JOSÉ MARIA NOVAES DOS SANTOS

UM FORMALISMO ADAPTATIVO COM MECANISMO DE SINCRONIZAÇÃO PARA APLICAÇÕES CONCORRENTES

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Mestre em Engenharia

São Paulo
1997

JOSÉ MARIA NOVAES DOS SANTOS

UM FORMALISMO ADAPTATIVO COM MECANISMO DE SINCRONIZAÇÃO PARA APLICAÇÕES CONCORRENTES

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Mestre em Engenharia

Área de Concentração:
Engenharia de Computação

Orientador:
Prof. Dr. João José Neto

São Paulo
1997

À minha esposa Juliana e
ao nosso filho Matheus

AGRADECIMENTOS

Ao Professor Dr. João José Neto, pela orientação prestativa e segura possibilitando o desenvolvimento deste trabalho.

Aos meus familiares, pelo apoio e incentivo.

Ao Professor Dr. José Sidnei Colombo Martini, pelo incentivo e ajuda prestados.

SUMÁRIO

RESUMO

Este trabalho apresenta uma contribuição para especificação de um conjunto de sistemas reativos complexos, sincronizados entre si. Esse formalismo, chamado ***Stad-Sinc*** se baseia numa composição das notações Rede de Petri, Statechart convencional e Statechart Adaptativo, e é representado por um diagrama que explicita os mecanismos de sincronização.

Desenvolve-se uma ferramenta para desenvolvimento e análise, chamado SAS, contituído de um editor de Statecharts e um simulador de Statecharts sincronizados.

Uilizando o Stad-Sinc um sistema pode ser considerado de diversos pontos de vista:

- Nos aspectos de hierarquia e concorrência, usam-se as características do Statechart
- Nos aspectos de sincronização, são utilizados os mecanismos das: Redes de Petri, e,
- Nos aspectos de aprendizagem, ose mecananismos de auto-modificação dos Statecharts Adaptativo

Com isso o formalismo permite a representação de cada um desses aspectos isoladamente, bastando omiti-lo se for desnecessário. Isso permite utilizar apenas os recursos estritamente necessários em cada aplicação, proporcionando maior clareza na notação, e portanto facilitando a sua utilização e manutenção.

1. CONCEITOS

Neste capítulo são apresentados um problema da Engenharia de Software e de Sistemas (que será o tema dessa dissertação), algumas soluções conhecidas na literatura e qual a proposta dessa dissertação.

1.1 Introdução

Um problema reconhecido pela Literatura na área de Engenharia de Software e de Sistemas é a especificação de sistemas reativos complexos de forma clara e formal. Um sistema reativo interage continuamente com o meio externo através de estímulos de entrada e saída. Uma maneira formal e clara de se descrever estes sistemas, é a utilização de estados e eventos (Harel, 1987). O formalismo Máquina de Estado Finito e diagramas de transição correspondentes, muitas vezes não resolvem os sistemas complexos de forma clara, devido à complexidade de seus diagramas resultantes.

Statechart é um formalismo gráfico, baseado em Máquinas de Estado Finito, utilizado para a especificação de sistemas reativos. O Statechart apresenta características que tornam seus diagramas mais concisos e claros que os equivalentes das Máquinas de Estados Finitos (Harel, 1987).

Autômato Adaptativo é um conceito baseado em Máquinas de Estados, que apresenta uma característica adaptativa, que permite a alteração dinâmica das propriedades do Autômato (José Neto, 1993).

Rede de Petri é um formalismo para sistemas de tempo real com características apropriadas à representação de sincronismo (Peterson, 1977).

Esta dissertação se baseia nesses três formalismos por serem importantes do ponto de vista teórico, e muito utilizados para a especificação formal e clara dos problemas a resolver.

Embora esses três formalismos sejam aplicáveis na especificação de sistemas reativos complexos, em alguns casos eles não propiciam uma solução gráfica simples, pois cada um possui características notacionais próprias. Essa dissertação propõe um formalismo que se aproveita das três características, cada qual empregada na situação em que se mostra mais adequada. Procura-se com isso porpor um formalismo que tenha uma visualização gráfica mais simples (quando comparado com as técnicas tradicionais), quando aplicado na especificação de um conjunto de sistemas reativos complexos e de tempo real, que envolvam operações de sincronização.

Ao definir um sistema, deparamo-nos com a necessidade de resolução de sua especificação, ou seja, da definição do comportamento do sistema em relação ao meio externo. Esta definição é muitas vezes chamada Especificação de Requisitos do Software (SRS - *Software Requirements Specification*). Sabe-se que a linguagem natural (Inglês, Português, etc.) apresenta muitos aspectos de ambigüidade, incompleteza e inconsistência. Como os softwares estão sendo usados cada vez mais em aplicações críticas, sistemas que podem comprometer a integridade física de pessoas como controle de tráfego aéreo, usina nuclear etc.; a redução de tais deficiências torna-se cada vez mais importante.

Segundo (Davis, 1988) um SRS deve ser elaborado com os seguintes propósitos:

- conter uma descrição completa sobre as funções a serem executadas pelo Software, sem descrever a forma como tais funções serão executadas,
- servir como base para toda a atividade de projeto, e
- servir de base para todos os testes do Sistema.

Atualmente, existem várias técnicas com este propósito. Neste capítulo, serão revistas as seguintes técnicas: Máquina de Estados Finitos, Tabela de Decisão, PDL (Program Design Language), Análise Estruturada, Statechart, Redes de Petri e Autômato Adaptativo.

O objetivo deste capítulo é apresentar algumas técnicas, utilizadas para a definição do aspecto comportamental de um sistema reativo, que são amplamente publicadas na literatura de engenharia de software: Máquina de Estados Finitos, Tabela de Decisão, Análise Estruturada, Statechart, Statechart Adaptativo, Redes de Petri e Autômato Adaptativo.

O segundo capítulo apresenta a definição de um formalismo proposto (STAD-Sinc) para a especificação de sistemas contendo sincronizações e alguns exemplos ilustrando a sua utilização.

O terceiro capítulo apresenta a definição da ferramenta desenvolvida (SAS) para a edição e simulação de sistemas contendo sincronizações e alguns exemplos ilustrando a utilização da ferramenta.

O quarto capítulo apresenta as conclusões dessa pesquisa, a sua contribuição e a indicações de futuras pesquisas futuras que poderão ser efetuadas.

O anexo A apresenta o manual de utilização da ferramenta SAS.

O anexo B apresenta a estrutura das tabelas utilizadas para o armazenamento das informações do sistema SAS.

Finalizando, são apresentadas as referências bibliográficas utilizadas na elaboração dessa pesquisa.

1.2 Sistemas Reativos

Um sistema reativo tem a característica de reagir continuamente ao meio externo em resposta a estímulos. Internamente ao sistema definem-se eventos, associados aos estímulos, por isso dizemos que são sistemas dirigidos por eventos. Os sistemas apresentam eventos e condições. Esses sistemas são comuns, pois temos inúmeros exemplos de interação entre um agente externo (por exemplo o homem) e um sistema como: televisão, vídeo cassete, estação meteorológica (nesse caso as condições climáticas interagem com o sistema), usina nuclear, etc.

Quando um sistema tem como característica apenas a reação a condições de entrada para que se produza a saída (resultado) dizemos que ele é um sistema transformacional. Como exemplo podemos citar: um programa que fornece número aleatório, um programa que fornece os juros cobrados a partir dos valores informados (valor normal e valor com juros), etc.

Nessa dissertação discutiremos os sistemas reativos, pois os sistemas transformacionais já dispõem de muito mais metodologias e ferramentas de especificação amplamente utilizados.

1.3 Máquinas de Estados Finitos

Máquinas de Estados Finitos são dispositivos utilizados para reconhecimento de linguagens regulares (geradas por gramáticas regulares ou descritas por expressões regulares).

Uma Máquina de Estados Finitos representa um sistema como um conjunto de configurações (também denominadas estados), cada qual identifica uma característica do sistema. Para cada configuração a Máquina de Estado Finito identifica as condições (estímulos) a que o sistema reage, indicando também a nova configuração que o sistema deverá assumir em decorrência da reação do sistema aos estímulos.

Definição

Máquinas de Estados Finitos são dispositivos que, em um determinado instante, ocupam apenas um estado entre os possíveis, e respondem a um conjunto de entradas. A cada resposta o sistema reage realizando uma transição para um novo estado.

Formalmente, uma máquina de Estados Finitos M é M=(Q, ,P,q_O,F) onde:

Q – É um conjunto não vazio de estados

– É um conjunto não vazio de estímulos (ou ações) de entrada

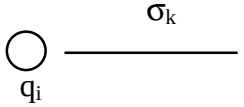
P – É uma função P: Qx que associa a cada par (estado, estímulo) um novo estado, que a máquina M passa a assumir

q_o – É o estado inicial ($q_o \in Q$)

F – Conjunto de estados finais ($F \subseteq Q$)

Uma descrição mais detalhada sobre as Máquinas de Estados Finitos pode ser encontrada em (Lewis, 1981).

Uma notação gráfica usada para representar as máquinas de Estados Finitos é o diagrama de Transição de Estados, cujos elementos são:

- círculos: representam os estados. O círculo identificado com uma seta no canto superior esquerdo, é denominado estado inicial. Os círculos com espessura maior são denominados estados finais.
- arco: conecta dois círculos, representando uma transição possível entre os estados.
- rótulo do arco - indica os eventos associados à transição.
-  estado inicial q_0
-  - estado não final ($q \in Q$, e $q \notin F$)
-  - estado final ($q \in F$)
-  - $\sigma \in \Sigma$, $P(qi, \sigma_k) = q_j$

Exemplo

Vamos considerar como exemplo um sistema de controle de ligações telefônicas em uma empresa em que podem ser feitas chamadas externas ou internas. O sistema a ser descrito se compõe de um telefone, que pode estar inerte (sem funcionamento) ou em tentativa de realizar uma ligação, interna ou externa à empresa. Ao se tentar realizar a ligação o sistema pode receber um sinal de linha ocupada, não completando-a. Quando se conseguir o sinal de chamada (destinatário recebendo o chamado) o sistema efetuará a conexão, início de conversação, somente após o atendimento do destinatário. O diagrama de transições do sistema está apresentado na figura 1.1.

Neste caso temos:

- Estado inicial q_0 - *Inativo*

- Estados $Q=\{q_0, q_1, q_2, q_3, q_4, q_5\}$

q_0 - *Inativo*	q_3 - *Linha Ocupada*
q_1 - *Ligação Externa*	q_4 - *Tocando*
q_2 - *Pronto para Discar*	q_5 - *Conectado*

- Estado final - q_0

$\Sigma = \{ \sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6, \sigma_7\}$

σ_1 - *Gancho retirado*	σ_5 - *Tocando*
σ_2 - *Ligação Externa*	σ_6 - *Conexão efetuada*
σ_3 - *Ligação Disponível*	σ_7 - *Gancho colocado*
σ_4 - *No. Linha Ocupada*	

- Produções (P)

$P(q_0, \sigma_1) = q_1$	$P(q_1, \sigma_2) = q_2$	$P(q_2, \sigma_3) = q_4$	$P(q_4,\sigma_6) = q_5$
$P(q_1, \sigma_7) = q_0$	$P(q_2, \sigma_7) = q_0$	$P(q_3, \sigma_7) = q_0$	$P(q_5, \sigma_7) = q_0$
$P(q_1, \sigma_3) = q_4$	$P(q_2, \sigma_4) = q_3$	$P(q_4, \sigma_7) = q_0$	

- Estado final - q_0

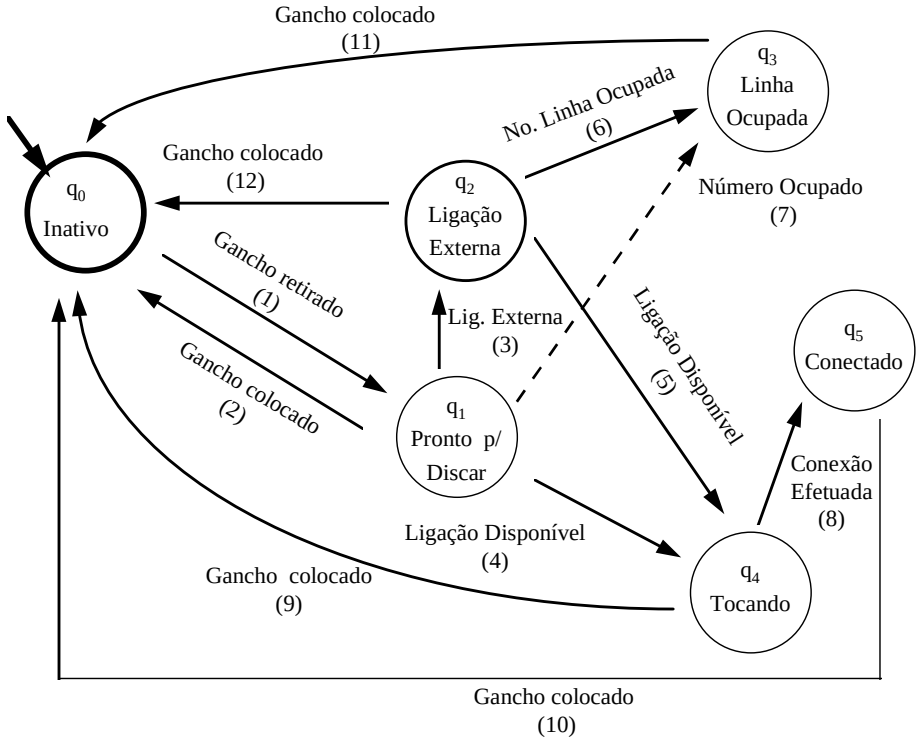


Fig 1.1 - Exemplo de um Diagrama de Estados Finitos

A seguir, são descritas as reações do sistema de acordo com o estado que ocupa.

a. Estado: q_0 - *"Inativo"*

É o estado inicial. Quando o ambiente externo gerar o estímulo (evento) *"Gancho Retirado",* o sistema efetuará a transição (1), passando ao estado *"Pronto p/ Discar"*.

b. Estado: q_1 - *"Pronto p/ Discar"*

Neste estado, se o ambiente externo gerar o evento *"Gancho colocado"*, o sistema efetuará a transição (2) voltando ao estado inicial *"Inativo"*. Se for gerado o evento *"Lig. Externa*", o sistema efetuará a transição (3), passando ao estado *"Ligação Externa"*. Se for gerado o evento *"Ligação Disponível"*, o sistema efetuará a transição (4), passando ao estado *"Tocando"*.

c. Estado: q_2 -*"Ligação Externa"*

Neste estado, se o ambiente externo gerar o evento *"Gancho colocado"*, o sistema efetuará a transição (12), voltando ao estado inicial *"Inativo"*. Se for gerado o evento *"No. Linha Ocupada*", o sistema efetuará a transição (6) passando ao estado *"Linha Ocupada"*. *S*e for gerado o evento *"Ligação Disponível*", o sistema efetuará a

transição (5), passando ao estado *"Tocando"*.

d. Estado q_3: "*Linha Ocupada*"

Neste estado, se o ambiente externo gerar o evento "*Gancho colocado"*, o sistema efetuará a transição (11), voltando ao estado inicial "*Inativo"*.

e.Estado q_4: "*Tocando"*

Neste estado, se o ambiente externo gerar o evento "*Gancho colocado"*, o sistema efetuará a transição (9), voltando ao estado inicial "*Inativo"*. Se for gerado o evento "*Conexão Efetuada*", o sistema efetuará a transição (8) passando ao estado *"Conectado"*.

f. Estado q_5: "*Conectado"*

Neste estado, se o ambiente externo gerar o evento "*Gancho colocado"*, o sistema efetuará a transição (10), voltando ao estado inicial "*Inativo"*.

