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.

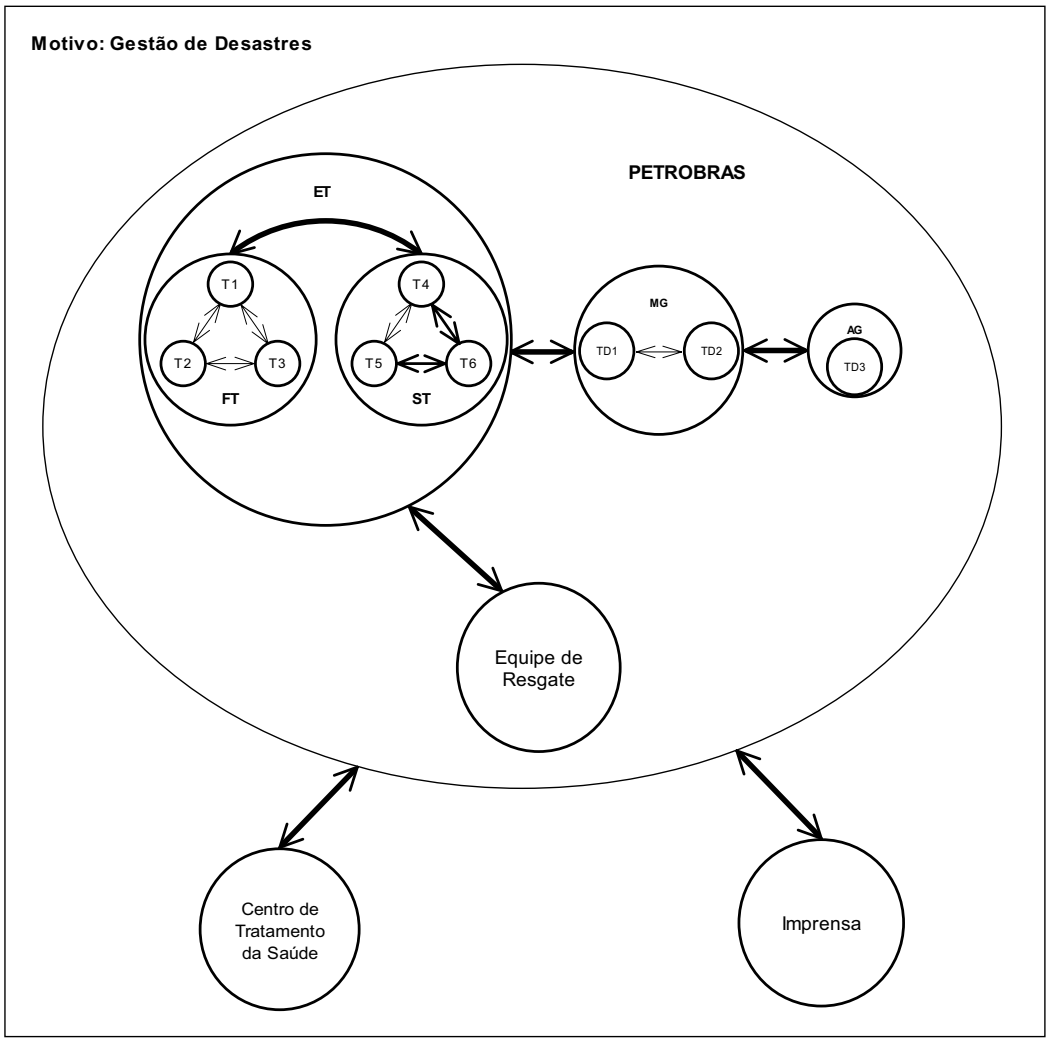

Figura 16 - O modelo colaborativo de gestão de desastres: figura global

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.

