoes* tem seu valor lido do console (usando a fórmula *read*) e é adicionado ao banco de conhecimentos (usando o predicado *assert*); então a seqüência de comandos a ser executada é determinada por duas regras *when*

```
role tecnico_1 //técnico responsável pela execução da simulação do SSTAB
{
 on-arrive(local, tab_mensagem(dummy, 1, source(self))) :-
   display(tab_mensagem(dummy, 1, source(self))), //mostra "Técnico 1, por favor inicie a simulação do SSTAB."
   read botao_inicio_SSTAB, //lê botao_inicio_SSTAB do console
   send(local, tab_mensagem(source(self), 2, dummy)), //envia "Simulação do SSTAB começou."
   read botao_fim_SSTAB, //lê botao_fim_SSTAB do console
   send(local, tab_mensagem(source(self), 3, dummy)). //envia "Fim da simulação do SSTAB."

 on-arrive(local, tab_mensagem(dummy, 4, source(self))) :-
   display(tab_mensagem(dummy, 4, source(self))). //mostra "Simulação do WAMIT começou."

 on-arrive(local, tab_mensagem(dummy, 5, source(self))) :-
   display(tab_mensagem(dummy, 5, source(self))). //mostra "Fim da simulação do WAMIT."

 on-arrive(local, tab_mensagem(dummy, 7, source(self))) :-
   display(tab_mensagem(dummy, 7, source(self))). //mostra "Simulação do DYNASIM começou."

 on-arrive(local, tab_mensagem(dummy, 8, source(self))) :-
   display(tab_mensagem(dummy, 8, source(self))). //mostra o resultado do DYNASIM

 on-arrive(local, tab_mensagem(dummy, 9, source(self))) :-
   display(tab_mensagem(dummy, 9, source(self))). //mostra "Um novo ciclo de simulações vai começar."

 on-arrive(local, tab_mensagem(dummy, 10, source(self))) :-
   display(tab_mensagem(dummy, 10, source(self))). //mostra "Tomador de Decisões, por favor aprove as simulações."

 on-arrive(remoto, tab_mensagem(dummy, 11, source(self))) :-
   display(tab_mensagem(dummy, 11, source(self))). //mostra a decisão tomada
}

role simulador_2 //simulador WAMIT
{
 on-arrive(local, tab_mensagem(dummy, 3, source(self))) :-
   wamit(), //inicia a simulação do WAMIT
   send(local, tab_mensagem(source(self), 4, dummy)). //envia "Simulação do WAMIT começou."

 when(fim_wamit, OK) :-
   send(local, tab_mensagem(source(self), 5, dummy)). //envia "Fim da simulação do WAMIT."
}
```

Tabela 12 - Regras de papéis para o operador do SSTAB e para o simulador WAMIT

subseqüentes, com base no valor de *aprovacao_simulacoes*. A regra de papel da Imprensa é simplesmente uma *on-arrive* disparada quando chega a mensagem 12, indicando que um novo relatório de notícias foi tornado disponível. Finalmente, o papel do grupo Equipes Técnicas é de destaque porque ele é o único neste

```
role tecnico_3 //técnico responsável pela execução da simulação do DYNASIM
{
 on-arrive(local, tab_mensagem(dummy, 2, source(self))) :-
  display(tab_mensagem(dummy, 2, source(self))). //mostra "Simulação do SSTAB começou."

 on-arrive(local, tab_mensagem(dummy, 3, source(self))) :-
  display(tab_mensagem(dummy, 3, source(self))). //mostra "Fim da simulação do SSTAB."

 on-arrive(local, tab_mensagem(dummy, 4, source(self))) :-
  display(tab_mensagem(dummy, 4, source(self))). //mostra "Simulação do WAMIT começou."

 on-arrive(local, tab_mensagem(dummy, 5, source(self))) :-
  display(tab_mensagem(dummy, 5, source(self))). //mostra "Fim da simulação do WAMIT."

 on-arrive(local, tab_mensagem(dummy, 6, source(self))) :-
  display(tab_mensagem(dummy, 6, source(self))), //mostra "Técnico 3, por favor inicie o DYNASIM com h e p."
  read botao_inicio_DYNASIM, //lê botao_inicio_DYNASIM do console
  send(local, tab_mensagem(source(self), 7, dummy)), //envia "Simulação do DYNASIM começou."
  read resultado_DYNASIM, //lê resultado_DYNASIM do console (vermelho, amarelo ou verde)
  assert resultado_DYNASIM. //grava resultado_DYNASIM no banco de conhecimentos

 when(resultado_DYNASIM, vermelho) :-
  write mensagem 8 com "Forças extremas: você deve começar um novo ciclo de simulações.",
  send(local, tab_mensagem(source(self), 8, dummy)). //envia mensagem 8

 when(resultado_DYNASIM, amarelo) :-
  write mensagem 8 com "Forças moderadas: você deve decidir se vai ou não executar um novo ciclo de simulações.",
  send(local, tab_mensagem(source(self), 8, dummy)). //envia mensagem 8

 when(resultado_DYNASIM, verde) :-
  write mensagem 8 com "Forças leves: você pode aprovar a simulação atual.",
  send(local, tab_mensagem(source(self), 8, dummy)). //envia mensagem 8

 on-arrive(local, tab_mensagem(dummy, 9, source(self))) :-
  display(tab_mensagem(dummy, 9, source(self))). //mostra "Um novo ciclo de simulações vai começar."

 on-arrive(local, tab_mensagem(dummy, 10, source(self))) :-
  display(tab_mensagem(dummy, 10, source(self))). //mostra "Tomador de Decisões, por favor aprove as simulações."

 on-arrive(remoto, tab_mensagem(dummy, 11, source(self))) :-
  display(tab_mensagem(dummy, 11, source(self))). //mostra a decisão tomada
}
```

Tabela 13 - Regras de papel para o operador do DYNASIM

```
role tomador_decisoes //gerente responsável pela tomada de decisão
{
  on-arrive(remoto, tab_mensagem(dummy, 2, source(self))) :-
    display(tab_mensagem(dummy, 2, source(self))). //mostra "Simulação do SSTAB começou."

  on-arrive(remoto, tab_mensagem(dummy, 3, source(self))) :-
    display(tab_mensagem(dummy, 3, source(self))). //mostra "Fim da simulação do SSTAB."

  on-arrive(remoto, tab_mensagem(dummy, 4, source(self))) :-
    display(tab_mensagem(dummy, 4, source(self))). //mostra "Simulação do WAMIT começou."

  on-arrive(remoto, tab_mensagem(dummy, 5, source(self))) :-
    display(tab_mensagem(dummy, 5, source(self))). //mostra "Fim da simulação do WAMIT."

  on-arrive(remoto, tab_mensagem(dummy, 7, source(self))) :-
    display(tab_mensagem(dummy, 7, source(self))). //mostra "Simulação do DYNASIM começou."

  on-arrive(remoto, tab_mensagem(dummy, 8, source(self))) :-
    display(tab_mensagem(dummy, 8, source(self))). //mostra o resultado do DYNASIM

  on-arrive(remoto, tab_mensagem(dummy, 9, source(self))) :-
    display(tab_mensagem(dummy, 9, source(self))). //mostra "Um novo ciclo de simulações vai começar."

  on-arrive(remoto, tab_mensagem(dummy, 10, source(self))) :-
    display(tab_mensagem(dummy, 10, source(self))), //mostra "Tomador de Decisões, por favor aprove as simulações."
    read aprovacao_simulacoes, //lê aprovacao_simulacoes (OK ou nao_OK) do console
    assert aprovacao_simulacoes. //grava aprovacao_simulacoes no banco de conhecimentos

  when(aprovacao_simulacoes, nao_OK) :-
    write mensagem 11 com "Pare as simulações: nível mais alto requisitado.",
    send(remoto, tab_mensagem(source(self), 11, dummy)). //envia mensagem 11

  when(aprovacao_simulacoes, OK) :-
    write mensagem 11 com "Simulações aprovadas.",
    send(remoto, tab_mensagem(source(self), 11, dummy)). //envia mensagem 11
}

role imprensa //a imprensa recebe o relatório de notícias
{
  on-arrive(remoto, tab_mensagem(dummy, 12, source(self))) :-
    display(tab_mensagem(dummy, 12, source(self))). //mostra "Novo relatório de notícias postado."
}

role equipes_tecnicas//nó Equipes Técnicas: papel processado pelo nó = atributo de nó de execução de pós-processamento
{
  on-arrive(remoto, tab_mensagem(dummy, 11, source(self))) :-
    display(tab_mensagem(dummy, 11, source(self))). //pos_11_ET determina que nós receberão a mensagem 11
}
}
```

Tabela 14 - Regras de papéis para Tomador de Decisões, Imprensa e Equipes Técnicas

exemplo associado a um grupo: quando a mensagem 11 chega, a regra *on-arrive* dispara e uma mensagem é mostrada no console informando que o módulo de pós-processamento *pos_11_ET* está determinando e re-roteando a mensagem 11 para os receptores apropriados dentro de ET que devem receber a mensagem 11 – no nosso exemplo, os nós folhas T0 (Piloto da Emergência), T1 (operador do SSTAB) e T3 (operador do DYNASIM), como podemos verificar nas Tabelas 11, 12 e 13, nas regras *on-arrive* recebendo a mensagem 11.

Na Tabela 15, apresentamos a tabela de atributos de mensagens completa para nosso primeiro modelo, incluindo o campo conteúdo da mensagem, para facilitar o entendimento. É importante notar que algumas mensagens possuem conteúdos constantes, enquanto outras possuem conteúdos que mudam à medida que o fluxo de trabalho vai sendo processado (no nosso exemplo, mensagens 8 e 11).

Nesta tabela, observamos duas colunas correspondentes a dois dos mais importantes elementos que fornecem flexibilidade ao nosso metamodelo: os módulos de pré e pós-processamento. Podemos ver, por exemplo, que a mensagem 9 é enviada para três diferentes receptores – T1, T3 e TD1 – com um módulo particular de pré-processamento e um módulo particular de pós-processamento associado a cada receptor. O mesmo ocorre com as mensagens 2, 3, 4, 5, 7, 8 e 10. Também podemos observar que, na linha associada à mensagem 11, TD1 está enviando a mensagem para o grupo ET, o que significa que *Pos_11_ET* (executado por T0) irá determinar, em tempo de execução, quais nós folhas irão receber a mensagem 11: T0, T1 e T3.

De modo a explorar melhor essas capacidades de pré e pós-processamento do nosso metamodelo, estamos considerando que T0, T1, TD1 e Imprensa usam língua portuguesa enquanto T3 usa língua inglesa. Isto significa que devemos ter traduções automáticas sendo executadas pelos seguintes módulos de processamento:

- *Pre_6_T3*, *Pre_9_T3*, *Pre_10_T3*, *Pre_2_T3*, *Pre_3_T3*, *Pre_4_T3* e *Pre_5_T3*: de português para inglês.
- *Pos_7_T0*, *Pos_7_T1*, *Pos_7_TD1*, *Pos_8_T0*, *Pos_8_T1* e *Pos_8_TD1*: de inglês para português.

remetente	id_mensagem	receptor	aresta	conteúdo da mensagem	pré-processamento	pós-processamento
T0	1	T1	a1	Técnico 1, por favor inicie a simulação do SSTAB.	Pre_1_T1	Pos_1_T1
T0	6	T3	a3	Técnico 3, por favor inicie o DYNASIM com h e p.	Pre_6_T3	Pos_6_T3
T0	9	T1	a1	Um novo ciclo de simulações vai começar.	Pre_9_T1	Pos_9_T1
T0	9	T3	a3	Um novo ciclo de simulações vai começar.	Pre_9_T3	Pos_9_T3
T0	9	TD1	a7	Um novo ciclo de simulações vai começar.	Pre_9_TD1	Pos_9_TD1
T0	10	TD1	a7	Tomador de Decisões, por favor aprove as simulações.	Pre_10_TD1	Pos_10_TD1
T0	10	T1	a1	Tomador de Decisões, por favor aprove as simulações.	Pre_10_T1	Pos_10_T1
T0	10	T3	a3	Tomador de Decisões, por favor aprove as simulações.	Pre_10_T3	Pos_10_T3
T0	12	IMPRENSA	a8	Novo relatório de notícias postado.	Pre_12_IMPRENSA	Pos_12_IMPRENSA
T1	2	T0	a1	Simulação do SSTAB começou.	Pre_2_T0	Pos_2_T0
T1	2	T3	a5	Simulação do SSTAB começou.	Pre_2_T3	Pos_2_T3
T1	2	TD1	a7	Simulação do SSTAB começou.	Pre_2_TD1	Pos_2_TD1
T1	3	T0	a1	Fim da simulação do SSTAB.	Pre_3_T0	Pos_3_T0
T1	3	T3	a5	Fim da simulação do SSTAB.	Pre_3_T3	Pos_3_T3
T1	3	TD1	a7	Fim da simulação do SSTAB.	Pre_3_TD1	Pos_3_TD1
S2	4	T0	a2	Simulação do WAMIT começou.	Pre_4_T0	Pos_4_T0
S2	4	T1	a4	Simulação do WAMIT começou.	Pre_4_T1	Pos_4_T1
S2	4	T3	a6	Simulação do WAMIT começou.	Pre_4_T3	Pos_4_T3
S2	4	TD1	a7	Simulação do WAMIT começou.	Pre_4_TD1	Pos_4_TD1
S2	5	T0	a2	Fim da simulação do WAMIT.	Pre_5_T0	Pos_5_T0
S2	5	T1	a4	Fim da simulação do WAMIT.	Pre_5_T1	Pos_5_T1
S2	5	T3	a6	Fim da simulação do WAMIT.	Pre_5_T3	Pos_5_T3
S2	5	TD1	a7	Fim da simulação do WAMIT.	Pre_5_TD1	Pos_5_TD1
T3	7	T0	a3	Simulação do DYNASIM começou.	Pre_7_T0	Pos_7_T0
T3	7	T1	a5	Simulação do DYNASIM começou.	Pre_7_T1	Pos_7_T1
T3	7	TD1	a7	Simulação do DYNASIM começou.	Pre_7_TD1	Pos_7_TD1
T3	8	T0	a3	Resultado do DYNASIM.	Pre_8_T0	Pos_8_T0
T3	8	T1	a5	Resultado do DYNASIM.	Pre_8_T1	Pos_8_T1
T3	8	TD1	a7	Resultado do DYNASIM.	Pre_8_TD1	Pos_8_TD1
TD1	11	ET	a7	Decisão tomada.	Pre_11_ET	Pos_11_ET

Tabela 15 - A tabela de atributos de mensagens do primeiro modelo para a aplicação de desastres

Com as regras de papéis e a tabela de atributos de mensagens descritas acima, somos capazes de armazenar os dados da sessão colaborativa em um banco de dados, tais como o nome do executor, a data e a hora de cada ação realizada. Esta é uma característica muito útil, auxiliando os especialistas quando da elaboração de um relatório de investigação a respeito do acidente, como também servindo para propósitos de treinamento.

5.1.1.1. Um Protótipo HLA para o Primeiro Modelo para a Aplicação Colaborativa de Gestão de Desastres

Agora alcançamos o último passo no desenvolvimento de nossa aplicação colaborativa de gestão de desastres, que é mapear nosso modelo em uma arquitetura no nível da implementação. Nesta primeira subseção, apresentamos um protótipo HLA para o nosso primeiro modelo (Figura 20).

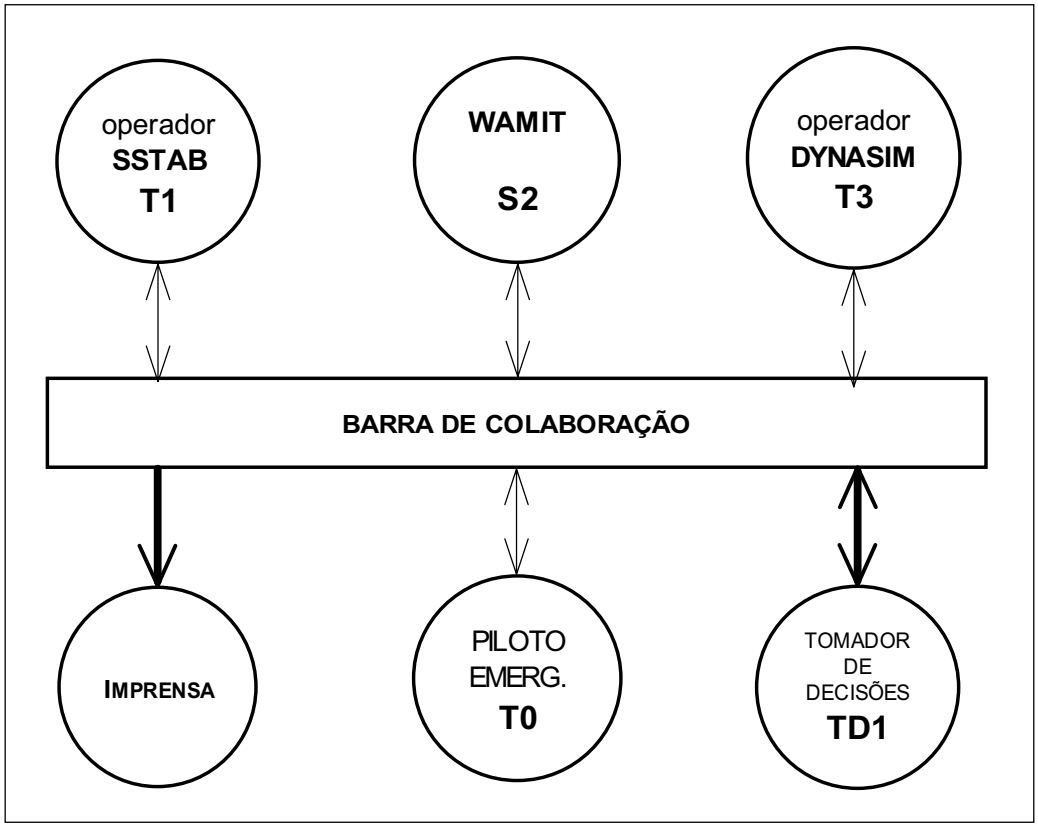

Figura 20 - Um protótipo HLA para o primeiro modelo para a aplicação de desastres

Observando nosso modelo Centrado nas Atividades da Figura 19, podemos ver que, como todos os nós estão conectados, todos os participantes podem constituir uma única Federação, que denominamos *simulacao_integrada* conforme a colaboração conceitual das Tabelas 11, 12, 13 e 14.

Associamos então um Federado a cada membro participante da Federação *simulacao_integrada*, isto é, T0, T1, T3 e TD1 (pessoas), S2 (agente de software), ET (grupo) e Imprensa (companhia). Cada código de Federado é um programa Java construído com base nas regras de fluxo de trabalho escritas em um programa baseado em lógica. Para aumentar a flexibilidade, o método principal de

cada Federado leva o nome *process_role* (processa_papel), que recebe como parâmetro o papel a ser desempenhado pelo Federado, codificado em um módulo Java separado. Usando esta estratégia, podemos codificar as regras de fluxo de trabalho, associadas com um papel específico, diretamente em um módulo separado dedicado a este papel.

Em termos da parte de rede do nosso modelo Centrado nas Atividades, de fato uma árvore, a implementamos usando listas encadeadas da linguagem Java. Um ponto interessante que deve ser considerado aqui é a implementação dos algoritmos *send* (envia) e *receive* (recebe) usados por cada Federado.

Primeiramente lembramos que a RTI escolhida da HLA, XRTI, possui como seu modelo de comunicação uma topologia cliente-servidor, sem comunicação ponto-a-ponto e simplesmente enviando mensagens através de um canal da barra de colaboração sem receptor específico. Então, de modo a implementar algumas regras de coordenação especificamente relacionadas a receptores particulares, tivemos que pensar em uma maneira de indicar os receptores nas mensagens.

Associado a cada mensagem, a XRTI tem um vetor de bytes denominado *userSuppliedTag*. Nosso protótipo conta com sete participantes (seis nós folhas e um nó grupo, ET) e então utilizamos um byte deste vetor para representar cada um dos receptores de mensagem, com cada posição do vetor de bytes correspondendo a um participante.

Agora explicamos cada um desses algoritmos:

- *Send* (Envia): este algoritmo pode ser implementado de duas formas básicas. Na primeira, implementamos diretamente o algoritmo conceitual descrito no Capítulo 3, enviando a mesma mensagem básica para cada receptor diferente associado a ela na tabela de atributos de mensagens, usando o *userSuppliedTag* para representar para qual receptor a mensagem está sendo enviada (além de também registrar nela a *id_mensagem*). Isto permite flexibilidade não apenas em termos de um *módulo de pós-processamento* diferente associado a cada par (*id_mensagem*, *receptor*) presente na tabela de mensagens, mas também em termos de *módulos de pré-processamento*, já que seremos capazes de executar diferentes módulos associados com cada receptor. Isto parece razoável para

nossa aplicação com poucos nós. Se considerarmos que é mais interessante perder alguma flexibilidade e melhorar a performance da rede, podemos adotar outra estratégia. Antes de enviar a mensagem através do canal, podemos processar cada linha da tabela de atributos de mensagens associada com o par particular (*remetente*, *id_mensagem*) e representar todos os receptores encontrados no vetor de bytes *userSuppliedTag*. Então enviamos através do canal apenas uma única mensagem associada com esta *id_mensagem*, com todos os seus receptores indicados no *userSuppliedTag*. Desse modo, ganhamos em performance de rede, diminuindo sua carga, e perdemos alguma flexibilidade por não sermos capazes de executar os módulos de pré-processamento prescritos para cada mensagem associada a cada receptor (observe que não perdemos toda a flexibilidade, visto que continuamos capazes de executar os pós-processamentos do lado do receptor). Provavelmente mais interessante do que usar as duas estratégias já descritas seria implementar uma combinação das duas: antes de enviar a mensagem através do canal, mais uma vez processamos cada linha da tabela de atributos de mensagens, mas desta vez, em vez de apenas agrupar mensagens usando o par (*remetente*, *id_mensagem*), também verificamos se elas usam o mesmo *módulo de pré-processamento*, caso em que combinamos as mensagens com os mesmos atributos *remetente*, *id_mensagem* e *pré-processamento* em uma única usando o *userSuppliedTag*. Utilizando esta estratégia, mantemos toda a nossa flexibilidade original, reduzindo a carga de rede ao mínimo possível.

- *Receive* (Recebe): simplesmente verifica por meio do vetor de bytes *userSuppliedTag* se a mensagem é destinada ao Federado presente e usa o par (*id_mensagem*, *receptor*) para acessar a tabela de atributos de mensagens e descobrir qual o *módulo de pós-processamento* a ser executado. Um caso interessante é aquele em que o receptor representado no *userSuppliedTag* é um nó grupo (ET no nosso protótipo). Relembrando o Capítulo 3, já sabemos que, para nós grupos, existe um atributo indicando qual nó folha (na nossa

implementação, um Federado) irá executar o algoritmo *receive* e o *módulo de pós-processamento* associado com o par (*id_mensagem*, *nó_grupo*). Este *módulo de pós-processamento* necessariamente determinará que receptores (nós folhas) irão receber a mensagem.

Algumas telas desta sessão colaborativa de simulação integrada usando a primeira versão do nosso protótipo HLA podem ser vistas no Apêndice C.

Além de uma aplicação de passagem de mensagens (restrição imposta pelos simuladores reais disponíveis), pensamos em formas de aprimorar a colaboração. Uma característica simples que melhorou bastante a informação de percepção (*awareness*) foi a adição de uma ferramenta de captura de vídeo em cada um dos dois simuladores interativos, transmitindo quadros periodicamente através da barra para os outros participantes. Desse modo, eles não apenas recebem mensagens sobre o início e o fim de uma simulação em particular, como também recebem quadros intermediários com a evolução da simulação.

Finalizamos esta subseção destacando dois pontos da implementação da XRTI:

- *Management Object Model* (MOM): cada conjunto de tabelas de modelos de objetos possui tabelas de estruturas de objetos e/ou classes de interação que descrevem as relações de herança entre as classes. Para cada classe, existe um sinal indicando se um Federado pode publicar, subscrever, publicar e subscrever, ou nem publicar nem subscrever instâncias da classe. Existem também tabelas descrevendo atributos de objetos e parâmetros de interação, tais como nome, tipo de dado, tipo de atualização, condição de atualização, capacidade de transferência de posse, capacidade de publicação e subscrição, dimensões disponíveis, transporte e tipos de ordem de cada atributo de cada classe de objeto.
- Método *mergeFDD* : FDD significa *Federation Object Model Document Data*. A XRTI implementa um método chamado *mergeFDD* que permite a Federados adicionarem conteúdos de outros FDDs ao FOM corrente durante a execução de uma Federação. Isto estimula o uso de modelos leves e componíveis. Então, sob a XRTI, Federados podem especificar um FDD contendo

somente o MOM (um componente obrigatório de cada FOM) como o FDD inicial e então fundir outros FDDs menores com o FOM à medida que for sendo necessário.

5.1.1.2. Um Protótipo InfoGrid para o Primeiro Modelo para a Aplicação Colaborativa de Gestão de Desastres

Apresentamos agora na Tabela 16 um protótipo InfoGrid para o nosso primeiro modelo para a aplicação colaborativa de gestão de desastres.

t_0: T0 cria um projeto
T0 autoriza T1 e T3 a acessarem o projeto
T0 envia notificações persistentes
T0 escreve dados no projeto (ex.: modelo da P-34)
t_1: T1 entra
T1 recebe notificações passadas e persistentes //aresta a1 assíncrona
T1 acessa remotamente o modelo no projeto e executa o SSTAB
T1 envia mensagem a todos notificando o início da simulação do SSTAB
t_2: T3 recebe notificação por *e-mail*
t_3: T1 escreve os resultados do SSTAB no projeto
T1 envia mensagem a todos notificando o fim da simulação do SSTAB
T1 ajusta parâmetros e ativa S2
$t_{3+\Delta}$: todos os participantes recebem notificação da ativação da simulação S2 (envio automático pelo sistema)
t_4: T3 entra
T3 recebe notificações
t_5: todos os participantes recebem notificação do fim da simulação S2 (envio automático pelo sistema)
$t_{5+\Delta}$: T3 acessa o resultado de S2 e inicia o DYNASIM
T3 envia mensagem a todos notificando o início da simulação do DYNASIM
t_6: T3 escreve os resultados do DYNASIM no projeto
T3 envia mensagem a todos notificando o fim da simulação do DYNASIM
t_7: T0 analisa os resultados do DYNASIM
T0 envia mensagem a TD solicitando sua aprovação
t_8: TD recebe notificação por *e-mail*
TD entra
TD analisa os resultados
TD notifica T0 a respeito de sua aprovação
t_9: T0 extrai os resultados a serem publicados pela Imprensa
T0 envia os resultados a serem publicados pela Imprensa

Tabela 16 - Um protótipo InfoGrid para o primeiro modelo para a aplicação de desastres

O protótipo InfoGrid é descrito em termos de eventos seqüenciais no tempo, com o primeiro deles responsável pela criação do projeto que conterá nosso modelo. Cada evento no tempo é iniciado por um termo que indica a hora em que

o evento ocorre: t_0, t_1, t_2, t_3, etc. Também, quando outro evento ocorre imediatamente após a ocorrência de um evento anterior, denotamos isto adicionando um Δ ao subscrito, tal como $t_{3+\Delta}$ e $t_{5+\Delta}$, que ocorrem, respectivamente, imediatamente após t_3 e t_5 em nosso exemplo. Para cada evento no tempo, definimos uma seqüência de comandos que são executados. O conjunto completo de eventos e comandos define a aplicação colaborativa.

Um aspecto interessante do protótipo InfoGrid é que ele permite tanto notificações persistentes quanto voláteis. Também todas as notificações podem ser enviadas por *e-mail*, o que pode ser o caso para as mensagens não críticas.

5.1.2. Um Segundo Modelo para a Aplicação Colaborativa de Gestão de Desastres

Derivamos nosso primeiro modelo para a aplicação de gestão de desastres considerando os simuladores reais disponíveis hoje na Petrobras. Imaginemos agora que pudéssemos configurar nosso modelo idealmente, sem nenhuma restrição de implementação. Uma arquitetura conceitual possível seria uma com todos os participantes separados dos motores de simulação e sendo capazes de ativá-los remotamente de outros locais e de mostrar as saídas usando um programa local de interface com o usuário. Teríamos então o grupo Força Tarefa novamente co-localizado em uma sala especial com todas as facilidades requeridas, mas desta vez ativando remotamente os simuladores disponíveis em outros locais da companhia. Novamente, visto que em nosso modelo FT é a única equipe dentro de ET, não a estamos representando explicitamente (se ela possuísse políticas de entrada e saída particulares, deveria também ser um nó do nosso modelo). O Tomador de Decisões permanece desempenhando seu papel de tomar a decisão remotamente situado em relação à Força Tarefa.

O modelo derivado do nosso metamodelo Centrado nas Atividades e correspondente a este novo cenário é o mostrado na Figura 21.

Podemos ver que não foi difícil acomodar o modelo anterior a esta nova abordagem. Tudo o que tivemos que fazer foi definir novos nós correspondentes a cada novo motor de simulação (S1 e S3) e definir as interações entre eles e com os outros nós. É interessante observar que neste segundo modelo temos tantos agentes (S1, S2 e S3) quanto pessoas (T0, T1 e T3).

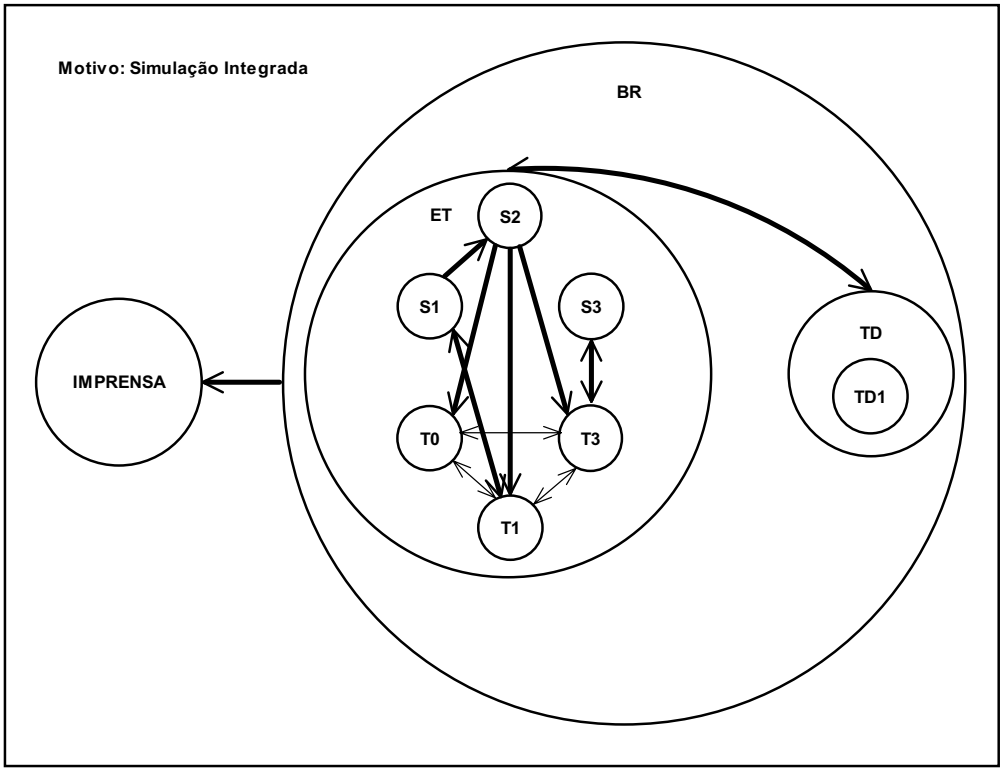

Figura 21 - Um segundo modelo para a aplicação colaborativa de gestão de desastres, focando na simulação integrada

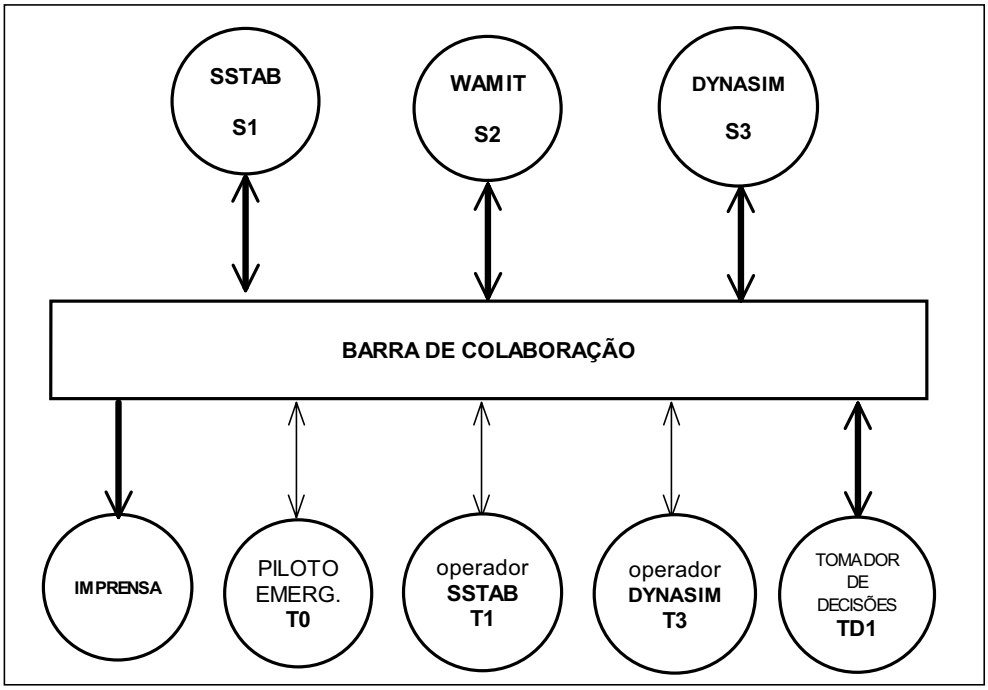

Figura 22 - Um protótipo HLA para o segundo modelo para a aplicação de desastres

5.1.2.1. Um Protótipo HLA para o Segundo Modelo para a Aplicação Colaborativa de Gestão de Desastres

Apresentamos agora o protótipo HLA implementando o segundo modelo para a aplicação de gestão de desastres (Figura 22).