A Máquina de Estado Finito é o dispositivo mais simples de reconhecimento de linguagens (do ponto de vista computacional) e pode também ser utilizada na especificação (aspecto comportamental) de sistemas simples, que não envolvam aspectos temporais.

1.4 Tabela de Decisão / Árvore de Decisão

- **Tabela de Decisão**

A tabela de Decisão pode ser de grande utilidade na especificação do aspecto comportamental de sistemas quando as combinações das condições envolvidas são muito complexas tornando a representação através de Máquina de Estados Finitos de difícil compreensão. A utilização de uma associação entre um conjunto de condições e suas respectivas ações se torna mais apropriada nesses casos.

Uma especificação de sistema através de uma tabela de decisão, consiste em definir as ações (respostas) que o sistema deverá efetuar como reação a cada possível conjunto de condições (estímulos) previamente definido gerados pelo meio externo

Existem sistemas com a característica de emitir diferentes reações (respostas aos estímulos do meio externo) a cada conjunto de condições, sendo então mais apropriada a especificação através da árvore de decisão (Davis, 1988).

Definição

Formalmente, uma tabela de decisão T é T=(C,A,R) onde:

C - Conjunto não vazio de condições (estímulos)

A - Conjunto não vazio de ações

R - R: C $\rightarrow$ A. É uma aplicação que associa sub-conjuntos das condições a ações que o sistema deve efetuar.

Para se criar uma tabela de decisão procede-se da seguinte forma:

a) definem-se as linhas com as condições (ou estímulos) usados pelo sistema

b) cada coluna representa uma combinação das condições

c) adicionam-se novas linhas representando as ações que o sistema deve tomar

Se uma linha estiver marcada (em uma determinada linha/coluna) com um *"V"*, indica que:

- se a linha representar uma condição: esta condição deve estar presente no conjunto representado pela coluna.

- se a linha representar uma ação: esta ação deverá ser realizada (quando todas as condições da coluna estiverem satisfeitas).

Exemplo

Vamos considerar como exemplo, o controle de uma porta automática de um caixa eletrônico de banco. Por segurança, o sistema deverá fechar a porta do caixa eletrônico na seguinte situação:

a) quando exceder o limite de tempo (por exemplo 5 segundos),

b) a porta está aberta e

c) o sensor (que indica se há alguém na frente da porta) não estiver sinalizado.

Sempre que o sensor estiver sinalizado (indicando a presença de uma pessoa querendo entrar no caixa eletrônico) e o caixa eletrônico estiver vazio, o sistema deverá abrir a porta.

Neste caso temos:

C = $\{c_1,c_2,c_3,c_4\}$, onde:

c_1 = Tempo Limite atingido (5 segundos)

c_2 = Botão acionado

c_3 = Local Vazio

c_4 = Porta Aberta

A = $\{a_1,a_2\}$, onde:

a_1 = Abrir a Porta

a_2 = Fechar a Porta

R =

$\{c_2,c_3\}$	$\{a_1\}$
$\{c_1,c_4\}$	$\{a_2\}$
$\{c_1,c_3,c_4\}$	$\{a_2\}$

Na figura 1.2 apresentamos a tabela de decisão para o sistema descrito acima.

Tempo Limite Atingido		V
Botão Acionado	V	
Local Vazio	V	
Porta Aberta		V
Abrir a Porta	V	
Fechar a Porta		V

Fig. 1.2 - Exemplo de uma Tabela de Decisão

- **Árvore de Decisão**

Pode-se criar uma árvore de decisão equivalente a uma tabela de decisão usando um fluxograma, onde cada condição da tabela é apresentada como uma pergunta do fluxograma e cada ação da tabela, corresponderá também a uma ação no fluxograma, na ramificação associada às respectivas condições da tabela.

A figura 1.3 representa a árvore de decisão equivalente à tabela da figura 1.2.

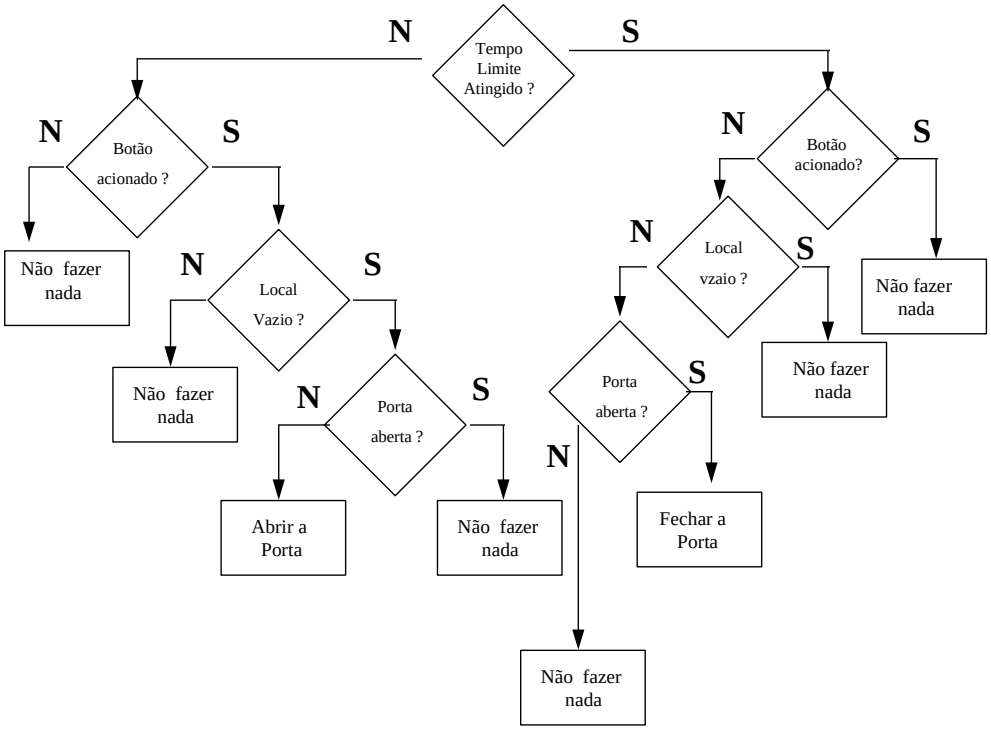


Fig. 1.3 - Exemplo de uma Árvore de Decisão

Recomenda-se portanto o uso da Tabela de Decisão para se definir as partes do sistema que envolvem muitas condições em sua lógica.

1.5 Diagrama de Fluxo de Dados

O Diagrama de Fluso de Dados (DFD) é utilizado geralmente quando se deseja ressaltar os processos internos e o fluxo das informações envolvidas em cada um.

Um Diagrama de Fluxo de Dados representa graficamente como os processos do sistema geram e manipulam as informações.

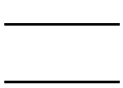
Definição

Um Diagrama de Fluxo de Dado (DFD) é uma notação gráfica representando os processos (funções) que o sistema exerce sobre os dados. Cada processo é representado identificando os dados de entrada e saída. Este tipo de diagrama mostra as transformações dos dados através dos processos. Neste diagrama, temos os seguintes elementos:

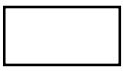
- Função / Processo: são representados por círculos. São as funções (processos) que transformam as informações (Dados) da entrada, gerando a partir deles a saída (Dado /

Informação).

- Fluxo: são representados por arcos direcionados interligando um processo com um outro elemento (Processo, Terminador ou Depósito de Dados). Quando a seta aponta para o processo, indica que a informação é uma entrada para o mesmo. Caso contrário, ela indica que a informação é uma saída do processo.



- Depósito de Dados: Identifica as informações que são armazenadas no diagrama. São representados por um par de linhas paralelas.



- Terminadores: Identificam as entidades externas que geram os estímulos (ações) do sistema. São representados por retângulos.

Para cada ocorrência de um desses elementos no diagrama associa-se um nome, identificando-o (por exemplo o nome do fluxo a que se refere, o nome do processo, etc.).

Exemplo

No exemplo da figura 1.4, temos a definição de um sistema destinado a emitir uma relação de produtos de um loja para os seus clientes, informando as características do produto e seu preço. Neste exemplo temos :

a) Função / Processo: *Cadastro de Produto, Cadastro de Cliente, Lista de Material*

b) Fluxo: *Solicita cadastro de Produto, Inf-Produto, Solicita cadastro de Cliente, Inf-Cliente, Lista de Produto*

c) Depósitos de dados: *Produto, Clientes*

d) Terminadores: *Operador* e *Cliente*

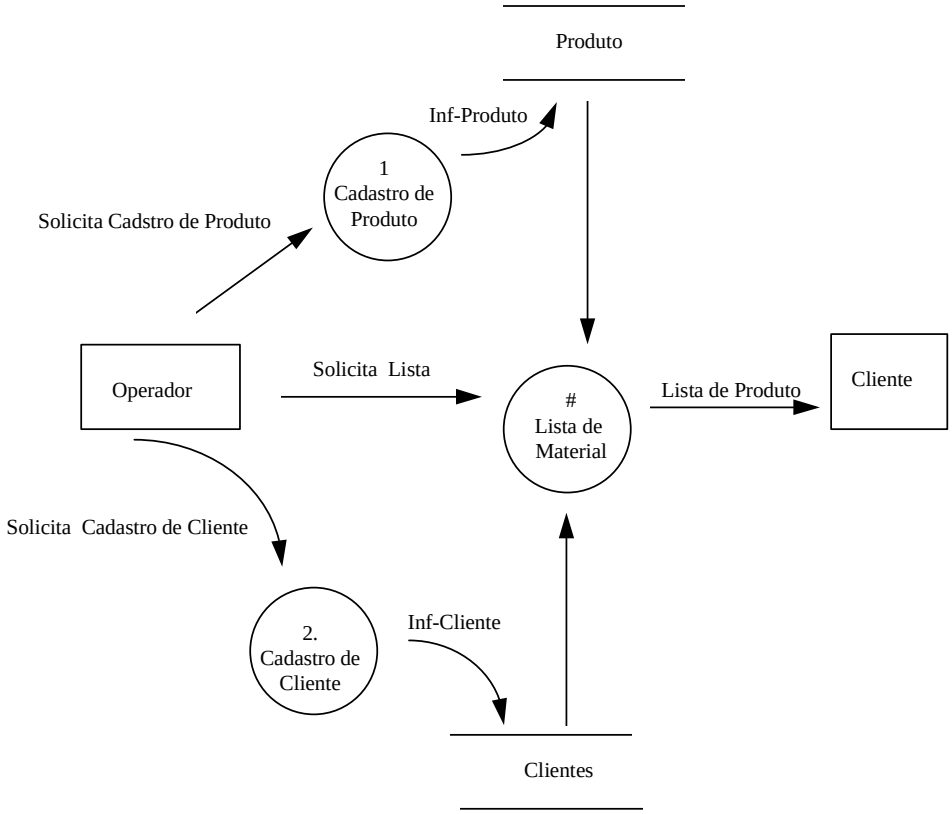


Fig 1.4 - Exemplo de um Diagrama de Fluxo de Dados

O Diagrama de Fluxo de Dados é amplamente utilizado para se representar os processos funcionais internos dos sistemas que manipulam uma grande variedade de dados (informações). Maiores detalhes sobre Diagramas de Fluxo de Dados podem ser encontrados em (Yourdon, 1992).

1.6 Statechart

Os Statechart são utilizados para especificar o aspecto comportamental de sistemas reativos complexos, tendo sido desenvolvido por David Harel em Israel, em 1982.

O Statechart é um formalismo baseado em Máquinas de Estados Finitos e seus diagramas de transição, mas os Statecharts apresentam algumas características adicionais que tornam mais clara sua representação gráfica (visual). Essas características são:

- Hierarquia: Uma bolha pode ser decomposta em várias sub-bolhas. Pode-se sumarizar bolhas, criar outras sub-bolhas detalhando a primeira, isso em tantos níveis quantos forem necessário. Com isso cria-se uma hierarquia entre as bolhas,

podendo-se criar uma quantidade qualquer de diagramas em vários níveis de detalhamento, permitindo uma maior clareza visual, o que facilita a compreensão dos diagramas.

- Ortogonalidade: Ao definir bolhas-filhas relacionadas a uma bolha (bolha-mãe) podemos identificá-las como ortogonais, significando que todas as bolhas-filhas sempre estarão ativas ou inativas simultaneamente (em conjunto). Essa característica permite identificar processos concorrentes com uma maior clareza.
- Comunicação: Uma transição pode ativar outra transição ao ser executada, permitindo assim uma influência entre as bolhas (comunicação).

Os Statecharts são constituídos de bolhas que podem estar ou não ativas em um determinado instante. As bolhas são interligadas por transições, as responsáveis por ativar e desativar as transições. Na notação gráfica adotada por Harel, temos as bolhas representadas por retângulos com bordas arredondadas, e as transições, por setas. A figura 1.5 mostra um exemplo de Statechart, que contém as bolhas *A*, *B*, *C*, *D*, *F* e as transições *x* e *y*.

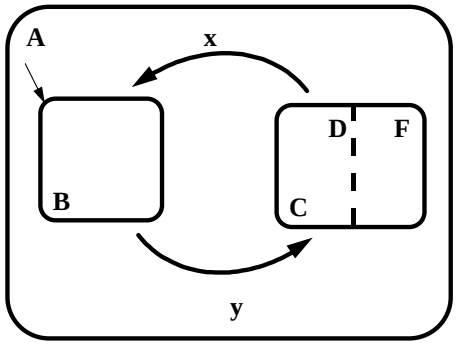


Figura 1.5 - Exemplo de um Statechart

Bolhas

As bolhas podem ser interligadas através de transições. Uma bolha tem a propriedade de estar ativa ou não.

Diferentemente dos estados das Máquinas de Estados Finitos, uma bolha de Statechart (bolha pai), pode ser decomposta em várias outras sub-bolhas (bolhas-filhas), em uma das seguintes formas:

1. Decomposição *OR:* neste caso, as sub-bolhas filhas são do tipo *ou-exclusivo*,

ou seja, quando a bolha mãe for ativada, apenas uma das sub-bolhas (bolhas-filhas) será ativada também, ou

2. Decomposição *AND*: neste caso, as bolhas-filhas operam em paralelo, ou seja, quando a bolha-mãe for ativada, todas as bolhas-filhas serão ativadas também. Este é o mecanismo usado para descrever comportamento paralelo (simultâneo). As bolhas-filhas são ditas ortogonais.

