SISTEMA INTEGRADO DE AUXÍLIO AO PROJETO DE PROTOCOLOS COM RESTRIÇÕES TEMPORAIS (ProtCAD/RT)

Gardel Moreira Delfino — **gardel@gta.ufrj.br**

José Valentim dos S. Filho — **valentim@gta.ufrj.br**

Rodrigo de Carvalho Roman — **roman@gta.ufrj.br**

Jorge Lopes de Souza Leão — **leao@gta.ufrj.br**

Aloysio de Castro Pinto Pedroza — **aloysio@gta.ufrj.br**

Grupo de Teleinformática e Automação (GTA)

Universidade Federal do Rio de Janeiro (UFRJ)

COPPE/PEE - Programa de Engenharia Elétrica

C.P. 68504 - CEP 21945-970 - Rio de Janeiro - RJ - Brasil

Tel: (021) 260-5010 Fax: (021) 290-6626

http://www.gta.ufrj.br

RESUMO

O objetivo deste trabalho é descrever a arquitetura de um ambiente integrado para especificação e simulação de protocolos de comunicação, que possibilite o uso de restrições temporais. Este sistema, em fase de desenvolvimento, tem como base a ferramenta CAD-Estelle, utilizada em diversos trabalhos acadêmicos. No ambiente que será descrito, poderão ser efetuadas todas as etapas de projeto a partir do modelo de Máquina de Estados Finitos Estendida. O ambiente apresenta ainda facilidades de edição visual baseada neste modelo e utiliza a extensão *Real-Time Estelle* para especificar sistemas distribuídos em tempo real. A parte visual está sendo desenvolvida na linguagem JAVA visando permitir o acesso através da WEB, bem como a independência de plataforma.

ABSTRACT

The purpose of this work is to describe the architechture of an integrated environment for specifying and simulating communication protocols, with timing constraints. The system, under development, is based on the CAD-Estelle toolset, which has been used in many academical works. In the proposed environment, all design steps can be done from the Extended Finite State Machine Model. The environment also has a visual editing tool based on this model and uses the Real-Time Estelle extension to specify distributed systems with timing constraints. This visual tool is being developed in JAVA language to allow WEB access, as well as making the system platform-independent.

Palavras Chaves: Protocolo, Especificação Formal, Tempo-Real, Estelle, Interface WEB

1. Introdução

Na última década houve um crescimento significativo de aplicações distribuídas onde parâmetros relacionados à qualidade de serviço (QoS) oferecida são fundamentais. Alguns exemplos que podem ser citados facilmente são as aplicações multimídia e o controle em tempo real.

As entidades que compõem estes sistemas realizam a troca de dados através de protocolos de comunicação que, devido às características destes novos sistemas, incorporaram restrições de QoS em suas especificações e implementações.

Neste contexto, as Técnicas de Descrição Formal (TDFs) assumem um papel importante. Elas apresentam a comodidade de se projetar uma determinada aplicação distribuída em um nível de abstração maior do que o obtido com o uso de uma linguagem de programação, pois esta última envolve detalhes muito específicos relacionados a um dado ambiente de implementação. Além disto, diversas ferramentas de projeto baseadas em TDFs permitem que se valide e teste uma determinada arquitetura antes de se partir para sua implementação.

Neste trabalho, apresentaremos um sistema integrado de auxílio ao projeto de protocolos (ProtCAD/RT – *Protocol Computer Aided Design/Real-Time*), que utiliza a TDF Estelle [Diaz89] e que se encontra em fase de desenvolvimento. Este sistema tem como base a ferramenta CAD-Estelle desenvolvida e utilizada em diversos trabalhos de graduação e de pós-graduação no Laboratório do Grupo de Teleinformática e Automação (GTA/COPPE-UFRJ) [PGVO89]. Entretanto, a padronização ISO para a linguagem Estelle não é capaz de expressar requisitos de tempo-real, o que torna necessário a utilização de extensões para esta finalidade. O sistema que apresentaremos utiliza a extensão *Real-Time Estelle* [Fisc96]. Este sistema tem como principal objetivo tornar a tarefa de especificação e simulação de protocolos com restrições de QoS mais simples e intuitiva.

Na seção 2, mostraremos algumas características importantes da linguagem Estelle dando ênfase à extensão para tempo-real. Na seção 3 descreveremos a estrutura do ProtCAD/RT, mostrando seus módulos básicos, suas funções e o relacionamento entre eles. Na seção 4 abordaremos o ambiente de implementação do sistema. Na seção 5 apresentaremos algumas observações e comentários.

2. A Linguagem de Descrição Formal Estelle e a Extensão Tempo-Real

A linguagem Estelle é uma TDF padronizada pela ISO. O uso de Estelle para especificação, validação e implementação semi-automática de protocolos tradicionais e sistemas distribuídos tem sido realizado com muito sucesso como se pode comprovar pela divulgação de algumas ferramentas de projeto comerciais [EDT99] e acadêmicas [SiSt91a, SiSt91b, ThGo98a, PGVO89].

Uma especificação realizada em Estelle é constituída por um conjunto de módulos que se comunicam seja através de filas *FIFO* (*First In First Out*), pelas quais é feito o envio de mensagens, utilizando primitivas específicas da linguagem, ou através de variáveis exportadas. O comportamento de um módulo é descrito por uma Máquina de Estados Finitos Estendida (MEFE), sendo o tratamento de dados efetuado com o uso da linguagem Pascal.