O que difere neste segundo protótipo HLA quando comparado com o primeiro é que agora temos dois novos Federados, cada um correspondendo a um dos novos motores de simulação: SSTAB e DYNASIM. Permanecemos com os operadores do SSTAB e do DYNASIM ainda se comunicando com os outros participantes e agora também com seus respectivos motores de simulação. Os nós remanescentes permanecem com seus comportamentos anteriores.

5.2. Visualização CAD em Ambientes Virtuais

De modo a validar a generalidade do metamodelo, delineamos agora um modelo para outra aplicação, isto é, a visualização CAD em ambientes virtuais.

Novamente consideramos um estudo de caso da companhia de óleo e gás brasileira Petrobras. Suponhamos que uma Unidade de Exploração & Produção (E&P) está desenvolvendo um projeto e que, durante uma fase específica, ela necessite atualizar e validar um modelo CAD em outra Unidade do E&P e pelo Gerente Geral do Departamento de E&P. O fluxo de trabalho típico para esta aplicação seria como se segue:

- O técnico da Unidade 1 do E&P envia o modelo CAD para a Unidade 2 do E&P de modo que ele possa ser atualizado considerando os aspectos particulares desta última. Uma primeira questão que surge é que a Unidade 1 do E&P usa um software de CAD (que chamaremos de A) enquanto que a Unidade 2 do E&P usa tanto este software de CAD quanto outro de outro fabricante (que chamaremos de B), dependendo da idade do modelo CAD (eles usam o software A para os modelos antigos e o software B para os novos). Isto significa que a aplicação pode ter que converter modelos do software A para o software B.
- Depois de atualizar o modelo CAD, o técnico da Unidade 2 do E&P (usando o software A ou o B) então retorna o modelo para o técnico

da Unidade 1 do E&P. Também podemos ter aqui a necessidade de converter modelos, dependendo do software que está sendo usado de cada lado.

- O técnico da Unidade 1 do E&P agora envia o modelo atualizado para ser validado pelo Gerente Geral, que está trabalhando em um ambiente de Realidade Virtual. Isto significa que mais uma vez necessitamos de conversão de modelos, desta vez de um ambiente CAD para um ambiente de Realidade Virtual.
- Finalmente o Gerente Geral envia de volta uma mensagem para o técnico da Unidade 1 do E&P informando se ele aprovou ou não o modelo atualizado.

Em uma condição de emergência, o sistema de visualização CAD deveria ser usado por especialistas para discutir sobre os resultados da simulação antes de mostrá-los para os tomadores de decisões.

5.2.1. Um Modelo para a Visualização CAD em Ambientes Virtuais

Iremos derivar agora um modelo Centrado nas Atividades completo para esta nova aplicação.

Primeiramente, definimos o componente de rede como mostrado na Figura 23. Nela podemos ver três nós principais: o nó E&P 1 liderando a aplicação colaborativa e, remotamente localizados em relação a ele, o nó E&P 2 e o nó Tomador de Decisões TD.

Dentro do nó E&P 1, temos um técnico do E&P (T0) executando o software de CAD A.

Dentro do nó E&P 2, temos dois técnicos do E&P, T1 executando o software A para modelos CAD antigos, e T2 executando o software B para modelos CAD novos.

Finalmente, dentro do nó TD, temos o Gerente Geral TD1 (Tomador de Decisões 1) usando um software de Realidade Virtual em um ambiente virtual.

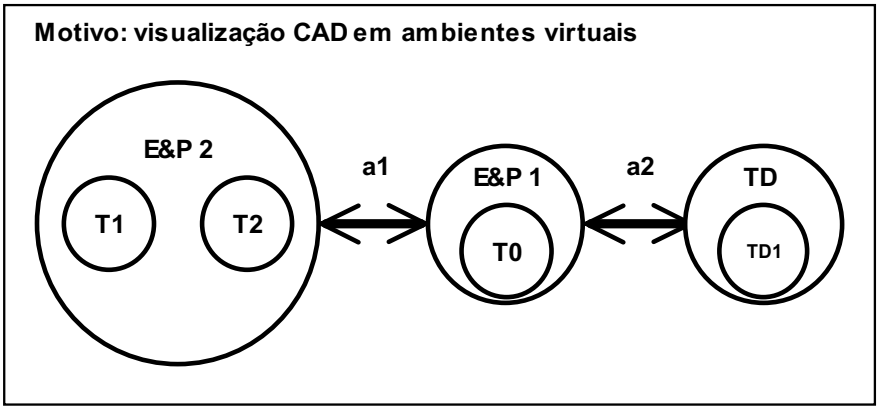

Figura 23 - Um modelo para a visualização CAD em ambientes virtuais

Tendo definido o componente de rede do nosso modelo, iremos agora apresentar as regras de papéis e a tabela de atributos de mensagens que definem o fluxo de trabalho da aplicação.

As regras de papéis para o técnico 0 do E&P, a Unidade 2 do E&P, o técnico 1 do E&P e o técnico 2 do E&P são mostradas na Tabela 17. As regras que definem o papel do técnico 0 do E&P são semelhantes às dos exemplos apresentados anteriormente. O papel de grupo da Unidade 2 do E&P tem uma regra *on-arrive* que é disparada quando recebe a mensagem 1, momento em que é informado no console que o módulo de pós-processamento *pos_1_EP2* está determinando e re-roteando a mensagem para os técnicos da Unidade 2 do E&P. Neste exemplo, não sabemos de antemão qual dos dois técnicos da Unidade 2 do E&P irá interagir com o técnico da Unidade 1 do E&P. Apesar disso, nosso modelo é ainda capaz de acomodar esta nova situação, com o módulo de pós-processamento *pos_1_EP2* sendo responsável por determinar qual dos dois técnicos da Unidade 2 do E&P irá participar da sessão colaborativa, dependendo da data do modelo CAD.

As regras de papel para o tomador de decisões estão mostradas na Tabela 18. O aspecto a destacar deste papel é que temos uma regra *on-arrive* que dispara uma seqüência de comandos ao chegar a mensagem 3. Esta seqüência primeiro mostra uma mensagem que requisita a validação do modelo, então apresenta o modelo CAD atualizado e espera até que o tomador de decisões valide ou não o modelo apresentado.

```
collaboration visualizacao_CAD_RV
{
 collaboration_bus {
            channel(remoto).
                    }

 role tecnico_0_Unidade_1_E&P //técnico responsável pela coordenação das atividades
 {
  on-init(visualizacao_CAD_RV) :-
   send(remoto, tab_mensagem(source(self), 1, dummy)). //envia "Técnico da Unidade 2, por favor atualize o modelo
                                                        CAD."

  on-arrive(local, tab_mensagem(dummy, 2, source(self))) :-
   display(tab_mensagem(dummy, 2, source(self))), //mostra "Modelo CAD foi atualizado."
   display o modelo CAD atualizado, //mostra o modelo CAD atualizado
   read botao_modelo_CAD_visto, //lê botao_modelo_CAD_visto do console
   send(remoto, tab_mensagem(source(self), 3, dummy)). //envia "Tomador de Decisões, por favor valide o modelo."

  on-arrive(remoto, tab_mensagem(dummy, 4, source(self))) :-
   display(tab_mensagem(dummy, 4, source(self))). //mostra a decisão tomada
 }

 role Unidade_2_E&P //unidade responsável pelo re-roteamento do modelo CAD para o técnico 1 ou 2
 {
  on-arrive(local, tab_mensagem(dummy, 1, source(self))) :-
   display(tab_mensagem(dummy, 1, source(self))). //pos_1_EP2 determina qual técnico irá receber a mensagem 1
 }

 role tecnico_1_Unidade_2_E&P //técnico responsável pela atualização do modelo CAD usando o software A
 {
  on-arrive(local, tab_mensagem(dummy, 1, source(self))) :-
   display(tab_mensagem(dummy, 1, source(self))), //mostra "Técnico da Unidade 2, por favor atualize o modelo CAD."
   read botao_modelo_CAD_atualizado, //lê botao_modelo_CAD_atualizado do console
   send(remoto, tab_mensagem(source(self), 2, dummy)). //envia "Modelo CAD foi atualizado."
 }
 role tecnico_2_Unidade_2_E&P //técnico responsável pela atualização do modelo CAD usando o software B
 {
  on-arrive(local, tab_mensagem(dummy, 1, source(self))) :-
   display(tab_mensagem(dummy, 1, source(self))), //mostra "Técnico da Unidade 2, por favor atualize o modelo CAD."
   read botao_modelo_CAD_atualizado, //lê botao_modelo_CAD_atualizado do console
   send(remoto, tab_mensagem(source(self), 2, dummy)). //envia "Modelo CAD foi atualizado."
 }
```

Tabela 17 - Regras de papéis para os técnicos do E&P 0, 1 e 2, e para a Unidade 2 do E&P

É importante notar que, em ambas as Tabelas, as mensagens 1, 2 e 3 são enviadas com o modelo anexado em um arquivo.

```
role tomador_decisoes //gerente responsável pela tomada de decisão
 {
  on-arrive(remoto, tab_mensagem(dummy, 3, source(self))) :-
    display(tab_mensagem(dummy, 3, source(self))), //mostra "Tomador de Decisões, por favor valide o modelo."
    display o modelo CAD atualizado, //mostra o modelo CAD atualizado
    read modelo_validado, //lê modelo_validado (OK ou nao_OK) do console
    assert modelo_validado. //grava modelo_validado no banco de conhecimentos

  when(modelo_validado, nao_OK) :-
    write mensagem 4 com "O modelo CAD não foi validado.",
    send(remoto, tab_mensagem(source(self), 4, dummy)). //envia mensagem 4

  when(modelo_validado, OK) :-
    write mensagem 4 com "O modelo CAD foi validado.",
    send(remoto, tab_mensagem(source(self), 4, dummy)). //envia mensagem 4
 }
}
```

Tabela 18 - Regras de papel para o tomador de decisões

Finalmente na Tabela 19 mostramos a tabela de atributos de mensagens para este modelo. Primeiramente notamos que, para fins de simplificação, estamos substituindo Unidade 2 do E&P por EP2. Em segundo lugar, apenas a *id_mensagem* juntamente com o *receptor* não foram suficientes para designar os nomes dos módulos de pré e pós-processamento – tivemos que levar em conta também o *remetente*. Isto ocorre porque, embora de fato não estejam interagindo simultaneamente durante a mesma sessão colaborativa, T1 e T2 estão representados na tabela de atributos de mensagens como remetentes de mensagens com a mesma *id_mensagem* (2). Simplesmente considerando também o *remetente* ao construir os nomes dos módulos de pré e pós-processamento, nosso metamodelo provou ser capaz de lidar com esta nova situação.

remetente	id_messagem	receptor	aresta	pré-processamento	pós-processamento
T0	1	EP2	a1	Pre_1_EP2	Pos_1_EP2
T0	3	TD1	a2	Pre_3_TD1	Pos_3_TD1
T1	2	T0	a1	Pre1_2_T0	Pos1_2_T0
T2	2	T0	a1	Pre2_2_T0	Pos2_2_T0
TD1	4	T0	a2	Pre_4_T0	Pos_4_T0

Tabela 19 - A tabela de atributos de mensagens para a aplicação de visualização CAD

O último aspecto a ser considerado sobre esta aplicação é a seleção apropriada dos módulos de processamento (pré e pós) que executarão a conversão

dos modelos de engenharia. Temos que considerar cada mensagem em particular para selecionar o módulo de processamento apropriado para cada uma delas:

- Mensagem 1: é enviada de T0 para a Unidade 2 do E&P. Como T0 não sabe de antemão qual o software que a Unidade 2 do E&P utilizará, ele simplesmente envia a mensagem com o modelo através do canal, sem nenhuma conversão, e deixa a Unidade 2 do E&P determinar se irá ocorrer alguma conversão. Do lado da Unidade 2 do E&P, o módulo de pós-processamento *Pos_1_EP2* determinará para qual dos dois técnicos da Unidade 2 do E&P a mensagem será encaminhada, com base na data do modelo CAD (isto determina qual software será usado).
- Mensagem 2: esta mensagem é enviada por um dos dois técnicos da Unidade 2 do E&P para T0. Se é T1 quem está participando da sessão colaborativa (usando o software A), não haverá nenhuma conversão ao se enviar a mensagem 2. Se é T2 quem está participando da sessão colaborativa (usando o software B), então neste caso o módulo de pré-processamento *Pre2_2_T0* deve converter o modelo do formato de B para o formato de A, garantindo que ele chegue do lado de T0 já devidamente convertido. Isto é importante porque, usando *dummy* como o remetente nas regras *on-arrive*, não seremos capazes de determinar precisamente neste caso qual linha da tabela corresponde à mensagem que está sendo recebida e portanto não seremos capazes de usar os módulos de pós-processamento. Outra alternativa seria usar *source*(*sender*) em vez de *dummy* nas regras *on-arrive* (a informação do remetente teria que ser incluída na mensagem), de modo que nos tornássemos capazes de determinar exatamente a linha da tabela de atributos de mensagens correspondente à mensagem que está chegando – neste caso, poderíamos usar tanto o módulo de pré quanto o módulo de pós-processamento para fazer a conversão dos formatos.
- Mensagem 3: esta é enviada de T0 para TD1. Em princípio, poderíamos usar tanto o módulo de pré quanto o módulo de pós-processamento para converter o modelo CAD para o modelo de ambiente de Realidade Virtual. Contudo, parece mais natural

permitir que o Tomador de Decisões use seu módulo de pós-processamento para fazer esta conversão, visto que ele é quem mais conhece as condições e recursos do ambiente de Realidade Virtual.

O objetivo deste segundo exemplo foi validar a generalidade do metamodelo, aplicando-o em outro cenário, não relacionado a emergências. Este novo cenário traz no mínimo duas situações interessantes que nosso metamodelo consegue abarcar. A primeira é o fato de que temos um nó grupo (Unidade 2 do E&P) contendo dois nós folhas que não se comunicam entre si – a única comunicação com eles vem através do nó pai. A segunda é o fato de que temos dois participantes pré-definidos no componente de rede e nas regras de papéis – Técnicos 1 e 2 da Unidade 2 do E&P – que não participam da mesma sessão colaborativa: existe uma seleção em tempo de execução baseada em um atributo de parte da mensagem (a data do modelo CAD transmitido como um arquivo anexo à mensagem). Isto causa o surgimento de duas linhas na tabela de atributos de mensagens com a mesma *id_mensagem* com dois remetentes diferentes, o que aparentemente não determina qual das linhas está relacionada com a mensagem que está sendo recebida. Entretanto, simplesmente usando o argumento opcional *remetente* da regra *on-arrive* com a informação de remetente incluída na mensagem, podemos lidar com esta situação.

6
Conclusões e Trabalhos Futuros

Neste trabalho, propusemos um metamodelo de multi-perspectiva – o metamodelo Centrado nas Atividades – que combina as perspectivas Centrada nos Lugares e Centrada nas Pessoas no grau desejado pelo projetista. O metamodelo emprega não uma abordagem dirigida por tecnologia, mas uma abordagem centrada no humano – ou melhor, como ele também enfatiza questões no nível de grupos e de organizações, uma abordagem centrada no social (Neale et al., 2004) ou uma combinação de ambas (Gutwin & Greenberg, 1998). O metamodelo permite experimentar e derivar modelos em duas dimensões ortogonais: com relação ao número de níveis de grupos e quebrando um nível de grupo em particular. Associando pré e pós-processamentos a cada um desses níveis, podemos acomodar políticas e regras de privacidade de organizações, permitindo inclusive trabalhos inter-organizacionais.

No nosso metamodelo, identificamos elementos associados com cada um dos três componentes do modelo de colaboração 3C: o componente de comunicação é representado pelas arestas; o componente de coordenação é representado pelos elementos de especialização de arestas, pelas regras de papéis e pela tabela de atributos de mensagens; e, finalmente, o componente de cooperação é formado pelos diferentes níveis de abstração, que representam toda a atividade que está sendo realizada, incluindo todas as aplicações executadas nos nós folhas e os artefatos produzidos ou manipulados. Os nós folhas podem ser, indistintamente, pessoas ou agentes de software.

O metamodelo permite flexibilidade em várias dimensões. Separar as características de abstração de alto nível das características de implementação de baixo nível permite ao projetista e ao desenvolvedor da aplicação se concentrarem no seu domínio particular de conhecimento. O uso de arestas permite trabalho síncrono e assíncrono. Separar o programa computacional e o programa de coordenação (as regras de papéis) permite aos programadores se concentrarem nas questões de coordenação com abstração de alto nível. As regras de papéis, escritas

em uma linguagem baseada em lógica, especificam políticas de coordenação e regras de participantes, acomodando tanto trabalhos fortemente quanto fracamente acoplados. Com os participantes divididos em um número de papéis fixo, o número de participantes que assumem estes papéis pode ser fluido e grande.

O metamodelo também provê um conjunto de outras características de flexibilidade, mais diretamente relacionado a características particulares de seus elementos. Ele é customizável no sentido de que permite associar processamentos de pré e pós-comunicação a cada mensagem enviada. Permite ainda mudanças paramétricas em tempo de execução, tais como a alteração dos nomes dos módulos de processamento pré e pós-comunicação na tabela de atributos de mensagens, ou até mesmo a possibilidade de mudança dos códigos dos módulos de processamento pré e pós-comunicação antes que eles tenham sido carregados durante uma sessão colaborativa. Os papéis são codificados em módulos separados do programa computacional, em consonância com o princípio de componentes de software colaborativos re-usáveis. Finalmente, ele também provê alguma extensibilidade, permitindo adicionar novas linhas na tabela de atributos de mensagens que associem uma mensagem específica a novos receptores, assim como carregar novos módulos de pós-processamento em tempo de execução.

O metamodelo parece ter elementos suficientemente genéricos para acomodar uma gama de aplicações, como também para ser especializável para um determinado domínio. Também concluímos que o metamodelo oferece condições para desenvolver um espaço de trabalho colaborativo: ele permite o compartilhamento de material em um grupo específico e provê comportamentos adequados e informações de percepção (*awareness*) para auxiliar a cooperação.

O metamodelo permite armazenar todos os elementos da sessão colaborativa, tais como as aplicações que estão sendo executadas nos nós folhas, os artefatos sendo produzidos ou manipulados, e o nome do executor da aplicação, assim como sua data e hora. Isto pode ser utilizado tanto no nível conceitual, como uma base de conhecimentos para auditoria ou reconfiguração do modelo em sessões futuras, como também no nível do mundo real, auxiliando especialistas quando da elaboração de um relatório de investigação sobre o acidente, ou ainda com objetivos de treinamento.

Outra característica importante deste trabalho é que ele foi motivado e conduzido em configurações do mundo real, isto é, em um cenário de desastre em

estruturas *offshore* de óleo e gás. Isto parece contribuir para o campo de CSCW, visto que um exame de estudos de avaliação de CSCW concluiu que menos da metade deles foram conduzidos em configurações do mundo real (Pinelle, 2000). Realizamos uma pesquisa sobre os principais sistemas comerciais de gestão de emergência disponíveis, concluindo que eles ainda carecem de uma integração plena e adequada de simuladores com sistemas de visualização de alta performance. A decisão de se usar a HLA (*High Level Architecture*) como uma plataforma para o nosso protótipo favorece a inclusão de simulações mais complexas, até mesmo aquelas com requisitos de visualização de alta performance.

Acreditamos que, considerando a generalidade do metamodelo, podemos suportar outros domínios da aplicação, mas apenas aqueles relacionados à área de óleo e gás, em particular os de cenários de emergência, foram exaustivamente testados.

Depois de derivar, a partir do nosso metamodelo, um modelo adequado para o cenário de desastre, implementamos nosso primeiro protótipo, como uma prova de conceito do nosso metamodelo, usando uma infra-estrutura em tempo de execução HLA, isto é, a XRTI. Ela contempla todos os requisitos de flexibilidade demandados, como também tem código fonte aberto e é distribuível gratuitamente. Ela provou ser particularmente adequada para acomodar um número de participantes variável (no nosso modelo associados a papéis), visto que a HLA permite a Federados livremente se associarem ou se desligarem de Federações, a qualquer hora durante a sessão colaborativa.

Também discutimos como o protótipo poderia ser implementado usando uma outra plataforma de implementação, o InfoGrid. Ainda para a aplicação de gestão de desastres, apresentamos um segundo modelo e seu protótipo, mostrando que podemos derivar diferentes modelos para a mesma aplicação. Finalmente, para validar a generalidade do metamodelo, também delineamos um modelo para uma outra aplicação, isto é, a visualização CAD em ambientes virtuais.

Em termos de contribuição para a Petrobras, este trabalho proporcionou:

- Um melhor entendimento dos requisitos de ICT da indústria de óleo e gás para apoiar a gestão de desastres envolvendo equipes de especialistas distribuídos.

- A criação de um ambiente de tomada de decisões, que integra simuladores, centros de visualização e vários especialistas para situações de emergência.
- Resultados da avaliação dos protótipos em configurações industriais reais usando especialistas envolvidos em situações de gestão de desastres reais.

6.1. Trabalhos Futuros

Existe uma série interessante de trabalhos futuros relacionados a este que poderiam ser desenvolvidos em várias dimensões.

Primeiramente, consideremos nosso metamodelo. Embora provendo o grau de flexibilidade mencionado acima, ainda existe muito trabalho a ser feito de modo a tornar nosso metamodelo uma arquitetura colaborativa plenamente flexível e evolutiva:

- Em primeiro lugar, seria interessante incluir um mecanismo de associação tardia como o registrador de Chung & Dewan (2004), capaz de reproduzir mensagens de saída gravadas, ou um método de cópia de imagem, já que é comum que pessoas se associem e se desliguem de sessões colaborativas.
- Em segundo lugar, particularmente na situação de cenário de emergência sendo considerada, seria muito importante incluir um sistema de Recomendação de Expertise, como o proposto por McDonald & Ackerman (2000), visto que, em uma situação de crise, é fundamental localizar a expertise necessária para resolver o problema no menor tempo possível.
- Em terceiro lugar, novamente relacionado diretamente ao caso particular do cenário de emergência, parece que seria interessante incluir algum tipo de redundância em uma ou mais das dimensões consideradas por Tjora (2004): redundância de funções, redundância de comunicação, redundância de competência ou redundância tecnológica.

- Em quarto lugar, deveria ser investigado o quanto o número de linhas da tabela de atributos de mensagens pode aumentar, considerando vários aspectos, tais como a quantidade de nós e arestas, a quantidade de mensagens diferentes, a quantidade de variações da mesma mensagem e a quantidade de nós grupos (que re-roteiam as mensagens).
- Finalmente, deveria ser investigado como promover nosso metamodelo de uma categoria customizável para uma categoria adaptável (Dourish, 1998), subindo da capacidade de ajustar controles paramétricos para a capacidade de reconfigurar seu comportamento de acordo com padrões de uso imediatos. Podemos pensar em implementar isto em duas fases. Em uma primeira fase, faríamos a monitoração das atividades atuais e das interações entre os usuários para derivar um diagnóstico, que serviria como a base para possivelmente ajustar o modelo para o padrão encontrado e então usar o modelo ajustado nas sessões colaborativas subseqüentes. Em uma segunda fase, usaríamos algum tipo de mecanismo de aprendizagem para fazer esses ajustes em tempo de execução.

A segunda dimensão a ser considerada é a de ferramentas para auxiliar a configuração e a implementação dos modelos:

- Já que identificamos a importância de sermos capazes de rapidamente reconfigurar o modelo, especialmente em situações de emergência, seria muito interessante ter algum tipo de editor de diagramas para configurar e reconfigurar modelos, assim como para automaticamente atualizar o banco de dados contendo os modelos de rede.
- Também parece ser muito útil ter mapeadores automáticos de nossas regras de papéis para as linguagens de programação que estiverem sendo utilizadas na implementação.

Uma terceira dimensão a ser considerada é a questão da validação do metamodelo. Validação é sempre um problema muito complicado quando se desenvolve um grande sistema. Sugerimos validar o metamodelo de acordo com diferentes aspectos:

- Primeiramente, para um cenário em particular e usando uma representação semi-formal, constituída pelo modelo de rede do nosso metamodelo, os usuários deveriam validar o modelo em particular que está sendo utilizado, verificando, por exemplo, se todos os grupos importantes estão representados, se os grupos estão dispostos nas suas posições corretas, se não existem arestas faltando e se a representação de arestas locais e remotas está correta.
- Em segundo lugar, os usuários deveriam analisar todas as regras de papéis para ver se realmente elas refletem o que está ocorrendo no mundo real e se não há nenhuma regra faltando.
- Em terceiro lugar, os usuários deveriam verificar se todos os módulos de pré e pós-processamento importantes estão representados na tabela de atributos de mensagens. Também, os usuários deveriam verificar se as mensagens que estão sendo enviadas para nós grupos são as apropriadas.
- Em uma segunda dimensão de validação, poderíamos implementar um protótipo para uma aplicação em particular usando diferentes plataformas de implementação para validar o metamodelo, como fizemos com a aplicação do cenário de emergência usando a HLA e o InfoGrid.
- Em uma terceira dimensão de validação, poderíamos experimentar diferentes modelos da mesma aplicação, para verificar quanto o metamodelo é flexível, de modo semelhante ao que foi apresentado nas Subseções 5.1.1 and 5.1.2.
- Finalmente, em uma quarta dimensão de validação, para validar a generalidade do metamodelo, deveríamos derivar modelos para diferentes tipos de aplicação. A aplicação de visualização CAD (Seção 5.2) é um exemplo deste tipo de validação.

Uma quarta dimensão a ser considerada em termos de trabalhos futuros é a relacionada com o protótipo HLA. Embora já tendo incluído algumas facilidades no nosso primeiro protótipo, como uma ferramenta de captura de vídeo e um tradutor de línguas automático, existem ainda várias novas facilidades que poderiam ser acrescentadas para melhorar o protótipo. Uma delas poderia ser uma facilidade de teleconferência, permitindo comunicação entre os participantes em tempo real.

Uma quinta dimensão consistiria em desenvolver completamente o protótipo InfoGrid, de modo a podermos fazer a comparação dele com o protótipo HLA, identificando quais são as vantagens de cada um deles.

A sexta dimensão está relacionada com o desenvolvimento de novas aplicações para usar e validar nosso metamodelo sob diferentes condições. Uma aplicação que deveria ser mais desenvolvida é a de visualização CAD em ambientes virtuais, descrita no Capítulo 5. Particularmente em uma sala multifuncional ou em um ambiente de realidade virtual, a interação CAD seria crítica. Outra aplicação a ser investigada no futuro próximo é novamente uma relacionada com a área de óleo e gás, isto é, a aplicação de Dimensionamento de Reservatório.

Finalmente, a sétima dimensão de trabalhos futuros é a relacionada com a melhoria do espaço de trabalho colaborativo para a aplicação de gestão de desastres. Isto poderia ser feito em duas direções principais:

- De modo a simular as ações do operador, o sistema deveria trabalhar com duas cenas diferentes, cada uma delas com a possibilidade de escolha entre os pontos-de-vista de primeira e terceira pessoa:
  - A primeira cena seria a cena normal dos simuladores interativos (SSTAB e DYNASIM), com os usuários visualizando os movimentos da unidade no mundo virtual baseados nas condições de estabilidade.
  - A segunda cena virtual seria a da sala de controle da unidade, de modo que os especialistas pudessem simular as reações da unidade baseadas na ativação de botões.
- Além dessas duas cenas virtuais, o sistema deveria ter ao menos uma e possivelmente duas projeções do mundo real:

- A mais importante seria a da imagem da sala de controle da unidade, capturada por uma câmera de vídeo. Com esta, os usuários poderiam observar e auxiliar as ações do operador nessas condições extremas.
- Uma segunda projeção possível seria a da imagem externa da unidade, novamente capturada por uma câmera de vídeo, possivelmente localizada em um navio ou helicóptero, de modo que os usuários pudessem observar a situação real da unidade.

7
Referências Bibliográficas

ARMSTRONG, C. R.; GANNON, D.; GEIST, A.; KEAHEY, K.; KOHN, S.; MCINNES, L.; PARKER, S.; SMOLENSKI, B. Towards a Common Component Architecture for High-Performance Scientific Computing. In: 8TH IEEE International Symposium on High Performance Distributed Computing – HPDC 1999, 1999. **Proceedings of 8th IEEE International Symposium on High Performance Distributed Computing – HPDC 1999**, 1999.

ARSENAULT, L. et al. DIVERSE: a Software Toolkit to Integrate Distributed Simulations with Heterogeneous Virtual Environments. Technical Report TR-01-10, Computer Science, Virginia Tech, 2001.

ATHANASOPOULOS, G.; LIASKOVITIS, P.; PILIOURA, T.; SOTIROPOULOU, A.; TSALGATIDOU, A.; YPODIMATOPOULOS, P. D8: Models and specification primitives for building dependable P2P Systems/Applications. University of Athens, Information Societies Technology (IST) Programme, IST-2001-32708, P2P Architect, jun. 2003, 106 p.

BENFORD, S.; BULLOCK, A.; COOK, N.; HARVEY, P.; INGRAM, R.; LEE, O. K. A spatial model of cooperation for virtual worlds. In: INFORMATIQUE'93: INTERFACE TO REAL & VIRTUAL WORLDS, 1993, University of Nottingham. **Proceedings of INFORMATIQUE'93**, 1993, p. 445-456.

BIERBAUM, A.; JUST, C. Software Tools for Application Development. In: SIGGRAPH'98 CONFERENCE, 19-24 jul. 1998, Orlando, Florida, EUA. **SIGGRAPH'98 Course 14**, 1998, p. 3-1, 3-45.

BIERBAUM, A.; JUST, C.; HARTLING, P.; MEINERT, K.; BAKER, A.; CRUZ-NEIRA, C. VRJuggler: A Virtual Platform for Virtual Reality Application Development. In: IEEE VR2001, 2001, Yokohama. **Proceedings of IEEE VR2001**, 2001.