Essa característica de as bolhas serem agrupadas em várias sub-bolhas é também conhecida como profundidade ou hierarquia dos Statecharts.

Ao se definir o conjunto das bolhas-filhas de uma bolha-mãe tipo *OR* deve-se indicar qual a bolha *default*, ou seja, qual a bolha-filha que deverá ser inicialmente ativada quando a bolha-mãe for ativada.

Uma outra possibilidade de se definir qual a bolha-filha que deve ser ativada quando da ativação da bolha-mãe é identificar a bolha-mãe com a característica *história*, que significa ativar a bolha-filha que foi mais recentemente ativada.

Transição e Evento

Cada transição está relacionada a um evento (um estímulo proveniente do meio ambiente ou gerado internamente pelo próprio Statechart) associado ou não a uma condição (proposição lógica). Uma transição pode disparar uma outra transição ativando um evento a ele associado.

As transições são arcos orientados que interligam duas bolhas, a *bolha-Origem* e a *bolha-Destino*.

Uma transição é disparada quando:

- o evento de entrada estiver ativado
- a condição estiver satisfeita e
- a bolha-origem estiver ativo.

Nessas condições a bolha-origem deixa de estar ativa, enquanto a bolha=destino passa a estar ativa, e o evento de saída é ativado (quando for o caso). O evento de saída pode ser um evento interno ao sistema, disparando eventualmente uma outra transição, ou então um evento a ser emitido ao meio externo (resposta).

Definição

Formalmente, um statechart *S* é *S*=(E, ρ, ψ, δ, H, L) onde:

. E: é o conjunto de bolhas do Statechart *S*

. ρ: função *hierarquia*. Essa função associa para cada bolha de *S* o conjunto de bolhas-filhas que a constituem. Existe apenas uma bolha (chamada raiz), que não pertence ao conjunto da função *hierarquia* de nenhuma bolha.

. ψ: função *tipo*. Essa função associa para cada bolha qual o seu tipo: *OR* ou *AND*

. δ: função *default*. Para as bolhas constituídas de bolhas-filhas, a função *default* indica qual a bolha *default* (única bolha que deve ser inicialmente ativada quando da ativação da bolha-mãe)

. H: função *história*. Identifica quais bolhas tipo *OR* têm ou não a característica de ativar a última bolha-filha ativada ao ser ativada.

. T: conjunto de transições $t = \{ t_i / t_i = (e_i, a_i, c_i, O_i, D_i) \text{ p/ } i=1,...,n \}$

As transições $t_i \in T$ são compostas de:

. evento $\boldsymbol{e_i}$ que dispara a transição

. ação a_i que será gerada pela transição

. condição c_i que deve ser satisfeita para se efetuar a transição

. bolha-origem O_i da transição

. bolha-destino D_i da transição

Ao se ativar uma bolha Y (ψ(Y)=OR), que seja composta por várias bolhas-filhas o sistema ativará a bolha-filha que foi definida com "*Default*", o que é indicado no diagrama com uma pequena seta. Ao se ativar uma bolha X (ψ(X)=AND), que seja composta de várias bolhas-filhas, todas essas bolhas-filhas serão ativadas.

Segundo (Huizing, 1991) os Statecharts são mais adequados à especificação de sistema reativos do que as Redes de Petri e as Máquinas de Estados Finitos, pois representam as especificações em diagramas gráficos mais simples e mais fáceis de serem compreendidos.

Exemplo 1

Vamos considerar o statechart do exemplo da figura 1.6

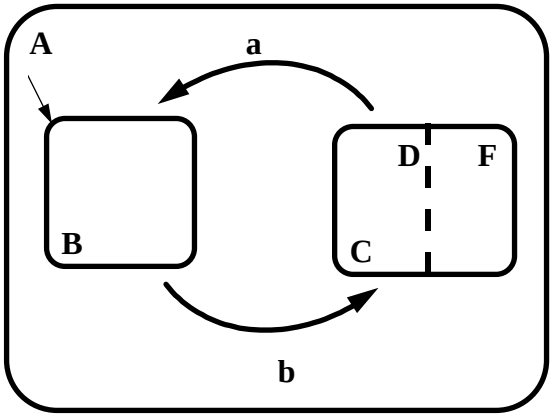


Figura 1.6 - Exemplo de Statechart

Neste caso temos:

E={A,B,C,D,F }

ρ(A) ={B,C,D,F}

ρ(C) ={D,F}

A - raiz do Statechart

ψ(A) = OR

ψ(C) = AND

δ(A) = B

Transições : (B,b) → (C)

(C,a) → (B)

Na figura 1.6, temos a bolha *A* (do tipo *OR*), que é decomposta em duas bolhas-filhas: uma de tipo *OR* bolhas *(B)* e uma do tipo *AND* (*C)*. Enquanto a bolha *A* estiver ativa, apenas uma das bolhas-filhas (*B* ou *C*) estará ativa em um determinado instante, ou seja, enquanto a bolha *A* estiver ativa, as suas bolhas-filhas *B* e *C* podem ser alternadamente ativadas e desativadas, de acordo com as transições que as interligam. A bolha *C* é constituída de duas bolhas-filhas (*D,E*) paralelas (decomposição *AND*). No exemplo, as bolhas-filhas *D* e *E* são ortogonais, ou seja, quando a bolha *C* é ativada, ambas as bolhas-filhas *D* e *E* são ativadas também, simultaneamente. Conceitualmente *B* e *C* são equivalentes a uma única bolha, como a bolha *A*. Elas próprias podem ter bolhas-filhas do tipo *OR* ou do tipo *AND,* transições internas, etc.

Exemplo 2

No exemplo da figura 1.7 temos:

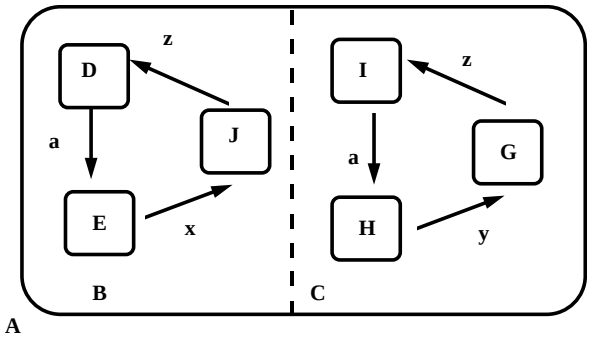


Fig 1.7 - Exemplo de Statechart representando Ortogonalidade / Sincronização

E={A,B,C,D,F,G,H,I,J}

ρ(A) ={B,C}

ρ(B) ={D,J,F}

ρ(C) ={G,H,I}

A - raiz do Statechart

ψ(A) = AND

ψ(B) = OR

ψ(C) = OR

δ(B) = D

δ(C) = I

Transições : (D,a) → (J) (F,z) → (D) (H,y) → (G)

(J,x) → (F) (I,a) → (H) (G,z) → (I)

Pode-se notar que a transição *a*, transfere a bolha-filha corrente de *B* (de *D* para *E*) e da bolha-filha *C* (de *I* para *H*) simultaneamente. Esta propriedade mostra como representar sincronização em um Statechart.

Se representarmos esse mesmo exemplo utilizando Máquinas de Estados Finitos teremos que representar as nove combinações possíveis das bolhas-filhas (3 x 3). O diagrama de transição seria como mostra a figura 1.8. Essa figura ilustra a simplicidade visual dos diagramas Statecharts em relação às Máquinas de Estados Finitos.

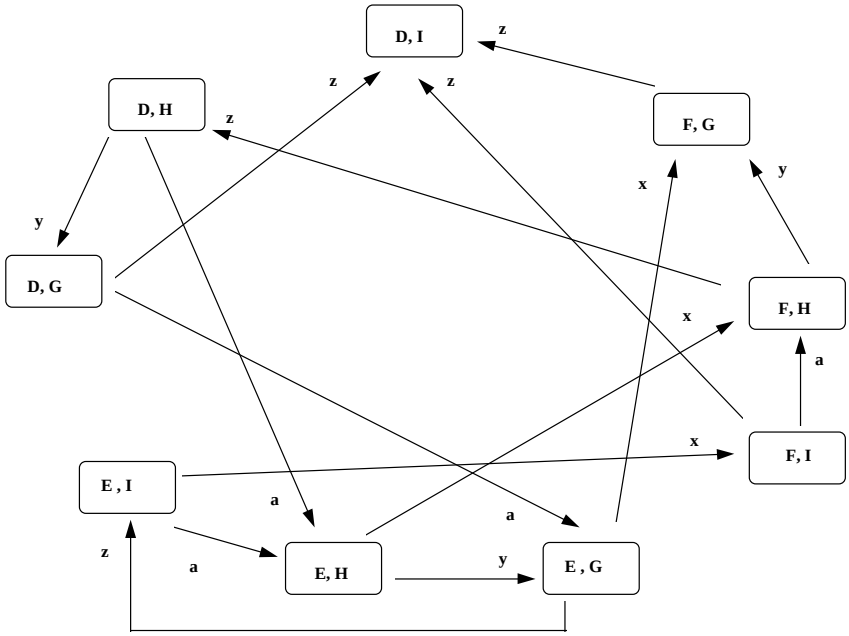


Fig 1.8 - Máquina de Estados Finitos Correspondente ao Statechart da Fig 1.7

Estado

Define-se como o estado de um Statechart em um determinado instante σ_i ($i \geq 0$) ao conjunto das bolhas ativas neste instante.

Configuração

Uma configuração de um Statechart em cada instante σ_i é uma quádrupla (X,Π,Θ,ξ) onde:

. X: é conjunto de bolhas ativas

. (Π,Θ,ξ): representa o estímulo externo associado, sendo Π o conjunto de eventos externos, Θ o conjunto das condições cujo valor é verdadeiro no instante e ξ é uma função determinada pelo ambiente externo

A reação do sistema em qualquer instante σ_i é composta pelo conjunto de transições realizadas naquele instante (Γ) e o conjunto de eventos gerados por essas transições (Π^e).

Resumindo, a execução de um sistema Statechart é a seqüencia de reações das configurações, ou seja, é uma seqüencia $\{(X_i,\Pi_i,\Theta_i,\xi_i, \Gamma_i,\Pi_i^e)\}$ para cada instante i, onde $i \geq 0$.

X_i	Conjunto de estados ativos
(Π_i,Θ_i,ξ_i)	Estímulo externo
Γ_i	conjunto de transições a serem realizadas
Π_i^e	conjunto de eventos a serem gerados pelas transições Γ_i

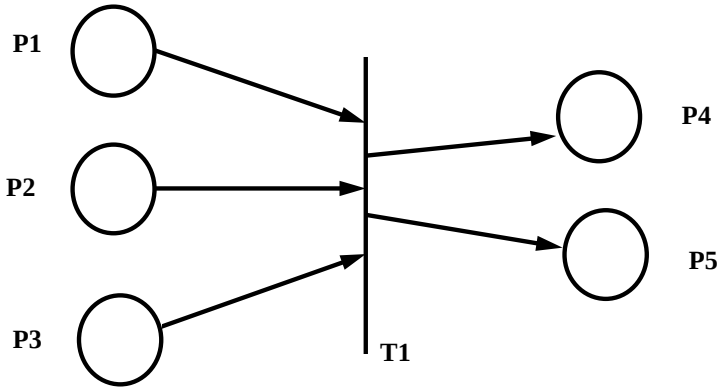
Finalizando, os Statecharts são largamente utilizados na especificação do aspecto comportamental de sistemas reativos, pois sua representação é de fácil compreensão, quando comparada às outras técnicas. Maiores detalhes sobre os Statecharts podem ser consultados em (Harel, 1987).

1.7 Redes de Petri

As Redes de Petri foram descritas em 1962 por Petri e tem sido largamente utilizada para especificar mecanismos de sincronização (Peterson, 1977). .

Rede de Petri é um modelo formal de fluxo de informações, utilizado para descrever e analisar o fluxo de informações no sistema, assim como o seu controle, particularmente em sistemas que apresentam atividades assíncronas e concorrentes

Basicamente podemos dizer que uma Rede de Petri é composta de lugares e transições. Uma diferença básica entre as Redes de Petri e Máquina de Estados Finitos é que uma transição pode ligar vários lugares de entrada a vários lugares de saída simultaneamente. Em cada instante um lugar pode estar ativo ou não. A ativação de uma transição só ocorre quando todos os lugares de entrada estiverem ativos. Quando uma transição ocorre todos os lugares de entrada são desativados, enquanto os lugares de saída são todos ativados simultaneamente. Essa característica permite a uma Rede de Petri representar fenômenos de sincronização de um modo explícito, pois uma transição só ocorre quando todos os eventos que se deseja sincronizar (representados pelos lugares de entrada da transição) tiverem acontecidos. Quando um lugar está ativo dizemos que ele contém uma marca. A partir de uma configuração inicial dos lugares ativos (marcados) uma Rede de Petri pode ser simulada mudando as posições das marcas (alterando os lugares ativos) de acordo com a transição ocorrida. Quando houver mais de uma transição habilitada em um determinado instante, apenas uma delas, aleatoriamente escolhida, é disparada (não determinismo).



Obs: T1 - Transição
P1,P2,P3,P4,P5 - lugares

Figura 1.9 - Exemplo de representação gráfica de uma Rede de Petri

Definição

Uma Rede de Petri é uma quádrupla *C=(P,T,I,O),* onde :

P : Conjunto dos lugares

T : Conjunto das transições

I : Função entrada. Função que associa a cada transição um conjunto de lugares de entrada (sub-conjunto de P)

O : Função Saída. Função que associa a cada transição o conjunto de lugares de saída (sub-conjunto de P)

Representação Gráfica

A representação gráfica das Redes de Petri, apresenta os seguintes elementos (chamados *nós*):

◯ - lugares - representado por uma circunferência

—— - transições - representado por uma barra

Esses dois *nós* (lugares ou transições), são conectados por meio de setas. Existem dois tipos de ligação:

- de um lugar para uma transição,

- de uma transição para um lugar.

Dizemos que, se um arco liga um *nó i* a um *nó j*, então *i* é uma entrada de *j* e *j* é uma saída de *i.*

Para se denotar as propriedades dinâmicas nas Redes Petri, definem-se as marcas

(círculos pretos dentro da circunferência), indicando que o lugar está sendo ocupado (ativo). A cada passo, as fichas podem mudar de lugar de acordo com as seguintes regras:

- As marcas são movidas pelo disparo da transição. Uma transição só é disparada se estiver habilitada, ou seja, se todas as entradas contiverem ao menos uma marca cada uma.
- A transição disparada retira as marcas das entradas colocando-as nos lugares de saída da transição.

Uma marcação é um vetor $u = (u_1,...,u_n)$ que indica o número de fichas em cada lugar p_i, para *i* de *1* a *n*, ou seja, $u(p_i) = u_i$., $1 \leq i \leq n$.