Apesar de sua ampla utilização em protocolos tradicionais, a padronização Estelle não é adequada para especificar restrições de tempo-real [Fisc95], pois com o uso de cláusulas *delay*, cujas informações se referem unicamente ao intervalo de disparo de uma dada transição, não se pode especificar garantias de requisitos temporais fim-a-fim. Isto levou ao desenvolvimento de uma extensão para a linguagem padrão Estelle denominada *Real-Time Estelle* [Fisc96], ou Estelle Tempo-Real.

2.1 A Extensão *Real-Time Estelle*

A sintaxe de *Real-Time Estelle* foi construída visando manter a mesma estrutura de Estelle padrão, o que não ocorre com outras técnicas, cuja inclusão de requisitos temporais alteram significativamente a estrutura da linguagem, como é o caso de uma extensão para SDL descrita em Fischer e Leue [FiLe97]. Todas as restrições de tempo-real referentes a um módulo são colocadas em uma nova seção dentro do corpo, denominada *TIME CONSTRAINTS.*

A descrição das restrições é realizada por fórmulas de lógica temporal, o que levou a incorporação de novas palavras-chave à linguagem, conforme exemplificamos na tabela 2.1.1.

Complemento à linguagem	*Lista de exemplos*
Palavras chave	TIME CONSTRAINTS
Operadores Lógicos	AND OR IMPLIES OTHERWISE EVENTUALLY HENCEFORTH LEADSTO FORBIDS
Descritores simples de estado	SENDING OF RECEIVING OF MAJOR_STATE

Tabela 2.1.1 – Exemplos de elementos da extensão Real-Time Estelle

Na figura 2.1.1 podemos visualizar um trecho de código Real-Time Estellle extraído de Fischer [Fisc96] que exemplifica a especificação de requisitos de tempo. Conforme verificamos no exemplo, a seção "*TIME CONSTRAINTS*" engloba uma restrição temporal referente a um módulo da especificação, denominada "*tc1*". A cláusula "*FORALL*" declara a variável "x" como sendo pertencente ao domínio do tempo. O operador lógico "*HENCEFORTH*" afirma que a fórmula entre parênteses é sempre verdadeira a partir do momento inicial de lançamento do módulo. Por outro lado, a fórmula entre parênteses afirma que qualquer transição do estado *"idle"* para *"connected"* deve acontecer em, no máximo, cinco unidades de tempo.

```
TRANS
     FROM idle TO pending
      begin
        output sap.REQUEST;
      end;

     FROM pending TO connected
        WHEN sap.CONFIRMATION
          begin
          end;

TIME CONSTRAINTS
     NAME tc1:
     FORALL x : time;
          HENCEFORTH (
               MAJOR_STATE(idle)    AT    x    LEADSTO
               MAJOR_STATE(connected) AT (now<=x+5));
```

Figura 2.1.1 – Trecho de código utilizando Real-Time Estelle

2.2 Aspectos Importantes

Utilizando a extensão *Real-Time Estelle* é possível, além de descrever formalmente os aspectos tradicionais dos protocolos, realizar a descrição quantitativa dos requisitos ou restrições de QoS, dentre os quais destacamos:

i) *Taxa de transmissão:* quantidade de banda que a aplicação vai utilizar, que pode ser expressa pelo número de mensagens por unidade de tempo;

ii) *Atraso:* atraso global introduzido durante o processo de transmissão e processamento dos pacotes;

iii) *Jitter: v*ariação do atraso;

iv) *Taxa de Erro:* confiabilidade do sistema, em termos de percentagem de pacotes não transmitidos com sucesso.

Utilizando a especificação de restrições temporais conforme mostrado na sintaxe do Real-Time Estelle todos estes parâmetros podem ser devidamente definidos. Exemplos mais detalhados podem ser encontrados em Fischer e Leue [FiLe97].

Um problema interessante relatado em Fischer [Fisc95] diz respeito à concorrência na linguagem Estelle padrão, que torna-se comprometida devido a hierarquia entre módulos pais e módulos filhos. Neste caso, as regras de disparo determinam que um módulo filho não pode ter nenhuma transição disparada enquanto houver alguma transição sensibilizada no módulo pai. Por outro lado, após o início da seqüência de disparos do módulo filho, o módulo pai deverá aguardar até que não haja mais nenhuma transição sensibilizada nos módulos filhos. Este tipo de situação dificulta a especificação de requisitos típicos de aplicações de alta velocidade.

Uma solução encontrada foi incorporar à linguagem Estelle uma nova palavra chave "***asynchronous***" [BrGo93]. Assim, módulos com o atributo "***asynchronous***" ***systemprocess*** tem uma semântica tal que os módulos filhos deste não obedecem a nenhuma hierarquia em relação aos módulos pais. Desta forma, a execução dos módulos pais torna-se independente dos módulos filhos.

3. O ProtCAD/RT

O sistema que está sendo descrito neste artigo tem como meta principal oferecer ao usuário recursos para efetuar o projeto de um protocolo de comunicação, visando :

i) Permitir a especificação de um protocolo de forma visual e desta obter, como saída, um arquivo textual contendo o código Estelle do modelo descrito;

ii) Compilar o código Estelle, verificando sintaticamente a especificação textual e gerando uma forma intermediária que permita realizar simulações, implementações e, eventualmente, utilizá-la em outros módulos adicionais existentes ou a serem desenvolvidos;

iii) Simular o protocolo de forma interativa, facilitando o processo de depuração do mesmo.