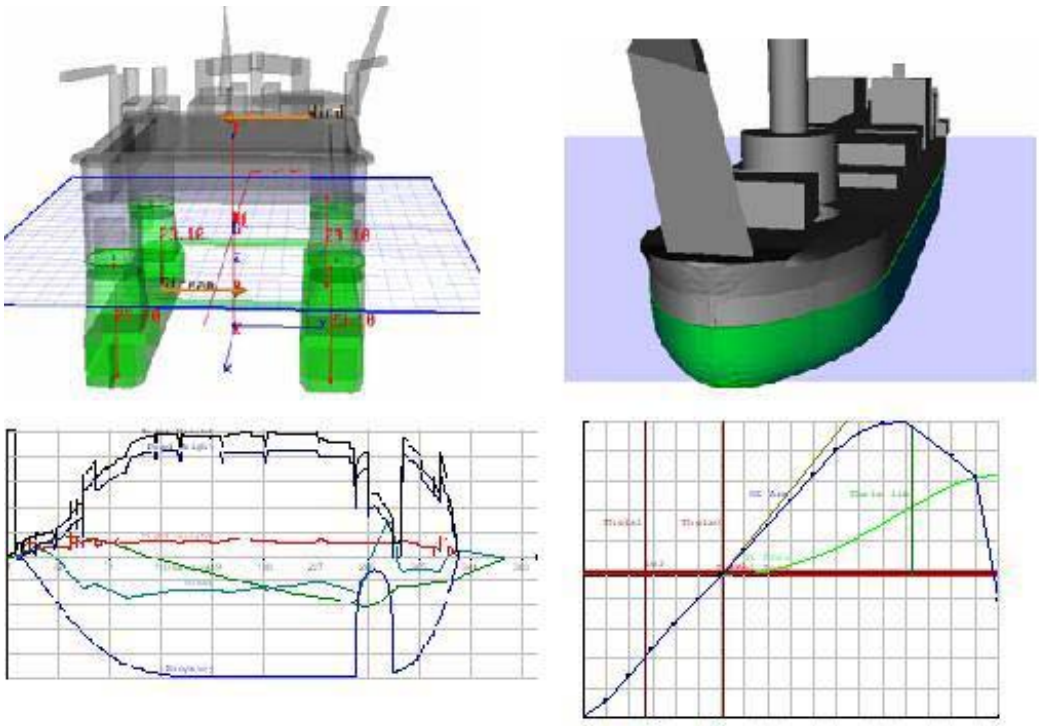

Figura 17 - SSTAB: sistema de Estabilidade de Unidades Flutuantes

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.