BOOZ ALLEN HAMILTON. Automated Resource Management System (ARMS), System Requirements Document (SRD). Federal Emergency Management Agency, Contact No.: GS-35F-0306J, FEMA BPA # EMV-2001-BP-0147, Task Order 11, EUA, 2003.

BOS, N.; SHAMI, N. S.; OLSON, J. S.; CHESHIN, A.; NAN, D. N. In-group/Out-group Effects in Distributed Teams: An Experimental Simulation. In: CSCW'04, 6-10 nov. 2004, Chicago, Illinois, EUA. **Proceedings of CSCW'04**, 2004, p. 429-436.

BRIGNULL, H.; IZADI, S.; FITZPATRICK, G.; ROGERS, Y.; RODDEN, T. The Introduction of a Shared Interactive Surface into a Communal Space. In: CSCW'04, 6-10 nov. 2004, Chicago, Illinois, EUA. **Proceedings of CSCW'04**, 2004, p. 49-58.

BUHR, R. J. A.; CASSELMAN, R. S. **Use CASE Maps for Object-Oriented Systems**. Prentice-Hall, 1996.

CAPPS, M.; MCGREGOR, D.; BRUTZMAN, D.; ZYDA, M. Npsnet-v a new beginning for dynamically extensible virtual environments. **IEEE Computer Graphics and Applications**, 2000, p. 12-15.

CHUNG, G.; DEWAN, P. Towards Dynamic Collaboration Architectures. In: CSCW'04, 6-10 nov. 2004, Chicago, Illinois, EUA. **Proceedings of CSCW'04**, 2004, p. 1-10.

COELHO, L. C. G.; JORDANI, C. G.; OLIVEIRA, M. C.; MASETTI, I. Q. Equilibrium, Ballast Control and Free-Surface Effect Computations Using The Sstab System. In: 8TH INTERNATIONAL CONFERENCE ON STABILITY OF SHIPS AND OCEAN VEHICLES – STAB, 2003. **Proceedings of the 8th International Conference on Stability of Ships and Ocean Vehicles - Stab**, 2003, p. 377-388.

COELHO, L. C. G.; NISHIMOTO, K.; MASETTI, I. Q. Dynamic Simulation of Anchoring Systems Using Computer Graphics. In: 20TH INTERNATIONAL CONFERENCE ON OFFSHORE MECHANICS & ARTIC ENGINEERING – OMAE 2001, 3-8 jun. 2001, Rio de Janeiro, RJ, Brasil. **Proceedings of the 20th International Conference on Mechanics & Artic Engineering – OMAE**, 2001.

COHEN, A. L.; CASH, D.; MULLER, M. J. Designing to Support Adversarial Collaboration. In: CSCW'00, 2-6 dez. 2000, Philadelphia, PA, EUA. **Proceedings of CSCW'00**, 2000, p. 31-39.

COMMON COMPONENT ARCHITECTURE FORUM, THE. Disponível em: <http://www.cca-forum.org>. Acesso em: 25 fev. 2006.

CORTÉS, M.; MISHRA, P. DCWPL: A Programming Language For Describing Collaborative Work. In: CSCW'96, 1996, Cambridge, MA, EUA. **Proceedings of CSCW'96**, 1996, p. 21-29.

COSTA, T. V. M. **Modelos de aprendizado organizacional de Chris Argyris: Uma investigação em segurança operacional e conservação do meio ambiente (SCA) com base em acidentes da indústria do petróleo**. 2004. 159 p. Tese de Doutorado – Coordenação dos Programas de Pós-Graduação de Engenharia, Engenharia de Produção, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2004.

COVER – COVISE VIRTUAL ENVIRONMENT. Disponível em: <http://www.hlrs.de/organization/vis/covise/features/cover/>. Acesso em: 2 mar. 2006.

CSBASE. Disponível em: <http://www.tecgraf.puc-rio.br/csbase/>. Acesso em: 25 fev. 2006.

DAHMANN, J.; WEATHERLY, R.; KUHL, F. **Creating Computer Simulation Systems: An Introduction to the High Level Architecture**. Upper Saddle River, NJ: Prentice-Hall, 1999.

DAVID, J. M. N.; BORGES, M. R. S. Selectivity of Awareness Components in Asynchronous CSCW Environments. In: 7TH INTERNATIONAL WORKSHOP ON GROUPWARE – CRIWG'2001, 2001. **Proceedings of the 7th International Workshop on Groupware – CRIWG'2001**, 2001, p. 115-124.

DEFENSE MODELING AND SIMULATION OFFICE HIGH LEVEL ARQUITECTURE Web Site, THE. Disponível em: <https://www.dmso.mil/public/transition/hla/>. Acesso em: 2 mar. 2006.

DEWAN, P. Architectures for Collaborative Applications. In: Beaudouin-Lafon (Ed.). **Computer Supported Cooperative Work**, John Wiley & Sons Ltd., 1999, p. 169-194.

DONGARRA, J. J.; HEMPEL, R.; HEY, A. J. G.; WALKER, D. W. A proposal for a user-level, message passing interface in a distributed memory environment. Technical Report TM-12231, Oak Ridge National Laboratory, 1993.

DOURISH, P. Using Metalevel Techniques in a Flexible Toolkit for CSCW Applications. **ACM Transactions on Computer-Human Interaction**, v. 5, n. 2, p. 109-155, 1998.

DOURISH, P.; BELLOTTI, V. Awareness and Coordination in Shared Workspaces. In: CSCW'92, nov. 1992. **Proceedings of CSCW'92**, 1992, p. 107-114.

DOURISH, P.; BLY, S. Portholes: Supporting Awareness in a Distributed Work Group. In: CHI'92, 3-7 mai. 1992. **Proceedings of CHI'92**, 1992, p. 541-547.

DYBVIG, R. K. **The Scheme Programming Language: ANSI Scheme**. 2.ed. Englewood Cliffs, NJ 07632, EUA: P T R Prentice-Hall, 1996.

EDWARDS, W. K. Policies and Roles in Collaborative Applications. In: CSCW'96, 1996, Cambridge, MA, EUA. **Proceedings of CSCW'96**, 1996, p. 11-20.

ELLIS, C. A.; GIBBS, S. J.; REIN, G. L. Groupware – some Issues and Experiences. **Communications of the ACM**, v. 34 , n. 1, p. 38-58, 1991.

FAGG, G. E.; MOORE, K.; DONGARRA, J. J.; GEIST, A. Scalable Network Information Processing Environment (SNIPE). In: INTERNATIONAL CONFERENCE FOR HIGH PERFORMANCE COMPUTING AND COMMUNICATIONS, 15-21 nov. 1997, San Jose, CA, EUA. **Proceedings of International Conference for High Performance Computing and Communications**, 1997.

FOSTER, I.; KESSELMAN, C. Globus: A Metacomputing Infrastructure Toolkit. **International Journal of Supercomputer Applications**, v. 11, n. 2, p. 115-128, 1997.

FOSTER, I.; KESSELMAN, C. (Ed.). **The GRID: Blueprint for a New Computing Infrastructure**. Morgan Kaufmann, 1998.

FUKS, H.; RAPOSO, A. B.; GEROSA, M. A.; LUCENA, C. J. P. Applying the 3C Model to Groupware Development. **International Journal of Cooperative Information Systems (IJCIS)**, World Scientific, v. 14, n. 2-3, p. 299-328, 2005.

FURUTA, R.; STOTTS, P. D. Interpreted Collaboration Protocols and their use in Groupware Prototyping. In: CSCW'94, 1994, Chapel Hill, NC, EUA. **Proceedings of CSCW'94**, 1994, p. 121-131.

FUSSELL, S. R.; KRAUT, R. E.; SIEGEL, J. Coordination of Communication: Effects of Shared Visual Context on Collaborative Work. In: CSCW'00, 2-6 dez. 2000, Philadelphia, PA, EUA. **Proceedings of CSCW'00**, 2000, p. 21-30.

GODEFROID, P.; HERBSLEB, J. D.; JAGADEESAN, L. J.; LI, D. Ensuring Privacy in Presence Awareness Systems: An Automated Verification Approach. In: CSCW'00, 2-6 dez. 2000, Philadelphia, PA, EUA. **Proceedings of CSCW'00**, 2000, p. 59-68.

GREENHALGH, C. Understanding the Network Requirements of Collaborative Virtual Environments. In: ___**Collaborative Virtual Environments**. London: Springer, 2001, p. 55-74.

GREENHALGH, C.; PUBRICK, J.; SNOWDON, D. Inside MASSIVE-3: Flexible Support for Data Consistency and World Structuring. In: ACM CONFERENCE ON COLLABORATIVE VIRTUAL ENVIRONMENTS (CVE2000), 2000, San Francisco, EUA. **Proceedings of ACM Conference on Collaborative Virtual Environments (CVE2000)**, 2000, p. 119-127.

GREENSPAN, S.; GOLDBERG, D.; WEISMER, D.; BASSO, A. Interpersonal Trust and Common Ground in Electronically Mediated Communication. In: CSCW'00, 2-6 dez. 2000, Philadelphia, PA, EUA. **Proceedings of CSCW'00**, 2000, p. 251-260.

GRIMSHAW, A. S.; WULF W. A. The Legion Vision of a Worldwide Virtual Computer. **Communications of the ACM**, v. 40, n. 1, 1997.

GROSS, T.; PRINZ, W. Modelling Shared Contexts in Cooperative Environments: Concept, Implementation, and Evaluation. **Computer Supported Cooperative Work**, Kluwer Academic Publishers, Países Baixos, v. 13, n. 3-4, p. 283-303, ago. 2004.

GUTWIN, C.; GREENBERG, S. Design for Individuals, Design for Groups: Tradeoffs Between Power and Workspace Awareness. In: CSCW'98, 1998, Seattle, Washington, EUA. **Proceedings of CSCW'98**, 1998, p. 207-216.

HANDEL, M.; HERBSLEB, J. D. What is Chat Doing in the Workplace? In: CSCW'02, 16-20 nov. 2002, New Orleans, Louisiana, EUA. **Proceedings of CSCW'02**, 2002, p. 1-10.

HART, S. V. Crisis Information Management Software (CIMS) Feature Comparison Report. U.S. Department of Justice, Office of Justice Programs, National Institute of Justice, NCJ 197065, EUA, 2002.

HAYNES, S. R.; PURAO, S.; SKATTEBO, A. L. Situating Evaluation in Scenarios of Use. In: CSCW'04, 6-10 nov. 2004, Chicago, Illinois, EUA. **Proceedings of CSCW'04**, 2004, p. 92-101.

HERBSLEB, J. D.; GRINTER, R. E. Splitting the Organization and Integrating the Code: Conway's Law Revisited In: ICSE'99, 1999, Los Angeles, CA, EUA. **Proceedings of ICSE'99**, 1999, p. 85-95.

HERBSLEB, J. D.; MOCKUS, A.; FINHOLT, T. A.; GRINTER, R. E. Distance, Dependencies, and Delay in a Global Collaboration. In: CSCW'00, 2-6 dez. 2000, Philadelphia, PA, EUA. **Proceedings of CSCW'00**, 2000, p. 319-328.

HUBBOLD, R.; COOK, J.; KEATES, M.; GIBSON, S.; HOWARD, T.; MURTA, A.; WEST, A.; PETTIFER, S. GNU/MAVERIK: A micro-kernel for large-scale virtual environments. **Presence: Teleoperators and Virtual Environments** 10, p. 22-34, 2001. ISSN 1054-7460.

IEEE - INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC. Disponível em: <http://www.ieee.org>. Acesso em: 3 mar. 2006.

ISHII, H.; KOBAYAHSI, M.; ARITA, K. Iterative Design of Seamless Collaboration Media. **Communications of the ACM**, v. 37, n. 8, p. 83-97, ago. 1994.

JOHNSEN, S. O.; BJØRKLI, C.; STEIRO, T.; FARTUM, H.; HAUKENES, H.; SKRIVER, J. CRIOP®: A scenario method for Crisis Intervention and Operability analysis. Report No. STF38 A03424, SINTEF, Noruega, 2004.

JONES, Q.; GRANDHI, S. A.; TERVEEN, L.; WHITTAKER, S. People-to-People-to-Geographical-Places: The P3 Framework for Location-Based Community Systems. **Computer Supported Cooperative Work**, Kluwer Academic Publishers, Países Baixos, v. 13, n. 3-4, p. 249-282, ago. 2004.

JUNGHYUN, K.; SUNGIL, K.; UNGYEON, Y.; NAMGYU, K. COVRA-CAD: A CORBA based Virtual Reality Architecture for CAD. In: INTERNATIONAL CONFERENCE ON VIRTUAL SYSTEMS AND MULTIMEDIA, 1998. **Proceedings of International Conference on Virtual Systems and Multimedia**, 1998.

KAPOLKA, A. **The Extensible Run-Time Infrastructure (XRTI): An Experimental Implementation of Proposed Improvements to the High Level Architecture**. 2003. 133 p. Master's thesis – Naval Postgraduate School, Monterey, CA, EUA, 2003.

LAUCHE, K. Collaboration Among Designers: Analysing an Activity for System Development. **Computer Supported Cooperative Work**, Kluwer Academic Publishers, Países Baixos, v. 14, n. 3, p. 253-282, jun. 2005.

LAURILLAU, Y.; NIGAY, L. Clover Architecture for Groupware. In: CSCW'02, 16-20 nov. 2002, New Orleans, Louisiana, EUA. **Proceedings of CSCW'02**, 2002, p. 236-245.

LEONTJEV, A. N. **Activity, Consciousness and Personality**. Englewood Cliffs, NJ, EUA: Prentice-Hall, 1978.

LI, D.; MUNTZ, R. COCA: Collaboration Objects Coordination Architecture. In: CSCW'98, 1998, Seattle, Washington, EUA. **Proceedings of CSCW'98**, 1998, p. 179-188.

LIMA, M. J.; MELCOP, T.; CERQUEIRA, R.; CASSINO, C.; SILVESTRE, B.; NERY, M.; URURAHY, C. CSGrid: Um Sistema para Integração de Aplicações em Grades Computacionais. In: SBRC'2005, SALÃO DE FERRAMENTAS, mai. 2005, Fortaleza, Ceará, Brasil.

LOCKE, J. An Introduction to the Internet Networking Environment and SIMNET/DIS. Computer Science Department, Naval Postgraduate School, EUA, 1993.

MACEDONIA, M. R.; BRUTZMAN, D. P.; ZYDA, M. J.; PRATT, D. R.; BARHAM, P. T.; FALBY, J.; LOCKE, J. NPSNET: A Multi-Player 3D Virtual Environment over the Internet. In: ACM 1995 SYMPOSIUM ON INTERACTIVE 3D GRAPHICS, 1995, Monterey, California, EUA. **Proceedings of the ACM 1995 Symposium on Interactive 3D Graphics**, 1995.

MACEDONIA, M. R.; ZYDA, M. J. A Taxonomy for Networked Virtual Environments. **IEEE Mutimedia**, v. 4, n. 1, p. 48-56, 1997.

MACEDONIA, M. R.; ZYDA, M. J.; PRATT, D. R.; BRUTZMAN, D. P.; BARHAM, P. T. Exploiting Reality with Multicast groups: A Network Architecture for Large-Scale Virtual Environments. **IEEE Computer Graphics and Applications**, v. 15, n. 5, p. 38-45, 1995.

MARSH, J.; PETTIFER, S.; WEST, A. J. A technique for maintaining continuity of perception in networked virtual environments. In: UKVRSIG'99, 1999, Salford, Reino Unido. **Proceedings of the UKVRSIG'99**, 1999.

MASTAGLIO, T. W.; CALLAHAN, R. A Large-Scale Complex Virtual Environment for Team Training. **Computer**, v. 28, n. 7, 1995.

MCDONALD, D. W.; ACKERMAN, M. S. Expertise Recommender: A Flexible Recommendation System and Architecture. In: CSCW'00, 2-6 dez. 2000, Philadelphia, PA, EUA. **Proceedings of CSCW'00**, 2000, p. 231-240.

MCLEOD INSTITUTE OF SIMULATION SCIENCES. California State University, Chico, California, EUA. Disponível em: <http://www.ecst.csuchico.edu/~mcleod/>. Acesso em: 3 mar. 2006.

MELO, R. N. Um ambiente de ferramentas integradas para desenvolvimento de sistemas interativos. In: I Simpósio Brasileiro de Engenharia de Software, out. 1987. **Anais do I Simpósio Brasileiro de Engenharia de Software**, 1987, p. 159-169.

MILNER, R. **Communication and Concurrency**. Prentice-Hall International, International Series on Computer Science, 1989.

MORRIS, M. R.; MORRIS, D.; WINOGRAD, T. Individual Audio Channels with Single Display Groupware: Effects on Communication and Task Strategy. In: CSCW'04, 6-10 nov. 2004, Chicago, Illinois, EUA. **Proceedings of CSCW'04**, 2004a, p. 242-251.

MORRIS, M. R.; RYALL, K.; SHEN, C.; FORLINES, C.; VERNIER, F. Beyond "Social Protocols": Multi-User Coordination Policies for Co-located Groupware. In: CSCW'04, 6-10 nov. 2004, Chicago, Illinois, EUA. **Proceedings of CSCW'04**, 2004b, p. 262-265.

MORRISON, J. The VR-Link Networked Virtual Environment Software Infrastructure. **Presence: Teleoperators and Virtual Environments**. Spring, 1995.

MPI-2. Extensions to the Message-Passing Interface. Technical Report, University of Tennessee, Knoxville, Tennessee, EUA. Disponível em: <http://www.mpi-forum.org/docs/mpi-20-html/mpi2-report.html>. Acesso em: 8 mar. 2006.

MPRI SHIP ANALYTICS. **L-3 CRISIS Brochure**. North Stonington, Connecticut, EUA, 2003.

MURATA, T. Petri Nets: properties, analysis and applications. **Proceedings of the IEEE**, v. 77, n. 4, p. 541-580, 1989.

NARDI, B. A.; WHITTAKER, S.; BRADNER, E. Interaction and Outeraction: Instant Messaging in Action. In: CSCW'00, 2-6 dez. 2000, Philadelphia, PA, EUA. **Proceedings of CSCW'00**, 2000, p. 79-88.

NEALE, D. C.; CARROLL, J. M.; ROSSON, M. B. Evaluating Computer-Supported Cooperative Work: Models and Frameworks. In: CSCW'04, 6-10 nov. 2004, Chicago, Illinois, EUA. **Proceedings of CSCW'04**, 2004, p. 112-121.

OMG – Object Management Group. Disponível em: <http://www.omg.org>. Acesso em: 20 mar. 2006.

OMG – Object Management Group. The Common Object Request Broker: Architecture and Specification. Revision 2.0. OMG Document, 1995.

PAEK, T.; AGRAWALA, M.; BASU, S.; DRUCKER, S.; KRISTJANSSON, T.; LOGAN, R.; TOYAMA, K.; WILSON, A. Toward Universal Mobile Interaction for Shared Displays. In: CSCW'04, 6-10 nov. 2004, Chicago, Illinois, EUA. **Proceedings of CSCW'04**, 2004, p. 266-269.

PALEN, L. Social, Individual & Technological Issues for Groupware Calendar Systems. In: CHI'99, 1999, Pittsburgh, PA, EUA. **Proceedings of CHI'99**, 1999, p. 17-24.

PANKOKE-BABATZ, U.; SYRI, A. Collaborative Workspaces for Time Deferred Electronic Cooperation. In: GROUP'97, 1997, Phoenix, Arizona, EUA. **Proceedings of GROUP'97**, 1997, p. 187-196.

PARK, K.; CHO, Y.; KRISHNAPRASAD, N.; SCHARVER, C.; LEWIS, M.; LEIGH, J.; JOHNSON, A. CAVERNSoft G2: A toolkit for High Performance Tele-Immersive Collaboration. In: ACM SYMPOSIUM ON VIRTUAL REALITY SOFTWARE AND TECHNOLOGY 2000 (VRST'00), 2000, Seoul, Coréia do Sul. **Proceedings of the ACM Symposium on Virtual Reality Software and Technology 2000 (VRST'00)**, 2000, p. 8-15.

PARKER, S. G. **The SCIRun problem solving environment and computational steering software system**. PhD Thesis, 1999.

PETROBRAS, REVISTA. Rio de Janeiro, ano 10, n. 96, p. 10-13, jan. 2004.

PETROBRAS MAGAZINE. Rio de Janeiro, v. 7, n. 33, p. 30-35, 2001.

PETTERSSON, M.; RANDALL, D.; HELGESON, B. Ambiguities, Awareness and Economy: A Study of Emergency Service Work. **Computer Supported Cooperative Work**, Kluwer Academic Publishers, Países Baixos, v. 13, n. 2, p. 125-154, abr. 2004.

PETTIFER, S.; COOK, J.; MARSH, J.; WEST, A. J. DEVA3: Architecture for a Large Scale Virtual Reality System. In: ACM SYMPOSIUM ON VIRTUAL REALITY SOFTWARE AND TECHNOLOGY 2000 (VRST'00), 2000, Seoul, Coréia do Sul. **Proceedings of the ACM Symposium on Virtual Reality Software and Technology 2000 (VRST'00)**, 2000.

PINELLE, D. A Survey of Groupware Evaluations in CSCW Proceedings. Technical Report HCI-TR-2000-01, Computer Science Department, University of Saskatchewan, Canadá, 2000.

RANTZAU, D.; FRANK, K.; LANG, U.; RAINER, D.; WOSSNER, U. COVISE in the CUBE: An Environment for Analysing Large and Complex Simulation

Data. In: 2ND WORKSHOP ON IMMERSIVE PROJECTION TECHNOLOGY (IPT'98), 1998, Ames, Iowa, EUA. **Proceedings of the 2nd Workshop on Immersive Projection Technology (IPT'98)**, 1998.

RAPOSO, A. B.; FUKS, H. Defining Task Interdependencies and Coordination Mechanisms for Collaborative Systems. In: A. M. Pinna-Dery, K. Schmidt and P. Zaraté (Eds.). **Cooperative Systems Design: A Challenge of the Mobility Age** (Frontiers in Artificial Intelligence and Applications, v. 74). Amsterdam: IOS Press, 2002. p. 88-103.

RIBAK, A.; JACOVI, M.; SOROKA, V. "Ask Before You Search" Peer Support and Community Building with ReachOut. In: CSCW'02, 16-20 nov. 2002, New Orleans, Louisiana, EUA. **Proceedings of CSCW'02**, 2002, p. 126-135.

ROUSSEV, V.; DEWAN, P.; JAIN, V. Composable Collaboration Infrastructures Based on Programming Patterns. In: CSCW'00, 2-6 dez. 2000, Philadelphia, PA, EUA. **Proceedings of CSCW'00**, 2000, p. 117-126.

RUSSO, E. E. R. **Um sistema de interpretação de diálogos gráficos**. 1988. 235 p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, mar. 1988.

RUSSO, E. E. R.; RAPOSO, A. B.; FERNANDO, T.; GATTASS, M. Workspace Challenges for the Oil & Gas Exploration & Production Industry. In: 4TH CONFERENCE OF CONSTRUCTION APPLICATIONS OF VIRTUAL REALITY - CONVR 2004, 14-15 set. 2004, Lisboa, Portugal. **Proceedings of the 4th Conference of Construction Applications of Virtual Reality - CONVR 2004**, 2004, p. 145-150.

SACHS, P. Transforming Work: Collaboration, Learning and Design, **Communications of the ACM**, v. 38, n. 9, p. 36-44, set. 1995.

SANTOS, A. G. **Sistema de gerência de interface do usuário: Especificação e interpretação de diálogos**. 1986. 93 p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, ago. 1986.

SCHMIDT, D. The adaptive communication environment: Object-orientated network programming components for developing client/server applications. Technical report, 11th and 12th Sun Users Group, 1994.

SCHMIDT, D. C. Our research on high-performance and real-time corba. Disponível em: <http://www.cs.wustl.edu/~schmidt/corba-research-overview.html>. Acesso em: 25 fev. 2006.

SCHUCKMANN, C.; KIRCHNER, L.; SCHÜMMER, J.; HAAKE, J. M. Designing object-oriented synchronous groupware with COAST. In: CSCW'96, 1996, Cambridge, MA, EUA. **Proceedings of CSCW'96**, 1996, p. 30-38.

SETLOCK, L. D.; FUSSELL, S. R.; NEUWIRTH, C. Taking It Out of Context: Collaborating within and across Cultures in Face-to-Face Settings and via Instant Messaging. In: CSCW'04, 6-10 nov. 2004, Chicago, Illinois, EUA. **Proceedings of CSCW'04**, 2004, p. 604-613.

SILICON GRAPHICS. Opengl scene graph / opengl++. Technical report, Silicon Graphics, 1997.

SINGHAL, S. K. **Effective Remote Modeling in Large-Scale Distributed Simulation and Visualization Environments**. PhD Thesis, Department of Computer Science, Stanford University, 1996.

SINGHAL, S.; ZYDA. M. **Networked Virtual Environments: Design and Implementation**. New York: ACM Press, 1999.

STEVENS, G.; WULF, V. A New Dimension in Access Control: Studying Maintenance Engineering across Organizational Boundaries. In: CSCW'02, 16-20 nov. 2002, New Orleans, Louisiana, EUA. **Proceedings of CSCW'02**, 2002, p. 196-205.

SWEDISH INSTITUTE OF COMPUTER SCIENCE (SICS): DIVE – A Toolkit for Distributed VR Applications. Disponível em: <http://www.sics.se/dce/dive>. Acesso em: 2 mar. 2006.

SWI-PROLOG. Disponível em: <http://www.swi-prolog.org>. Acesso em: 3 mar. 2006.

TJORA, A. Maintaining Redundancy in the Coordination of Medical Emergencies. In: CSCW'04, 6-10 nov. 2004, Chicago, Illinois, EUA. **Proceedings of CSCW'04**, 2004, p. 132-141.

TRAMBEREND, H. Avocado: A Distributed Virtual Reality Framework. In: IEEE VIRTUAL REALITY'99, 1999. **Proceedings of IEEE Virtual Reality'99**, 1999.

TUIKKA, T. Remote Concept Design from An Activity Theory Perspective. In: CSCW'02, 16-20 nov. 2002, New Orleans, Louisiana, EUA. **Proceedings of CSCW'02**, 2002, p. 186-195.

UML – UNIFIED MODELING LANGUAGE. OMG Documentation. Disponível em: <http://www.omg.org/technology/documents/formal/uml.htm>. Acesso em: 20 mar. 2006.

WAMIT. Disponível em: <http://www.wamit.com>. Acesso em: 9 mar. 2006.

WATSEN, K.; ZYDA, M. Bamboo – A Portable System for Dynamically Extensible, Real-time, Networked Virtual Environments. In: VRAIS'98, 1998. **Proceedings of VRAIS'98**, 1998, p. 260-267.

WEST, A.; HUBBOLD, R. System Challenges for Collaborative Virtual Environments. In: ___**Collaborative Virtual Environments**. London: Springer, 2001. p. 44-54.

XCAT. The Indiana University, Extreme Lab implementation of the Common Component Architecture (CCA). Disponível em: <http://www.extreme.indiana.edu/xcat/>. Acesso em: 25 fev. 2006.

YANG, Y.; LI, D. Separating Data and Control: Support for Adaptable Consistency Protocols in Collaborative Systems. In: CSCW'04, 6-10 nov. 2004, Chicago, Illinois, EUA. **Proceedings of CSCW'04**, 2004, p. 11-20.

YOUNGMAN, N. DIS Frequently Asked Questions. Disponível em: <http://www.crg.cs.nott.ac.uk/~dxl/DIS/dis.faq>. Acesso em: 2 mar. 2006.

Apêndice A: Entrevistas

Neste Apêndice, relatamos as entrevistas que foram realizadas com os usuários-chave quando do levantamento dos requisitos do sistema.

No dia 13/10/2004, reuni-me com o engenheiro Álvaro Maia, Gerente dos Métodos Científicos do Centro de Pesquisas e Desenvolvimento da Petrobras (CENPES), abordando os acidentes na área de óleo e gás:

- os maiores foram o do Exxon Valdez (4 a 8 bilhões de dólares) e o do Piper Alpha;
- Álvaro falou que o pior é o lucro cessante: 150.000 barris por dia por 3 anos de uma P-40 são US$ 3 bilhões (com um cálculo bem conservador considerando US$ 20 o barril; hoje, ele já passa dos US$ 40), fora US$ 700 milhões do empreendimento.

Em 15/10/2004, o Prof. Terrence Fernando da University of Salford teve uma reunião com o engenheiro Álvaro Maia, discutindo detalhes sobre a operação de plataformas e sobre as situações de emergência:

- as plataformas da Bacia de Campos pertencem à Unidade de Negócios do Rio (UN-Rio);
- os dados sobre as plataformas ficam armazenados no sistema GIEN;
- o controle de lastro e estabilidade da plataforma é feito utilizando-se o software SSTAB;
- em situações de emergência, uma equipe típica é constituída de três especialistas de hidrodinâmica, três especialistas de análise de *risers* e 10 especialistas de análise de estabilidade;
- o sistema InfoPAE possui os procedimentos a serem seguidos em casos de emergência;
- a esposa do Álvaro Maia fez uma tese de doutorado sobre comportamentos em empresas em situações de emergência (Costa, 2004).

Ainda em 15/10/2004, o Prof. Terrence e eu nos reunimos com o engenheiro Isaias Masetti, também do CENPES, para discutirmos requisitos do sistema:

- Masetti deseja que o sistema se pareça com um jogo de computador;
- deve tratar de estabilidade e hidrodinâmica;
- no mínimo deve contemplar:
  - SSTAB + DYNASIM (código aberto) + dinâmica;
  - para cada inclinação no tempo:
    - análise hidrodinâmica;
    - forças;
    - cada passo no tempo.

  Não seria para um caso de emergência (isto é feito pelo SSTAB);
- Masetti falou sobre o Tanque de Provas Numérico (TPN) ou *Numerical Offshore Tank* (NOT);
- falou também sobre o *Mesh Generator* (MG);
- falou ainda sobre o *Web Analysis*, desenvolvido por uma parceria entre a UFAL e a USP:
  - eles programariam e eu elaboraria o conceito (seria o arquiteto do sistema);
  - envolveria realidade virtual + análise hidrodinâmica + análise de linhas;
  - seria para *benchmark* com modelos de pequena escala.

  Seriam elaborados modelos para verificar o sistema;
- SSTAB -> WAMIT (do MIT) -> TPN: não é tempo real;
- TPN = DYNASIM + WAMIT (coeficientes de forças hidrodinâmicas).

  Pelo menos, deveria ser feita a integração do SSTAB com estes sistemas;
- a idéia seria fazer uma grande cooperação envolvendo a UFAL (análise de linhas), a USP (DYNASIM + WAMIT + realidade virtual), a PUC (MG + SSTAB) e também a University of Salford:
  - a análise de linhas é a mais pesada (as outras demoram apenas 5 minutos);
  - tempo real;

  - mudanças repentinas, linhas danificadas;
  - cada sistema foi desenvolvido no Mestrado de alguém:
    - DPS: *Dynamically Positioning System* (ao invés de sistema de movimento);
    - WAMIT: do engenheiro Donato, do CENPES;
  - o sonho do Masetti é fazer tudo via Web, sem a necessidade de uma grande estação de trabalho;
- alta tecnologia para o futuro seria um grande aglomerado de computadores: verificar os requisitos e treinamento;
- o sistema é distribuído: temos que ver as interfaces com as pessoas;
- SSTAB + DYNASIM + WAMIT integrados:
  - primeiro passo: colocá-los juntos;
  - propriedades de estabilidade:
    - quase-estática (passo a passo);
    - depois disso, dinâmica (ex.: *Particle Method System* – PMS);
- se o sistema for um sucesso, será passado para o GIEN e todo mundo o utilizará:
  - o melhor seria fazer um pré-processador para importar os dados de entrada reais (faríamos uma *shell* sem provocar nenhum ruído):
    - eles podem assumir o novo formato no futuro;
    - sistema especializado para danos.