Para uma Rede de Petri *C=(P,T,I,O)* com uma marcação *u*, temos a Rede de Petri marcada *M=(P,T,I,O,u).*

Exemplo 1

Na figura 1.10 temos uma Rede de Petri R={P,T,I,O}onde:

$P = \{P_1, P_2, P_3, P_4\}$

$T = \{t_1, t_2, t_3\}$

$I(t_1)=\{P_2\}$ $I(t_2)=\{P_1, P_4\}$ $I(t_3)=\{P_3,\}$

$O(t_1)=\{P_1\}$ $O(t_2)=\{P_2, P_3\}$ $O(t_3)=\{P_4\}$

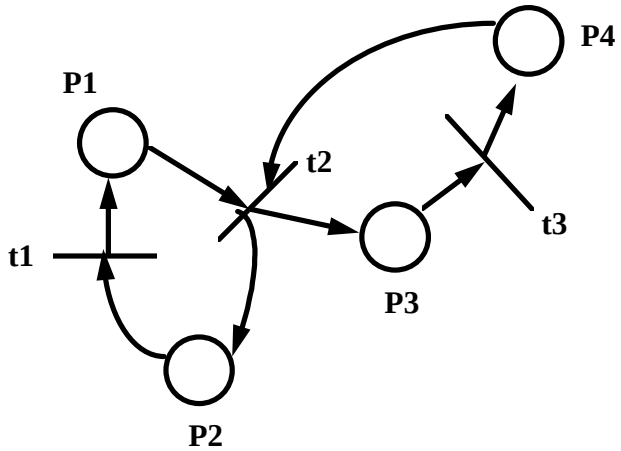


Fig 1.10- Representação Gráfica de uma Rede Petri

A figura 1.11 mostra a configuração da Rede de Petri R antes do disparo da transição t_2, e a figura 1.12 a mesma rede após o disparo dessa transição. Antes do disparo existem marcas nos lugares P_1 e P_4, enquanto que após o disparo não existem

marcas nos lugares P_1 e P_4, e passam a existir nos lugares P_2 e P_3.

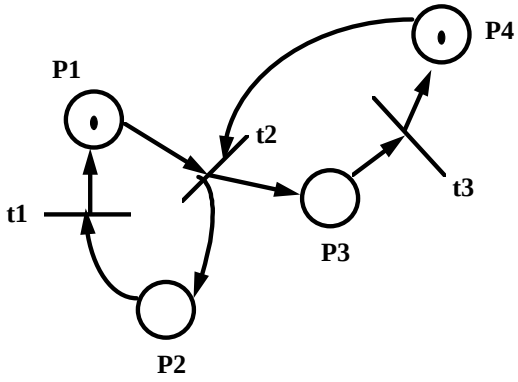


Fig 1.11- Exemplo de disparo de transição (situação inicial)

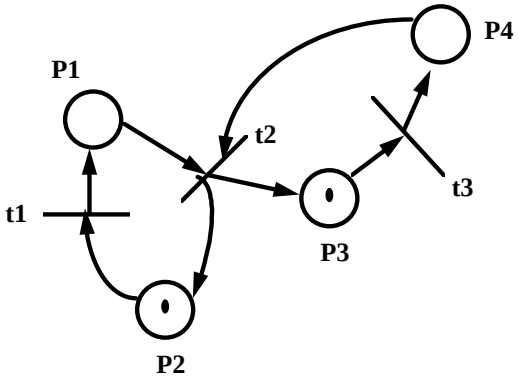


Fig 1.12- Exemplo de disparo de transição (situação após o disparo da transição t2)

Exemplo 2

Podemos utilizar as Redes de Petri para modelar sistemas de software. Como exemplo, vamos considerar o clássico problema de dois processos simultâneos (*A e B*), cada um com uma seção crítica. O processo *A* é composto de duas seções representados pelos lugares P_1 e P_2 enquanto que o processo *B* é composto de duas seções representadas pelos lugares P_3 e P_4. Tanto P_2 quanto P_4 são seções críticas, ou seja, não podem ser executadas simultaneamente.

No exemplo ilustrado (figura 1.13), temos que as seções (processos) P_2 e P_4 são exclusivos. Só é possível executar um deles em um determinado instante. Quando um é executado (habilitado), o outro passa a ficar desabilitado (figura 1.14 e 1.15).

Nessa rede de Petri $R_1 = \{P,T,I,O\}$ temos:

$P = \{P_1, P_2, P_3, P_4, P_5\}$

$T = \{t_1, t_2, t_3, t_4\}$

$I(t_1)=\{P_1, P_5\}$	$I(t_2)=\{P_2\}$	$I(t_3)=\{P_3, P_5\}$	$I(t_4)=\{P_4\}$
$O(t_1)=\{P_2\}$	$O(t_2)=\{P_1, P_5\}$	$O(t_3)=\{P_4\}$	$O(t_4)=\{P_3, P_5\}$

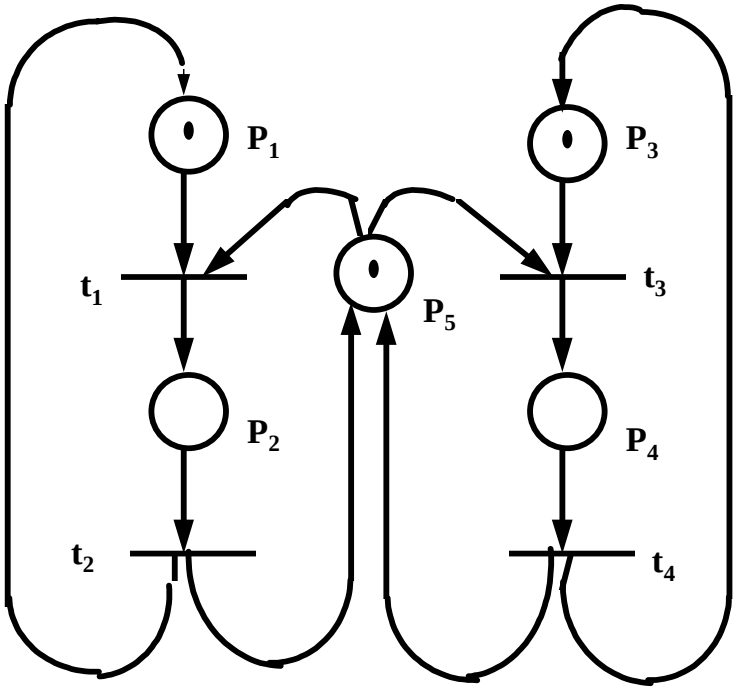


Fig 1.13 - Exemplo de uma Rede Petri com dois Processos Exclusivos.

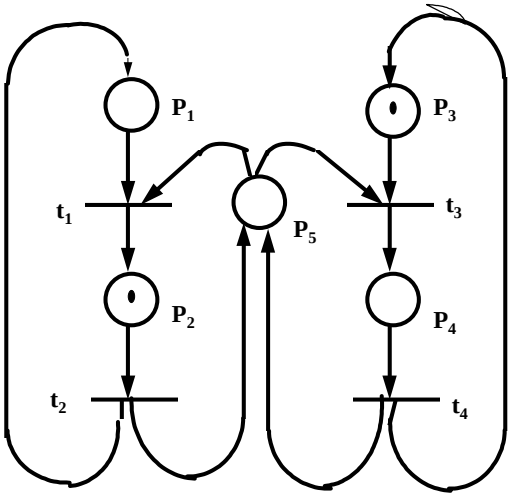


Fig 1.14 Execução do processo P2.

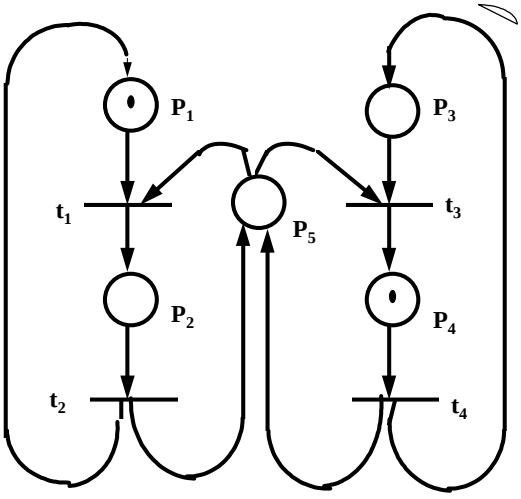


Fig 1.15 Execução do processo P4.

Concluindo, podemos dizer que a Rede de Petri é uma técnica apropriada para descrever fenômenos de sincronização.

1.8 Autômato Adaptativo

Originalmente os Autômatos Adaptativos foram criados para reconhecimento de linguagens dependentes de contexto, desenvolvido por José Neto (1993).

O Autômato Adaptativo, é uma extensão do conceito de Autômatos Finitos,

capaz de reconhecer linguagens sensíveis ao contexto, através de sua auto-configuração como resposta aos estímulos de entrada.

O seu modo de execução é similar à Máquina de Estados Finitos acrescido da capacidade de executar algumas transições especiais chamadas transições adaptativas. Ao se executar essas transições adaptativas o autômato altera a sua configuração de estados e transições, modificando dinamicamente a sua estrutura.

A cada transição adaptativa associa-se uma função adaptativa, que basicamente identifica:

- o conjunto de produções a serem eliminadas

- o conjunto de padrões de produções a serem inseridas

A transição adaptativa pode executar uma função antes ou após o seu disparo.

Cada transição que compõe o autômato adaptativo é denotada por uma produção adaptativa da seguinte forma:

$(e) : A \rightarrow (e') : B$

Os símbolos *e* e *e'* identificam o estado corrente e o estado posterior à transição; A e B são opcionais, e correspondem à funções adaptativas, executadas respectivamente antes e depois da mudança de estado determinada pela aplicação da produção.

Definição

Um autômato adaptativo ***M*** apresenta os seguintes componentes:

w - cadeia de entrada

E_0 - Máquina de estados inicial de ***M***

E_m - Máquina de estados que implementa ***M*** após a aceitação da cadeia de entrada w_m (m = 0)

E_i - Máquina de estados que implementa ***M*** após a execução de i transições adaptativas (0 = i = m)

Para que o autômato aceite a cadeia de entrada $w=\alpha_0\alpha_1...\alpha_m$ ele passa pelas configurações $< E_0, \alpha_0> \rightarrow < E_1, \alpha_1> \rightarrow ... < E_m, \alpha_m>$ onde $< E_j, \alpha_j> \rightarrow < E_{j+1}, \alpha_{j+1}>$ $(0 \leq j < m)$ representa o reconhecimento de α_j por E_j, onde é executada a transição adaptativa P_j que altera o autômato para a configuração E_{j+1}. Uma transição adaptativa pode ser representada por $P_j = (t_j, A_j, u_j, B_j)$, onde:

- t_j é a configuração anterior do autômato M

-A_j é uma função Adaptativa anterior, executada antes da mudança de configuração (opcional)

-u_j é a configuração posterior do autômato M

-B_j: é uma função adaptativa posterior, executada após a mudança de configuração (opcional)

O autômato passa pelas diversas configurações t_i após a execução de produções, representadas acima, com pelo menos uma das componentes adaptativa *A* ou *B*.

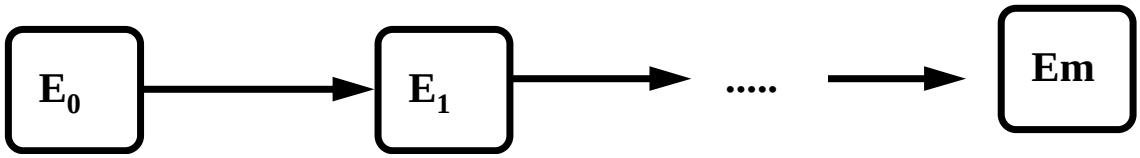


Fig 1.16 Representação da alteração da configuração do Autômato Adaptativo

Exemplo

Vamos ilustrar com um exemplo adaptado de (José. Neto,1993). O autômato representa a simulação de uma pilha, pois ele reconhece qualquer seqüencia do tipo $({}^{n}w)^{n}$.

Na figura 1.17 temos a configuração inicial do Autômato. A figura 1.18 apresenta a configuração do autômato após uma realização da transição que liga o estado 1 ao estado 2.

Pode-se perceber que a cada execução da transição associada a uma função adaptativa o autômato aumenta a quantidade de estados podendo reconhecer uma cadeia maior de parênteses, de acordo com a entrada dos dados.

<u>- Produções iniciais:</u>

(1, “w”) : 4

(2, “w”) : 3

(3, “)”) : 4

(1, , “(”) : 2, *A* (2, 3, 1)

<u>Função Adaptativa:</u>

A (i, j, n) = { k*, m* :

+ [(k, “w”) : m]

+ [(m, “)”) : j]

+ [(i, , “(”) : k , *A* (k, m, i)]
- [(n, “(”) : i, *A* (i, j, n)]
+ [(n, “(”) : i] }

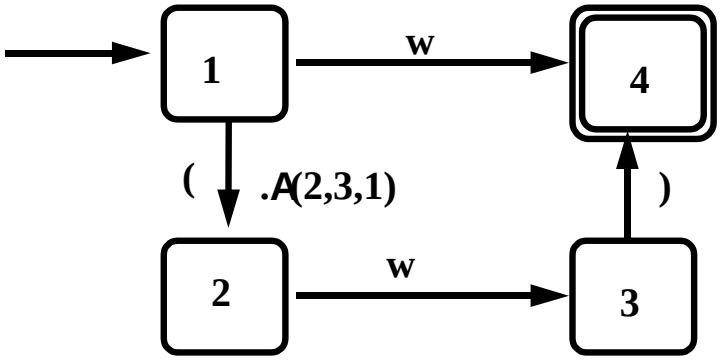


Figura 1.17 - Exemplo de Autômato Adaptativo (configuração inicial)

Após a primeira execução da transição associada à função adaptativa (transição que liga os estados 1 e 2) teremos a execução da função adaptativa com os parâmetros 2,3 e 1.

A (2, 3, 1) = { 5*, 6* :
+ [(5, “w”) : 6]
+ [(6, “)”) : 3]
+ [(2, “(“) : 5 , *A* (5, 6, 2)]
- [(1, “(“) : 2, *A* (2, 3, 1)]
+ [(1, “(“) : 2] }

A função adaptativa cria dois novos estados (5 e 6), ligando-os ao autômato original (conforme figura 1.16) eliminando a função adaptativa *A* (2, 3, 1) e criando a função adaptativa *A* (5, 6, 2).