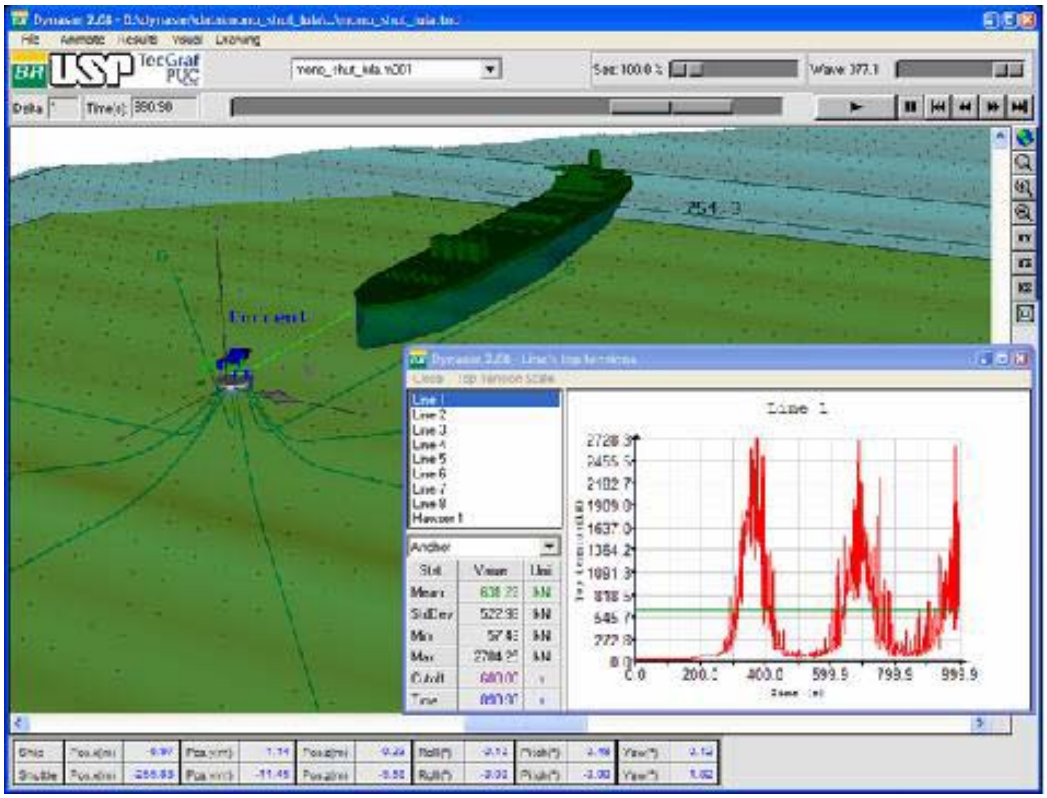

Figura 18 - DYNASIM: sistema de Estabilidade Dinâmica

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.

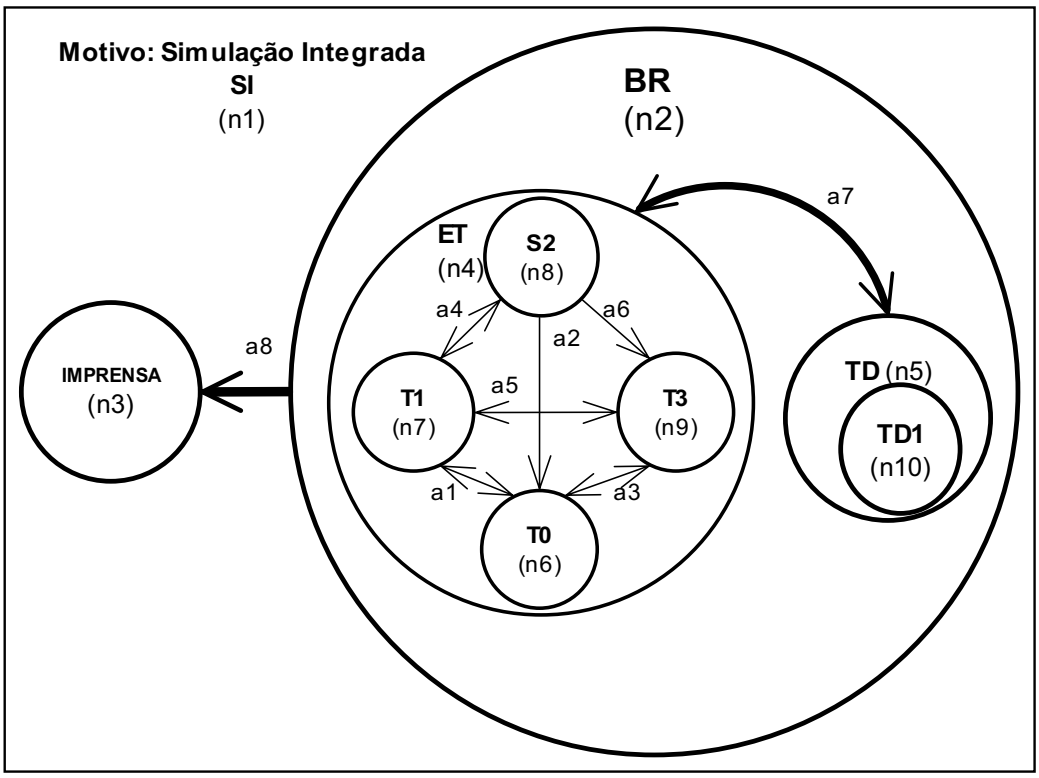

Figura 19 - Um primeiro modelo para a aplicação colaborativa de gestão de desastres, focando na simulação integrada

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).