Em 31/01/2005, o Prof. Terrence e eu apresentamos a Proposta de Tese ao engenheiro Álvaro Maia, que, após ler detalhadamente a Proposta, nos deu o seguinte retorno:

- considerou a Proposta muito bem organizada e disse nunca ter visto uma Proposta tão bem descrita;
- aconselhou investigar os Sistemas de Tratamento de Crises existentes;

- no acidente da P-34, teve que re-executar todas as simulações de novo para preparar o relatório; agora tem todas as simulações armazenadas no sistema INFOPAE;
- depois do acidente da P-36, o sistema GIEN da Petrobras passou a armazenar todos os modelos;
- disse para se prever até 10 a 15 pessoas acessando o mesmo modelo, ao se fazer simulações via o INFOPAED (INFOPAE Dinâmico);
- disse que a tese deve enfatizar o aspecto de visualização.
  O Prof. Terrence disse que as pessoas poderão fazer suas próprias simulações e mostrar seus resultados;
- disse que o principal objetivo é mostrar os resultados via o INFOPAED, que utiliza o banco de dados do INFOPAE com simulação em tempo real (todos os modelos seriam executados por meio do INFOPAED);
- fez uma correção na Proposta: a decisão de alto nível não está na Sede da empresa, mas sim na Unidade Operacional;
- disse para reduzir o escopo da tese, fazendo apenas uma definição em nível mais amplo e se concentrando na parte técnica, se possível com um exemplo prático (um programa com visualização ou um algoritmo);
- O Prof. Terrence disse que a tese definiria um terceiro paradigma (após os acidentes da P-36 e da P-34) para tratamento de emergências como seu resultado e também que o desenvolvimento não partiria do zero, aproveitando-se a tecnologia já disponível.

Em 02/02/2005, o Prof. Terrence e eu conversamos com o engenheiro Luiz Cristovão Coelho do Tecgraf/PUC-Rio, que participou das operações de salvamento tanto da P-34 quanto da P-36:

- a *task force* da P-36 estava baseada em Macaé e a da P-34 na UN-Rio;
- o modelo de estabilidade da P-36 foi feito em dois dias com mais volumes e baseado em modelo de empresa terceirizada;

- o modelo de estabilidade da P-34 foi feito em duas horas com uns 10 ou 15 volumes;
- não havia como salvar a P-36 sem estabilidade, mesmo com mergulhadores.

Em 04/02/2005, o Prof. Terrence e eu fomos recebidos pela Taciana Melcop no Tecgraf 2 da PUC-Rio, que nos falou sobre o CSGRID (o *framework* utilizado pelo INFOPAED):

- os programas são acionados em algum servidor e lêem e escrevem na área central de dados, que é protegida;
- vários simuladores do Álvaro Maia estão nas listas e podem ser acionados (alguns podem ser executados em aglomerados de computadores): ANFLEX, ANPEC, SSTAB, WAMIT, etc;
- SSTAB hoje só pode ser executado como programa externo:
  - carrega o dado da área central em disco local;
  - copia do disco local para a memória e apaga o disco local;
  - no final da simulação, copia da memória para o disco local;
  - finalmente, copia do disco local para a área central e apaga do disco local.

Em 26/09/2005, reuni-me com o engenheiro Mauro Costa Oliveira no CENPES, discutindo sobre o uso integrado dos simuladores SSTAB, WAMIT e DYNASIM em situações de emergência. O engenheiro Mauro disse que teria interesse em testar a arquitetura sendo proposta também em trabalhos de rotina.

Em 05/10/2005, fiz a apresentação sobre a tese para o Gerente de Pesquisa, Engenharia e Corporativo da Tecnologia da Informação da Petrobras (TI/TI-PEC), Roberto Murilo, na presença dos orientadores Prof. Terrence Fernando e Prof. Alberto Raposo. O retorno dado pelo Gerente Roberto Murilo foi bastante positivo, com ele se mostrando interessado no uso da arquitetura proposta também em outras situações de emergência, como por exemplo em refinarias.

Em 06/12/2005, reuni-me com o engenheiro Mauro Costa Oliveira no CENPES, discutindo formas de se acionar automaticamente o simulador WAMIT a partir da obtenção do resultado do simulador SSTAB.

Apêndice B: Artigos Publicados

Reproduzimos no presente Apêndice os quatro artigos que já foram elaborados a respeito deste trabalho e o resumo da palestra proferida durante uma oficina internacional no Reino Unido.

Os artigos serão apresentados na ordem seguinte:

1. *A Multiple-Perspective Architecture for CSCW Applications*.
   Co-autores: Alberto Raposo, Terrence Fernando e Marcelo Gattass.
   Este artigo será apresentado durante a *7th International Conference on the Design of Cooperative Systems* (COOP'06), a ser realizada em Provence, França, de 9 a 12 de maio de 2006.
2. *Configuring a Collaborative Virtual Workspace for Disaster Management of Oil & Gas Offshore Structures*.
   Co-autores: Alberto Raposo, Terrence Fernando, Marcelo Gattass e Börje Karlsson.
   Este artigo será apresentado durante a *11th International Conference on Computing in Civil and Building Engineering* (ICCCBE-XI), a ser realizada em Montreal, Canadá, de 14 a 16 de junho de 2006.
3. *Workspace Challenges for the Oil & Gas Exploration & Production Industry*.
   Co-autores: Alberto B. Raposo, Terrence Fernando e Marcelo Gattass.
   Este artigo foi apresentado durante a *4th Conference of Construction Applications of Virtual Reality* - CONVR 2004, realizada em Lisboa, Portugal, de 14 a 15 de setembro de 2004.
4. *Emergency Environments for the Oil & Gas Exploration and Production Industry*.
   Co-autores: Alberto Raposo, Terrence Fernando e Marcelo Gattass.
   Embora tendo sido um dos artigos selecionados dentre 1200 outros para ser apresentado durante o *18th World Petroleum Congress*, realizado em Johannesburg, África do Sul, de 25 a 29 de setembro

de 2005, este Poster não foi apresentado nem publicado porque nenhum dos autores participou do Congresso.
(Disponível em: <http://www.world-petroleum.org/18thwpc/18th%20WPC%20Programme%20Master%2029-03-05.doc> e <www.18wpc.com/about_block1.html>. Acesso em: 4 de março de 2006.)

Finalmente, reproduzimos o resumo da palestra proferida durante o *International Workshop on Virtual Prototyping* realizado de 10 a 11 de março de 2005 em Salford Quays, Manchester, Reino Unido.

- Título: *Virtual Prototyping Challenges for the Oil & Gas Exploration & Production Industry*.
  Co-autor: Terrence Fernando.
  Resumo (Disponível em: <http://www.avprc.ac.uk/abstracts.shtml>. Acesso em: 4 de março de 2006):
  "The oil & gas industry has been a leading player in exploiting the power of virtual reality technology to enhance its business processes. The Virtual Reality Centres (VRCs), large projection rooms with features such as 3D and stereoscopic images, soon became very popular in the oil & gas industry, since they gave specialists the ability to quickly and comprehensively interpret large volume of data, thus significantly reducing cycle time for prospect generation.

  However, due to ever increasing business pressures, there are further demands on researchers to extend the capabilities of the VR technologies, so that they can be fully utilised in all the oil & gas exploration and production (E&P) phases - reservoir exploration, design and construction of the production facilities, and production and transportation of the oil & gas - and their activities, such as 3D geomodelling, seismic interpretation, real-time drilling follow-up and correction, offshore structures' design, static and dynamic simulations of these offshore structures, oil pipelines' monitoring and emergency situations' handling.

This talk presents the main E&P processes of the oil & gas industry that can benefit from the VR technologies and discusses the research challenges emerging from these processes while defining and building virtual prototypes for their activities.”

A Multiple-Perspective Architecture for CSCW Applications

Enio Emanuel Ramos RUSSO [a, b, 1], Alberto RAPOSO [a],
Terrence FERNANDO [c] and Marcelo GATTASS [a]
[a] *Tecgraf, Department of Computer Science, PUC-Rio, Brazil*
[b] *PETROBRAS Research and Development Centre (CENPES), Brazil*
[c] *School of Construction and Property Management, University of Salford, UK*

Abstract. We present a mutiple-perspective collaboration metamodel, which mixes Place-Centred and People-Centred perspectives. It allows instances of the metamodel to be derived and experimented until the more adequate to a particular situation is found. It also allows parametric changes in run-time, enhancing the flexibility of the metamodel. The motivation for this work was extracted from the necessity of developing for a global oil & gas company a collaborative virtual workspace for disaster management of oil & gas offshore structures.

Keywords. collaboration modeling, metamodel, collaboration architecture, multiple perspectives, oil & gas

Introduction

Many companies have been creating virtual teams that bring together geographically dispersed workers with complementary skills, increasing the demand for CSCW applications. In order to make the development of a wide range of these collaborative applications more effective, we should offer a general architecture that is adaptable to different situations, tasks, and settings in a flexible way.

CSCW research to date on how to address the architecture characteristic mentioned above has largely focused on issues concerning differences between: *(i)* co-located work and working across distance; or *(ii)* work with people from the same culture or common ground and work with people from different cultures. The previous perspectives have been named, respectively: *Place-Centred* and *People-Centred* [1].

We propose to adopt a different view on the problem based on the activities carried out by the teams participating in the collaborative work. We name it an *Activity-Centred* perspective, which may be seen as a multi-perspective concept since it not only encompasses the Place-Centred and the People-Centred perspectives, but also allows adopting each one or both of them (in a hybrid way) to the desired extent, and admits seamless change from one perspective to another.

The motivation for this work has been the necessity of developing a collaborative virtual workspace for disaster management of oil & gas offshore structures for a global company [2].

[1] Corresponding Author: Centro de Pesquisas e Desenvolvimento da PETROBRAS (CENPES), Cidade Universitária Quadra 7, Fundão, Rio de Janeiro, RJ, 21949-900, Brasil; E-mail: enio@tecgraf.puc-rio.br.

1. Activity-Centred Metamodel

Dewan's generic collaborative architecture [3] structures a groupware application into a variable number of layers from the domain-dependent level to the hardware level, where a layer is a software component corresponding to a specific level of abstraction. Similarly, the Clover architectural metamodel [4] also structures a groupware application into a variable number of layers, decomposing each layer into three functional sub-components dedicated to production, communication and coordination.

Our proposed metamodel adopts a similar multi-level approach, accordingly to Leontjev's [5, 6] activity theory version in which a three-level scheme describes the hierarchical structure of activity. Orthogonally to this approach, similarly to the Clover metamodel, the Activity-Centred metamodel also allows the breakdown of the components correspondent to a specific level. These two orthogonal approaches applied together contribute to the generality of the metamodel.

1.1. Metamodel Abstraction Levels

The top-most level is represented by a complex *node* which encompasses the whole activity. This level can be as diverse as the elaboration of this paper or the disaster management of an oil & gas offshore structure. The level immediately below contains the main actions that should be performed in order to accomplish the activity. These actions are the result of the interactions of groups, with each group represented by a complex node and the interactions among them represented by *edges*.

Splitting downwards each complex node of the upper abstraction level in more elementary nodes, we reach a *leaf node*, which will typically be a *person* or a *software agent*. To those leaf nodes we then associate implementation and hardware attributes such as the application to be executed and the host in which it should be run.

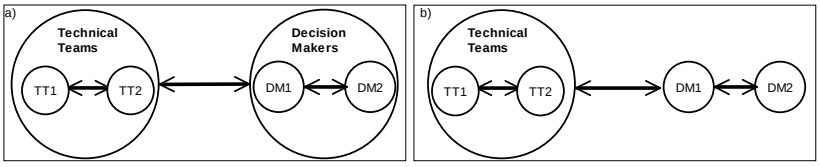

Figure 1. a) The first downward level of the oil & gas company from the disaster management collaborative application. b) Now Decision Maker 1 is placed between the Technical Teams node and Decision Maker 2.

Orthogonally to the top-down process, the Activity-Centred metamodel also allows the breakdown of the components correspondent to a specific level. Let's consider one level of the disaster management example, namely the first one downward of the oil & gas company (Figure 1a). We can observe two main groups: Technical Teams (TT) and Decision Makers (DM). TT is decomposed into sub-groups, and DM, also decomposed into sub-groups. In Figure 1a, both DMs have the same background and level of interaction with TT, while in Figure 1b DM2 has a higher organisational level, with DM1 making the link between TT and him.

1.2. Metamodel Components

1.2.1. Nodes and Edges

Nodes are essential components of our metamodel, going from the top-most node representing the whole activity through many nodes of different levels representing

groups and sub-groups until the *leaf nodes* representing a *person* or a *software agent*. Nodes have a set of *attributes* such as user interface preferences and language used, which are applied using a hierarchical class concept.

Nodes also have an attribute called *artifacts* defined as "all objects on which users can operate" [7]. Examples of artifacts are drawings, physical models, prototypes, and documents. Following the class concept, an artifact associated with a group node is shared by all members in the group, unless otherwise explicitly stated. In this case, a mechanism such as an access control list will determine who share access to the artifact.

Edges in our metamodel represent the *interaction paths* among nodes, which can be uni or bi-directed. When an edge is represented by a thin arrow, this means that the nodes on its extremities are co-located. When the arrow is thick, the nodes are placed remotely to each other. Edges have one important element, *channel*, which represents the electronically mediated channel that allows communication between two nodes.

1.2.2. Edge Especialisation Elements

We have identified the need for additional *edge especialisation elements*, namely *pre-* and *post-communication processings*, which are separated into two different classes. The first class is constituted by the ones directly associated with the leaf nodes. They represent the processings to be executed particularly onto a specific message being passed between two nodes and are stored in an especial table with key (message_id, receiver). The second class is constituted by the ones associated with groups on different levels of the metamodel hierarchy, representing the policies of these groups when respectively sending (*out-policies*) and receiving (*in-policies*) messages.

In Figure 2, we show possible pre- and post-communication processings that could be executed while sending a message from a Computer Science Researcher CR1 of the Computer Science Dept. CD1 of University U1 to Researcher CR2 of University U2.

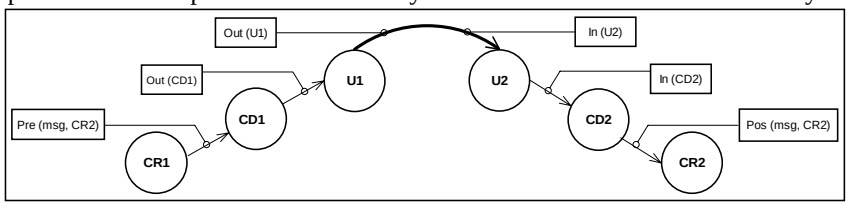

Figure 2. Activity-Centred metamodel: pre- and post-communication processings.

At the sender side, the natural candidate to execute the pre-processings is the leaf node who is sending the message. At the receiver side, this could be accomplished adding an attribute to the first group node pointed by the edge (in our example, U2) corresponding to the leaf node of this group to execute the post-processings.

Regarding the algorithms to be executed when sending a message, it is important to note that each message has one initial sender, which is necessarily a leaf node, and one or more final receivers, which can be either leaf or group nodes. The algorithms to be executed at either side are shown in Table 1.

1.2.3. Role Rules and Message Attributes Table

Role rules for coordination structure have been employed in CSCW for more than one decade [8, 9]. According to the majority of these studies, we adopted the strategy of separating the coordination structure and the computational program, using a logic-based specification language for specifying coordination policies.

Table 1. Activity-Centred metamodel: sender and receiver algorithms.

sender (sender, receiver, flag)

- until the receiver is found repeat
  - at the current level, search for the sub-tree that contains the receiver
  - if the receiver is found (and all the path from the sender to the receiver is determined)
    - if flag = in_table
      - execute the pre-processing associated with the pair (message, receiver)
    - else
      - create a new line in the message attributes table with pair (message, receiver) indicating the post-processing to be executed
    - execute all the out-policies associated with groups on levels in the path beginning at the sender until the communication edge is reached
    - send the message with the receiver to the leaf node which is assigned to the post-processing attribute of the receiver group node, or to the receiver itself
  - else
    - go to the upper level

Sender side of the communication edge (executed by the initial_sender leaf node):

- for each final_receiver associated with the message
  - sender (initial_sender, final_receiver, in_table)

Receiver side of the communication edge:

- receive the message
- execute all the in-policies associated with groups on levels in the path beginning at the present node until the final_receiver node is reached
- if the final_receiver is a leaf node
  - if it is equal to the post-processings execution node:
    - execute the post-processing associated with the pair (message, final_receiver)
  - else
    - send the message to the final_receiver
- else
  - execute the post-processing associated with the pair (message, final_receiver), which in this case should determine the leaf node(s) or group node(s) to receive the message
  - for each of the node(s) determined above (current_node)
    - sender (final_receiver, current_node, not_in_table)

We declare a *collaboration bus*, used to connect all participants, having at least one *channel* declaration. Different collaborations may be executed at the same time, each with its correspondent collaboration bus. The set of participants who are governed by the same set of coordination policies is playing the same *role*. When these policies also define the order in which the events occur, they can be considered workflow rules. Communication among participants occurs through one or more message channels associated with one collaboration bus. Similarly to COCA [9], the basic tasks of receiving messages and sending out messages are: *(i)* for receiving messages, an active rule named *on-arrive* with arguments *channel*, *receiver*, *message_id* (and *sender*); *(ii)* for sending out messages, a *send* formula with arguments *channel*, *sender*, *message_id* (and *receiver*).

We also build a *message attributes table* to enhance the flexibility of the coordination program, separating coordination rules from data related specifically to each message. This table provides an indirection that enables dynamic reconfiguration.

2. Instanciating the Activity-Centred Metamodel: Activity-Centred Models

Sometimes the most important aspects of our collaborative application are related to the place where people are effectively working. A model using this *Place-Centred*

perspective for a paper elaboration collaborative application is shown in Figure 3a. There, we have three main nodes: PUC University, Salford University and Petrobras (BR, Brazilian oil & gas company). The central node, playing the main role in writing the paper, is PUC, which communicates remotely with both Salford University and Petrobras. Within PUC, we have two sub-groups: one is the Computer Science (CP, C for Computer and P for PUC) Department, which has two co-located researchers, and the other is the Engineering department, which has one single Engineer (EP1). The two departments, being in different buildings, also communicate remotely. In Salford, there is only one Computer Science researcher, and in Petrobras, two co-located Engineers.

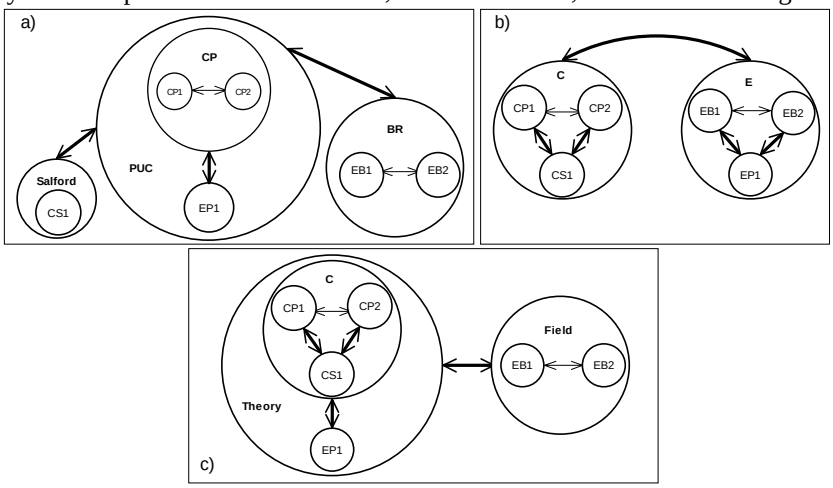

Figure 3. The paper elaboration collaborative application: a) a Place-Centred perspective; b) a People-Centred perspective; c) an Activity-Centred perspective.

Now consider that the main concerning issues of our collaborative application are related to culture and common ground barriers. In this case, we should derive a model with a *People-Centred* perspective (Figure 3b). We now have only two main nodes: the Computer Science researchers' (C) group and the Engineers' (E) group, communicating remotely. Within C group, we have three researchers: CP1 and CP2 work co-located and CS1 works remotely. It is important to note that, although CS1 is from a different university than CP1 and CP2, their common ground is so intense that they belong to the same sub-group. The same reasoning is applied to the E group.

We now mix the two previous perspectives in what we call an *Activity-Centred* perspective. In the present collaborative application, it seems more adequate to focus on the whole activity being performed – the paper elaboration – and then derive the groups to be formed. To elaborate the paper, authors CP1, CP2, CS1 and EP1 try to derive a new theoretical model based on the requirements' identified through field study, working together with Engineers EB1 and EB2. So we aggregate those people in two main groups formed based on their main activity: the Theory group and the Field group, which communicate remotely (Figure 3c).

3. Case Study

We now focus on the case study that motivated the creation of our metamodel: the development of a collaborative virtual workspace for disaster management of oil & gas

offshore structures for the Petrobras Research and Development Centre.

The disaster management of an oil & gas offshore structure is a complex operation involving three main groups: the oil & gas company, the Rescue Team and the Health Care Centre. This is an inter-organisational complex activity led by the oil & gas company, whose node will be detailed. An overall picture of the disaster management collaborative application is depicted in Figure 4.

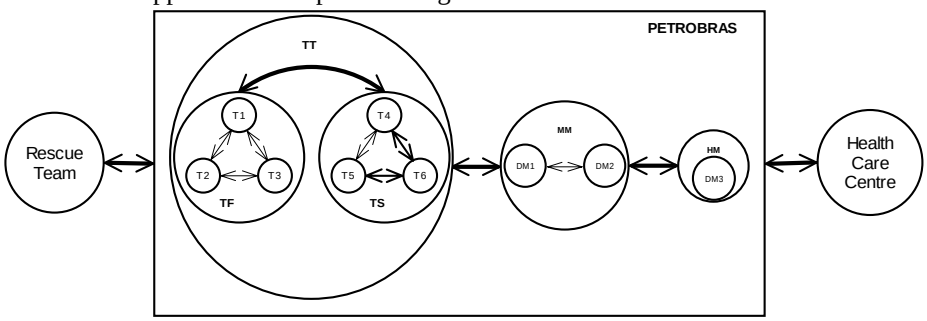

Figure 4. The disaster management collaborative application: overall picture.

Within Petrobras node, we identify three main groups: the Technical Teams (TT), the Middle-level Managers (MM) and the High-level Managers (HM), each one remotely located to the other. TT is formed by two technical sub-groups: the Task Force (TF) team and the Technical Support (TS) team, also remotely located.

TF plays the main role, leading the make-decision process. It is constituted by three co-located technicians, such as naval engineers, structural engineers, risers analysts or oceanographers. TF runs different simulators to derive the best solution to save the offshore unit, permanently communicating with TS. They also maintain contact with MM informing about their work evolution and asking for approval for their derived solution. Once their solution is approved, they pass the sequence of commands to be executed to the unit operator (not represented in our picture).

TS team, with technicians working in the same fields as TF team, can be invoked by TF team to perform especialised simulations focusing on some particular issues that would not be possible to be done by TF, or to obtain another opinion about the problem.

MM is constituted by middle-level managers working co-located in a company office, with one of them usually being the responsible to make the final decision. They have an overall knowledge about the technical issues and work constantly interacting with the TT group. They also communicate with the HM group, informing about the work evolution and eventually when they need to make a more critical decision.

3.1. Prototype

After investigating the activities involved in this disaster scenario, identifying their requirements in terms of ICT, we decided to concentrate on the Technical Teams group to develop a prototype of collaborative application implementing a particular model of our Activity-Centred metamodel. This prototype is particularly related to the work performed by the Task Force group (TF), including the simulators they run, their mutual communication and their interaction with the Middle-level Manager group.

We first investigate how TF runs the different simulators and what are the relationships among them. During a crisis situation, Petrobras typically uses three simulators. The first simulator to be run is SSTAB [10], the Floating Units Stability

system. The second simulator is called WAMIT and uses as input the results from SSTAB. The third simulator is DYNASIM [11], for Dynamic Stability. It uses as input the results obtained from WAMIT as well as additional parameters related to environmental conditions. DYNASIM calculates the forces acting on the mooring lines and risers. When these forces are considered extreme, a retrofeedback process is started, performing all the simulations again, beginning with SSTAB, to find another stable condition of the unit.

An Activity-Centred model representing this crisis situation (Figure 5a) can be derived based on the participants' roles. We created two remote groups: Technical Teams (TT) and Decision Makers (DM). TT is constituted by the Task Force (TF) team with members T0, T1 and T3, and the agent S2. DM is constituted by a single manager, a representative of all participants not directly involved with the technical part of the simulation activity such as operators and other managers, who only receive from TT follow-up messages, commands to be executed or approval requests.

Other than the interaction network part of the model just described, we also define rules and the message attributes table in order to represent the following workflow.

The Crisis Pilot T0 plays the main role in this disaster application, coordinating the collaborative session and leading the make-decision process. He asks for the SSTAB operator (T1) to begin his simulation. After receiving a message from agent S2 indicating the end of its simulation, he asks for DYNASIM operator (T3) to begin his simulation. On receiving a simulation conclusion message from T3, he makes a decision based on the force values acting on mooring lines and risers. If he understands that these forces are extreme, he asks for T1 to begin all the process again, in order to find a new stable condition of the unit, and this loop continues until he is satisfied with the force values obtained. In this case, he makes contact with DM1, asking for his approval to their solution.

The basic conceptual level architecture of our collaborative application is shown in Figure 5b.

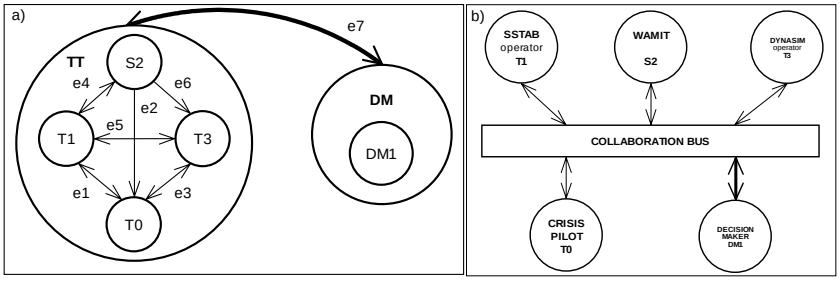

Figure 5. A first model of the disaster management collaborative application (a) and its prototype (b).

In order to map our model into an implementation-level architecture, we investigated different approaches, having in mind two main requirements: real-time support and open-source standard to develop prototypes. We chose HLA – High Level Architecture [12, 13], which not only fulfils our requirements but also is a flexible component-based architecture, in accordance to the principles we have been pursuing.

4. Conclusions and Future Work

We propose a multiple-perspective metamodel, which mixes Place-Centred and People-

Centred perspectives. It employs not a technology-driven but a human- and socially-centred approach. Associating pre- and post-communication processings to each of these levels, we could accommodate policy and privacy rules of organisations, even allowing inter-organisational work.

The metamodel allows flexibility in many dimensions. Separating high-level abstraction features from low-level implementation features allows the designer and the application developer to concentrate on their particular domain of expertise. Separating the computational program and the coordination program allows programmers to concentrate on coordination issues with high-level abstraction.

The metamodel is also customisable in the sense that it allows associating pre- and post-communication processings with each message sent. It allows parametric run-time changes such as changing names of pre- and post-communication processings in the message attributes table, or even changing the pre- and post-communication codes before they have been loaded during a collaborative session.

There is still a lot of work to do in order to make our metamodel a fully flexible and evolving collaborative architecture. For example, we should investigate how to promote our metamodel from a customisable category to an adaptable category [14], upgrading from the capability of adjusting parametric controls to the capability of reconfiguring its behaviour according to immediate patterns of use. We could accomplish this using a learning mechanism to monitor the users' activities.

References

[1] Q. Jones, S.A. Grandhi, L. Terveen, and S. Whittaker, People-to-People-to-Geographical-Places: The P3 Framework for Location-Based Community Systems, *Computer Supported Cooperative Work* **13** (2004), 249-282, Kluwer Academic Publishers.

[2] E.E.R. Russo, A.B. Raposo, T. Fernando, and M. Gattass, Workspace Challenges for the Oil & Gas Exploration & Production Industry, in *Proceedings of CONVR 2004 - 4th Conference of Construction Applications of Virtual Reality* (2004), 145-150.

[3] P. Dewan, Architectures for Collaborative Applications, in Beaudouin-Lafon (eds.), *Computer Supported Cooperative Work*, 1999, 169-194, John Wiley & Sons Ltd.

[4] Y. Laurillau and L. Nigay, Clover Architecture for Groupware, in *Proc. of CSCW´02* (2002), 236-245.

[5] A.N. Leontjev, *Activity, Consciousness and Personality*, Prentice-Hall, Englewood Cliffs, USA, 1978.

[6] T. Tuikka, Remote Concept Design from An Activity Theory Perspective, in *Proceedings of CSCW´02* (2002), 186-195.

[7] T. Gross and W. Prinz, Modelling Shared Contexts in Cooperative Environments: Concept, Implementation, and Evaluation, *Computer Supported Cooperative Work* **13** (2004), 283-303, Kluwer Academic Publishers.

[8] M. Cortés and P. Mishra, DCWPL: A Programming Language For Describing Collaborative Work, in *Proceedings of CSCW´96* (1996), 21-29.

[9] D. Li and R. Muntz, COCA: Collaboration Objects Coordination Architecture, in *Proceedings of CSCW´98* (1998), 179-188.

[10] 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. *8th Int. Conf. Stability of Ships and Ocean Vehicles - Stab* (2003), 377-388.

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

[12] IEEE 1516, The Institute of Electrical and Electronic Engineers, IEEE Std 1516-2000, *IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)-Framework and Rules* (2000).

[13] A. Kapolka, The Extensible Run-Time Infrastructure (XRTI): An Experimental Implementation of Proposed Improvements to the High Level Architecture, *Master's thesis*, Naval Postgraduate School, Monterey, CA, USA, 2003.

[14] P. Dourish, Using Metalevel Techniques in a Flexible Toolkit for CSCW Applications, *ACM Transactions on Computer-Human Interaction* **5, 2** (1998), 109-155.

CONFIGURING A COLLABORATIVE VIRTUAL WORKSPACE FOR DISASTER MANAGEMENT OF OIL & GAS OFFSHORE STRUCTURES

Enio Emanuel Ramos Russo [1], Alberto Raposo [2], Terrence Fernando [3], Marcelo Gattass [4], and Börje Karlsson [5]

ABSTRACT

There are serious risks involved in running offshore units, with many reported disasters. These disasters can not only cause deaths and important environmental impacts, but also have a strong impact on business. Oil & gas companies are thus continuously seeking to employ processes and technologies to respond to such events in order to ensure safety. Such processes involve collaboration among a large number of groups and resources from different natures and geographically distributed, in order to make appropriate decisions within a short period of time. These groups are comprised of many technical experts and decision makers such as naval engineers, structural engineers, risers analysts and oceanographers, as well as managers. They need to be in constant communication with operators inside the unit, divers, security team, and, perhaps, with experts who are travelling to execute the rescue plan.

This work investigates how a distributed workspace environment can support disaster management, involving distributed collaborative technical teams. We first identify the requirements for the distributed workspace, from the stakeholders involved in a disaster, and analyse the commercial emergency systems available. We then elaborate a multi-perspective metamodel to support configuring this collaborative virtual workspace. Finally a prototype for oil & gas offshore structures disaster management based on our multi-perspective metamodel is derived and an HLA-compliant implementation for this prototype is developed as a proof-of-concept of the metamodel.