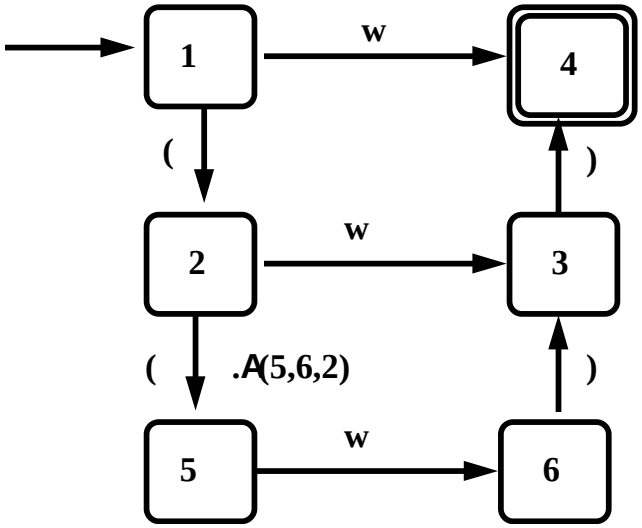


Figura 1.18 - Autômato Adaptativo, após a execução da funcão adaptativa.

Finalizando, podemos utilizar os Autômatos Adaptativos para a especificação do aspecto comportamental de sistemas onde se possa representar sua configuração dinamicamente. Maiores detalhes referentes aos Autômatos Adaptativos podem ser encontrados em (José Neto, 1993)

1.9 Statechart Adaptativo

O Statechart Adaptativo é uma extensão dos Statechart convencional (desenvolvido por Harel), incorporando a propriedade Adaptativa (análogo ao Autômato Adaptativo) sendo definido em (Rady, 1995).

O Statechart Adaptativo é constituído de um statechart convencional inicial, que transita entre as suas bolhas de maneira convencional até a execução de alguma transição que contenha uma chamada à uma função adaptativa. Similiarmente às funções adaptativas dos Autômatos Adaptativos elas alteram a configuração do Statechart, podendo incluir e excluir bolhas e transições.

Formalmente um Statechart Adaptativo ***SA*** compõe-se de:

- **w**, que é a seqüência dos eventos externos independentes a serem tratados
- $\mathbf{SA_0}$, que é o statechart convencional que representa ***SA*** no início da operação
- $\mathbf{Sa_m}$, que é o statechart convencional que representa ***SA*** no final da operação
- $\mathbf{Sa_i}$, que é o statechart convencional que representa ***SA*** após **i** transições adaptativas

As transições realizadas em decorrência da sequência de eventos externos independentes $w = \alpha_0\ \alpha_1\ \alpha_2\ \alpha_3 \ldots\ \alpha_n$ fazem com que se percorra uma trajetória

$<SA_0,\alpha_0> \rightarrow <SA_1,\alpha_1> \ldots \rightarrow <SA_m,\alpha_m>$, onde $<Sa_j,\alpha_j> \rightarrow <Sa_{j+1},\alpha_{j+1}>$ $(0 \leq j < m)$ representa a transição realizada em decorrência do evento α_j por ***SA*** ao final da qual é executada uma transição adaptativa $\mathbf{P_j}$ que altera $\mathbf{SA_j}$ para $\mathbf{SA_{j+1}}$.

Uma transição em um statechart adaptativo pode ser representado por
$(X_i, \Pi_i, \Theta_i, \xi_i, \Gamma_i, \gamma_i, \tau_i) \rightarrow (X_{i+1}, \Pi_{i+1}, \Theta_{i+1}, \xi_{i+1}, \Gamma_{i+1}, \gamma_{i+1}, \tau_{i+1})$ sendo:

- $(X_i, \Pi_i, \Theta_i, \xi_i, \Gamma_i, \gamma_i, \tau_i)$ é a configuração do statechart antes da aplicação de um passo
- $(X_{i+1}, \Pi_{i+1}, \Theta_{i+1}, \xi_{i+1}, \Gamma_{i+1}, \gamma_{i+1}, \tau_{i+1})$ é a configuração do statechart após a aplicação de um passo
- γ_j, τ_j representam o conjunto de eventuais chamadas de funções adaptativas para cada passo.

Os tipos possíveis de transições são:

- Transições adaptativas - contêm ações γ ou τ
- Transições não adaptativas - com γ e τ omitidos

Estando o Statechart Adaptativo em sua situação inicial, recebe a seqüencia de eventos externos. As transições são executadas conforme essa seqüencia, consumindo os eventos externos, repetindo-se o processo até o fim da seqüencia. Para cada evento recebido verifica-se qual ou quais são as produções a serem executadas.

Exemplo

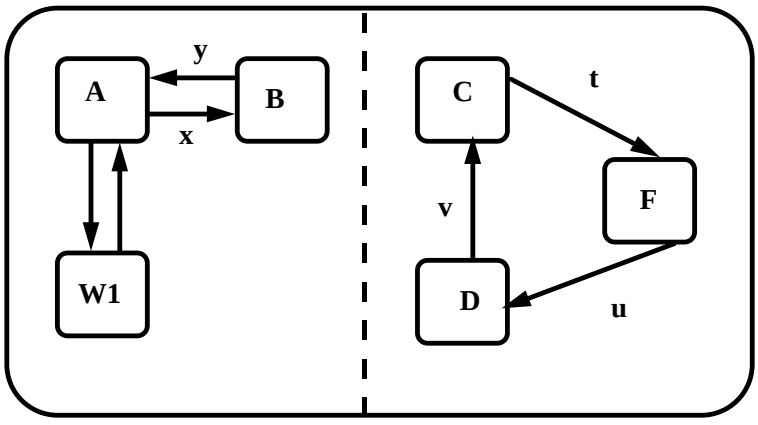


Fig 1.19 - Statechart Adaptativo. Configuração inicial

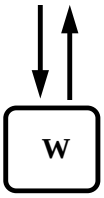


Fig 1.20 - Função Adaptativa

Esse exemplo ilustra um Statechart Adaptativo. A figura 1.19 apresenta a configuração inicial do Statechart. A ligação x está associada a uma função adaptativa, representada na figura 1.20. A função adaptativa acrescenta uma bolha e duas ligações

ao Statechart da forma mostrada no exemplo seguinte: a figura 1.21 apresenta a configuração do Statechart após a primeira execução da transição x, e a figura 1.22 após a segunda transição, ilustrando a dinâmica das configurações do Statechart Adaptativo.

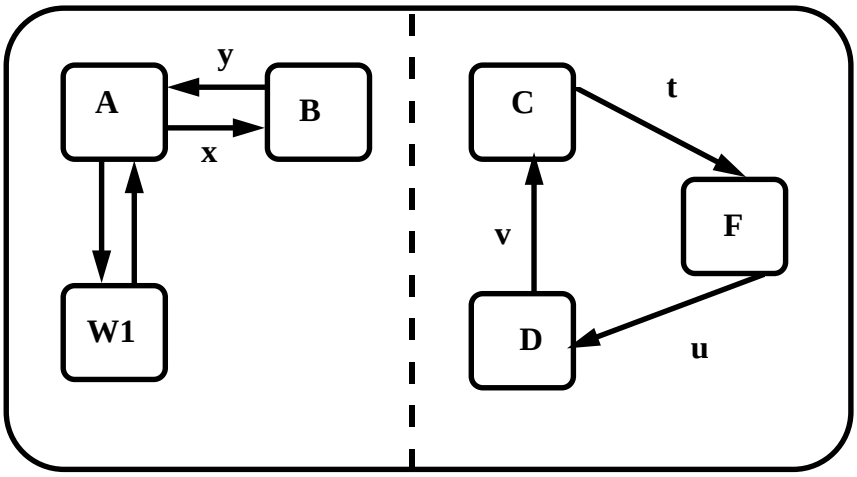


Fig 1.21 Statechart Adaptativo após uma execução da função adaptativa.

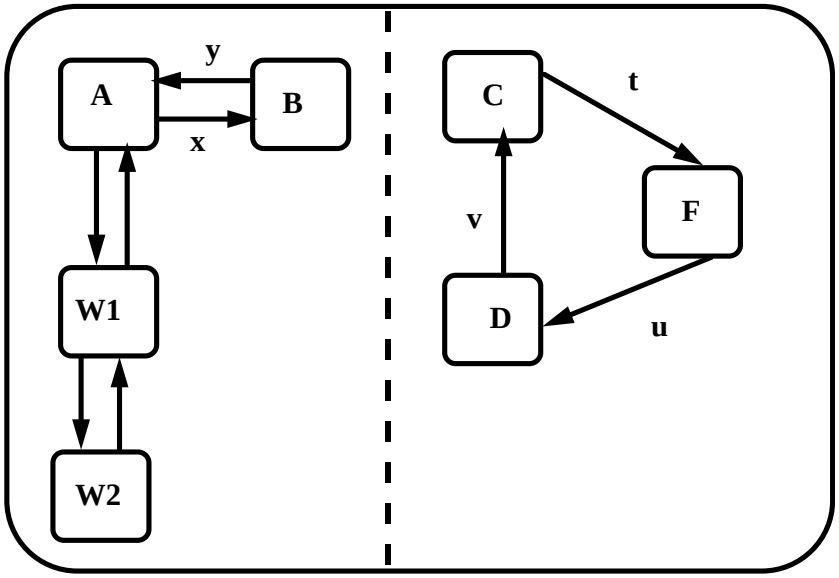


Fig 1.22 Statechart Adaptativo após duas execuções da função adaptativa.

Com a incorporação da característica Adaptativa os Statecharts Adaptativos podem ser utilizados na especificação do aspecto comportamental de sistemas reativos complexos com aprendizagem, ou seja, nos casos em que o comportamento do sistema varia dinamicamente em função da sua história.

1.10 Comentários

Pode-se concluir que existem vários métodos para se definir o comportamento de um sistema em relação ao meio externo, sendo que a escolha do método a utilizar dependerá do sistema e de sua finalidade. Do ponto de vista da facilidade de compreensão por parte de usuários as mais indicadas são as Máquinas de Estados Finitos e as Árvores de Decisão por serem amplamente utilizados para diversas finalidades, em várias áreas. De um modo geral, temos:

Máquinas de Estados Finitos: indicadas para aplicações sequenciais em sistemas de tempo de pequeno porte.

Árvore de Decisão: indicadas para as aplicações nas quais as repostas do sistema são, passo a passo, baseadas em decisões sobre algumas condições conhecidas.

Diagramas de Fluxo de Dados (DFD): apropriadas para o caso de haver necessidade de identificar os dados internamente através dos processos do sistema.

Statechart: indicado para aplicações de tempo real e sistemas reativos complexos.

Rede de Petri: indicado para sistemas onde se queira especificar os mecanismos de sincronização.

Autômato Adaptativo: indicados para aplicações com aprendizagem, ou seja, nos casos em que o comportamento do sistema varia dinamicamente em função da sua história.

Statechart Adaptativo: indicados para aplicações com aprendizagem (como os Autômatos Adaptativos) e que necessitem de uma representação hierárquica para seu melhor entendimento.

Vamos apresentar um quadro resumindo as características de cada uma das técnicas apresentada.

Técnica	Aplicação
Máquinas.de Estados Finitos	aplicações sequenciais em sistemas de tempo de pequeno porte.
Tabela. de Decisão	aplicações nas quais as repostas do sistema são, passo a passo, baseadas em decisões sobre algumas condições conhecidas.
Diagram de Fluxo de Dados (DFD)	aplicações onde há a necessidade de identificar os dados internamente através dos processos do sistema
Statechart	aplicações de tempo real e sistemas reativos complexos.
Redes de Petri	aplicações onde se queira especificar os mecanismos de sincronização.
Autômatos Adaptativos	aplicações com aprendizagem, ou seja, nos casos em que o comportamento do sistema varia dinamicamente em função da sua história.
Statecharts Adaptativos	indicados para aplicações com aprendizagem (como os Autômatos Adaptativos) e que necessitem de uma representação hierárquica para seu melhor entendimento.

Fig 1.23 - Resumo das técnicas / aplicação

Essa dissertação abordará os aspectos formais dos Statecharts, Redes de Petri e Autômato Adaptativo criando um formalismo baseado nas característica adaptativa e de sincronismo.

2. STATECHARTS ADAPTATIVOS SINCRONIZADOS

2.1 Introdução

Essa dissertação se propõe a definir um formalismo (***Stad-Sinc***) gráfico que seja aplicável à representação do sincronismo entre vários sistemas reativos e uma ferramenta (baseada no formalismo ***Stad-Sinc***) de simulação desses sistemas. Esse formalismo tem como principal característica permitir a representação explícita e sucinta aspectos de sincronização entre dois ou mais sistemas distintos. Essas interações que o formalismo abrangerá entre os vários sistemas podem ser representadas como do tipo *cliente-servidor*. Ou seja, um sistema solicita um serviço ao outro, e este executa e responde assim que estiver disponível (sincronização entre um recurso e um processo).

Baseado nesse formalismo, foi desenvolvida uma ferramenta (***SAS***), descrita no capítulo 3, que permite a sua edição e simulação. Com o auxílio dessa ferramenta podemos simular a interação entre vários sistemas sincronizando os processos, permitindo:

a) uma visualização gráfica clara dos processos sincronizados

b) análise do desempenho entre a solicitação e a resposta.

O formalismo proposto é baseado em:

1. Statechart Adaptativo: onde cada um dos sub-sistemas que interagem é representado isoladamente.

2. Rede de Petri: onde são feitas as ligações dos processos que interagem entre si

efetuando a sincronização entre eles.

Esse tipo de representação é de difícil interpretação visual quando efetuada com os formalismos tradicionais da literatura.

Baseado na ferramenta (***SAS***) poderemos responder à seguinte pergunta:

- *"Quantos recursos é necessário disponibilizar em um sistema para que se*

atenda a uma determinada demanda de solicitação?"

Quando temos vários sistemas interagindo entre si, essa questão é fundamental para se analisar a performance do sistema, pois esse desempenho está ligado no binômio disponibilidade/solicitação dos recursos.

2.2 Motivação

Vamos considerar um exemplo para ilustrarmos a aplicação do formalismo que será proposto e definido. Nosso exemplo será composto de 2 Statecharts. Um Statechart representa um sistema de controle de um hospital. Esse sistema controla vários segmentos: pagamento de funcionários, escala de horário, controle de horário (através de leitura magnética) e a autorização para internação dos pacientes. O outro Statacehart representa o sistema que controla um convênio médico. O sistema controla vários segmentos do convênio como: pagamento de médicos e hospitais credenciados, cobrança e cadastramento dos usuários e autorização de internação dos usuários segundo alguns critérios (pagamento em dia, carência, etc.).

O nosso enfoque estará no processo de autorização de internação.

A seguir vamos representar a interação (sincronismo) entre esses processos através de um Statechart convencional.