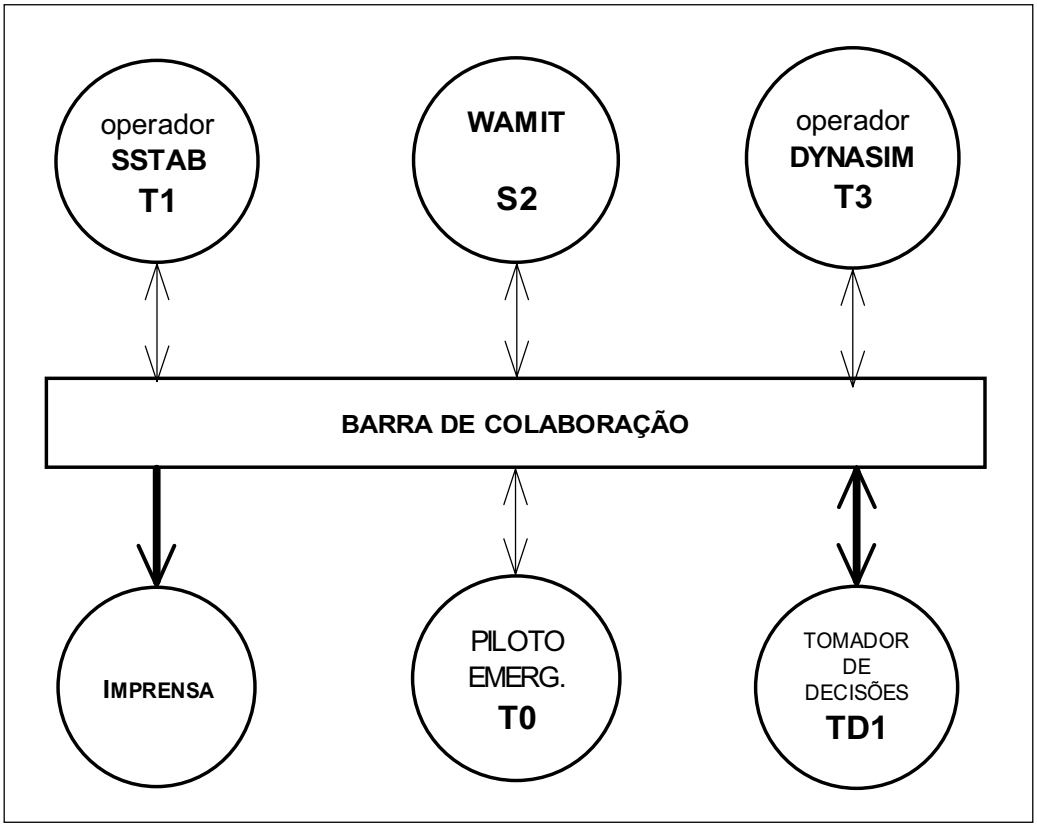

Figura 20 - Um protótipo HLA para o primeiro modelo para a aplicação de desastres

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).

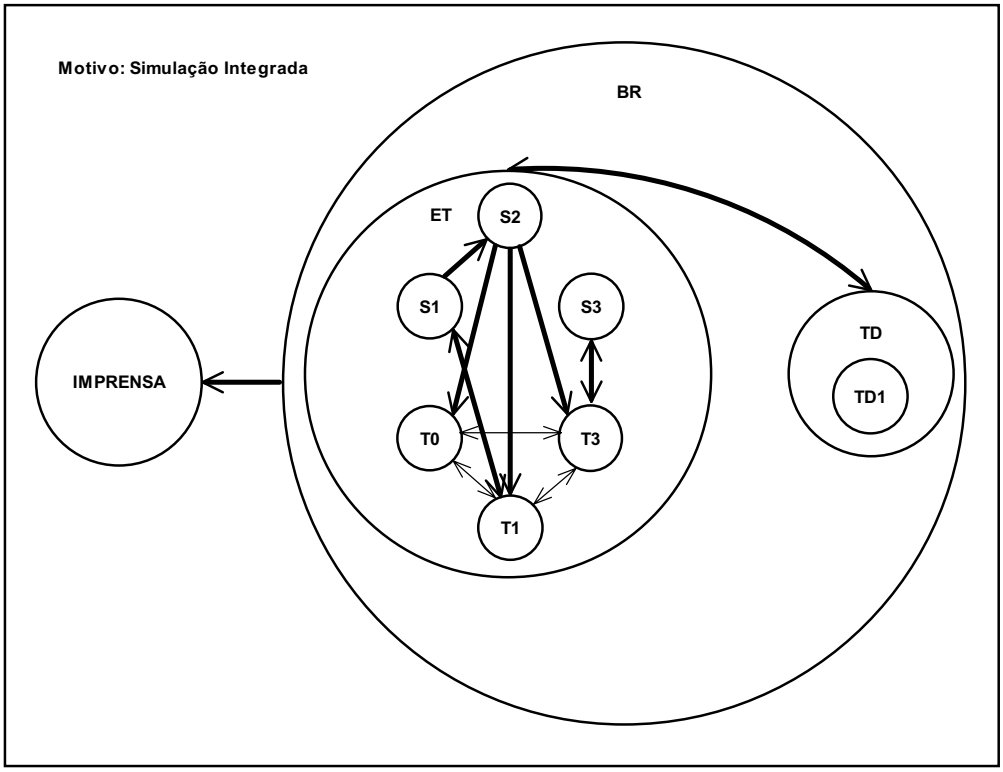

Figura 21 - Um segundo modelo para a aplicação colaborativa de gestão de desastres, focando na simulação integrada