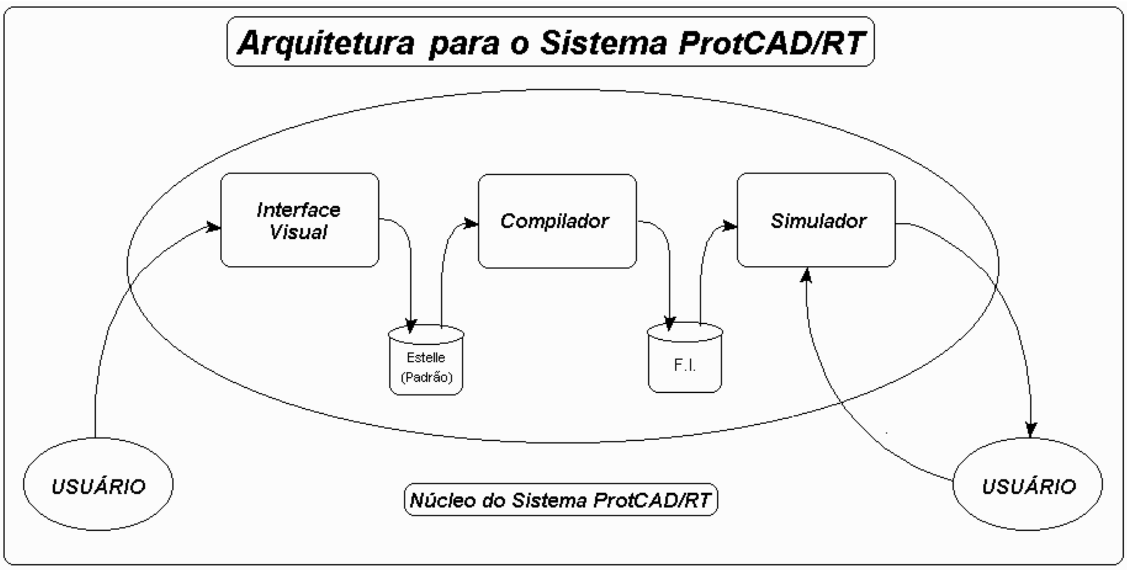

Figura 3.1 – Núcleo do ProtCAD/RT

A arquitetura resultante dos componentes do sistema em questão, é esquematizada na figura 3.1. Nesta podemos visualizar módulos referentes à Interface Visual, ao Compilador e ao Simulador respectivamente. Esses módulos serão detalhados nas subseções a seguir.

3.1 Interface Visual

O objetivo que se pretende alcançar com a construção da interface visual é tornar a especificação de um protocolo algo mais simples, rápido e intuitivo do que a codificação puramente manual, tentando alcançar o maior nível de abstração possível, dadas as características da linguagem Estelle padrão. Não podemos nos esquecer que esta interface deve ser operacionalmente simples também, de modo a não ser ela própria uma outra camada que exigirá uma familiarização demorada e complexa.

3.1.1 Especificando um protocolo de comunicação

A Interface Visual permite ao projetista efetuar a especificação da arquitetura estática dos módulos, dos canais de comunicação, dos pontos de interação e do comportamento interno dos módulos. Para as partes relativas ao tratamento de dados (que serão basicamente trechos de programa em linguagem Pascal), às restrições temporais (que serão expressões de lógica temporal) e ao comportamento dinâmico dos módulos serão oferecidos recursos para que o usuário entre com a parte textual de forma organizada. O código texto gerado deverá estar no formato padronizado Estelle, com as respectivas extensões temporais descritas em Fischer [Fisc96].

Uma implementação de um sistema com propósito parecido ao nosso foi desenvolvida por Goulart [GoPe89]. Neste trabalho o comportamento dos módulos podia ser descrito através de uma notação visual derivada de SDL. Um outro trabalho afim [TRKB98] permitia a especificação dinâmica dos módulos graficamente, resultando em uma forma visual bastante complexa, em virtude de tentar contemplar todas as características da linguagem Estelle.

A especificação de um protocolo se inicia pela identificação do mesmo, seguido do tipo e de alguns outros parâmetros como a escala de tempo, os módulos externos requisitados e a declaração dos procedimentos a serem utilizados, por exemplo. A seguir define-se, no formato da linguagem Pascal, os tipos de variáveis que serão utilizados. Todos estes dados devem ser fornecidos na forma textual, já que não podem ser facilmente representados de forma visual. A interface prevê esta etapa da especificação fornecendo um formulário a ser preenchido, como o exemplo mostrado na figura 3.1.1.1.

Os canais de comunicação são definidos de maneira bastante simples, identificando-se o nome do canal e as primitivas aceitas em cada uma das extremidades, que também devem ser nomeadas. Para a enumeração das primitivas e para a definição do nome do canal e das extremidades serão fornecidas caixas de texto a serem preenchidas pelo usuário. A interface prevê esta etapa da especificação conforme esquematizado na figura 3.1.1.2.

Também será necessário definir os cabeçalhos existentes e que poderão ser utilizados pelos corpos de módulos a serem definidos posteriormente. Para isto deveremos fornecer a identificação, o tipo do módulo, os parâmetros de inicialização, as variáveis exportadas e especificar os seus pontos de interação, que estão associados a extremidades dos canais que foram previamente definidos. A especificação de um cabeçalho de módulo é prevista como esquematizado na figura 3.1.1.3. Note que, neste exemplo, um dos pontos de interação está sendo editado (o que está marcado), de modo que visualizamos dois "*combo-box*" que contém o nome dos canais predefinidos e os nomes das extremidades, respectivamente.