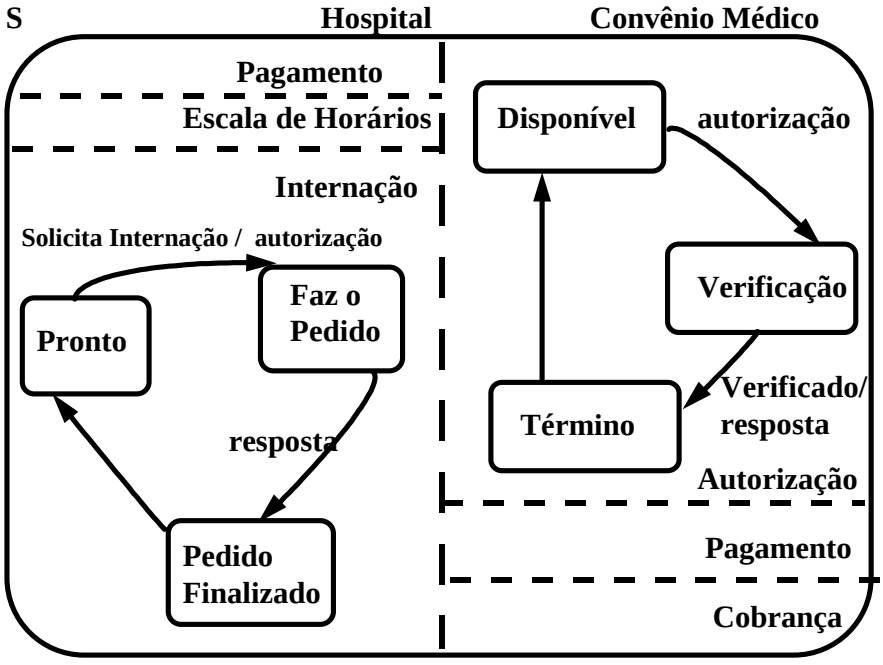


Figura 2.1 - Statechart representando o sistema Convênio Médico / Hospital

No exemplo acima temos um sistema representando o sincronismo entre o hospital e o convênio médico, utilizando-se statechart convencional. Quando a bolha "*pronto*" (da bolha hospital) estiver ativa e for gerado o *evento "solicita internação*", é disparado o evento "a*utorização*" que ativará a bolha "*verificação*" quando a bolha "*disponível*" estiver ativa (da bolha convênio médico). Quando for gerado o evento "verificado" e houver a transição para a bolha "*Término*" será gerado o evento "*resposta*" que ativará a bolha "*pedido finalizado*" (da bolha "*hospital*").

Nessa representação temos os seguintes problemas:

- o sincronismo entre os processos não é explícito (processo "*Internação*" e "*Autorização*"). Ao se visualizar a figura não é imediato notar a dependência entre os dois processos.
- os dois sistemas foram representados em apenas um Statechart, quando na verdade seria melhor se vistos como dois sistemas distintos que se relacionam apenas através de um processo comum. Essa representação descaracteriza a independência dos sistemas.
- sabemos que existe a dependência entre dois processos mas não sabemos como será o comportamento do sistema em função dos recursos disponíveis (tempo de resposta, desempenho).

2.3 Proposta

A figura abaixo mostra a representação do mesmo sistema Convênio Médico / Hospital utilizando a idéia do formalismo ***Statecharts Adaptativos Sincronizados.*** Resumidamente, a representação incorpora uma Rede de Petri que promove a sincronização entre dois processos.

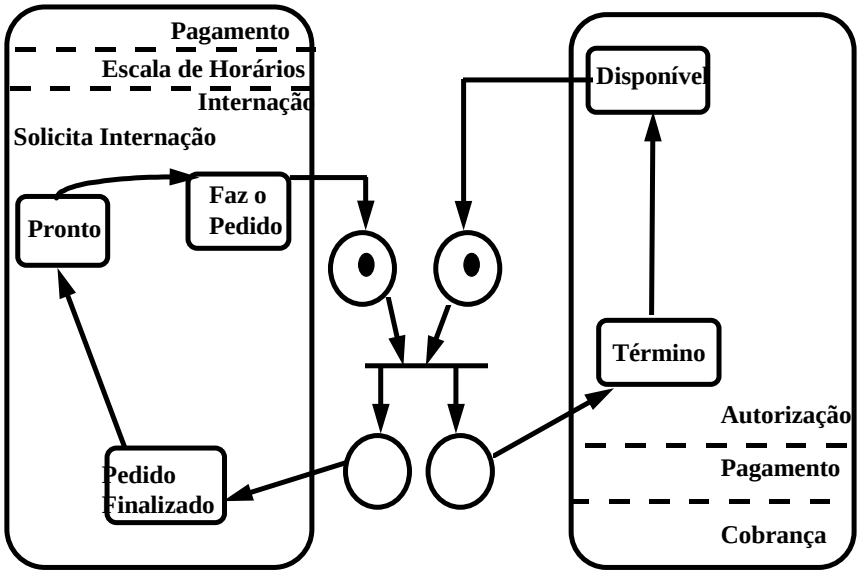


Figura 2.2 - Dois Statecharts independentes sincronizados com Rede de Petri

Essa representação mostra um instante em que:

a) está disponível o recurso do Convênio Médico para autorizar a internação e

b) o processo do Hospital está solicitando uma autorização de internação.

Nesse caso há a sincronização (os dois processos necessários estão disponíveis) e será executada a operação de autorização (incorporada na Rede de Petri) e os processo continuarão independentes após a autorização.

Avaliações da Alternativa Proposta

Como vantagens dessa representação temos:

- Maior visualização do sincronismo entre os dois processos
- Os dois sistemas podem ser considerados distintos (representados por Statecharts distintos) visto que existe apenas uma relação de sincronismo entre os dois processos.
- Com o auxílio da ferramenta (***SAS***) para a simulação desses sistemas poderemos ter informações sobre a performance desse processo de sincronismo. Por exemplo, se há uma solicitação de internação a cada 2 segundos em média, qual será o tempo de espera (médio) se houver um tempo de resposta (médio) de 3 segundos?

Com isso teremos um formalismo e uma ferramenta que terão grande utilidade para representar e simular sistemas com essas características (com sincronismo),

podendo ser enfatizado aspectos de performance entre os processos de respostas visando melhorar a qualidade dos sistemas.

2.4 Definição do formalismo dos Statecharts Adaptativos Sincronizados

A seguir vamos descrever o formalismo proposto, que permite estabelecer a sincronização entre dois Statechart independentes, através de uma Rede de Petri.

Esse formalismo pode ser representado como ***Stad-Sinc= (SA,RP,TE,TS)***, onde SA é um conjunto de Statechart Adaptativos, RP é um conjunto de Rede de Petri, TE é um conjunto de transições de entrada e TS é o conjunto de transições de saída, de tal forma que:

a) $SA=\{SA_1,SA_2,...,SA_n\}$ (n>1): são Statecharts Adaptativos

$SA_k=(X_k, \Pi_k, \Theta_k, \xi_k, \Gamma_k, \Pi_k^e, A_k, B_k)$ para k=1,2, ...,n ($n \geq 1$)

X_k	Conjunto de bolhas ativas
(Π_k, Θ_k, ξ_k)	Estímulo externo
Γ_k	conjunto de transições a serem realizadas
Π_k^e	conjunto de eventos a serem gerados pelas transições de Γ_k
A_k	função adaptativa, executada antes da transição
B_k	função adaptativa, executada depois da transição

b) $RP=\{RP^1, RP^2, RP^3,..., RP^m\}$, ($m \geq 1$), é um conjunto de Redes de Petri onde para cada $RP^j = (P^j, T^j, I^j, O^j)$, ($1 \leq j \leq m$), temos:

$P^j=\{P^j_1, P^j_2, ,..,P^j_{sj}\}$, ($sj \geq 3$)

$T^j=\{t^j_1\}$

$I^j = \{ P^j_1, P^j_2 ,.., P^j_{rj}\}$, ($0 < rj < sj$)

$O^j = \{ P^j_{rj+1}, P^j_{rj+2} ,..,P^j_{sj} \}$

c) $TE=\{TE^1,TE^2,...,TE^{sj}\}$, ($sj \geq 3$), é um conjunto de transições que ligam bolhas do Statechart Adaptativo aos estados (lugares) da Rede de Petri. Essas transições são acionadas assim que a bolha origem (do Statechart) é ativada, acionando assim o lugar da Rede de Petri que representa a bolha do Statechart. O evento que aciona a transição então é o evento In(X), onde In(X) indica se a bolha X do Statechart Adaptativo é ativo ou não. Se um lugar da Rede de Petri puder ser ativado por 2 ou mais bolhas de

Statechart Adaptativo em um instante, apenas uma transição será disparada.

Temos $TE^j=\{TE^j_1,TE^j_2, ...,TE^j_{rj}\}$ para $(1 \leq j \leq m)$

$TE^j_1(A^j_1) = P^j_1$

$TE^j_2(A^j_2) = P^j_2$

...

$TE^j_r(A^j_r) = P^j_{rj}$

onde A^j_1 ,...., A^j_{rj} são bolhas que pertencem ao conjunto de bolhas dos Statecharts Adaptativos $SA_1,SA_2,...,SA_n$.

d) $TS=\{TS^1,TS^2,...,TS^{s-r}\}$, é um conjunto de transições que ligam lugares da Rede de Petri às bolhas dos Statecharts Adaptativos. Essas transições são acionadas assim que o lugar da Rede de Petri for ativado, ativando a bolha do Statechart e desativando o lugar da Rede de Petri.

Temos $TS^j=\{TS^j_1,TS^j_2, ...,TS^j_{sj-rj}\}$ para $(1 \leq j \leq m)$

$TS^j_1(P^j_{rj+1}) = A^j_{rj+1}$

$TS^j_2(P^j_{rj+2}) = A^j_{rj+2}$

...

$TS^j_{sj-rj}(P^j_{sj}) = A^j_{sj}$

onde A^j_{rj+1} ,...., A^j_{sj} são bolhas que pertencem ao conjunto de bolhas dos Statecharts Adaptativos $SA_1,SA_2,...,SA_n$.

e) Para todo Statechart SA_k (k=1,...,n) existe uma uma bolha B_{xk} tal que

$\{B\} \subset \{ \cup_{(j=1,...,m\ ;\ k=1,....,sj)} A^j_k$

Ou seja, existe ao menos uma bolha de cada um dos n Statecharts presentes nas transições que ligam as Redes de Petri aos Statecharts e vice-versa..

Resumindo, podemos representar graficamente (para o caso de m=1) como:

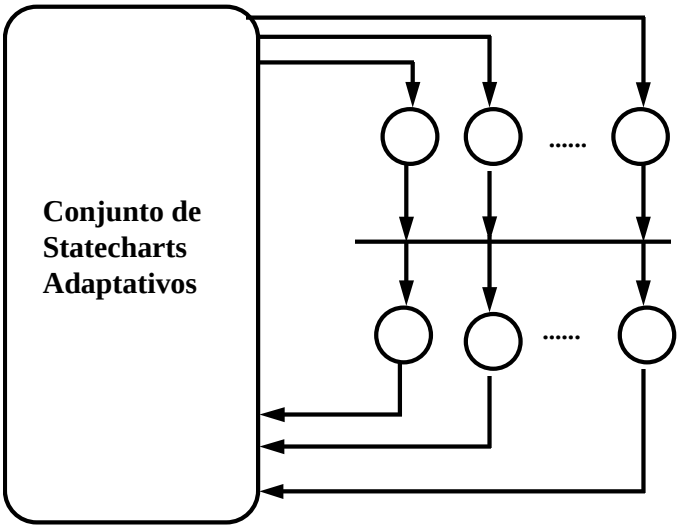


Fig. 2.3 - Representação do Stad-Sinc

2.5 Exemplo

Para ilustrar o uso do formalismo ***Stad-Sinc*** vamos considerar como exemplo a interação entre dois sistemas O primeiro é um sistema de monitoramento de aprendizado para alunos exercitarem uma disciplina qualquer enquanto o segundo é um sistema que auxilia os monitores a corrigirem exercícios e elaborarem os capítulos da disciplina.

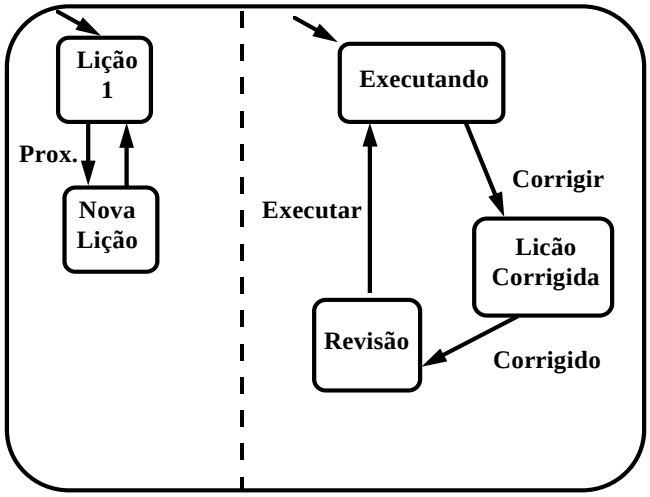


Fig 2.4 - Statechart representando sistema do Aluno

É claro a dependência entre os dois sistemas, pois os monitores serão solicitados a corrigirem as lições dos alunos, mas os dois sistemas podem ser considerados distintos,visto que cada um tem uma aplicação distinta do outro em muitos aspectos.

O sistema do aluno permite que este solicite uma nova lição, que será criada de acordo com o aproveitamento alcançado em lições já realizadas. Isso permitirá ao

sistema fornecer um atendimento com características próprias a cada aluno, personalizando as lições de acordo com o aprendizado do aluno. Essa característica fica associada à uma função adaptativa do sistema que cria estados (cada um associado a uma lição nova). O sistema também permite o aluno solicitar correção e solicitar auxílio a um monitor (externo ao sistema). Quando uma nova lição é solicitada, o sistema cria a lição de acordo com os resultados obtidos e desempenho das lições já realizadas. Portanto se o aluno não solicitar a correção de uma lição, ele não conseguirá efetuar as lições dos assuntos posteriores, pois não haverá nota que o habilite. Mas isso pode ser uma escolha, para que um aluno realize vários exercícios de uma matéria que ele julgar ainda não assimilada.

O sistema do monitor apresenta a possibilidade de se criar novas questões e lições de acordo com a percepção dos monitores, em relação às dúvidas dos alunos.

O sistema permite corrigir as lições efetuadas pelos alunos.

Um par de sistemas que se interagem continuamente só terá sucesso no desempenho das respostas se estiver bem equacionada a relação *solicitação / resposta*.

Como representar os sistemas e visulizar a interação entre ambos de um modo claro?