KEY WORDS

collaborative virtual workspaces, distributed environments, HLA, decision making, oil & gas

[1] Systems Analyst, Tecgraf, Dept. of Computer Science, PUC-Rio, Rua Marquês de São Vicente 225, Rio de Janeiro, RJ, 22453-900, Brasil, Phone 55-21-2512-5984, enio@tecgraf.puc-rio.br

[2] Professor, Tecgraf, Dept. of Computer Science, PUC-Rio, Rua Marquês de São Vicente 225, Rio de Janeiro, RJ, 22453-900, Brasil, Phone 55-21-2512-5984, abraposo@tecgraf.puc-rio.br

[3] Professor, School of Construction and Property Management, University of Salford, Maxwell Building, The Crescent, Salford, M5 4WT, U.K., Phone 44-161-295-2914, t.fernando@salford.ac.uk

[4] Professor, Tecgraf, Dept. of Computer Science, PUC-Rio, Rua Marquês de São Vicente 225, Rio de Janeiro, RJ, 22453-900, Brasil, Phone 55-21-2512-5984, mgattass@tecgraf.puc-rio.br

[5] Computer Science Researcher, Tecgraf, Dept. of Computer Science, PUC-Rio, Rua Marquês de São Vicente 225, Rio de Janeiro, RJ, 22453-900, Brasil, Phone 55-21-2512-5984, borje@tecgraf.puc-rio.br

INTRODUCTION

There are serious risks involved in running offshore units, with many reported disasters. Companies can lose billion of dollars by losing an offshore unit and further billions of dollars due to the cease of the oil production. As a direct result of these huge accidents, the oil & gas companies usually take actions in two main directions: *(i)* one that has the objective of correcting and improving the operational procedures; and *(ii)* a second one that has the aim of planning a set of projects to improve the technological level of the company in order to minimize the risk of future accidents (Costa 2004).

Considering the second aspect and the necessity of minimizing disaster impacts, we verify the need to develop a system architecture capable of bringing people together to work as a virtual team to explore various rescue plans and work towards consensus.

Many companies have been creating virtual teams that bring together geographically dispersed workers with complementary skills, increasing the demand for CSCW (Computer Supported Cooperative Work) applications. In order to make the development of a wide range of these collaborative applications more effective, we should offer a general architecture that is adaptable to different situations, tasks, and settings in a flexible way. The motivation for this work has been the necessity of developing a collaborative virtual workspace for disaster management of oil & gas offshore structures for a global company (Russo et al. 2004).

The main aim of this work is to investigate how a distributed workspace environment can support disaster management, involving distributed collaborative technical teams. Specifically, this research will focus on a distributed workspace for technical groups to work as a collaborative virtual team to explore various simulation options and to communicate their results to the decision makers. This aim will be achieved through the following objectives: *(i)* to conduct a survey to identify the requirements for the distributed workspace, from the stakeholders involved in a disaster scenario; *(ii)* to elaborate a metamodel to configure collaborative virtual workspaces; and *(iii)* to define a distributed workspace environment based on this metamodel for the technical team to engage in the rescue efforts.

REQUIREMENTS GATHERING

Petrobras, Brazilian Oil & Gas Company, faced two major accidents in the beginning of this decade. In 2001, the largest semi-submersible platform in the world P-36 sunk, killing 11 employees and ceasing a daily production of 84,000 barrels of oil and 1.3 million cubic meters of natural gas. In 2002, the FPSO (Floating Production, Storage and Offloading) unit P-34 with a daily production of 35,000 barrels and a storage capacity of 58,000 m^3 of oil had a stability problem and almost sunk, immediately ceasing its operation. At this time, Petrobras managed to rescue the unit without loss of lives.

The requirements gathering for the distributed workspace has been obtained through Petrobras case studies P-36 and P-34. These case studies have been used to identify the roles and attributes of people involved in a typical disaster management operation. Structured interviews have also been carried out to identify procedures and the users' expectations about the collaborative workspace. In this type of environment, it is important to model the users' relationships and to identify the main collaborative features that the users would like to have.

Once the users' requirement capture phase was completed, the next step was to define the technical requirements in terms of collaboration models, simulation steering, personalised and global workspaces, synchronised viewing, video-streaming, etc. We then conducted a survey on the main commercial emergency management systems available to gather their main characteristics and the main features still underdeveloped.

EVOLUTION OF DISASTER MANAGEMENT IN PETROBRAS

This section illustrates the complexity of the problem in terms of processes and groups of people involved in such disaster incidents. From this discussion, we show that Petrobras has been continuously active in improving its disaster management program.

During the P-36 disaster, there was a mechanical explosion and a chemical explosion with loss of lives, which caused difficulty in acting quickly to save the unit. During the P-34 disaster, there was no explosion, enabling the teams to react quickly, although the communication among them could still be improved. This research aims to make the next step change in terms of using ICT (Information and Communication Technology) to improve the collaboration between the stakeholders involved in disaster incidents.

In the case of the P-36 disaster, Petrobras identified the need for updated emergency procedures and for executing the actions within a short period of time in order to save the unit. This case aroused the need to investigate collaborative and decision-making models to help complex teams in avoiding disasters. In the case of P-34, there was already an updated model of the offshore unit and a form of distributed working that did help the rescue team to act quickly. There was also a static simulator that allowed the specialists to run different simulations. Nevertheless, the team still did not have an adequate environment to work as a virtual team to share knowledge, jointly discuss possible rescue plans, and to work quickly towards consensus.

As a result, it was necessary to bring people together into the same physical location with some delay in the process. Furthermore, some of the information was not directly available to the decision makers. This incident showed the necessity to strengthen the collaboration among the distributed teams providing better interaction, simulation and discussion during the whole rescue operation.

DISTRIBUTED NATURE OF THE TEAMS AND THE RESOURCES

In the case of Petrobras, when an accident occurs, the head office is immediately contacted and the General Manager of the operational unit is in charge of crisis management. All the work will be under his control in the decision workspace. The Security, Environmental and Health Dept. then starts emergency procedures and at the same time the technical specialists begin to act. In the technical workspace, there are naval engineers, structural engineers, risers analysts and oceanographers. When working together in a collaborative way there are usually the following main distributed groups: *(i)* the high-level decision team at the operational unit; *(ii)* a task force group leading the make-decision process; *(iii)* a technical support team at the company headquarters, at the Business Unit, and at the research center; and *(iv)* mobile experts, who sometimes are overseas or travelling and who must also be connected.

In addition to these groups, and working together with them, there are security teams in rescue units which are moved towards the region of the accident and give help during all the crisis period.

Not only the experts, but also the system resources are distributed in this scenario. For example, the computer intensive simulators may have to remotely run on a super computer or on a cluster of computers to get quick results. Also, the environment may need access to remote databases which maintain CAD models and simulation models of the unit.

In terms of configurations, each site participating in the crisis solution can have different ones, such as a Virtual Reality Centre, an intranet desktop and a laptop connected to the network. Moreover, experts who are travelling may have to be linked via mobile technologies and the connection between the unit and the people on earth may vary.

COMMERCIAL EMERGENCY MANAGEMENT SYSTEMS

After having determined the collaborative disaster management workspace requirements, we conducted a survey on the main commercial emergency management systems available. We identified the main characteristics of those systems, the main areas already covered, what is the state-of-the-art and what are the main features which are still underdeveloped.

While performing this survey, existent Emergency Management Systems from some vendors were investigated: L-3 CRISIS Command and Control System (MPRI Ship Analytics 2003); Oil Spill Crisis Management Simulator, also from Ship Analytics; and U.S. Automated Resource Management System (ARMS) Systems Requirements Document (Booz Allen Hamilton 2003). Crisis Intervention methods – the Crisis Intervention and Operability (CRIOP) Analysis (Johnsen et al. 2004) – being practiced in companies such as Statoil, Norsk Hydro, Elf and BP, were also investigated.

From this survey, we concluded that most of the Emergency Management Systems have some common characteristics, such as: serving as an incident management as well as a training and planning tool; having capability of integration, not only with internal databases and systems, but also with public emergency management systems; normally providing a Geographical Information System (GIS), which is responsible for displaying real-time data of the incident; and providing logging and tracking capabilities of resources and activities, as well as checklists as an efficient method to address the multiple simultaneous requirements.

In spite of all the features listed above, we identified two main drawbacks of current Emergency Management Systems: *(i)* lack of suitable integration of simulators with high performance visualisation systems; and *(ii)* inadequate security and access control features.

The survey demonstrated that, in spite of the integration of most of the Emergency Management Systems with simulators, there is the need to develop a system architecture capable of supporting distributed resources, mainly distributed simulators running on *high performance* visualisation systems. This architecture should also provide synchronous communication among different equipments with virtual co-location as one feature.

The integration of simulators using high performance visualisation systems in a synchronous distributed environment is the aspect of the emergency scenario on which we are going to focus. In order to support the definition of the architecture of this environment, a metamodel will be elaborated.

A METAMODEL TO CONFIGURE COLLABORATIVE VIRTUAL WORKSPACES

CSCW applications have largely focused on issues concerning differences between: *(i)* co-located work and working across distance; or *(ii)* work with people from the same culture, or common ground, and work with people from different cultures. The previous perspectives have been named, respectively: Place-Centered and People-Centered (Jones et al. 2004). We propose to adopt a different view on the problem based on the activities carried out by the teams participating in the collaborative work. We name it an *Activity-Centered* perspective, which may be seen as a multi-perspective concept since it not only encompasses the Place-Centered and the People-Centered perspectives, but also allows adopting each one or both of them in a hybrid way, and admits seamless change from one perspective to another.

Nodes are essential components of our metamodel, going from the top-most node representing the whole activity through many nodes of different levels representing groups and sub-groups until the leaf nodes representing a person or a software agent. Nodes also have an attribute called *artefacts* defined as "all objects on which users can operate" (Gross and Prinz 2004). Examples of artefacts are drawings, physical models, prototypes, and documents. Following the class concept, an artefact associated with a group node is shared by all members in the group, unless otherwise explicitly stated. In this case, a mechanism such as an access control list will determine who share access to the artefact.

Edges in our metamodel represent the interaction paths among nodes, which can be uni- or bi-directed. When an edge is represented by a thin arrow, this means that the nodes on its extremities are co-located. When the arrow is thick, the nodes are placed remotely to each other. Edges have one important element, *channel*, which represents the electronically mediated channel that allows communication between two nodes.

We take an overall picture of the disaster management collaborative application (Figure 1) to illustrate the metamodel components. The disaster management of an oil & gas offshore structure is a complex operation involving several groups, such as the oil & gas company, the rescue team, the health care centre, the press, among others. This is an inter-organizational complex activity led by the oil & gas company, whose node will be detailed.

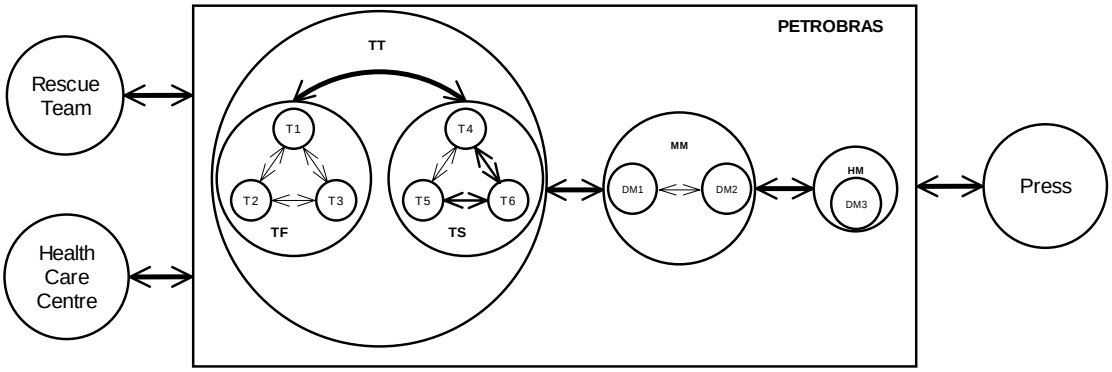

Figure 1: The disaster management collaborative application: overall picture

Within Petrobras node, we identify three main groups: the Technical Teams (TT), the Middle-level Managers (MM) and the High-level Managers (HM), each one remotely located to the other. TT is formed by two technical sub-groups: the Task Force (TF) team and the

Technical Support (TS) team, also remotely located.

TF plays the main role, leading the decision making process. It is constituted by three co-located technicians, such as naval engineers, structural engineers, risers analysts or oceanographers. TF runs different simulators to derive the best solution to save the offshore unit, permanently communicating with TS. They also maintain contact with MM informing about their work evolution and asking for approval for their derived solution. Once their solution is approved, they pass the sequence of commands to be executed to the unit operator (not represented in our picture).

TS team, with technicians working in the same fields as TF team, can be invoked by the latter to perform specialized simulations focusing on some particular issues that would not be possible to be done by TF, or to obtain another opinion about the problem.

MM is constituted by middle-level managers working co-located in a company office, with one of them usually being the responsible to make the final decision. They have an overall knowledge about the technical issues and work constantly interacting with the TT group. They also communicate with the HM group, informing about the work evolution and eventually when they need to make a more critical decision.

In our metamodel, we have also identified the need for additional *edge specialization elements*, namely *pre-* and *post-communication processing*, which are separated into two different classes. The first class is constituted by the ones directly associated with the leaf nodes. They represent the processing to be executed particularly onto a specific message being passed between two nodes. The second class is constituted by the ones associated with groups on different levels of the metamodel hierarchy, representing the policies of these groups when respectively sending (*out-policies*) and receiving (*in-policies*) messages. In Figure 2, we show possible pre- and post-communication processings that could be executed while sending a message from a Computer Science Researcher CR1 of the Computer Science Dept. CD1 of University U1 to Researcher CR2 of University U2.

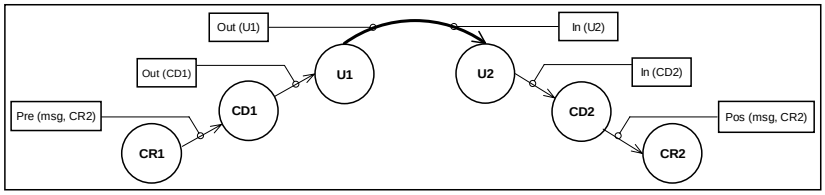

Figure 2: Activity-Centered metamodel: pre- and post-communication processings

According to the majority of CSCW studies (e.g., Cortés and Mishra 1996, Li and Muntz 1998), we adopted the strategy of separating the coordination structure and the computational program, using *role rules* with a logic-based specification language for specifying coordination policies. We also built a *message attributes table* to enhance the flexibility of the coordination program, separating coordination rules from data related specifically to each message. This table provides an indirection that enables dynamic reconfiguration.

PROTOTYPE

After investigating the activities involved in the disaster scenario and identifying their requirements in terms of ICT, we decided to concentrate on the Technical Teams group to

develop a prototype of collaborative application implementing a particular model of our Activity-Centered metamodel. This prototype is particularly related to the work performed by the Task Force group (TF), including the simulators they run, their mutual communication and their interaction with the Middle-level Manager group.

During a crisis situation, the Task Force group typically uses three simulators. The first simulator to be run is SSTAB (Coelho et al. 2003), the Floating Units Stability system, used to analyse the static conditions of the floating unit (Figure 3a). SSTAB uses as its inputs the unit model obtained from a centralized system and updated data from the unit obtained through a monitoring system. It gives as outputs five files, including the inertia matrix.

The second simulator is called WAMIT and uses as inputs the output files generated by SSTAB. It works in the frequency domain, deriving the excitation forces of the unit and water forces reactions to lateral displacement. WAMIT is activated by a user interface program called WMG.

Finally the third simulator to be executed is DYNASIM (Coelho et al. 2001), for Dynamic Stability (Figure 3b). It uses as inputs the results obtained from WAMIT as well as the parameters representing the *height* and the *period* of the waves at the moment of the disaster. DYNASIM calculates the forces acting on the mooring lines and risers. When these forces are considered extreme, a retrofeedback process is started, performing all the simulations again, beginning with SSTAB, to find another stable condition of the unit.

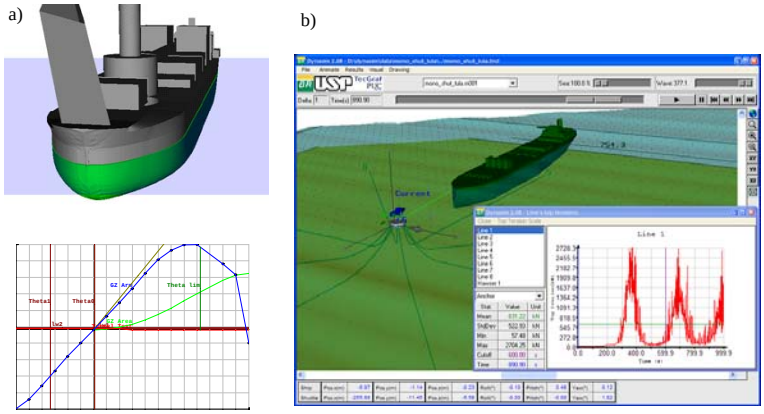

Figure 3: a) SSTAB; b) DYNASIM

The method used to save an offshore unit has the goal of defining a sequence of commands to be passed to the unit operators or to the rescue team so that they can move the unit in a step by step mode from its initial unstable condition until it reaches back its normal equilibrium state. It is based on the following workflow:

- We first use these three simulators to derive the initial conditions of the offshore unit.
- We then define a next step configuration of tanks (e.g., moving water from a ballast tank of one side to a ballast tank of the opposite side) and simulate the unit in this new condition using again the simulators. If we are not satisfied with the results, we define another configuration of tanks and continue this process, experiencing

iteractively configurations, until we are satisfied.

- From the configuration of the previous step, we now try to derive a new step configuration of tanks, using a process analogous to the one just described.
- We repeat this process of deriving step configurations of tanks using our three simulators until we reach back a normal equilibrium state.

At the end of this whole process, we have a sequence of commands in terms of tanks' valves operations, correspondent to the achievement of each of the step configurations described above, in a step by step mode, which was exactly our goal.

It is important to note that the executions of simulators SSTAB and DYNASIM are highly interactive visualisation processes, mainly in a crisis situation, when we need to rapidly experiment many alternatives to respond to the disaster. Also we have to consider that, in emergency situations, it is very important to be as fast as possible. Then, searching for points where we could save time, we found that, if WAMIT receives the results from SSTAB, it can be activated automatically on ending the SSTAB simulation.

An Activity-Centered model representing this crisis situation (Figure 4a) can be derived based on the participants' roles. We created two remote groups: Technical Teams (TT) and Decision Makers (DM). TT is constituted by the Task Force (TF) team with members T0, T1 and T3, and the software agent S2. DM is constituted by a single manager, a representative of all participants not directly involved with the technical part of the simulation activity such as operators and other managers, who only receive follow-up messages, commands to be executed or approval requests.

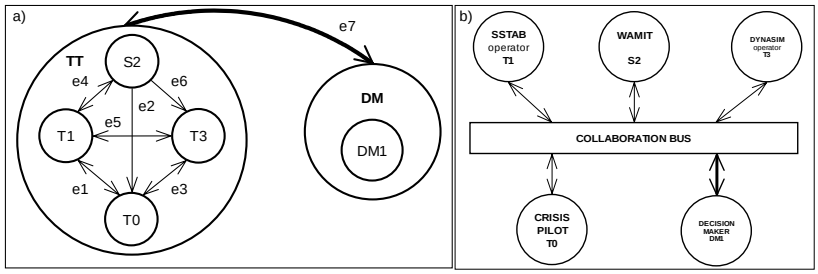

Figure 4: A first model of the disaster management application (a) and its prototype (b)

Other than the interaction network part of the model just described, we also define role rules and the message attributes table in order to represent the following workflow.

The Crisis Pilot T0 plays the main role in this disaster application, coordinating the collaborative session and leading the make-decision process. He asks for the SSTAB operator (T1) to begin his simulation. After receiving a message from agent S2 indicating the end of its simulation, he asks for the DYNASIM operator (T3) to begin his simulation. On receiving a simulation conclusion message from T3, he makes a decision based on the force values acting on mooring lines and risers. If he understands that these forces are extreme, he asks for T1 to begin the whole process again, in order to find a new stable condition of the unit, and this loop continues until he is satisfied with the force values obtained. In this case, he makes contact with DM1, asking for his approval to their solution. The basic conceptual level architecture of our collaborative application is shown in Figure 4b.

In order to map our model into an implementation-level architecture, we chose HLA – High Level Architecture (IEEE 2000), with real-time support and a flexible component-based architecture. The fundamental concepts in HLA are: *(i)* Federate – a simulation implemented as part of an HLA-compliant simulation; and *(ii)* Federation – a collection of federates working together. We use XRTI – The Extensible Run-Time Infrastructure (Kapolka 2003) as the HLA run-time infrastructure, an open-source and freely distributable implementation, written in Java and using XML object models. Among its basic characteristics we have: *(i)* a dynamic object model extension and composition support; *(ii)* a pure client-server topology in which federates only communicate with one another through the XRTI Executive, a server application; and *(iii)* Federates maintain two channels to the Executive: a TCP channel for reliable communication and a UDP channel for unreliable messaging.Observing the model of Figure 4a, we conclude that all participant members can constitute a single Federation. We then associate a Federate with each participant of this Federation. Each Federate code is a Java program built based on the workflow rules written in a logic-based program. To enhance flexibility, the main method of each Federate is the one named *process_role*, which receives as parameter the role to be played by the Federate, coded in a separate Java module. Using this strategy, we can code the workflow rules associated with a specific role directly into a separate module dedicated to this role.

CONCLUSIONS

This work was motivated by and was conducted in real-world settings, namely an oil & gas offshore structure disaster scenario. This seems to contribute to the CSCW field, since a review of CSCW evaluation studies concluded that less than half were conducted in real-world settings (Pinelle 2000). An adequate model to the disaster scenario was derived from our multi-perspective metamodel. We also implemented a first prototype as a proof-of-concept of our metamodel, using an HLA run-time infrastructure.

The metamodel allows flexibility in many dimensions. Separating high-level abstraction features from low-level implementation features allows the designer and the application developer to concentrate on their particular domain of expertise. Separating the computational program and the coordination program allows programmers to concentrate on coordination issues with high-level abstraction.

The metamodel is also customisable in the sense that it allows associating pre- and post-communication processings with each message sent. It allows parametric run-time changes such as changing names of pre- and post-communication processings in the message attributes table, or even changing the pre- and post-communication codes before they have been loaded during a collaborative session.

There is still a lot of work to do in order to make our metamodel a fully flexible and evolving collaborative architecture. Particularly to the situation of an emergency scenario being considered, it would be very important to include an Expertise Recommender system, such as the one proposed by McDonald and Ackerman (2000), since in a crisis situation it is fundamental to locate the expertise necessary to solve the problem in the lesser possible time. We should also investigate how to promote our metamodel from a customisable category to an adaptable category (Dourish 1998), upgrading from the capability of adjusting parametric controls to the capability of reconfiguring its behaviour according to immediate patterns of

use. We could accomplish this using a learning mechanism to monitor the users' activities.

REFERENCES

Booz Allen Hamilton (2003). "Automated Resource Management System (ARMS), System Requirements Document (SRD)." *Contact No.: GS-35F-0306J, FEMA BPA # EMV-2001-BP-0147, Task Order 11*, Federal Emergency Management Agency, USA.

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

Coelho, L.C.G., Nishimoto, K., and Masetti, I.Q. (2001). "Dynamic Simulation of Anchoring Systems Using Computer Graphics." *OMAE Conference*.

Cortés, M. and Mishra, P. (1996). "DCWPL: A Programming Language For Describing Collaborative Work." *Proceedings of CSCW´96*, 21-29.

Costa, T.V.M. (2004). *Modelos de Aprendizado Organizacional de Chris Argyris: Uma Investigação em Segurança Operacional e Conservação do Meio Ambiente (SCA) com Base em Acidentes da Indústria do Petróleo*. D.Sc. Thesis, Production Engineering, Federal University of Rio de Janeiro (UFRJ), 159 pp.

Dourish, P. (1998). "Using Metalevel Techniques in a Flexible Toolkit for CSCW Applications." *ACM Transactions on Computer-Human Interaction*, 5 (2) 109-155.

Gross, T. and Prinz, W. (2004). "Modelling Shared Contexts in Cooperative Environments: Concept, Implementation, and Evaluation." *Computer Supported Cooperative Work*, Kluwer Academic Publishers, 13 283-303.

IEEE – The Institute of Electrical and Electronic Engineers (2000). *IEEE Std 1516-2000, IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)-Framework and Rules*.

Johnsen, S.O., Bjørkli, C., Steiro, T., Fartum, H., Haukenes, H., and Skriver, J. (2004). "CRIOP®: A scenario method for Crisis Intervention and Operability analysis." *Report No. STF38 A03424*, SINTEF, Norway.

Jones, Q., Grandhi, S.A., Terveen, L., and Whittaker, S. (2004). "People-to-People-to-Geographical-Places: The P3 Framework for Location-Based Community Systems." *Computer Supported Cooperative Work*, Kluwer Academic Publishers, 13 249-282.

Kapolka, A. (2003). *The Extensible Run-Time Infrastructure (XRTI): An Experimental Implementation of Proposed Improvements to the High Level Architecture*. Master's thesis, Naval Postgraduate School, Monterey, CA, USA.

Li, D. and Muntz, R. (1998). "COCA: Collaboration Objects Coordination Architecture." *Proceedings of CSCW´98*, 179-188.

McDonald, D.W., and Ackerman, M.S. (2000). "Expertise Recommender: A Flexible Recommendation System and Architecture." *Proceedings of CSCW´00*, 231-240.

MPRI Ship Analytics (2003). *L-3 CRISIS Brochure*. North Stonington, Connecticut, USA.

Pinelle, D. (2000). "A Survey of Groupware Evaluations in CSCW Proceedings." *Technical Report HCI-TR-2000-01*, Computer Science Dept., University of Saskatchewan, Canada.

Russo, E.E.R., Raposo, A.B., Fernando, T., and Gattass, M. (2004). "Workspace Challenges for the Oil & Gas Exploration & Production Industry." *Proceedings of CONVR 2004 - 4th Conference of Construction Applications of Virtual Reality*, 145-150.

Workspace Challenges for the Oil & Gas Exploration & Production Industry

Enio Emanuel Ramos Russo
PUC-Rio
Rio de Janeiro - Brazil
enio@inf.puc-rio.br

Alberto B. Raposo
PUC-Rio
Rio de Janeiro – Brazil
abraposo@tecgraf.puc-rio.br

Terrence Fernando
The University of Salford
Salford – United Kingdom
t.fernando@salford.ac.uk

Marcelo Gattass
PUC-Rio
Rio de Janeiro - Brazil
mgattass@inf.puc-rio.br

Abstract
The objective of this paper is to present some of the key challenges faced when defining and building virtual workspaces for oil & gas Exploration & Production (E&P) activities. First, we present the main E&P processes that can benefit from the VR technology. Secondly, we classify and describe the different challenges.

Keywords
virtual environments, virtual workspaces, oil & gas, E&P processes.

1. INTRODUCTION

The oil & gas industry has been a leading player in exploiting the power of virtual reality technology to enhance its business processes. The motivation for deploying such advanced technology in this industry is due to the difficulties that the companies were facing in the late nineties, with the price of oil hovering near all-time lows. At that time, the pressure to reduce exploration and development costs of new reserves and existing fields was immense and the immersive virtual reality technology was identified, by the oil & gas industry, as one of the key tools which can meet these challenges. The Virtual Reality Centres (VRCs), large projection rooms with features such as 3D and stereoscopic images, soon became very popular in the oil & gas industry, since they gave specialists the ability to quickly and comprehensively interpret large volumes of data, thus significantly reducing cycle time for prospect generation [American98].

However, due to ever increasing business pressures, there are further demands on researchers to extend the capabilities of the VR technologies, so that it can be fully utilised in all the oil & gas exploration and production (E&P) phases. This paper presents various E&P processes of the oil and gas industry and discusses research challenges emerging from these processes.

The structure of this paper is as follows. Initially, Section 2 presents the key E&P processes and their application demands. Section 3 presents the classes of technology challenges emerging from these E&P processes for VR. The final conclusion of this paper is presented in Section 4.

2. TYPES OF E&P PROCESSES

This section discusses the main processes of the oil & gas E&P industry and the main challenges within each process. The work presented here is based on the authors' experience with oil & gas projects at Petrobras in Brazil.

Figure 1 shows the main resources involved in the production of oil & gas. The typical E&P processes in the oil & gas industry are: (i) reservoir exploration through 3D geomodelling and seismic interpretation; (ii) design and construction of the production facilities based on the results of the first phase; and (iii) production and transportation of the produced oil & gas.

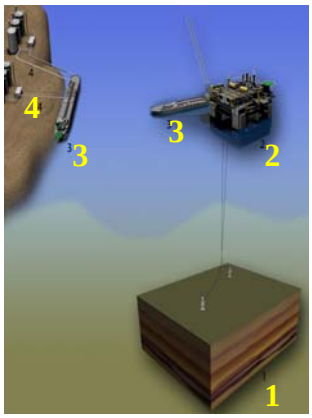

Figure 1: (1) reservoir; (2) offshore platform; (3) transportation ships; (4) oil pipelines.

The following subsections describe how virtual reality can enhance each of these E&P phases.

2.1 Reservoir Exploration Phase

During this exploration phase, the goal is to elaborate the subsurface models that best represent the reservoirs.

Whether it is a seismic cube or a stratigraphic geological model, what is important in this phase is to build an individual mental representation of the model. Therefore the key tasks in this phase are 3D geomodelling and 3D seismic interpretation.

Drilling wells for crude may consume up to 85% of the total exploratory funds. Thus, the decision to drill should be taken in a sensible way based on studies that provide detailed knowledge of the area's geologic conditions, both on the surface and in the subsurface. Of all such studies, the seismic method is more decisive to select the drilling spots. Seismography makes use of subsurface ultrasonography, generating seismic logs that provide an approximate image of the configuration of several subsurface strata.

3D geomodelling involves a large spectrum of skills, spread over different domains (geophysics, geology, reservoir and petroleum engineering). During its lifetime, a numerical earth model is shared by people with different types of specializations. The model evolves continuously over time, by absorbing various inputs from the team members [Reis01].

Seismic interpretation in the late seventies used to be made over a stack of paper maps, from which the interpreter would pinpoint areas of interest for drilling by creating a mental 3D image about thickness, constitution, depth and performance of rock beds. However, the advent of VRCs and stereoscopic images opened a door to a new world for seismic interpretation, allowing the users to visualise and explore in an interactive manner. The work became much more easier since specialists no longer need to use their knowledge and imagination to draw a mental picture of the area and to feel immersed in it. A mapping that used to take months began to be drawn in just a few hours [Petrobras99].

The viability to use 3D imaging fosters a more accurate and faster interpretation of the external geometry and internal architecture of reservoirs. With all the participants of a project having access to the same shared viewing, one can have a better interpretation of a large pile of data, achieving more reliable simulations of the oil output performance of that reservoir and analysis of its results. The team can calculate curves for future production, forecast the number of wells for drilling and devise the whole project for an oilfield development [Petrobras99].

The images can be studied until specialists are able to determine the best way to exploit the reservoir they represent. Well location, rock qualities and the distribution of well fluids (water, gas and oil) can be analysed more efficiently with the purpose of ascertaining the best distribution patterns for production and injection wells [Petrobras01].

One of the current key challenges in this area is the development of collaborative workspaces for supporting truly collaborative geomodelling and visualization for distributed users. Given the geographical dispersion of experts in the oil & gas E&P industry, remote collaboration offers great benefits, particularly in activities involving continuous model refinement and decision taking.

Another challenge is to develop better interaction facilities with real-time performance to explore the seismic data in a more interactive manner. This requires both heavy processing power and intuitive interfaces designed for team work. Two approaches could be explored in providing the computer power necessary for real-time seismic data exploration: PC cluster approach and computational steering from super-computers. Although both approaches are sensible from the research point of view, the PC cluster approach seems to be favoured by many due to the cost factor. The interfaces for controlling the simulation and visualisations, generated by these computers, need to be enhanced to provide natural interaction. The deployment of emerging technologies such as wireless tracking, PDAs, gesture-based interaction to develop natural interfaces for the team collaboration is still a challenging research problem.