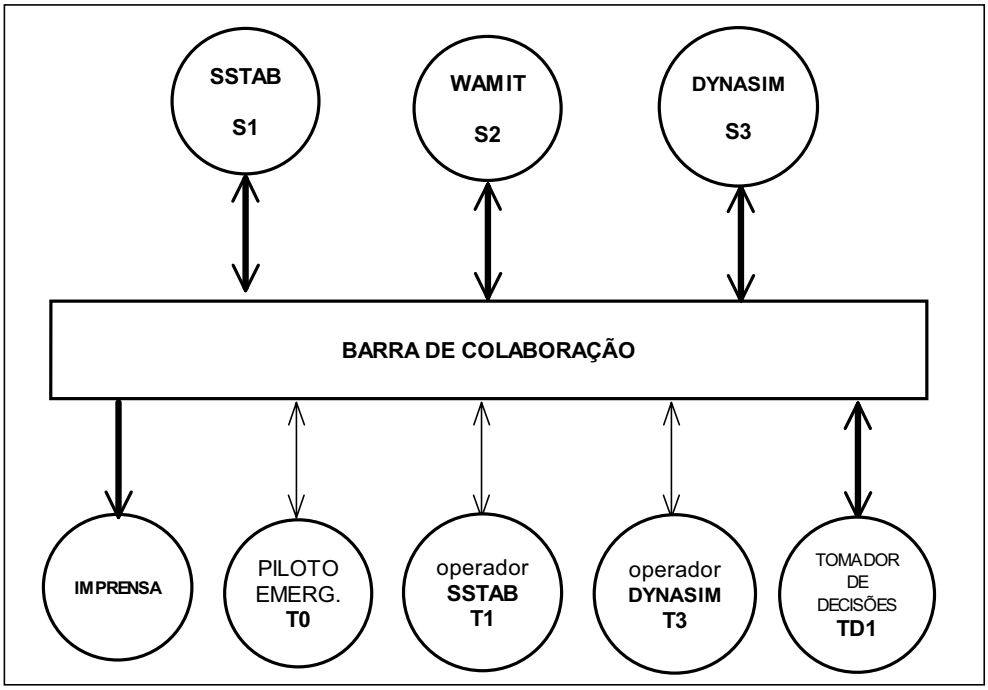

Figura 22 - Um protótipo HLA para o segundo modelo para a aplicação de desastres

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.

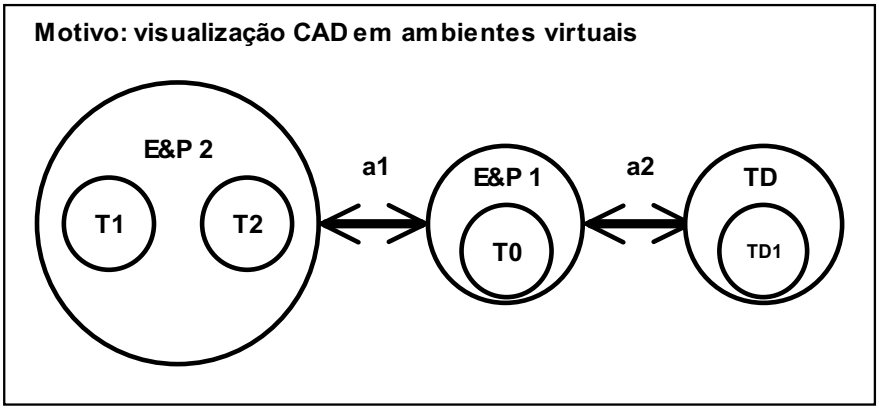

Figura 23 - Um modelo para a visualização CAD em ambientes virtuais

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.