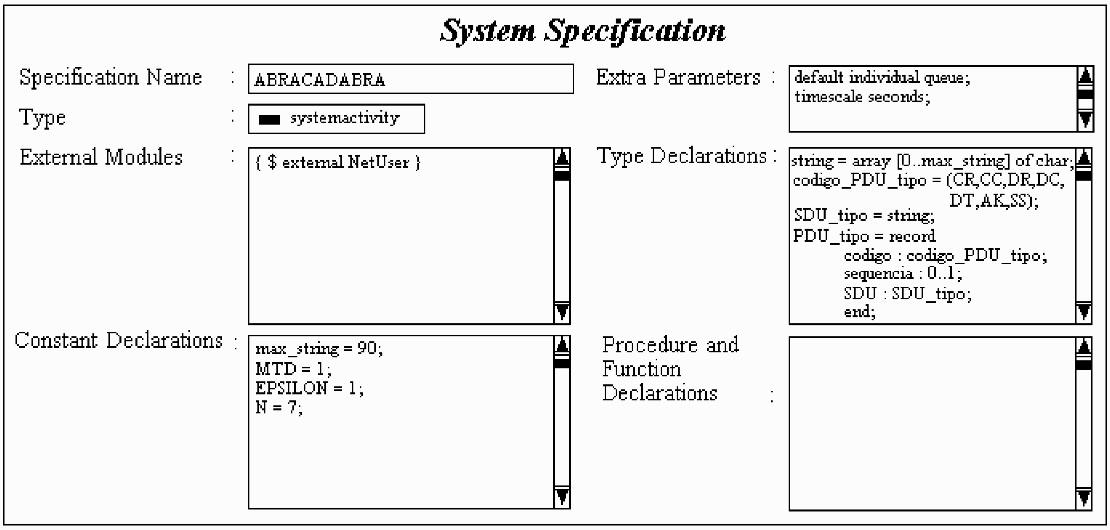

Figura 3.1.1.1 – Iniciando a especificação de um protocolo

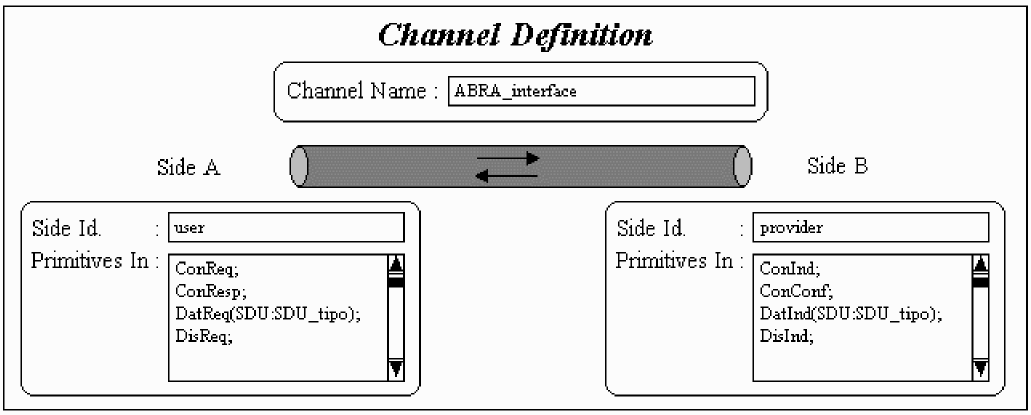

Figura 3.1.1.2 – A especificação de um canal de comunicação

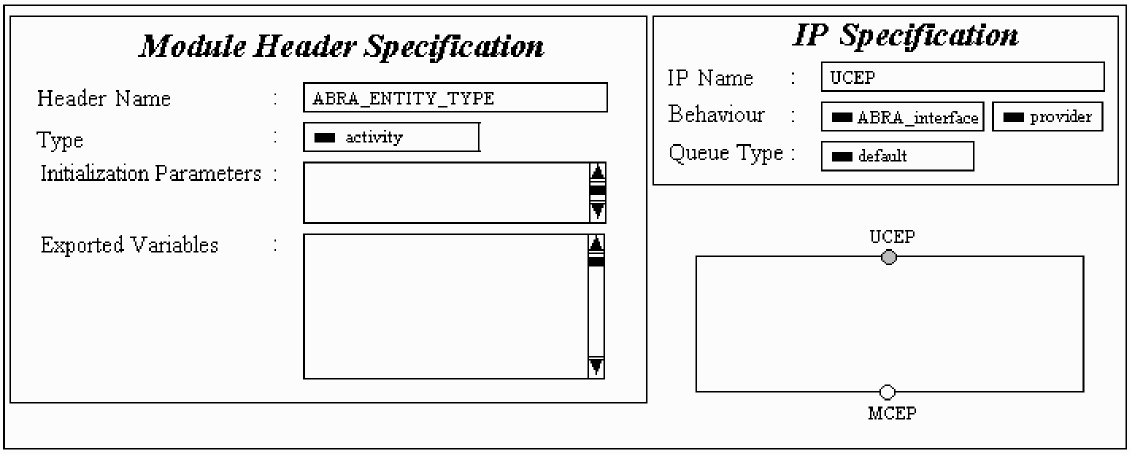

Figura 3.1.1.3 – A especificação de um cabeçalho de módulo

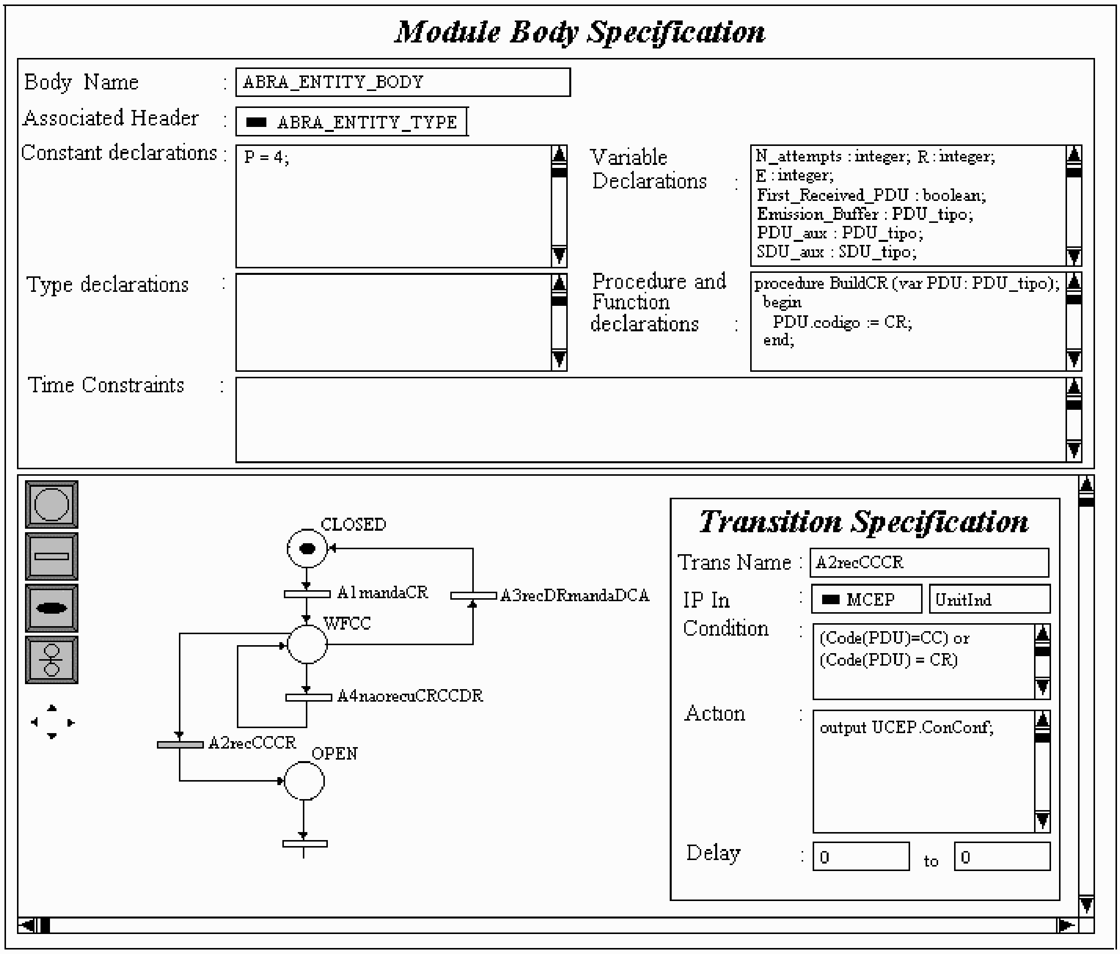

Figura 3.1.1.4 – A especificação de um corpo de módulo

Os corpos de módulos, que estão associados aos cabeçalhos que foram previamente definidos, podem ser especificados de maneira simples. Inicialmente deve-se declarar as constantes, os tipos de variáveis, as variáveis que serão utilizadas, os procedimentos e funções necessárias. O bloco contendo as restrições de tempo-real também é especificado neste momento, através de uma caixa de texto que deve ser preenchida. A seguir deve-se especificar, de forma visual, a MEFE que definirá o comportamento do módulo. Para cada transição inserida deve-se fornecer, através do preenchimento de uma caixa de texto, a condição de disparo e as ações a serem realizadas através de trechos de código em linguagem Pascal.

Para os estados principais deve-se fornecer os apenas os nomes associados. A figura 3.1.1.4 mostra um exemplo de especificação de um corpo de módulo, onde ressaltamos a edição de uma transição (no caso a que está evidenciada). Note que neste caso também temos dois "*combos*" que serão utilizados para selecionar a condição de recebimento de primitivas através dos pontos de interação (que está associada aos pontos de interação definidos no cabeçalho escolhido para este corpo).

Finalmente, após a definição de todos os módulos e canais constituintes do sistema que se quer especificar, deve-se especificar, o comportamento dinâmico dos módulos, definindo de que maneira eles devem ser lançados ou finalizados e as conexões entre os mesmos. Em uma área de trabalho que é aberta após se clicar no botão apropriado deve-se preencher as caixas de texto com as especificações requisitadas, conforme é esquematizado na figura 3.1.1.5.

Todos os trechos de código visualizados nas figuras presentes nesta subseção foram retirados da especificação do protocolo "ABRACADABRA" [GoPe89].