Real-time follow-up and correction of the course of a deepwater horizontal well is one of the activities that can also take advantage of the VRCs' features. Although this technology may be used in any kind of well, its potential is clearly shown in horizontal drilling in the need to navigate the reservoir as it is drilled. Mostly in the early stages of the oil field's development, the reservoir may not always be found as forecast and as a result a well of about US$ 20 million may be lost. One of the challenges is to explore the use of optic fibre cables, connected to a VRC, to monitor the real-time drilling to make sure the rig will hit its target and will not skip the reservoir [Petrobras99]. This application obviously requires real-time features of the virtual reality system, as usually rig information is sent from the field at regular time intervals.

2.2 Design and Construction Phase

During the design phase, the oil & gas industry is interested in visualizing offshore structures, performing static and dynamic simulations of these structures to ensure their stability, examining the construction processes, analysing procedures for monitoring oil pipelines and emergency situations etc. The construction phase will only be executed, once these issues are fully analysed to the satisfaction of all the stakeholders.

2.2.1 Reviewing the construction process

Offshore structures, modelled using CAD systems, have every single component highly detailed, since the goal here is to analyse the construction process.

The engineers need not only to have access to every single part of the model and its characteristics, but also to review the model from different perspectives. Therefore the key challenge in this process is to develop a dynamic virtual environment to allow the designers to assess the construction of the offshore structures from their own perspectives. This will require a flexible software framework which can provide access to various simulations with personalised interfaces.

The installation of subsea equipments is a challenging task during the construction of offshore structures, requiring precise manipulation inside a complex environment. This requires highly skilled people to ensure the operations can be done efficiently without damaging the surrounding equipments. The challenge here is to enhance the current capabilities of the VR technology to allow engineers to rehearse such intricate operations in advance to avoid costly mistakes. The use of robots is also being investigated to conduct such operations remotely.

2.2.2 Stability analysis

Thorough analysis of stability of the offshore structures is an important aspect to be considered during the design phase, where thousands of barrels of oil are produced daily in the open sea. The stability analysis needs to take into account the stress conditions, sea currents, waves and wind pressures on semi-submersible platforms and FPSOs (Floating Production, Storage and Offloading unit). Additionally, these production units may be floating in the sea, which is more than two thousand meters deep, and therefore requiring the deployment of complex mooring systems.

Most of the current simulators are still static [Coelho03], but the demand for dynamic simulations is growing in the oil & gas industry to conduct rich simulation of offshore structures to ensure safe operation. Examples of such dynamic simulators are Dynasim [Coelho01] and NOT (Numerical Offshore Tank) from Petrobras. The Dynasim system has been designed to compute the supervening forces and consequential movements on anchored structures, where as NOT has been designed to simulate waves, currents, the line dynamics and the damping of floating production and storage oil & gas systems. The key challenge which is being explored in this work is the deployment of massively parallel computing with PC clusters to support interactive visualisation and simulation. Another key challenge in this area is the deployment of such dynamic simulations to give designers and engineers the feeling of the movements suffered by the unit, using hardware simulators. Such a simulator could be used for assessing various issues in operation, maintenance and emergency scenarios.

2.3 Production Phase

The main aim of this phase is to support efficient and safe production of oil & gas. This requires putting in place a well trained work force for operation, plant monitoring, maintenance, emergency handling etc. This section discusses how virtual reality technology could be used for supporting these key activities.

However, the application of VR in this phase requires an up to date virtual model of the plant. As a result, any changes to the plant need to be captured and be used to maintain a valid virtual representation of the real plant. This could be done by means of a 3D laser scanner that is capable of acquiring a cloud of points from the real structure. This section describes few examples to illustrate the use of VR in the production phase.

2.3.1 Monitoring

During the production phase, the virtual reality technology has the potential for supporting better monitoring of plants. Examples of such monitoring tasks include remote monitoring of oil pipelines to avoid oil spillages, stability of the offshore structures and off loading operations.

To better analyse oil pipeline deformations, it is possible to use post-videos over the structural analysis results. Also the manager or the specialist could be allowed to receive a visual representation of the oil pipeline directly from the field, in case of an accident or during a maintenance operation. However, in order to transmit data from the field to the expert's virtual workspace, the equipments used by the field engineers must not be heavy and should be based on mobile technology to work on difficult terrain conditions.

2.3.2 Emergency scenario

The importance of rigorous procedures for handling emergency situation is now becoming extremely important due to ever growing environmental concerns. An oil spillage could have a devastating consequence on the environment costing millions of dollars to constrain the damage. Virtual reality can play a significant role in developing systems for training people for handling such situations and also for connecting experts during such a disaster situation to advise the workers, on the ground, to control the situation.

One such system, which has been developed to manage and control actions during a leakage of a pipeline is InfoPAE [Carvalho02]. It provides facilities to manage conventional and geographical data, associating them with the plans. The system has been developed to help and minimise the response time, to validate and optimise the emergency plan's logic and to train the teams responsible for the actions.

Another typical emergency scenario in the oil & gas area is a crisis situation in an offshore structure, when the structure becomes unstable. In this scenario, there are two main possibilities:

- If the unit is heavily damaged and has security problems, then the unit is abandoned and no person remains inside the offshore platform. In such a situation, divers are called to do possible rescue operations.
- If the unit has a minimal security condition, it usually remains with two or three operators. In such a situation, operators receive instructions from the experts on the ground to stabilise the offshore structure.

During such emergency situations, several expertises are brought together to provide advise. Typically the specialists involved are naval engineers, structural engineers, risers analysts and oceanographers. These specialists are geographically distributed and therefore in need of an efficient IT environment to support the collaborative decision making process.

3. CLASSES OF WORKSPACE CHALLENGES

In order to develop usable industrial solutions, it is important to first identify and analyse the industrial processes and the requirements and expectations from the specialists. Such an analysis for the oil & gas industry was presented in Section 2 in this paper.

From this analysis, it is apparent that the oil & gas industry needs a suit of virtual workspaces for supporting various tasks such as seismic exploration, design reviews including dynamic simulation of offshore structures, training environments for subsea offshore equipment installation and disaster management, monitoring of real-time follow-up and correction of the course of a deepwater horizontal well, decision making environment for emergency situations, monitoring environments for oil pipelines, offshore structures and off loading operations etc. When constructing virtual workspaces for these applications, great care must be given to the user's expectations, appropriate collaboration operations, interaction metaphors, appropriate display environments and visualisation techniques to suit the tasks and the expert teams. Since most of these workspaces will be used by multi functional teams, it is important to deal with different levels of perception and perspectives that users are expecting to conduct their tasks. The VR technology used for building such workspaces should fit the user in terms of intuition, attention and productivity [Parkin99].

Although each application requires specific functionality and interfaces, the following generic classes of virtual workspaces, for the oil & gas industry, can be identified from the analysis given in Section 2:

- Distributed Design Review Workspaces.
- Co-located Design Review Workspaces.
- Field Activity Monitoring Workspaces.
- Disaster Management Workspaces.
- Training Workspaces.

The following subsections discuss the generic technology challenges faced when building these generic and specific workspaces.

3.1 Real-time Visualization and Interaction

A common characteristic of a typical virtual workspace, constructed for supporting an E&P process, is the enormous amount of data it has to deal with. The type of data could vary from seismic data for reservoir exploration to CAD and simulation data for the design and construction of offshore structures. For tasks such as monitoring of oil pipelines over a mountain, GIS, CAD and video data need to be brought together to support the construction of the monitoring virtual workspaces. Such scenes could be out door environments or complex structures (offshore structures) with different spatial characteristics. Furthermore, the data produced for supporting certain design functionality may not have an efficient representation for achieving the best visualisation performance, requiring certain pre-processing techniques.

The challenge here is how to decide what part of the data to visualize at each time. This is not only because of performance and real-time constraints but also to avoid cluttering the scene with unnecessary data. Therefore model simplification algorithms which do not eliminate key features of the structures are important to provide usable real-time visualisation services for E&P processes. In addition, real-time performance for visualising such large data sets needs to be gained by utilising the power of specialised hardware solutions or PC clusters.

In these virtual workspaces, the construction of interfaces for supporting specific activities for specific experts is extremely important to achieve user acceptance of the technology. Such interfaces need to be natural and simple without requiring any training. Although some advanced interaction technologies are now becoming available, it is important to research and build simple interfaces appropriate for a task. In [Froehlich99], Froehlich claims that the geoscientists found the Cubic Mouse, an input device specially tailored to geo-scientific data, is very natural and effectively performs their tasks. Its interesting characteristics are the sensation it gives the user of having the whole model in his hand and the possibility of easily moving through 2D slices of the model by simply sliding small bars of the cube. It allows the users to focus on their exploration tasks rather than on operating the computer. Further research is required to identify interface devices and paradigms for supporting natural interaction within E&P virtual workspaces.

3.2 Collaboration

One of the most important challenges in constructing oil & gas E&P virtual workspaces is the development of efficient collaborative virtual environments (CVE). This is because most of the projects involve many professionals who are geographically dispersed over a country or even across different countries, who need to work together as a virtual team. These cross-functional team members need to collaborate effectively and make decisions quickly and accurately to support various stages of the E&P process.

3.2.1 Distribution support

For a virtual environment to be collaborative, it must be distributed between the participants who wish to share it. The choice between communication architectures is parameterized by the degree to which the data structures representing the virtual environment are replicated or cached between the computing nodes and the underlying transportation technology [West01].

However, whatever the technology, communication latencies are an important factor in building usable collaborative systems. If it is not possible to achieve the adequate synchrony, one solution is to at least focus resources upon those activities which are most sensitive to lag, i.e. those which produce the most pronounced discontinuities of perceptual experience when lag is present. For the moment, it is fair to say that there is no universal choice of distribution or communication architecture, but

rather a range of trade-offs in performance and deployment issues [Singhal99].

It is impossible to predict the network requirements of CVEs in isolation; rather, we need a model of CVE operation which encompasses the application, user, software and hardware concerns. In this paper we follow the model proposed by Greenhalgh [Greenhalgh01], which has six layers:

1. Task/application/collaboration requirements: what do people want or try to do? For each virtual workspace, it is important to identify the exploration or design tasks that the user is expecting to perform.
2. User behaviour: what particular actions do people do and when? For example, if users speak only rarely, and never at the same time, then the network requirement for audio could be very limited. On the other hand, for some scenarios, there must be enough bandwidth for every user to speak at the same time. This could be the case of the emergency scenario, for example.
3. Process behaviour: how does the application respond? Once again, the emergency scenario could be a good example: while people heading the whole operation could execute any command, the other specialists could only execute the tasks they were asked to.
4. Distribution architecture: what communicates with what? The choice of distribution architecture determines which information must be communicated to which parts of the system. Typically, communication will be necessary between both people and simulators. Typically oil & gas applications nowadays are held in no more than a half dozen visualization rooms simultaneously, with no more than 20 specialists in each one. In the case of dynamic simulation of offshore structures, simulators performing various analyses need to communicate their data to each other and/or to a central controller to produce the final results of the simulation.
5. Communication protocols: how is information exchanged? Protocols can be either unicast or multicast or a combination to achieve both performance and reliability.
6. Network communication: what actually happens in the network? In the particular case of oil & gas applications, as presented in Section 2, it is not unusual to have one or more specialists out in the field who need to be somehow connected to the collaboration environment. This could be done by a mobile system. Therefore it is important to support both fixed and mobile communication for E&P workspaces.

Due to the high commercial value of their data, oil & gas industry has imposed strict data base consistency and security requirements. As a result, the data is typically at various sources which need to be brought together to support innovative virtual workspace concepts discussed in this paper. The grid concept seems to match those requirements, since it is conceptually centralized with real data, distributed at various places transparently to the application.

3.2.2 Collaboration metaphors

While it is important to develop a flexible and open software platform for supporting collaboration, the human factors issues for supporting tighter interaction between the team members should not be ignored. Due to space limitations, only few interaction considerations important for supporting collaborative working within the E&P workspaces are summarised below:

- In some cases, experts would need the possibility of having a copy of the data model in their private workspaces to explore their ideas individually, and to take their views to the shared workspace for discussion. Such a facility is important for applications such as modelling or interpreting an oil reservoir or dealing with an emergency scenario.
- During collaborative discussions or training, it is important to control and share various viewing points to communicate ideas to each other. Some key viewing support necessary within collaborative working could be summarised as: (i) sharing of each other's viewing point (look over the other's shoulder) [Cheng98]; (ii) mirrored viewing point (the opposite side of the situation). Furthermore, in some emergency training situations, the trainees may want to observe the simulation result from various view points in parallel. For example, one might want to observe the simulation effect of a possible emergency operation using an exocentric point-of-view (outside in) and another may want to observe the simulation effect using an egocentric point-of-view (inside out). Such parallel observation could lead to better understanding of the emergency situation and to work learn to as a team.

The next generation of collaborative workspaces will provide much more realistic face-to-face tele-immersive environments, integrated with appropriate simulations and data bases [Johnson01]. Such mixed-reality workspaces, created by combining virtual workspaces and video avatars of users, have the potential for mimicking co-located meeting metaphors. However, the human factors issues, performance issues and business benefits of such environments will need to be addressed properly to ensure their acceptance by the oil & gas industry.

4. CONCLUSIONS

This paper discussed E&P processes of the oil & gas industry with the view to identifying how VR technology can be used to build better virtual workspaces for these processes. Several generic virtual workspaces were identified which are specific for the oil & gas industry. Finally the paper presented some of the generic technology challenges in building virtual workspaces for the oil & gas industry.

This paper emphasised the need for developing virtual workspaces with a thorough understanding of the processes and the user expectations to ensure their acceptance

by the oil & gas industry. Furthermore, the paper argued that the interfaces of these virtual workspaces need to be mapped onto the roles and the tasks of the users.

However, the construction of virtual workspaces for every possible application and various users can be a tedious and expensive task. Therefore it is important that future research lays foundation for creating reconfigureable and dynamic software architectures to facilitate easy construction of various virtual workspaces on demand.

5. ACKNOWLEDGEMENTS

We would like to thank Petrobras and Tecgraf/PUC-Rio for their support, and, for their great contribution, Petrobras' engineers Álvaro Maia, Heitor Araújo and Isaias Masetti, and Prof. Markus Endler and engineer Luiz Cristovão Gomes Coelho from PUC-Rio.

6. REFERENCES

[American98] *The American Oil& Gas Reporter* (March 1998).

[Carvalho02] Carvalho, M.T., Casanova, M.A., Torres, F. and Santos, A.: INFOPAE – An Emergency Plan Deployment System. *Proceedings of the Int. Pipeline Conference* (2002).

[Cheng98] Cheng, D.Y.: Design of a Virtual Environment that Employs Attention-Driven Interaction and Priorization. *Proceedings of the 4th Eurographics Workshop on Virtual Environments* (1998), 114-123.

[Coelho01] Coelho, L.C.G., Nishimoto, K. and Masetti, I.Q.: Dynamic Simulation of Anchoring Systems Using Computer Graphics. *OMAE Conference* (2001).

[Coelho03] Coelho, L.C.G., Jordani C.G., Oliveira, M.C. and Masetti, I.Q.: Equilibrium, Ballast Control and Free-Surface Effect Computations Using The Sstab System. *8th Int. Conf. Stability of Ships and Ocean Vehicles - Stab* (2003), 377-388.

[Froehlich99] Froehlich, B., Barrass, S., Zehner, B., Plate, J. and Goebel, M.: Exploring GeoScience Data in Virtual Environments. *Proc. Visualization 99* (1999).

[Greenhalgh01] Greenhalgh, C.: Understanding the Network Requirements of Collaborative Virtual Environments. *Collaborative Virtual Environments*. London: Springer, 2001.

[Johnson01] Johnson, A. and Leigh, J.: Tele-immersive Collaboration in the CAVE Research Network. *Collaborative Virtual Environments*. London: Springer, 2001.

[Parkin99] Parkin, B.: The Human Sphere of Perception and Large-Scale Visualization Techniques. *Conference Guide of the 1999 High Performance Visualization and Computing Summit Oil & Gas*, Silicon Graphics, 15.

[Petrobras99] *Petrobras Magazine*, 7, 26 (1999), 20-23.

[Petrobras01] *Petrobras Magazine*, 7, 33 (2001), 14-17.

[Reis01] Reis, L.P., Bosquet, F. and Paul, J.C.: Toward collaborative geomodeling. *Proceedings of the 21st GOCAD Meeting* (2001).

[Singhal99] Singhal, S. and Zyda, M.: *Networked Virtual Environments: Design and Implementation*. New York: ACM Press, 1999.

[West01] West, A. and Hubbold, R.: System Challenges for Collaborative Virtual Environments. *Collaborative Virtual Environments*. London: Springer, 2001.

Emergency Environments for the Oil & Gas Exploration and Production Industry

Enio Russo Petrobras
Rio de Janeiro Brazil

Alberto Raposo PUC-Rio
Rio de Janeiro Brazil

Terrence Fernando The University of Salford
Salford – Greater Manchester United Kingdom

Marcelo Gattass PUC-Rio
Rio de Janeiro Brazil

Abstract:

The objective of this poster is to present some of the key challenges faced when defining and building virtual workspaces for oil & gas Exploration & Production (E&P) activities, such as 3D geomodelling, seismic interpretation, real-time drilling follow-up and correction, offshore structures' design, static and dynamic simulations of these offshore structures, oil pipelines' monitoring and emergency situations' handling. Also a case study focusing on emergency scenarios with extreme conditions is discussed in details.

---

1. INTRODUCTION

The oil & gas industry has been a leading player in exploiting the power of virtual reality technology to enhance its business processes. The motivation for deploying such advanced technology in this industry is due to the difficulties that the companies were facing in the late nineties, with the price of oil hovering near all-time lows.

The Virtual Reality Centres (VRCs), large projection rooms with features such as 3D and stereoscopic images, soon became very popular in the oil & gas industry, since they gave specialists the ability to quickly and comprehensively interpret large volumes of data, thus significantly reducing cycle time for prospect generation [American98].

Petrobras built the first Latin America VRC in its R&D centre (CENPES) in 1998. The idea was to test moderate-priced configurations, show the benefits of this technology and encourage the installation of similar VRCs at other operational units of Petrobras.

Another VRC was built in the company's headquarters in 1999 and now Petrobras has already ten VRCs being used all over the country, including a holo-space installed in its headquarters.

However, due to ever increasing business pressures, there are further demands on researchers to extend the capabilities of the VR technologies, so that it can be fully utilised in all the oil & gas exploration and production (E&P) phases.

2. TYPES OF E&P PROCESSES

This section discusses the main processes of the oil & gas E&P industry and the main challenges within each process. The work presented here is based on the authors' experience with oil & gas projects at Petrobras in Brazil.

2.1 Reservoir Exploration Phase

During this exploration phase, the goal is to elaborate the subsurface models that best represent the reservoirs. Whether it is a seismic cube or a stratigraphic geological model,

what is important in this phase is to build an individual mental representation of the model. Therefore the key tasks in this phase are 3D geomodelling and 3D seismic interpretation.

2.1.1 3D geomodelling

3D geomodelling involves a large spectrum of skills, spread over different domains (geophysics, geology, reservoir and petroleum engineering). During its lifetime, a numerical earth model is shared by people with different types of specializations. The model evolves continuously over time, by absorbing various inputs from the team members [Reis01], as shown in Figure 1.

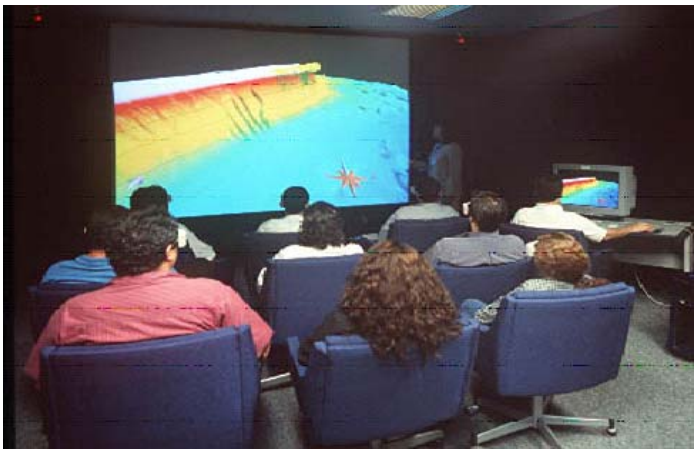

Figure 1: Team members discussing a 3D geological model.

The main software that have been used in 3D geomodelling until now include GOCAD, Landmark, Schlumberger and Earth Vision from Dynamic Graphics. Petrobras is already using synchronized viewing of the earth model among different specialists.

2.1.2 3D seismic interpretation

The advent of VRCs and stereoscopic images opened a door to a new world for seismic interpretation, allowing the users to visualise and explore in an interactive manner. The work became much more easier since specialists no longer need to use their knowledge and imagination to draw a mental picture of the area and to feel immersed in it. A mapping that used to take months began to be drawn in just a few hours [Petrobras99].

In terms of software, several geophysical visualisation programs have been developed, namely, in-depth Reverse Time Migration, 2D and 3D acoustic and elastic seismic modelling, and seismic volume visualisation [Silva03], such as the one seen in Figure 2.

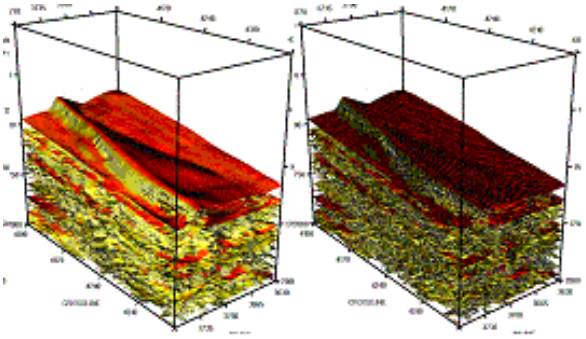

Figure 2: Seismic volume visualization with different methods.

2.1.3 Real-time follow-up and correction of the course of a deepwater horizontal well

Mostly in the early stages of the oil field's development, the reservoir may not always be found as forecast and as a result a well of about US$ 20 million may be lost. One of the challenges is to explore the use of optic fibre cables, connected to a VRC, to monitor the real-time drilling to make sure the rig will hit its target and will not skip the reservoir [Petrobras99].

Petrobras is already using this technology in the exploration phase, including an in house development known as gWLog [Campos02].

2.2 Design and Construction Phase

During the design phase, the oil & gas industry is interested in visualising offshore structures, performing static and dynamic simulations of these structures to ensure their stability, examining the construction processes, analysing procedures for monitoring oil pipelines and emergency situations etc.

2.2.1 Reviewing the construction process

The engineers need not only to have access to every single part of the model and its characteristics, but also to review the model from different perspectives.

When dealing with virtual reality applications over these types of data, the focus is in visualising and walking through the facility with good performance and sufficient realism. It is necessary to treat the data before visualising them.

For this purpose, many recent works have been developed searching for efficient solutions for the conversion of CAD models to VR models. An example is the ENVIRON tool [Corseuil04], shown in Figure 3.

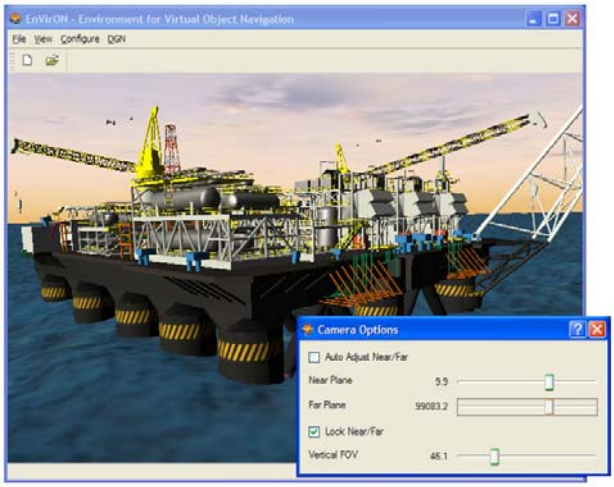

Figure 3: ENVIRON screenshot.

2.2.2 Stability analysis

The stability analysis needs to take into account the stress conditions, sea currents, waves and wind pressures on semi-submersible platforms and FPSOs (Floating Production, Storage and Offloading unit). Additionally, these production units may be floating in the sea, which is more than two thousand meters deep, and therefore requiring the deployment of complex mooring systems.

Most of the current simulators (Figure 4) are still static such as Sstab [Coelho03], but the demand for dynamic simulations is growing in the oil & gas industry to conduct rich simulation of offshore structures to ensure safe operation. Examples are Dynasim [Coelho01] and NOT (Numerical Offshore Tank) from Petrobras.

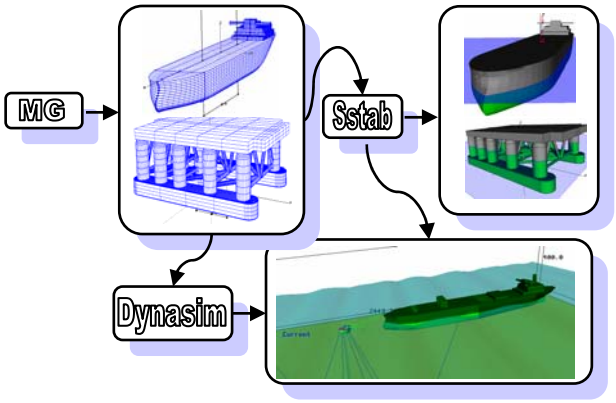

Figure 4: Integration among simulation tools.

2.3 Production Phase

The main aim of this phase is to support efficient and safe production of oil & gas. This requires putting in place a well trained work force for operation, plant monitoring, maintenance, emergency handling etc.

2.3.1 Monitoring

During the production phase, the virtual reality technology has the potential for supporting better monitoring of plants. Examples of such monitoring tasks include remote monitoring of oil pipelines (Figure 5) to avoid oil spillages, stability of the offshore structures and off loading operations, all of them already being employed by Petrobras.

Figure 5: Virtual pipeline trajectory over a satellite image.

To better analyse oil pipeline deformations, it is possible to use post-videos over the structural analysis results. Petrobras is also analysing solutions to allow the manager or the specialist to receive a visual representation of the oil pipeline directly from the field, in case of an accident or during a maintenance operation.

2.3.2 Emergency Scenarios

An oil spillage could have a devastating consequence on the environment costing millions of dollars to constrain the damage. Virtual reality can play a significant role in developing systems for training people for handling such situations and also for connecting experts during such a disaster situation to advise the workers, on the ground, to control the situation.

One such system, which has been developed to manage and control actions during a leakage of a pipeline is InfoPAE [Carvalho02], shown in Figure 6. It provides facilities to manage conventional and geographical data, associating them with the plans.

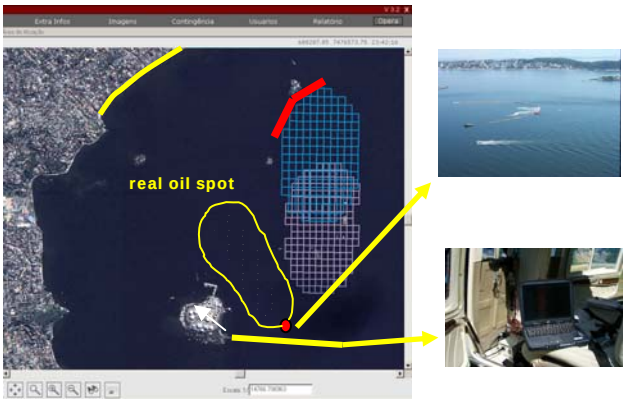

Figure 6: InfoPAE: emergency scenario application.

Another typical emergency scenario in the oil & gas area, described in the next section, is a crisis situation in an offshore structure, when the structure becomes unstable.

3. DISASTER MANAGEMENT OF OIL & GAS OFFSHORE STRUCTURES

Offshore units disasters can not only cause deaths and important environmental impacts, but also have a strong impact on business. Companies can lose billion of dollars by losing an offshore unit and further billions of dollars due to the cease of the oil production.

Petrobras faced two major accidents in the beginning of this decade. In 2001, the largest semi-submersible platform in the world P-36 (40 story-high, weighing 31,000 tons) sunk, killing 11 employees and ceasing a daily production of 84,000 barrels of oil and 1.3 million cubic meters of natural gas.

In 2002, the FPSO (Floating Production, Storage and Offloading) unit P-34 with a daily production of 35,000 barrels and a storage capacity of 58,000 m^3 of oil, weighting 62,000 tons, had a stability problem and almost sunk, immediately ceasing its operation. Fortunately at this time, Petrobras managed to rescue the unit without loss of lives.

Petrobras, as one of the oil & gas companies seeking to employ efficient processes and technologies to respond to such events, had taken important actions in order to ensure safety.

The implementation of such processes involves bringing together a large number of diverse and geographically distributed groups and resources to make appropriate decisions within a short period of time. Such groups are comprised of many technical experts and decision makers such as naval engineers, structural engineers, risers analysts and oceanographers, as well as managers.

The high-level decision group will operate from the operational unit headquarters. The technical staff, running various simulation programmes which take into account the waves, wind, currents and other forces on the unit, operates either from a base on earth near the disaster, from the company's headquarters and/or from various research centres.

These groups need an efficient communication media with the operators inside the unit, divers and security team, and perhaps with experts who are travelling to execute the rescue plan and work towards consensus.

Petrobras has an on going project to develop a distributed ICT environment for the technical groups to work as a virtual team to explore various simulation options and to communicate their results to the high-level decision makers (Figure 7). To achieve this aim, there are some actions involved:

- a survey is being conducted to identify the requirements for the distributed workspace, from the stakeholders involved in a disaster scenario;
- a distributed workspace environment is being designed and built for the technical team to engage in the rescue efforts;
- the usability and functionality of this environment will be evaluated for training and operational purposes.

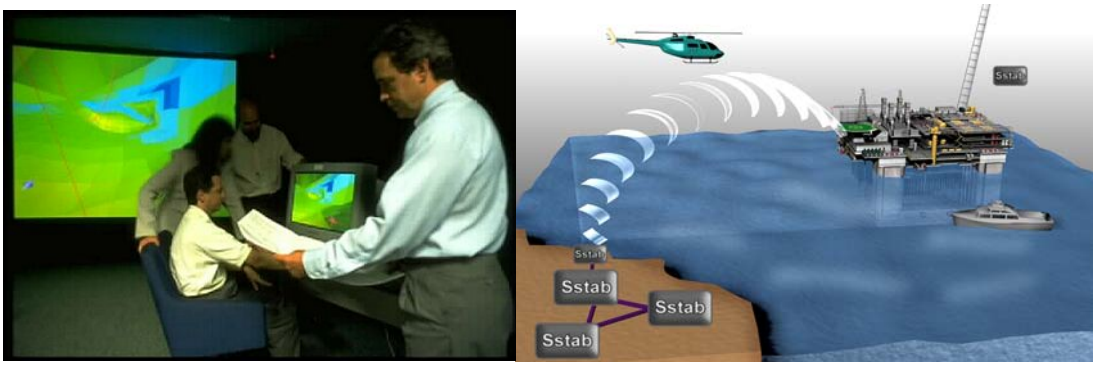

Figure 7: Emergency oil & gas E & P scenario with distributed people and resources.

4. ACKNOWLEDGEMENTS

The authors would like to thank Petrobras, Tecgraf/PUC-Rio and The University of Salford for their support, and, for their great contribution, Petrobras' engineers Álvaro Maia and Isaias Masetti, and engineer Luiz Cristovão Gomes Coelho from PUC-Rio.

5. REFERENCES

[American98] The American Oil& Gas Reporter (March 1998).

[Campos02] Campos, J.L.E.: Real-Time Well Drilling Monitoring using gOcad. *Proceedings of the 22nd Gocad Meeting* (2002), 10.

[Carvalho02] Carvalho, M.T., Casanova, M.A., Torres, F. and Santos, A.: INFOPAE – An Emergency Plan Deployment System. *Proceedings of the Int. Pipeline Conference* (2002).

[Coelho01] Coelho, L.C.G., Nishimoto, K. and Masetti, I.Q.: Dynamic Simulation of Anchoring Systems Using Computer Graphics*. OMAE Conference* (2001).

[Coelho03] Coelho, L.C.G., Jordani C.G., Oliveira, M.C. and Masetti, I.Q.: Equilibrium, Ballast Control and Free-Surface Effect Computations Using The Sstab System. *8th Int. Conf. Stability of Ships and Ocean Vehicles - Stab* (2003), 377-388.