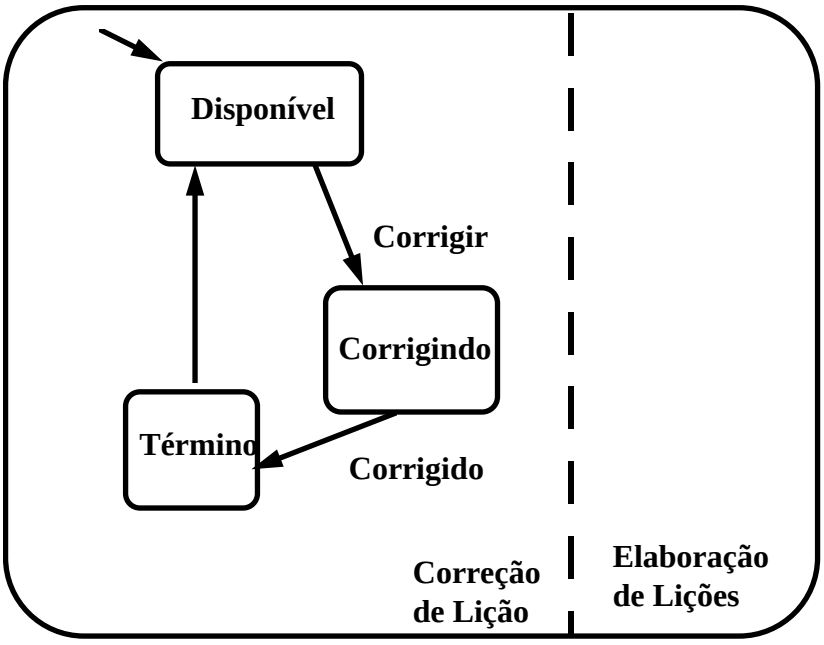


Fig. 2.5 - Statechart representando o sistem do Monitor

A figura 2.4 ilustra a representação do sistema do aluno e a figura 2.5 ilustra o sistema do monitor (ambas representadas em Statechart convencional). Embora

implicitamente esteja representada a relação de dependência entre ambos, a figura 2.6 (baseada em **Statecharts Adaptativos com Sincronismo**) ilustra com maior clareza os aspectos de dependência entre os dois sistemas.

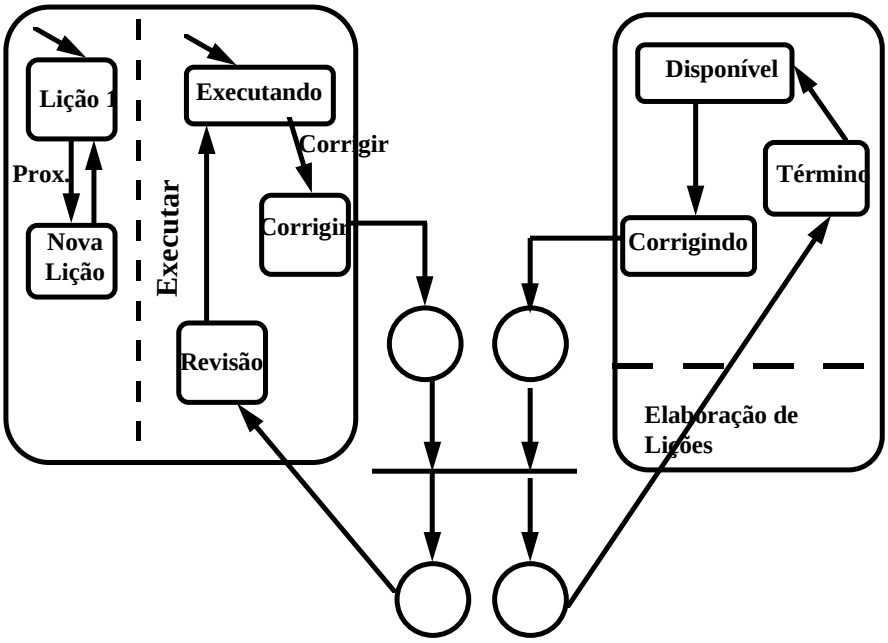


Fig. 2.6 - Sistema Aluno / Monitor representado por Stad-Sinc

Dessa forma o resultado apresentado (figura 2.6) identifica com maior clareza a sincronização entre os sistemas (Monitor e Aluno).

2.6 Conclusão

O formalismo definido apresenta de um modo formal e claro a sincronização existente entre um conjunto de sistemas, através de uma notação que exibe uma visualização mais fácil de ser compreendida, em comparação às outras técnicas tradicionais (quando aplicados em um conjunto de sistemas reativos complexos que apresentem sincronismos).

Para se desenvolver uma especificação de um sistema é necessário realizar as seguintes etapas:

- lógica do sistema: desenvolvimento de como será a lógica do funcionamento do sistema
- Aspectos de hierarquia: definição da hierarquia para se agrupar partes que se interligam
- Aspectos de concorrência: definição de processos que podem ser concorrentes
- Aspectos de sincronização: definição de processos que devem ser

sincronizados

- Aspectos de aprendizagem: definição de processos que podem ser dependentes do contexto (da história) para se auto-configurar.

Utilizando o Statechart Adaptativo com Sincronismo podemos utilizar as etapas acima em separado, utilizando para cada uma a técnica mais apropriada. Dessa forma, utilizamos:

- lógica do sistema: sem auxílio de uma técnica específica
- Aspectos de hierarquia: Statechart
- Aspectos de concorrência: Statechart
- Aspectos de sincronização: Rede de Petri
- Aspectos de aprendizagem: Statechart Adaptativo

Com isso o formalismo permite a representação de cada uma dessas componentes isoladamente, segmentando cada uma em uma etapa distinta. Quando uma das componentes não tiver aplicação em um determinado sistema, basta não usá-la que o formalismo se comporta como um caso particular. Por exemplo, se utilizarmos o Statechart Adaptativo com Sincronismo para representar um sistema que apresenta apenas heirarquia e concorrência (como o exemplo da figura 1.5) a nossa representação não precisará das componentes de sincronização e aprendizagem, sendo necessário só a utilização do Statechart. Portanto a especificação será composta apenas de um Statechart convencional.

Isso permite utilizar apenas as técnicas necessárias em cada aplicação, conseguindo desde uma representação simples quando a aplicação permitir, abrangendo também sistemas mais complexos no caso geral (utilizando todas as componentes), apresentando mesmo nesses casos uma solução clara. Essa representação evidencia cada componente do sistema, facilitando assim a sua manutenção e aperfeiçoamento. Por exemplo, quando há a necessidade de se analisar os processos concorrentes basta utilizar a representação do Statechart convencional. Quando há a necessidade de se alterar as componentes de aprendizado basta utilizar as componentes adaptativas. Quando há a necessidade de se alterar a sincronização de processos basta utilizar a representação da Rede de Petri.

CAPÍTULO 3 -

SAS - GERADOR E SIMULADOR DE STATECHARTS ADAPTATIVOS COM SINCRONISMOS

Neste capítulo apresentaremos o Sistema ***SAS*** (***S*tatehcarts *A*daptativos com *S*incronismos**), que é um gerador e simulador de de um conjunto de Statecharts Adaptativos que apresenta sincronizações entre si. O ***SAS*** é um software baseado no ***STAD*** (Rady,1995), ferramenta para representação e simulação de sistemas através de de Statecharts Adaptativos, com a capacidade adicional de representar e simular sincronização entre vários Statecharts distintos. Esse capítulo descreve suas características e o seu funcionamento. Finalizando o capítulo serão apresentados alguns exemplos caracterizando a sua capacidade de representar Statecharts com Sincronismos, ilustrando a sua utilização.

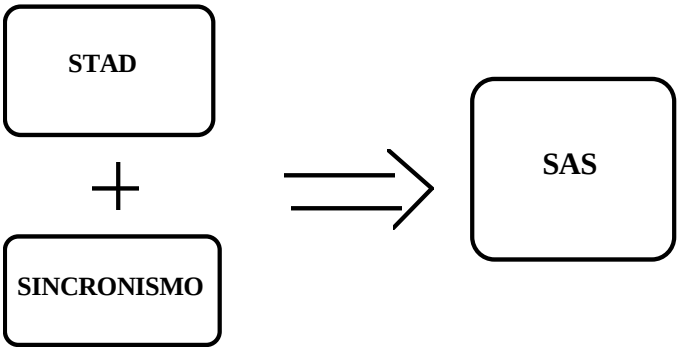


Fig 3.1 - Diagrama ilustrando a composição do SAS

3.1 Descrição da Ferramenta SAS

O ***SAS*** foi desenvolvido com o software Microsoft Visual Basic, utilizando Banco de Dados Microsoft Access para armazenar as informações dos Statecharts e de suas Sincronizações.

O ***SAS*** se compõe de dois módulos principais: Statechart e a Sincronização. O primeiro módulo (Statechart) é composto por funções que permitem a definição e simulação de um Statechart conforme citado anteriormente é a primeira versão, identificada como *STAD*, (Rady,1995). O segundo módulo (Sincronização) se compõe das funções que permitem a definição das sincronizações existente entre vários Statecharts, já definidos e projetados pelas funções do primeiro módulo (Statechart), e sua simulação.

3.1.A Módulo - Statechart

Esse módulo permite criar e simular Statecharts Adaptativos através de simples manipulação gráfica. O ***SAS*** apresenta uma interface gráfica (ambiente *Windows*), permitindo uma interação intuitiva através do *mouse*. As bolhas são representadas por retângulos com bordas arredondadas, e as ligações, por uma linha terminada por um losango em uma extremidade, indicando a orientação da ligação. As bolhas contêm um quadrado interno para identificar visualmente o seu tipo: as bolhas tipo *OR* são identificados pela cor azul, as bolhas do tipo *AND*, pela cor amarela e as bolhas de funções adaptativas, pela cor cinza. Por limitação física da dimensão da tela, foi adotado no sistema que cada nível de detalhamento seja efetuado em uma tela distinta. Em outras palavras, o detalhamento das sub-bolhas de uma bolha-mãe é feito em um outro diagrama, separadamente visualizado em uma outra janela do sistema.

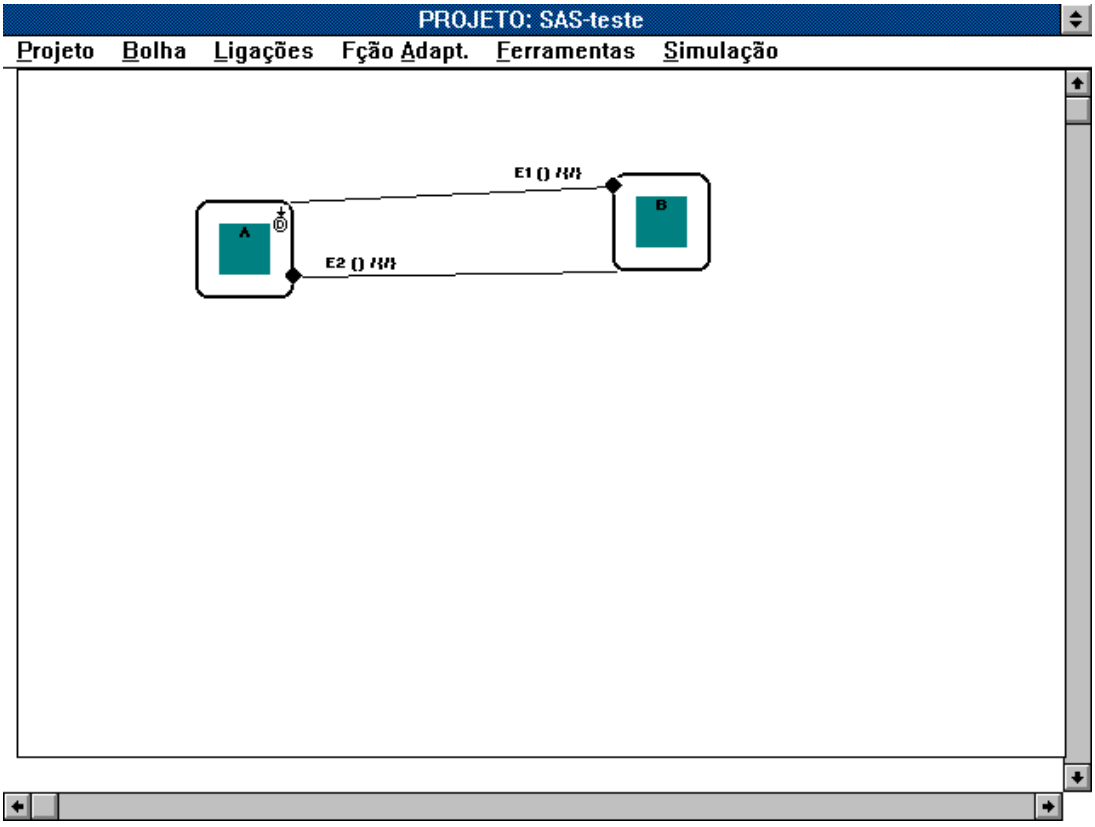


Figura 3.2 - Exemplo de Statechart representado no SAS

Para a edição dos Statecharts é utilizado um editor gráfico. Cada bolha pode ser detalhada em um número arbritário de níveis (detalhamento), conforme seja necessário. Para representar Statecharts complexos, com vários níveis de detalhamento, cada nível é representado em uma janela de edição distinta. Cada Statechart no sistema é considerado como um *Projeto*, sendo identificado por um nome único no sistema, ou

seja, um nome que é utilizado para identificar univocamente o Projeto.

Na figura 3.2 temos um exemplo de um Statechart muito simples, contendo as bolha *A* e *B*, e as transições (ligações) E1 e E2.

A seguir serão apresentados resumidamente as características do módulo Statechart, extraídas de (RADY, 1995), onde podem ser encontrados maiores detalhes.

Bolha

Para se criar as bolhas do Statechart, deve-se informar seu nome e tipo: OR, AND ou PRIMITIVA O nome da bolha é único, ou seja, um dado nome pode identificar apenas uma única bolha do Statechart. As bolhas do tipo OR e AND são iguais aos tipos definidos no capítulo 1, enquanto que o tipo PRIMITIVA é um tipo criado apenas para possibilitar a visualização de valores e efetuar cálculos aritméticos durante a simulação, enriquecendo-a com a identificação de alguns valores. Para as bolhas do tipo OR deve-se ainda fornecer as seguintes característica: Default, History ou Simples. Como opções de edição pode-se alterar qualquer uma das características acima, assim como incluir e excluir bolhas do Statechart.

Ligação

Para se criar as ligações deve-se obrigatoriamente informar o seu nome e o tipo da ligação. O nome da ligação também é único. O tipo pode ser: Dependente Interna, Dependente Externa Entrada, Dependente Externa Saída, Independente Interna, Independente Externa Entrada, Independente Externa Saída. As do tipo Dependente têm associado um evento originado interno ao sistema, enquanto as do tipo Independente têm associado um evento externo ao sistema. As ligações Dependentes são representadas no gráfico por uma linha contínua, e as Independentes, por uma com linha tracejada. Interna significa que tanto a bolha-origem quanto a bolha-destino são bolhas do Statechart corrente. Externa Entrada significa que a bolha-origem é uma bolha externa ao Statechart, não sendo, portanto, visualizada no diagrama, enquanto que Externa Saída siginifica que a bolha-destino é externa ao Statechart não sendo visualizada também.

Ao se criar uma ligação, pode-se ainda informar as seguintes características: Evento, Condição, Disparo, Tempo de Atraso, Tempo Mínimo, Valor e Tipo da Ligação. Apenas a característica Evento é obrigatória. A seguir vamos especificar cada uma dessas características:

Evento: indica o elemento que dispara a transição representada pela ligação.

Condição: indica o nome de uma bolha que deve estar ativa para que a transição seja efetuada.

Disparo: indica o nome de uma outra ligação que deverá ser ativada também, sempre que a ligação em questão for ativada

Tempo de Atraso: Indica quanto tempo deve decorrer a partir do instante corrente para que a transição seja disparada.