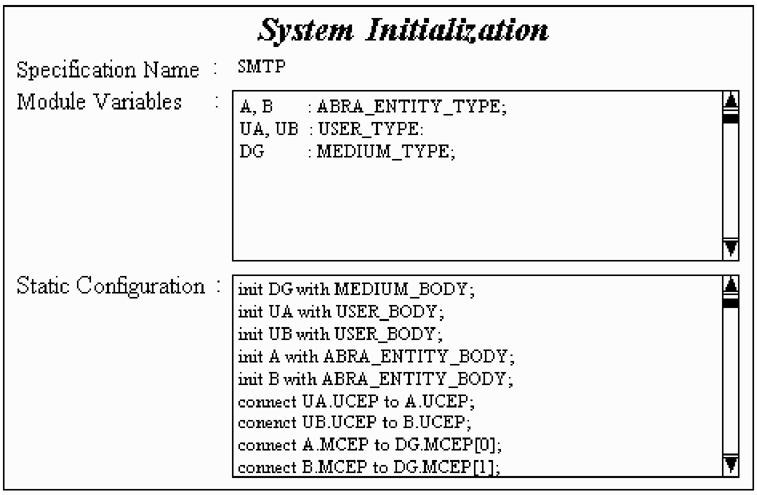

Figura 3.1.1.5 – Especificação da configuração estática do sistema

3.2 Compilador de EstelleTempo-Real

O compilador para protocolos descritos em Estelle com restrições temporais tem por finalidade gerar as formas intermediárias (FI) que servem de arquivos de entrada para o Simulador e, eventualmente, para outros módulos do sistema que venham a ser desenvolvidos. Estas formas seguem o padrão definido em Oliveira [Oliv91]. Neste contexto, a especificação em Estelle de um determinado protocolo, gera duas FIs que descrevemos a seguir:

i) *FI estática*: contém os dados referentes ao sistema de transições do protocolo, como seus estados, seus módulos constituintes, condições de habilitação das transições, entre outros. Em suma, trata-se de uma formalização textual da regras do sistema de transições do protocolo, bem como a estrutura hierárquica desta especificação.

ii) *FI dinâmica*: contém as informações a respeito do protocolo como as ações efetuadas durante o disparo de uma transição. Além disto, são introduzidas as funções e procedimentos em Pascal referentes à manipulação dos dados de sistemas e subsistemas referidos na especificação.

Com base nesta descrição podemos dizer que a FI estática determina como o protocolo progride, e a dinâmica, o que ele realiza durante esta progressão. A FI estática é na verdade um arquivo textual com uma sintaxe própria descrita em Oliveira [Oliv91]. A FI dinâmica seria uma tradução das ações efetuadas pelos módulos de Estelle para uma linguagem de programação existente, o que permite a geração de arquivos executáveis.

Existem outros compiladores Estelle, que seguem uma abordagem ligeiramente diferente, tendo objetivos diversos dos nossos. O compilador PET (Portable Estelle Translator) [SiSt91a], por exemplo, visa gerar uma forma intermediária portável que permita realizar o desenvolvimento de ferramentas baseadas na TDF Estelle e que o utilize como ponto de partida. Uma destas ferramentas, o XEC (eXperimental Estelle Compiler) [ThGo98a], se vale de uma versão modificada do PET, e tem por finalidade gerar implementações distribuídas da maneira mais eficiente possível, tornando possível a realização de experiências de implementação com a linguagem Estelle. Este sistema tem suporte à uma extensão de Estelle que permite a especificação de sistemas distribuídos abertos (Open Estelle [ThGo98b]).

Como a meta deste compilador é a geração de código que permita realizar a simulação de protocolos com restrições temporais, decidimos por efetuar uma implementação deste compilador baseada na estrutura original do compilador do CAD-Estelle [Oliv91]. Desta forma, é necessário incluir o tratamento dado às restrições temporais, realizando modificações na parte do sistema que realiza a geração de código, que, no caso, são as formas intermediárias. Em virtude das características inerentes à sintaxe e à semântica das restrições temporais, estas serão mapeadas em objetos da FI estática. Podemos verificar esquematicamente este processo na figura 3.2.1.