[Corseuil04] Courseuil, E.T.L., Raposo, A.B., Silva, R.J.M., Pinto, M.H.G., Wagner, G.N. and Gattass, M.: ENVIRON – Visualization of CAD Models In a Virtual Reality Environment. *Proceedings of the Eurographics Symposium on Virtual Environments* (2004), 79-82.

[Petrobras99] *Petrobras Magazine*, 7, 26 (1999), 20-23.

[Reis01] Reis, L.P., Bosquet, F. and Paul, J.C.: Toward collaborative geomodeling. *Proceedings of the 21st GOCAD Meeting* (2001).

[Silva03] Silva, P.M., Machado, M. and Gattass, M.: 3D Seismic Volume Rendering. *8th Int. Congress of The Brazilian Geophysical Society* (2003).

Apêndice C: Telas do Protótipo HLA

Neste Apêndice, apresentamos algumas telas mostrando a execução do nosso protótipo HLA.

Na Figura 24, podemos ver a sessão colaborativa sendo iniciada. Ela possui sete janelas contíguas representando, respectivamente, de cima para baixo: *(i)* no lado esquerdo, o servidor *Executive*, o Piloto da Emergência T0, o operador do SSTAB T1 e a Imprensa P1 (*Press*); *(ii)* no lado direito, o agente de software S2 executando o simulador WAMIT, o operador do DYNASIM T3 e o Tomador de Decisões DM1 (*Decision Maker*). Deve ser observado que cada um desses participantes poderia estar utilizando diferentes máquinas, mas eles estão utilizando a mesma máquina neste exemplo apenas para mostrar a sessão colaborativa na mesma tela. As Figuras 25 a 45 mostram, em ordem cronológica, a seqüência de telas da execução do protótipo.

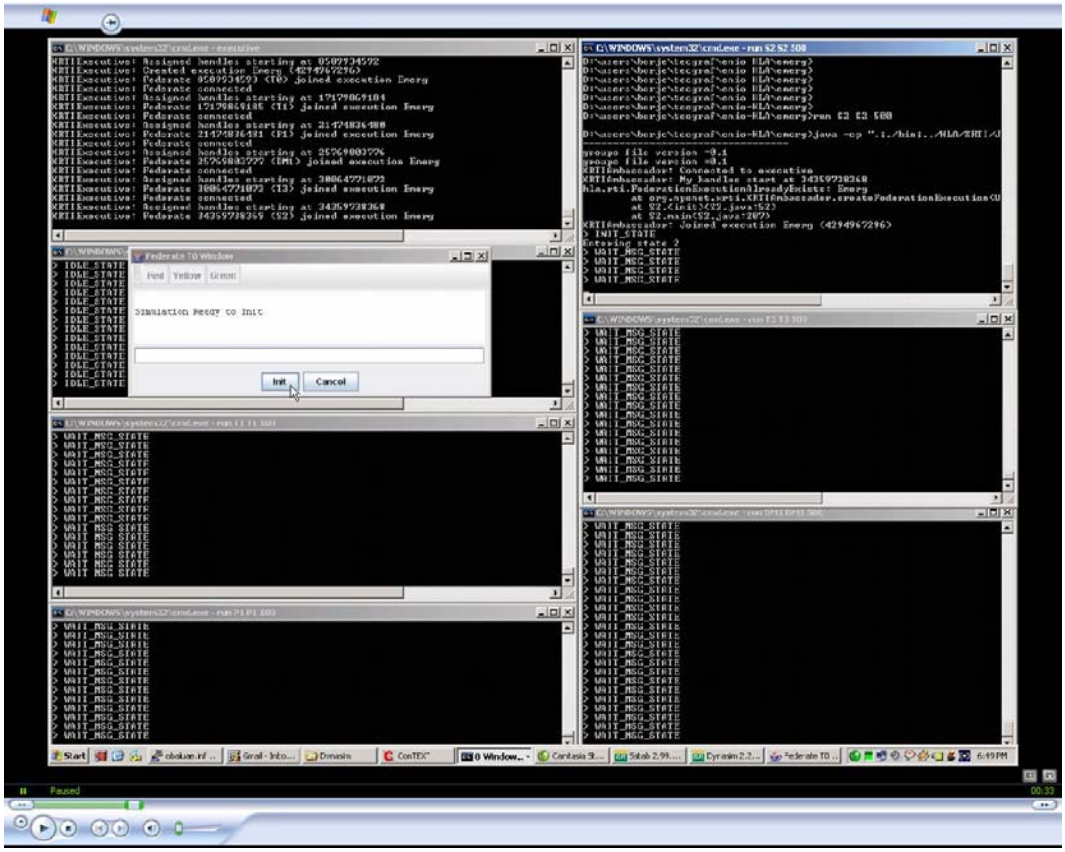

Figura 24 - Sessão colaborativa sendo iniciada

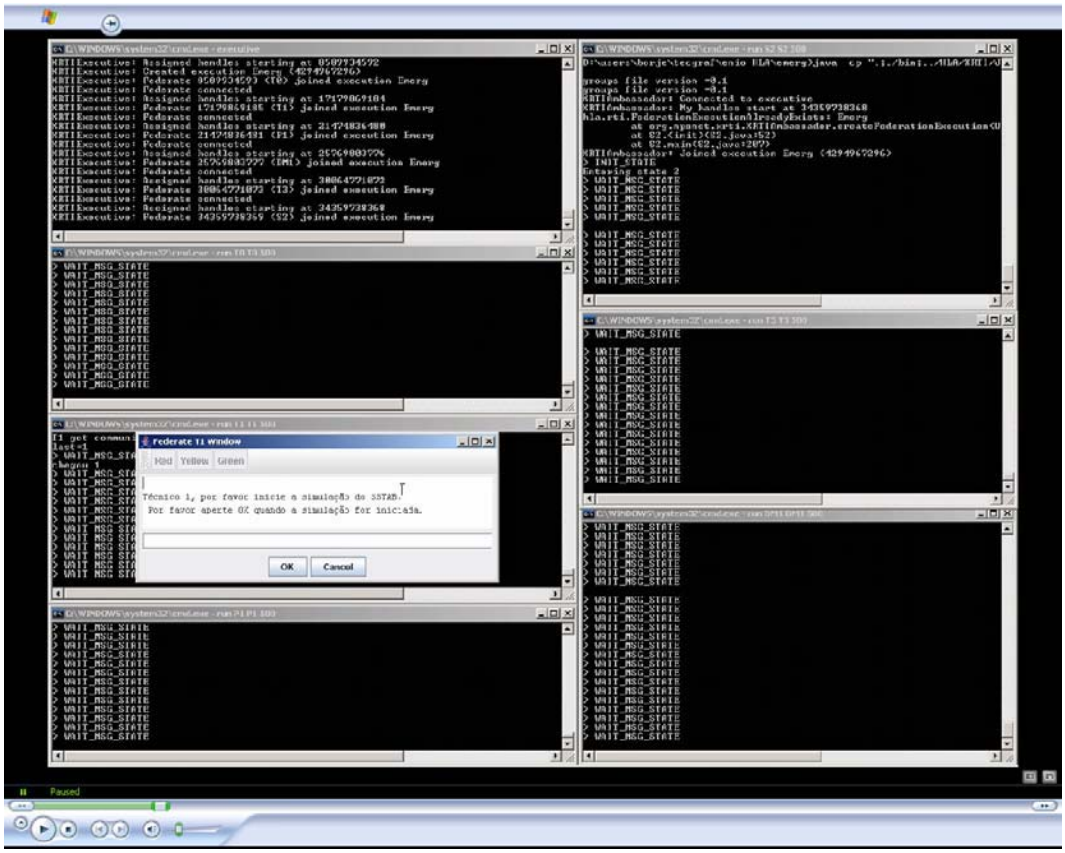

Figura 25 - O operador do SSTAB T1 recebe uma mensagem de T0 para iniciar a simulação do SSTAB

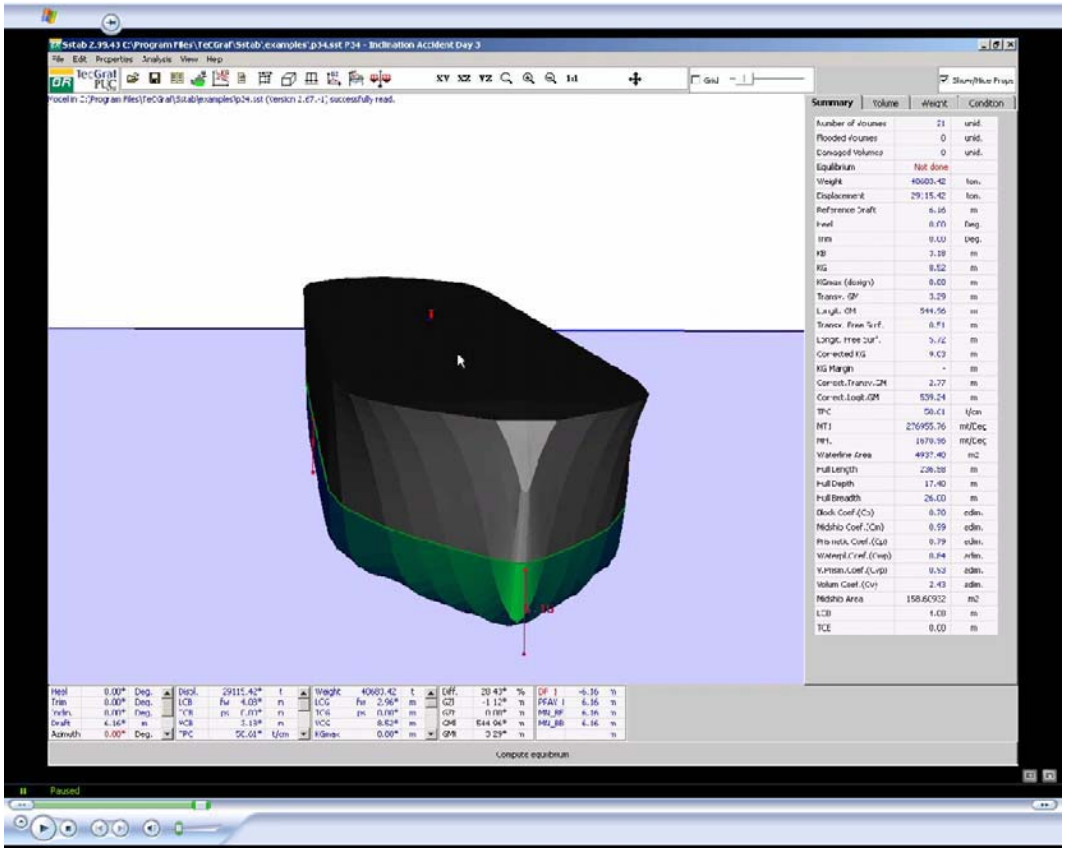

Figura 26 - O operador do SSTAB T1 inicia a simulação do SSTAB

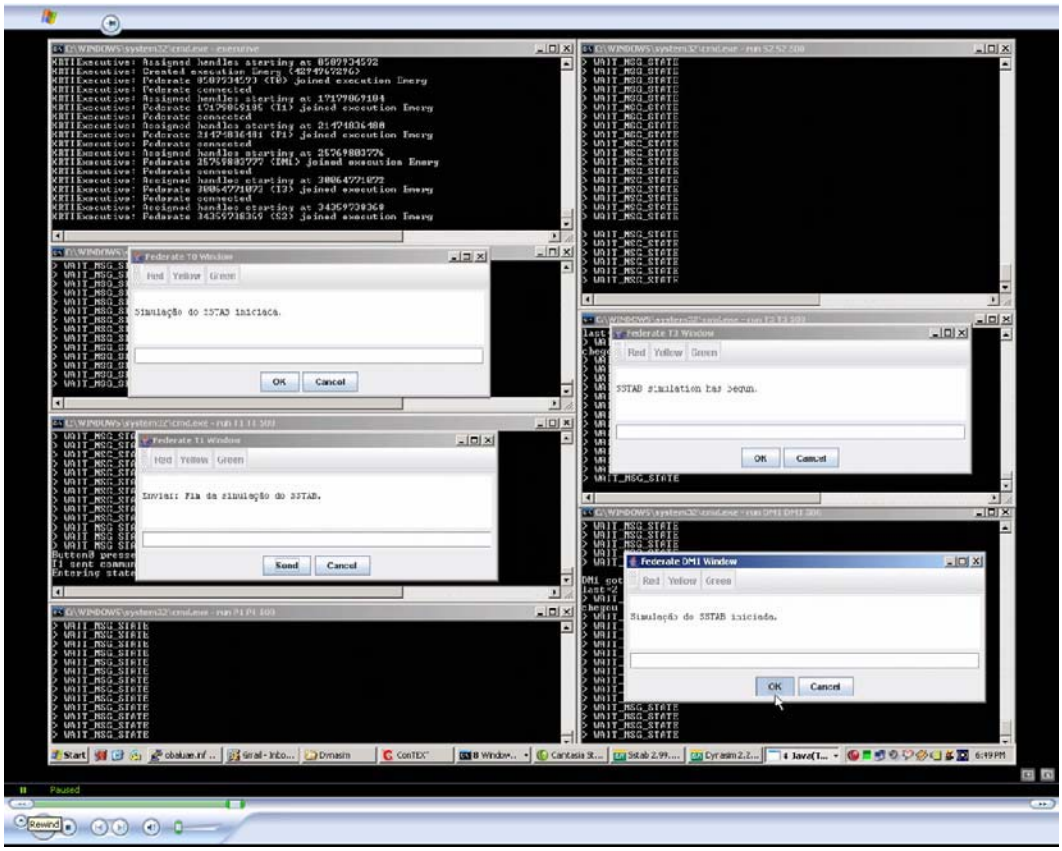

Figura 27 - Os federados T0, T3 e DM1 recebem uma mensagem de T1 informando que ele começou a simulação do SSTAB

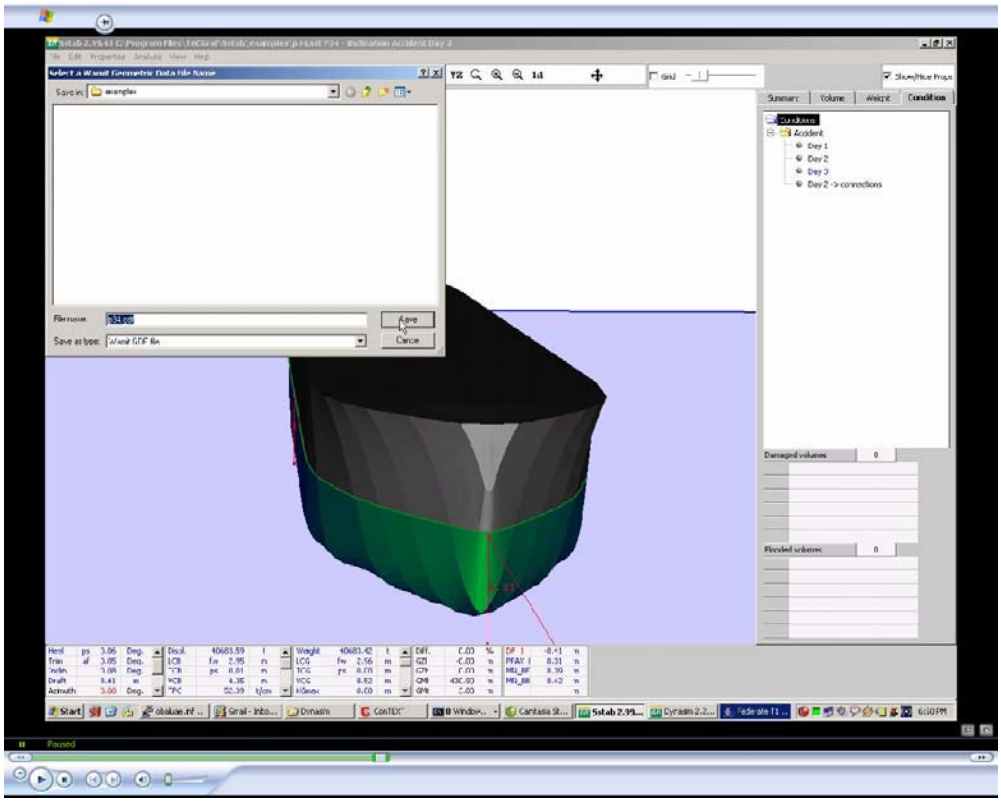

Figura 28 - T1 exporta um arquivo geométrico do WAMIT e termina a simulação do SSTAB

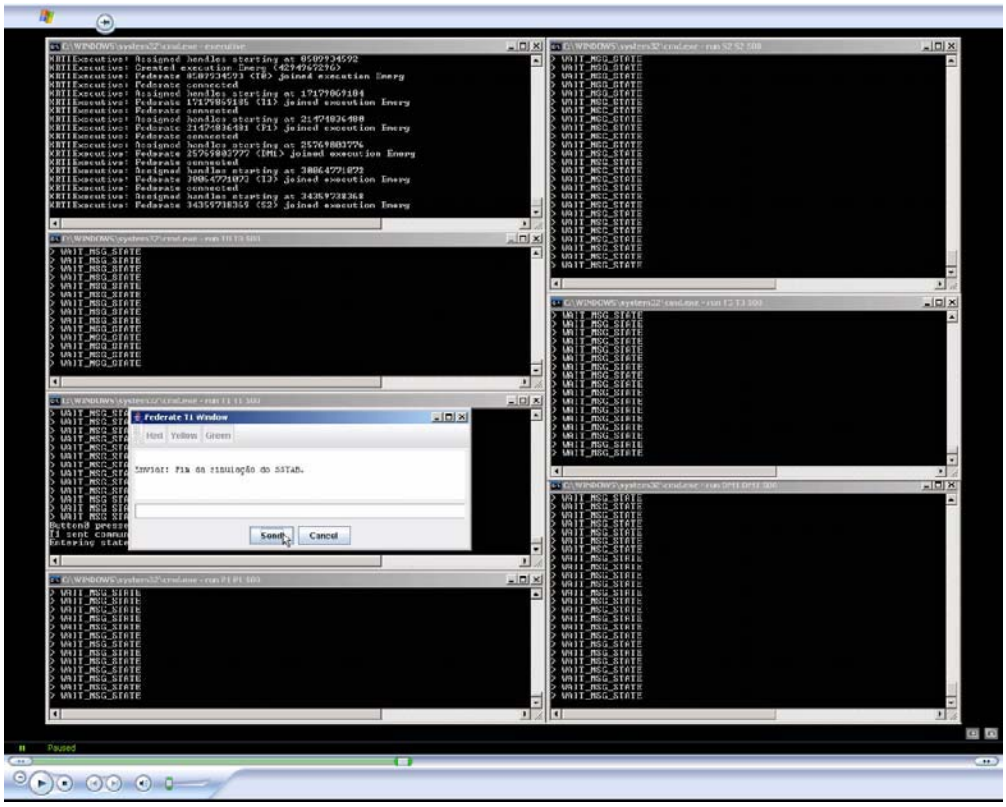

Figura 29 - T1 envia uma mensagem informando o fim da simulação do SSTAB, com S2 ativando automaticamente o simulador WAMIT

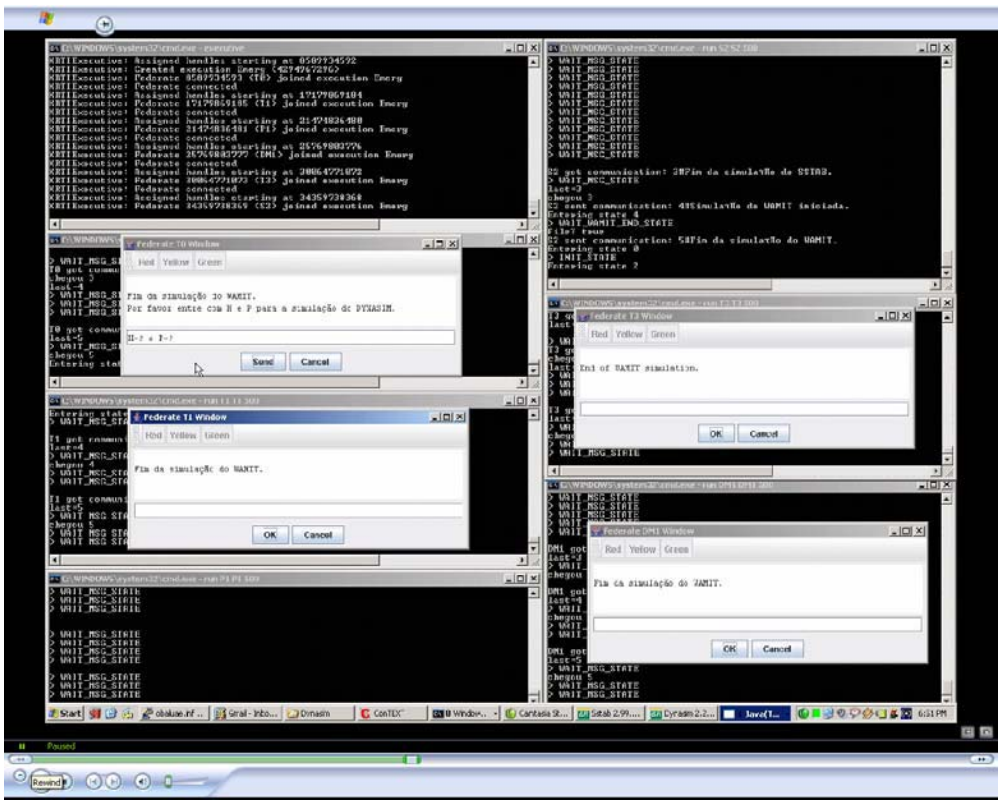

Figura 30 - S2 envia automaticamente para os federados T0, T1, T3 e DM1 uma mensagem informando o fim da simulação do WAMIT

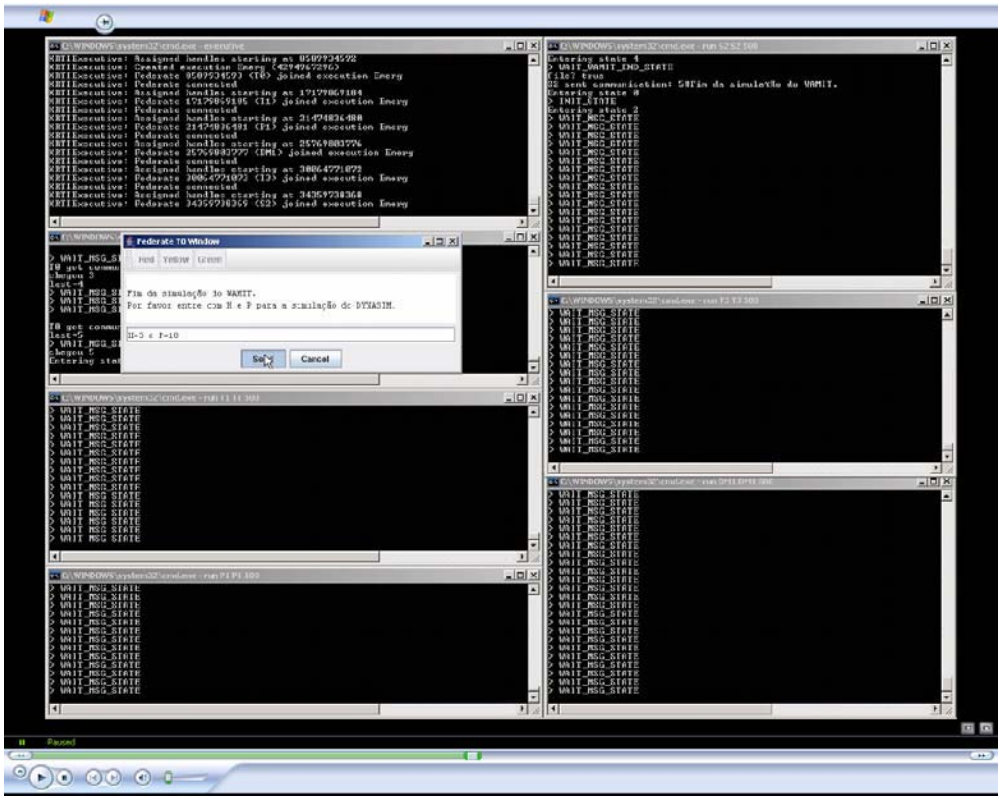

Figura 31 - O Piloto da Emergência T0 envia os dados ambientais (H=5 e P=10) para o operador do DYNASIM T3

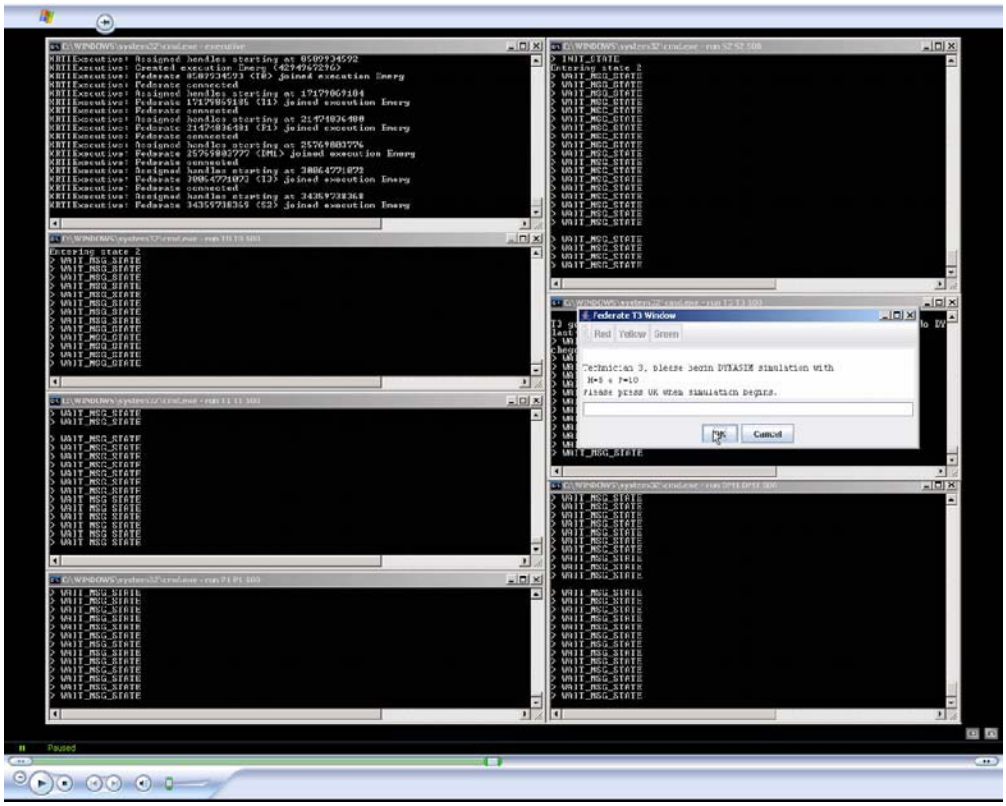

Figura 32 - O operador do DYNASIM T3 recebe os dados ambientais (H=5 e P=10) de T0

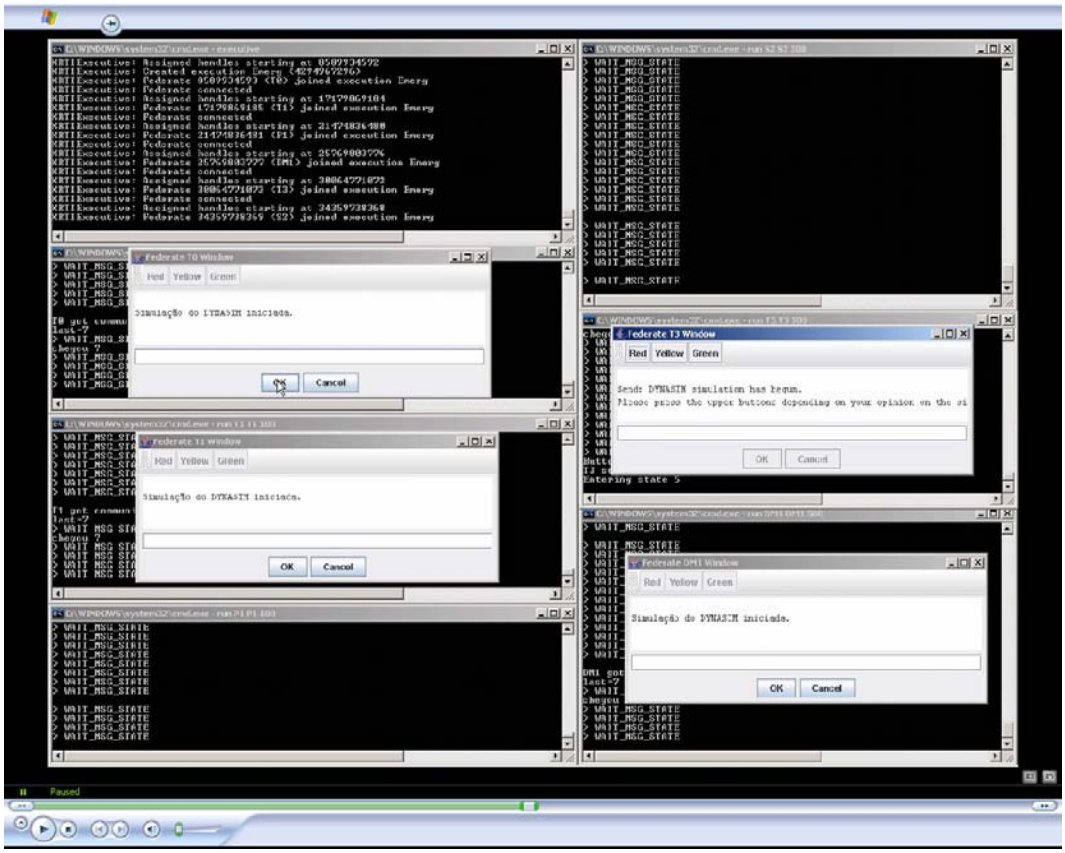

Figura 33 - O operador do DYNASIM T3 envia uma mensagem informando que ele irá iniciar a simulação do DYNASIM

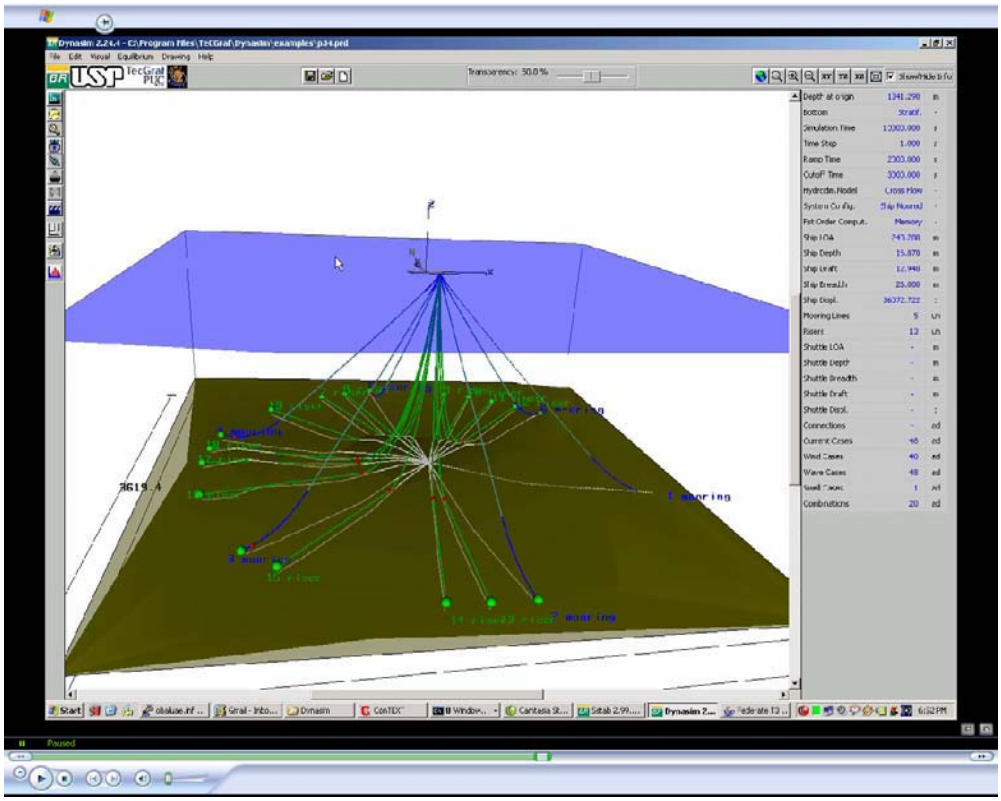

Figura 34 - A simulação do DYNASIM é iniciada

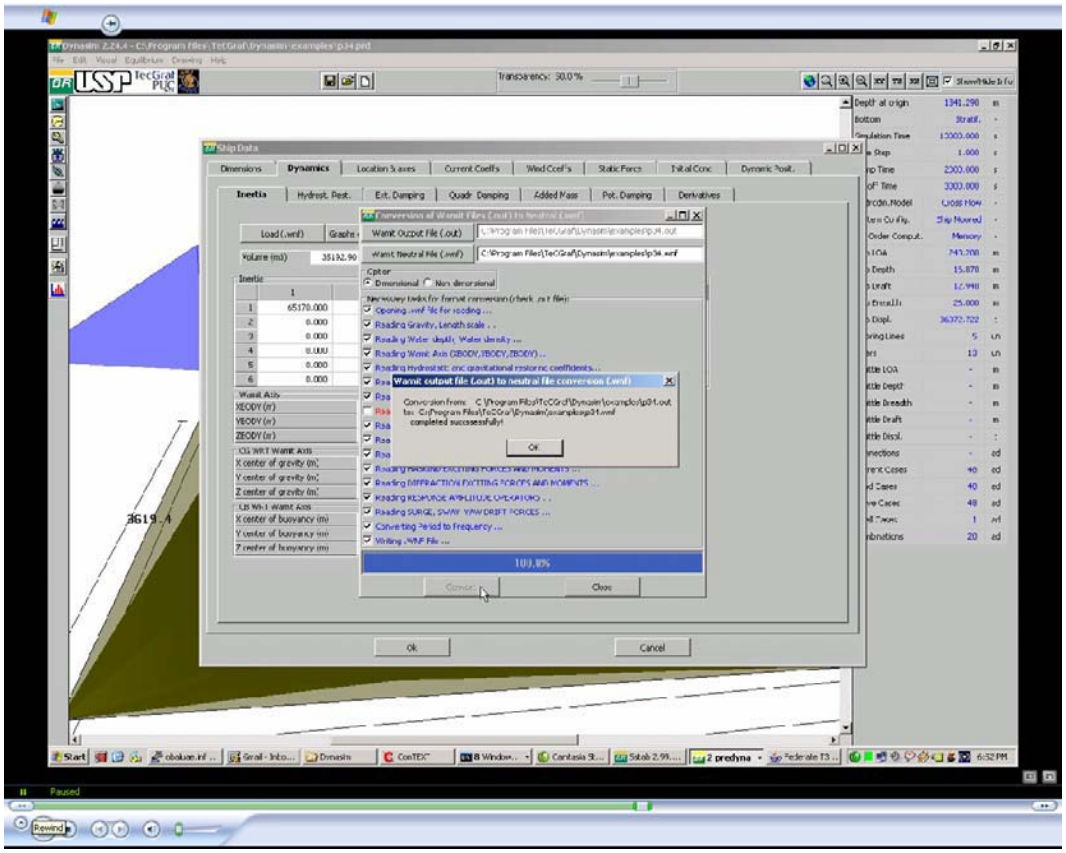

Figura 35 - O DYNASIM lê e converte o arquivo de saída do WAMIT em um arquivo neutro do WAMIT

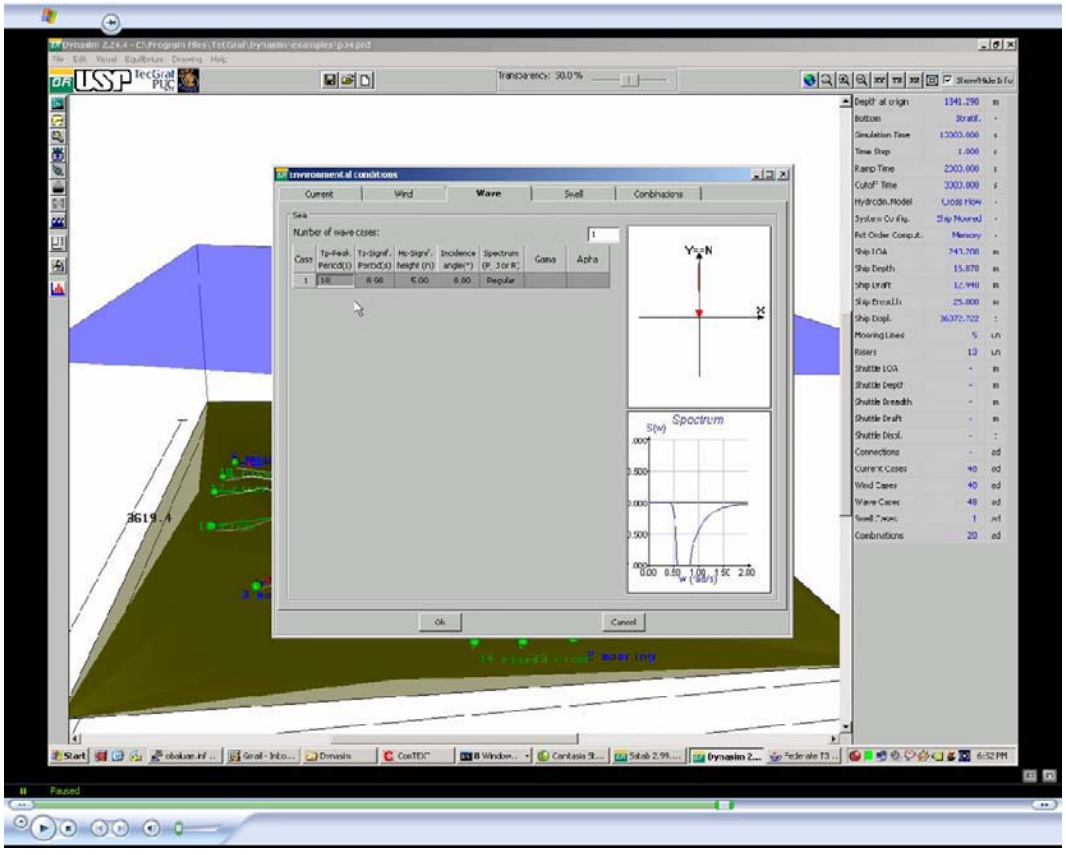

Figura 36 - T3 entra com os dados ambientais (H=5 e P=10) dentro do DYNASIM

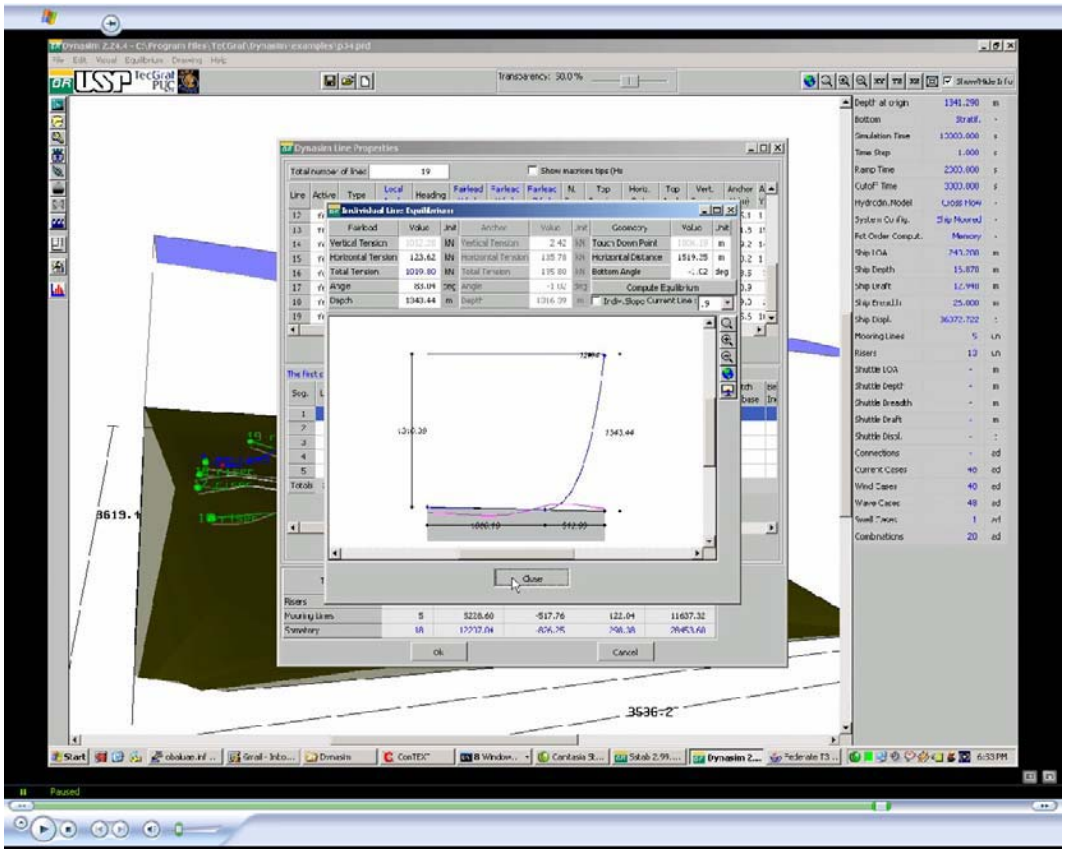

Figura 37 - T3 termina a simulação do DYNASIM

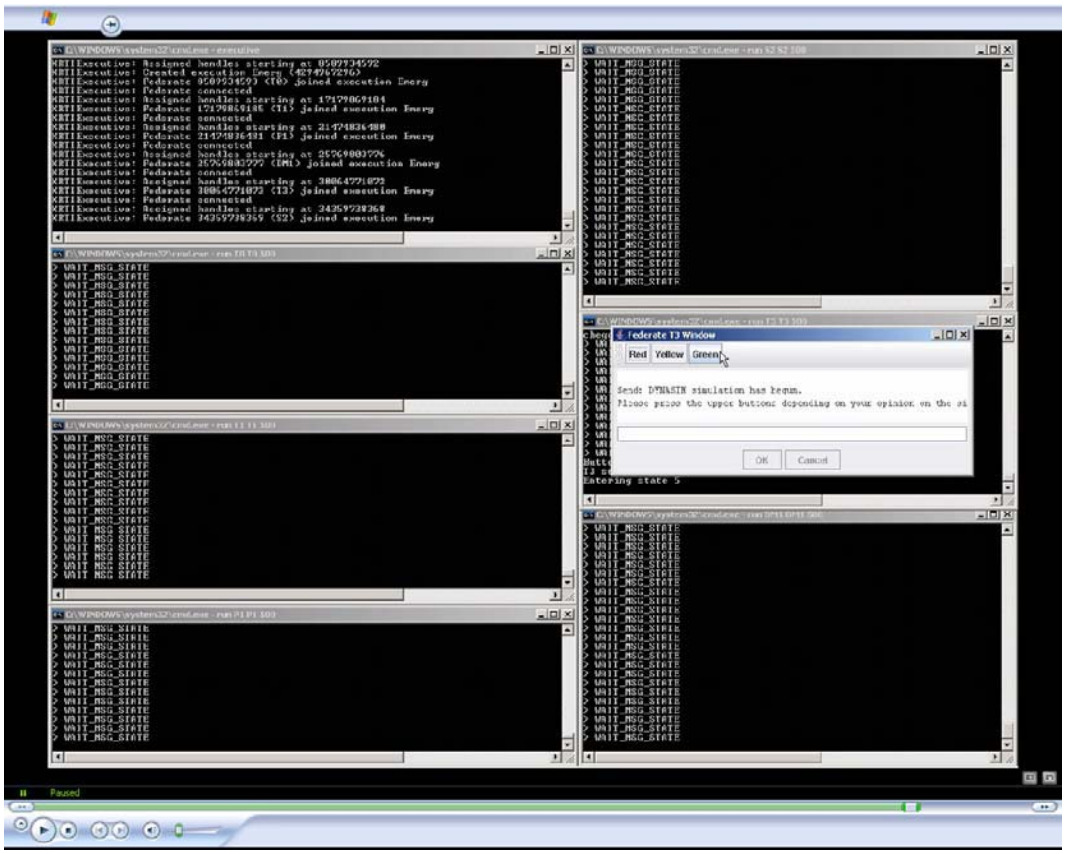

Figura 38 - T3 envia um sinal verde referente à simulação do DYNASIM

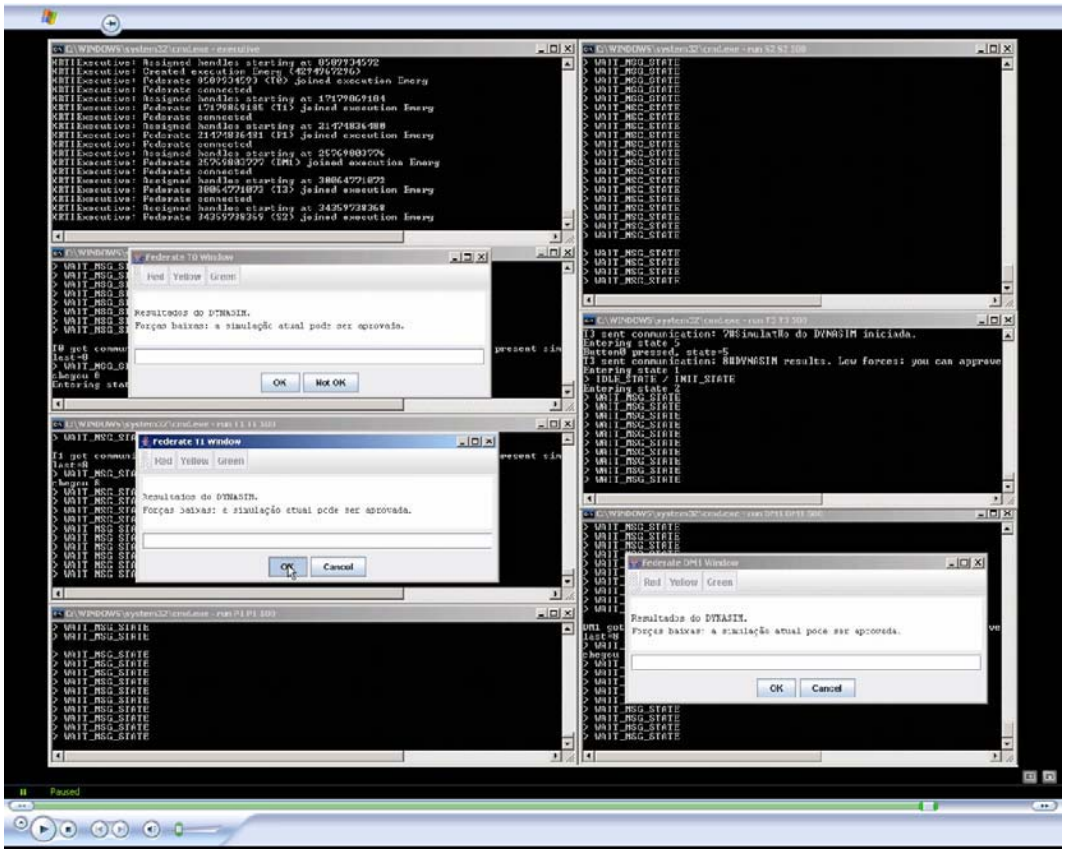

Figura 39 - Os federados T0, T1 e DM1 recebem uma mensagem de T3 informando o resultado da simulação do DYNASIM

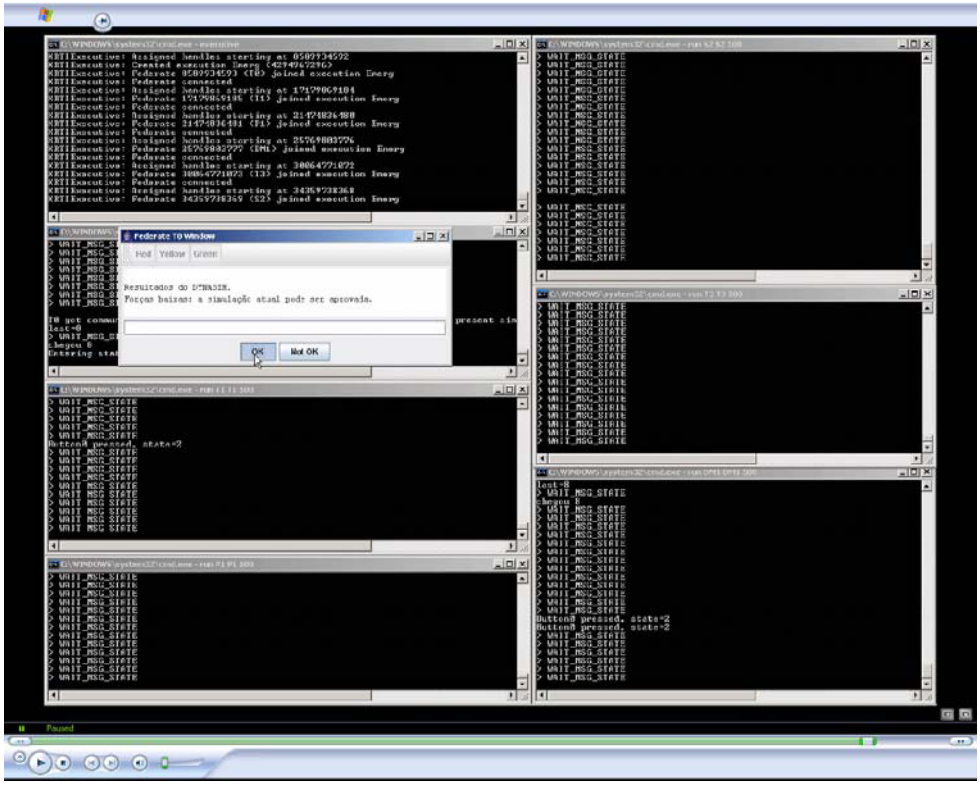

Figura 40 - O Piloto da Emergência T0 aprova os resultados das simulações

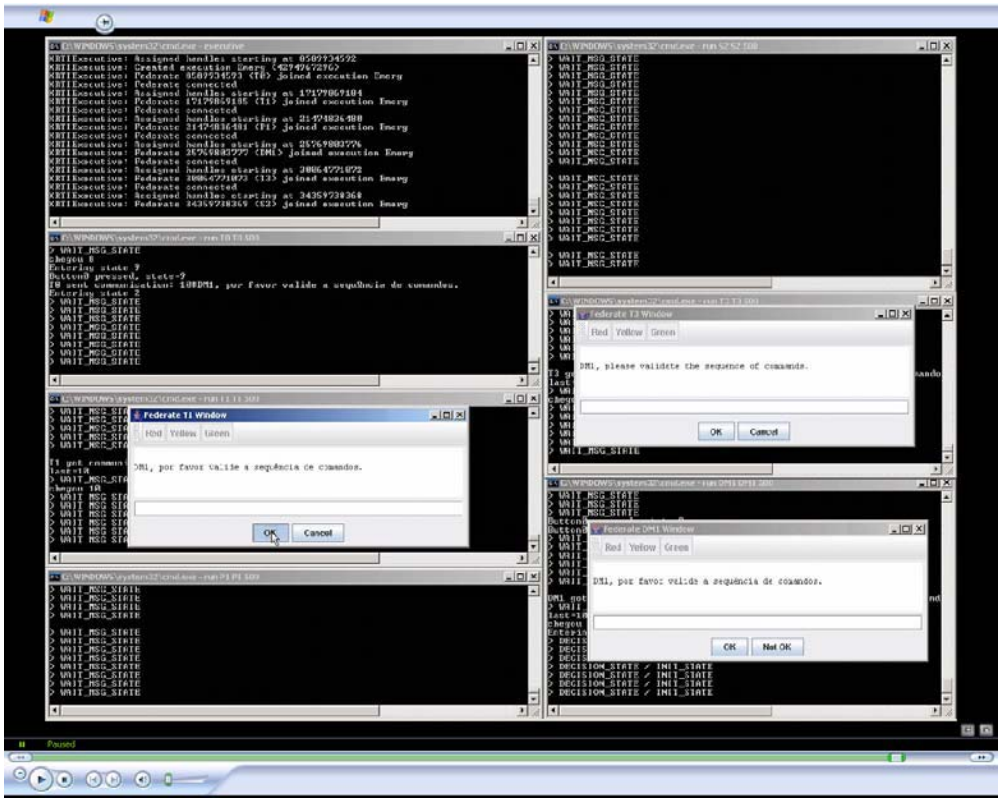

Figura 41 - O Tomador de Decisões DM1 (e os federados T1 e T3) recebe uma mensagem de T0 solicitando que ele valide a seqüência de comandos a ser executada

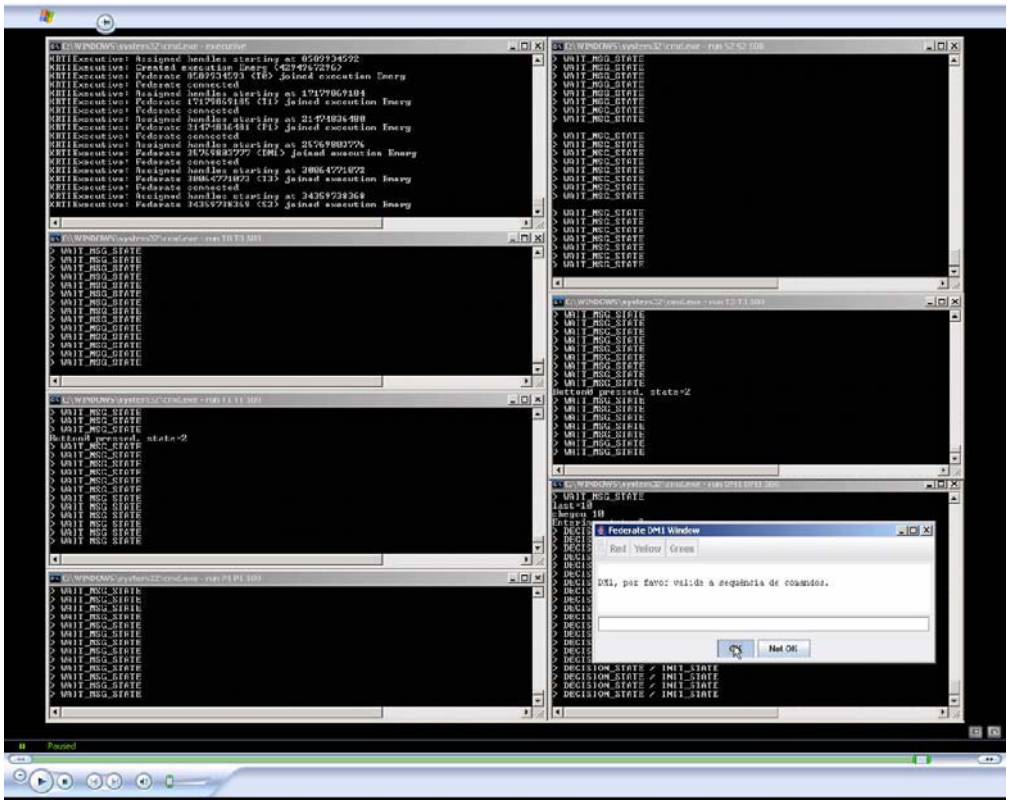

Figura 42 - O Tomador de Decisões DM1 valida a seqüência de comandos a ser executada

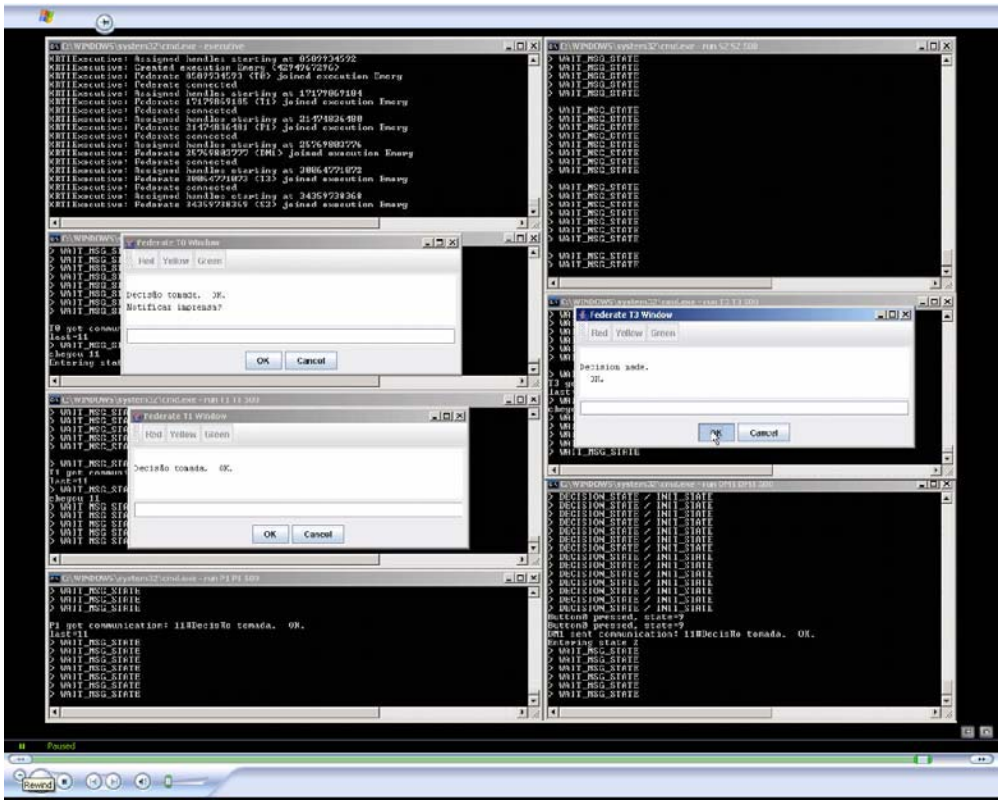

Figura 43 - Os federados T0, T1 e T3 recebem uma mensagem de DM1 informando que ele validou a seqüência de comandos a ser executada

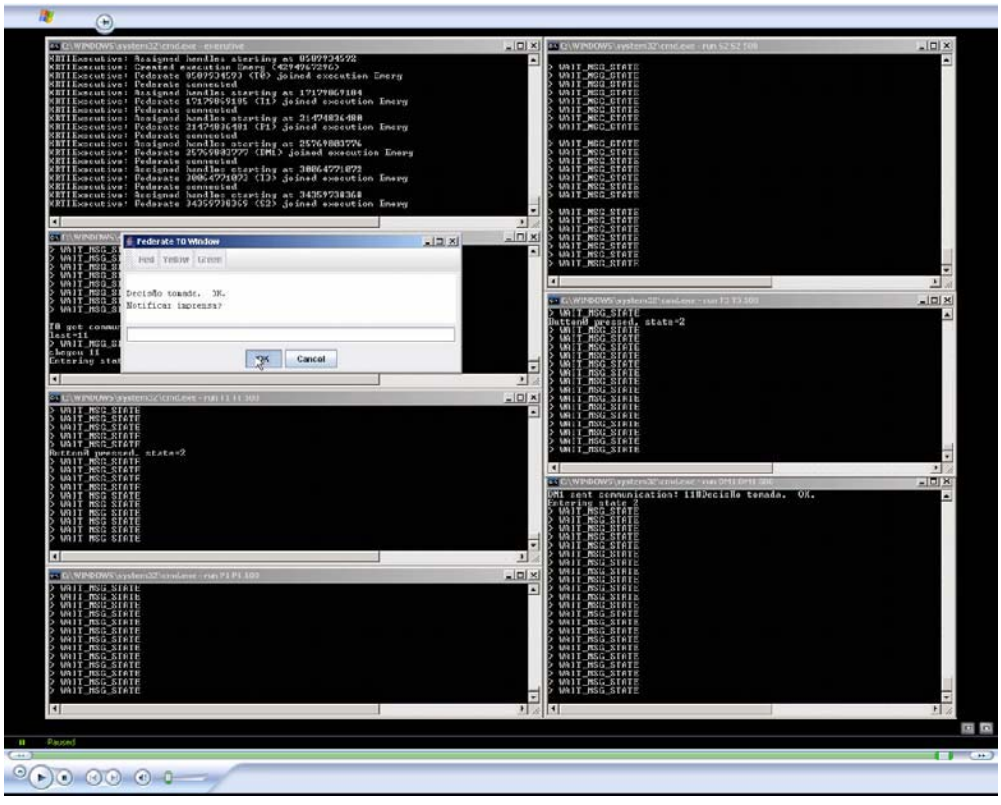

Figura 44 - O Piloto da Emergência T0 notifica a Imprensa a respeito da decisão tomada, enviando um relatório

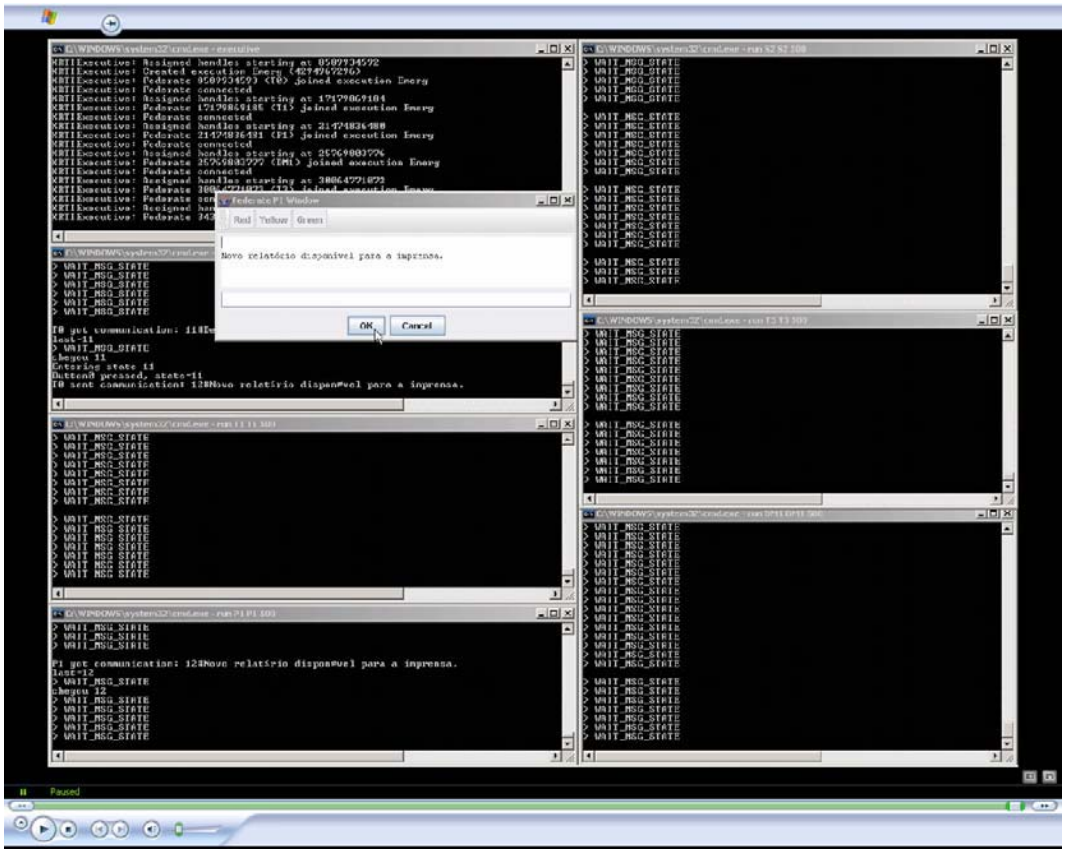

Figura 45 - A Imprensa recebe uma mensagem de T0 informando que um novo relatório está disponível