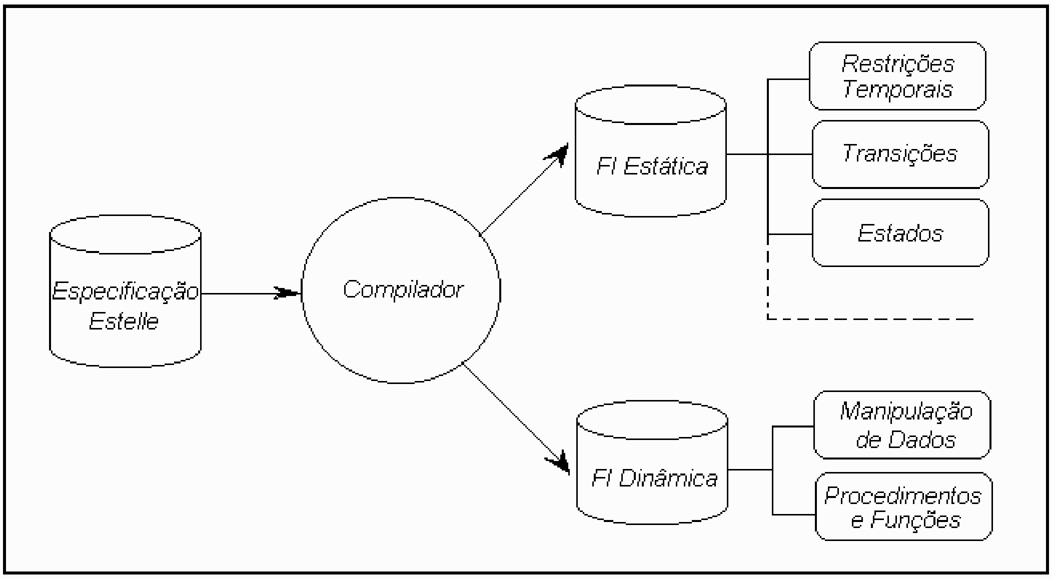

Figura 3.2.1 – Compilação de um Protocolo com Restrições Temporais

3.3 Simulador de Protocolos em Tempo Real

O objetivo do simulador é propiciar ao projetista uma maneira deste certificar-se do perfeito funcionamento do protocolo, fazendo uma análise lógica do seu comportamento antes de iniciar a fase de implementação num determinado sistema hospedeiro.

Durante a simulação são criadas instâncias de módulos que podem, eventualmente, ser destruídas e recriadas novamente. A criação de uma instância de módulo corresponde à criação de um conjunto de instâncias de objetos associados ao dado módulo, tais como as filas correspondentes aos pontos de interação, as variáveis e os módulos declarados em seu interior.

A arquitetura do simulador está dividida em um conjunto de módulos que se relacionam segundo uma regra hierárquica [Vali90].

Requisitos de tempo quantitativos devem ser avaliados na comunicação entre os módulos, fazendo com que seja necessária a implementação de um mecanismo que possibilite detectar quando uma mensagem sai de um módulo e chega a um outro. As mensagens devem ser monitoradas também dentro do mesmo módulo, verificando se foram satisfeitos os requisitos de tempo para aquela mensagem ou ação a ser executada.

O motor do simulador é responsável pela execução do modelo de base do ambiente de simulação. O simulador cria filas interconectando as diversas Máquinas de Estados Finitos.

O motor usa o algoritmo básico de simulação de sistemas a eventos discretos. É criada uma lista de eventos onde serão colocados os eventos que ocorrerão em um tempo futuro, vale ressaltar que a fila de eventos está organizada cronologicamente.

Na fase de inicialização, coloca-se o evento de tempo igual a 0 (zero) na fila de eventos, iniciando o estado global. O algoritmo executa o seguinte laço:

i) Retira os eventos que ocorrem num mesmo instante da lista de eventos, atualiza o estado global e acerta o relógio;

ii) Para o estado global atual, cria uma lista de todas as transições habilitadas;

iii) Escolhe uma transição desta lista para ser disparada, esta tarefa poderá ser feita pelo usuário;

iv) Executar a ação correspondente, colocando os eventos gerados na fila de eventos, inclusive o evento de término de execução da ação

v) Depois da ação referente ao disparo da transição ser executada, as restrições temporais são avaliadas. O modelo passa então a um próximo estado;

vi) É feito o teste do ponto de parada. Se o teste é verdadeiro, o controle é passado para o usuário, senão, retorna ao item (i).

Além do simulador gastar um tempo na execução da ação relacionada a uma dada transição, ele também escala o evento que determina o término do disparo desta transição na fila de eventos baseado no desempenho modelado do sistema-alvo.

As operações de envio de mensagens também vão produzir um evento de chegada da mensagem ao seu destino baseado na modelagem do desempenho do sistema de comunicação.

O simulador oferece ao usuário, entre outras, às seguintes facilidades:

- Inserção de pontos de parada e obtenção de traços de simulação;
- Execução passo a passo da simulação, permitindo ao usuário escolher a transição que irá ser disparada, facilitando o processo de depuração do protocolo.
- Introdução de seqüências de teste durante uma parada programada da simulação;
- Acesso às medidas dos requisitos de tempo-real: *Atraso, Taxa de Transmissão , Jitter e Taxa de Erros*, fornecendo um quadro comparativo entre os requisitos especificados e os obtidos durante a simulação.

4. Ambiente de Implementação

O sistema operacional Linux foi escolhido por ser uma plataforma de distribuição gratuita e com arquitetura aberta (filosofia "*freeware*") que possui versão para ser utilizada em computadores da linha Intel ou compatíveis (a partir do 386), para o qual encontramos amplo suporte divulgado e documentado na própria **INTERNET** e por se constituir em um sistema "*UNIX-Like*" bastante eficiente. O desenvolvimento deste sistema em um ambiente de distribuição gratuita e aberta pode minimizar os riscos de que sejam lançadas novas versões incompatíveis com as anteriores, tornando o sistema obsoleto. Além disso é notória a rápida difusão deste sistema operacional no meio acadêmico e comercial.

A necessidade de se criar um sistema independente de plataforma e de se divulgar mais amplamente o trabalho nos levou a escolher a **INTERNET** como meio de acesso e utilização deste sistema. A escolha de um navegador visual qualquer como elemento de interação com o usuário é natural, pois ele fornece recursos visuais e textuais suficientes para o nosso propósito, além de ser bastante utilizado para navegação na **INTERNET**.

Para gerar as formas e manipulações visuais requeridas, obter um nível elevado de interação entre o usuário e o ambiente de desenvolvimento e gerar uma aplicação que pudesse ser executada independentemente de plataforma, optou-se por utilizar a linguagem **JAVA**. Esta linguagem permite também a execução local na máquina "cliente", razão pela qual podemos projetar aplicações mais interativas.

5. Comentários

Este trabalho apresentou a descrição de um sistema de desenvolvimento de protocolos com a possibilidade do uso de restrições temporais utilizando a TDF Estelle e uma extensão desta técnica para tempo real. Este sistema possui um núcleo composto por três módulos : uma interface visual de especificação, um compilador tempo real e um simulador tempo real.

A interface visual irá gerar uma especificação textual Estelle que será compilada e simulada com o auxílio dos outros dois módulos.

A parte relativa à Interface Visual do **ProtCAD/RT** está sendo implementada de forma a permitir a sua utilização através da **WEB**, oferecendo alta interatividade e independência de plataforma devido a escolha da linguagem **JAVA**. Esta arquitetura possibilita a execução remota, sendo adequada para o ambiente acadêmico e passível de divulgação e utilização via **INTERNET**.

O sistema **ProtCAD/RT**, que está em fase de desenvolvimento, leva em consideração a especificação de restrições temporais. Além disso, a concepção dentro de uma filosofia mais aberta deverá garantir uma sobrevida maior do que a do sistema atual em termos de ambiente de implementação.

Atualmente, já está em fase final de desenvolvimento um implementador semi-automático que contempla as restrições tempo real e que poderá ser perfeitamente integrado ao novo conjunto de ferramentas. Com base neste sistema, outros módulos adicionais poderão ser desenvolvidos futuramente. Dentre estas opções podemos destacar um analisador de desempenho de protocolos e um verificador de propriedades relacionadas aos parâmetros de **QoS**.

6. Referências

[Diaz89] DIAZ, M., et al. *The Formal Description Technique ESTELLE.* 1 ed.North-Holland.USA, 1989.

[SiSt91a] SIJELMASSI, R., STRAUSER, B. *The Portable Estelle Translator: an Overview and User Guide*. Technical Report of National Institute of Standards and Technology, System and Network Architechture Division, Gaithersburg, USA, janeiro 1991.

[SiSt91b] SIJELMASSI, R., STRAUSER, B. *The Distributed Implementation Generator: an Overview and User Guide*. Technical Report of National Institute of Standards and Technology, System and Network Architechture Division, Gaithersburg, USA, janeiro 1991.

[EDT99] INSTITUT NATIONAL DES TELECOMMUNICATIONS. *Estelle Development Toolset.* http://www-lor.int-evry.fr/lor/edt/, setembro 1999.

[GoPe89] GOULART, C. C., PEDROZA, A. C. P. "Proposta de uma Forma Gráfica para a Linguagem Estelle" In: *7º Simpósio Brasileiro de Telecomunicações 7º SBT.* Fortaleza, setembro 1989.

[Oliv91] OLIVEIRA Jr., R. C. *Um Compilador para a Linguagem Estelle.* Tese MSc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, 1991.

[Lour95] LOUREIRO, F. R. *Um Simulador para a Linguagem Estelle.* Tese MSc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, 1995.

[Vali90] VALIM P. R. O. *Um Simulador de Protocolos de Comunicação para a Linguagem Estelle.* Tese MSc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, 1990.

[BrGo93] BREDEREKE J., GOTZHEIN R. "Increasing the Concurrency in Estelle" In: 6th International Conference on Formal Description Techniques (FORTE 93), Boston, EUA, 1993.

[TRKB98] TEMPLEMORE-FINLAYSON J., RAFFY J., KRITZINGER P., BUDKOWSKI S. "A Graphical Representation and Prototype Editor for the Formal Description Technique Estelle" In: FORTE/PSTV 98, França, novembro 1998.

[ThGo98a] THEES J., GOTZHEIN R. "The eXperimental Estelle Compiler – Automatic Generation of Implementations from Formal Specifications" In: The 2nd Workshop of Formal Methods in Software Practice (FMSP 98), Clearwater Beach, Florida, EUA, março 1998.

[ThGo98b] THEES J., GOTZHEIN R. "Open Estelle – An FDT for Open Distributed Systems" In: Formal Description Techniques and Protocol Specification, Testing and Verification (FORTE XI/PSTV XVIII), 1998.

[PGVO89] PEDROZA A. C. P., GOULART C. C., VALIM P. R. O., OLIVEIRA JR. R. C. "Um Sistema de Auxílio ao Projeto de Protocolos de Comunicação para Redes de Computadores" In: Seminário Franco-Brasileiro de Sistemas de Informação Distribuídos (SFBSID 89), Florianópolis, SC, Brasil, setembro 1989.

[FiLe98] FISCHER S., LEUE S. "Formal Methods for Broadband and Multimedia Systems" In: Computer Networks & ISDN Systems, vol. 30, nº 9-10, p. 865-899, 1998.

[Fisc95] FISCHER S. “On the Suitability of Estelle for Multimedia Systems” Relatório Técnico 5/95, Universität Manheim, Germany, 1995.

[Fisc96] FISCHER S. “Real-Time Estelle” Relatório Técnico 4/96, Universität Manheim, Germany, 1996.

[Srin98] SRINIVASAN B. *“A Firm Real-Time System Implementation using Commercial Off-The-Shelf Hardware and Free Software”* Tese MSc., Department of Electrical Engineering and Computer Science, University of Kansas, Lawrence , Kansas, EUA, 1998.