Tempo Mínimo: Indica o intervalo mínimo de tempo necessário para que a transição possa ser disparada.

Valor: Indica um valor para a ligação, utilizado quando a bolha-destino for uma bolha *Primitiva.*

Tipo da Ligação: Identifica se a ligação é normal ou *Adaptativ*a. Quando for do tipo *Adaptativa* deve-se associar à ligação a chamada de uma *Função Adaptativa*

Como recursos de edição, é possível alterar essas características, assim como excluir qualquer ligação do Statechart.

Função Adaptativa

Essa opção permite que se criem as *Funções Adaptativas*, que são compostas de bolhas e ligações, sendo portanto a sua criação idêntica à do Statechart comum. Uma diferença é a opção de se definir a exclusão de uma ligação quando a função for executada. Após a definição de uma função, é possível associá-la a uma ligação do Statechart original, onde será colocada uma *Ação Adaptativa* (chamada de *Função Adaptativa*). Deve-se informar todos os parâmetros da *Função Adaptativa*: as origens das ligações, a identificação das ligações que devem ser excluídas quando da execução da *Ação Adaptativa* e a indicação de que a *Ação* será anterior ou posterior à execução da ligação. Para facilitar a criação das funções pode-se fazer cópia das *Funções Adaptativas*, facilitando assim a edição de funções semelhantes. Qualquer função pode ser alterada ou excluída.

Funções Auxiliares

Para auxiliar na criação dos diagramas do Statechart existem as seguintes operações:

Exibição de Zoom

Essa operação permite que se visualize o diagrama corrente em uma vista reduzida, facilitando a visualização de diagramas complexos (que ultrapassam as dimensões de uma tela).

Lista de Inconsistências

Essa opção permite que se verifiquem várias consistências no Statechart corrente, como a existência de todas as bolhas utilizadas nas condições das ligações, a existência de todas as ligações referenciadas no campo Disparo das ligações, etc.

Impressão do Statechart

Essa opção permite que se imprima a parte do Statechart que está sendo visualizado na tela.

Simulação

Essa opção permite que sejam visualizadas as alterações dos estados ativos, através das ativações das ligações. Quando uma ligação ou bolha se torna ativa, a sua borda passa a ser representada, na ferramenta, em cor azul, caso contrário é representada em cor preta. Pode-se escolher entre simular ou não os níveis de detalhamento.

3.1.B Módulo - Sincronismo

Esse módulo foi desenvolvido com a finalidade de permitir a criação e a simulação de sincronizações envolvendo vários Statecharts.

Cadastro

Essa opção permite que se crie os sincronismos entre vários processos (bolhas) de vários Statecharts (distintos ou não) que já foram definidos no sistema através das opções do módulo Statechart.

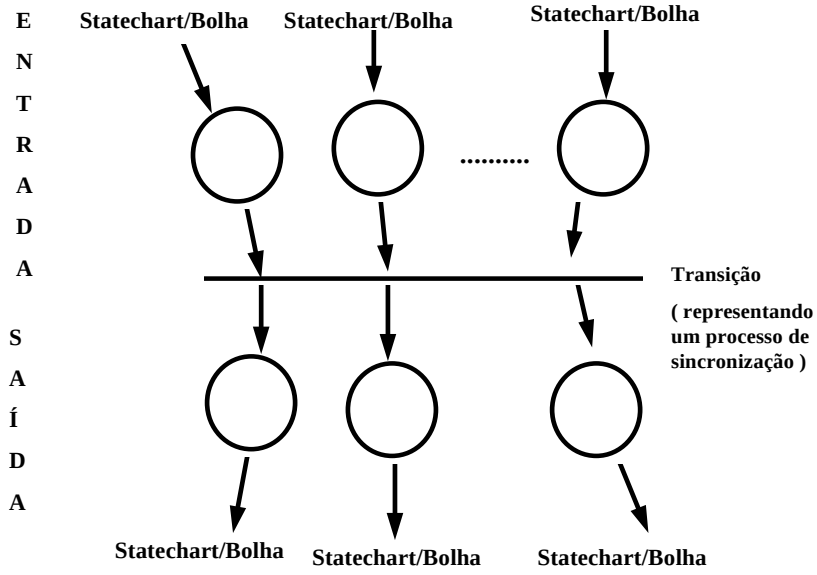


Fig. 3.3 - Representação gráfica de um sincronismo

Cada sincronização é representada da forma similar a uma transição de Rede de Petri, ou seja, é composta de vários lugares entrada (cada um representando uma bolha de algum Statechart) e vários lugares de saída, conforme definido no capítulo 2. Revisando, podemos representar graficamente cada sincronização como mostra a figura

3.3.

Para se definir uma **Sincronização** deve-se fornecer as seguintes informações:

Nome: qual o nome do sincronismo, que deve ser único.

Lista de processos de entrada: Uma lista contendo os nomes das bolhas (e os nomes dos respectivos Statecharts) que serão sincronizados.

Lista de processos de saída: Uma lista contendo os nomes das bolhas (e os nomes dos respectivos Statecharts) que deverão ser ativados após a execução da transição (processo de sincronismo).

Para se escolher os pares bolha/Statechart que compõem as listas, indicadas anteriormente, o sistema apresenta uma relação dos Statecharts cadastrados, para que seja feita a escolha. Após a escolha do Statechart é apresentado o seu diagrama, para que seja feita a escolha da bolha, através de simples manipulação do mouse. Após terem sido feitas todas as escolhas de bolha/Statechart o sistema cria o sincronismo, podendo sempre ser alterado o conteúdo das listas (incluindo ou excluindo pares bolha/Statechart). A figura 3.4 mostra um Statechart com uma bolha identificada como bolha de entrada no sincronismo (traço superior direito), e a figura 3.5, a definição de um sincronismo para ilustração.

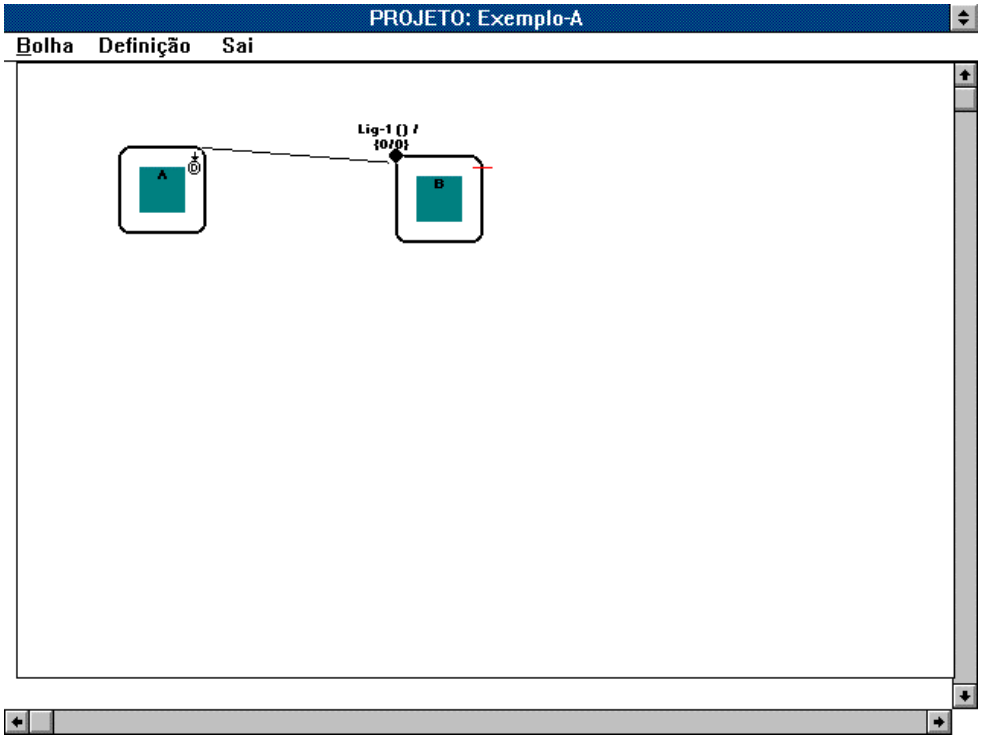


Fig 3.4 - Statechart identificando uma bolha de entrada de um sincronismo

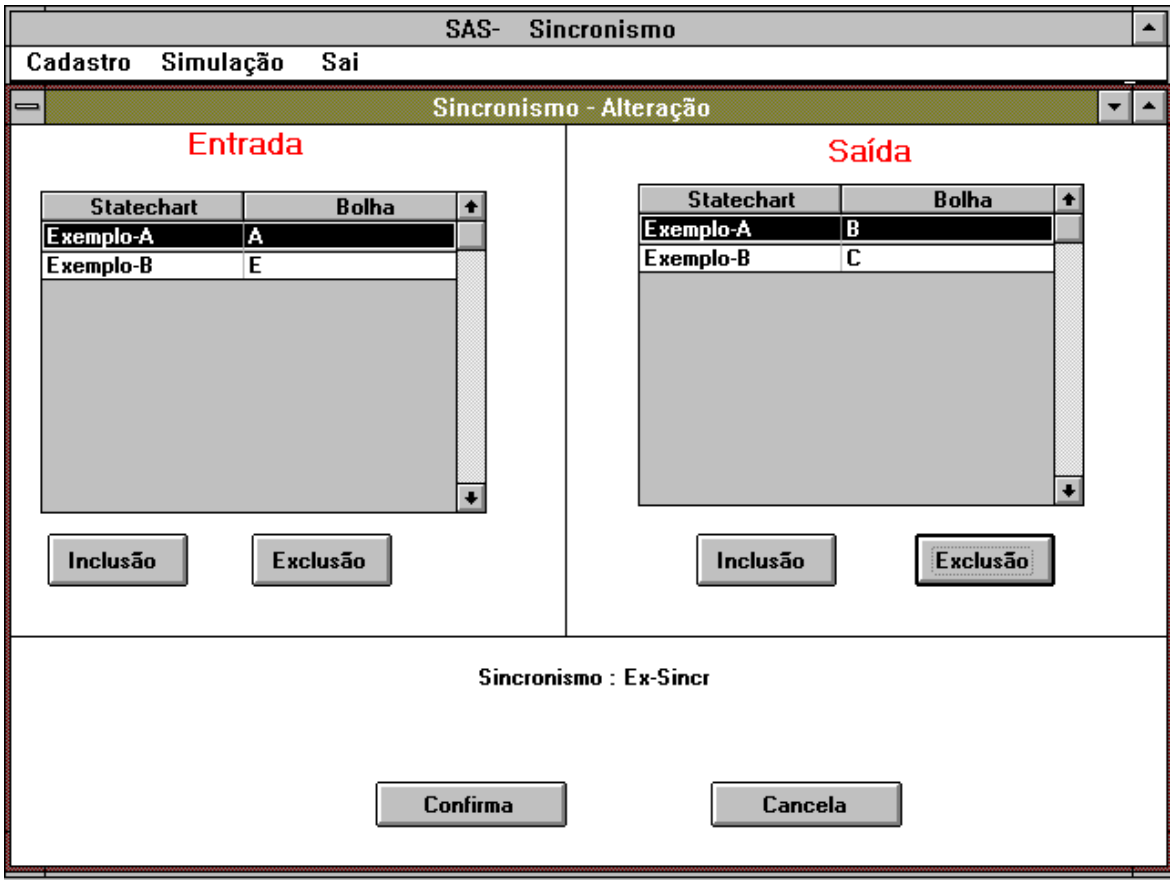


Fig 3.5 - Exemplo de cadastro de um Sincronismo

Simulação

Essa opção permite simular simultaneamente um conjunto de Statecharts e todos os sincronismos envolvidos. Para se fazer a simulação o sistema apresenta uma lista contendo todos os Statecharts cadastrados, para que seja feita a escolha dos statecharts que serão simulados. Após a confirmação da lista escolhida, o sistema verifica automaticamente quais sincronismos serão simulados. Para que um sincronismo seja simulado é necessário que todos os Statecharts que o compõem estejam na lista de simulação.

A simulação é feita ativando-se alternadamente as ligações e as bolhas. Na ferramenta desenvolvida, os elementos ativos são identificados com a borda desenhada em azul, e as demais com na cor preta. Quando as ligações se ativam, as bolhas se desativam e vice-versa. Cada statechart é simulado em uma janela separada, assim como cada sincronismo. Cada Statechart é simulado em todos os níveis possíveis, ou seja, quando uma bolha que for ativada contiver algum possível detalhamento será criada uma janela com o diagrama de detalhamento ativando as suas bolhas *default* ou a *história* do diagrama, e assim succesivamente para todos os níveis. Pode-se visualizar apenas um diagrama (uma janela de simulação) de cada vez na tela, podendo ser alternadamente escolhido qualquer um dos diagramas correntes, bastando para isso fazer a escolha através de uma simples operação com o *mouse*.

Convém ressaltar que na ferramenta desenvolida as bolhas-filhas, diagramas de detalhamento, deixam de ser visualizadas a partir do instante que a bolha-mãe é desativada, pois o sistema só mantém disponível as janelas dos sub-diagramas enquanto a bolha-mãe estiver ativa.

Nas telas do sincronismo pode-se visualizar em cada instante quais processos já estão ativos, esperando a sincronização (graficamente representados com a borda azul), e quais ainda não foram ativados (representados com a borda preta). A cada instante são verificados todos os lugares (processos) dos sincronismos, ativando-os quando a bolha correspondente do Statechart for ativada. Durante a simulação pode-se visualizar uma tabela contendo as estatíticas de acesso para cada sincronismo, podendo facilitar a compreensão dos processos responsáveis pelo atraso da sincronização. Cada transição do processo pode ter uma duração determinada pelo usuário, permitindo assim representar várias configurações sobre a disponibilidade do processo sincronizador (tempo de resposta para se efetuar o processo, após todas as condições estarem

satisfeitas). Ou seja, a cada instante pode-se visualizar qualquer diagrama, de qualquer Statechart e qualquer nível, assim como também qualquer sincronismo.

Cada Statechart ativa inicialmente as bolhas *defaults.* A qualquer instante da simulação o usuário pode disparar qualquer transição definida como Independente (externa ao meio ambiente). Enquanto não é ativada nenhuma transição independente (externa) o sistema simula as transições dependentes (interna ao sistema), sendo possível portanto uma simulação reativa com o meio ambiente (representadas pelas transições independentes).

3.2 Estrutura do SAS

Neste item será apresentada a estrutura interna do sistema ***SAS***, descrita com o auxílio dos Statecharts. O ***SAS*** pode ser representado como um conjunto de módulos manipulando as suas informações, que são armazenadas em um Banco de Dados.

A figura 3.6 representa a estrutura geral do software ***SAS***. O primeiro módulo do sistema é Entrada do Sistema. Após a solicitação para prosseguir, o sistema passa para o módulo Seleção de Arquivos, em que deve ser selecionado o Banco de Dados, que contém as informações dos Statecharts e Sincronismos. Após escolhido o Banco de Dados, deve-se escolher entre os módulos: Statecharts Adaptativos ou Sincronismo.

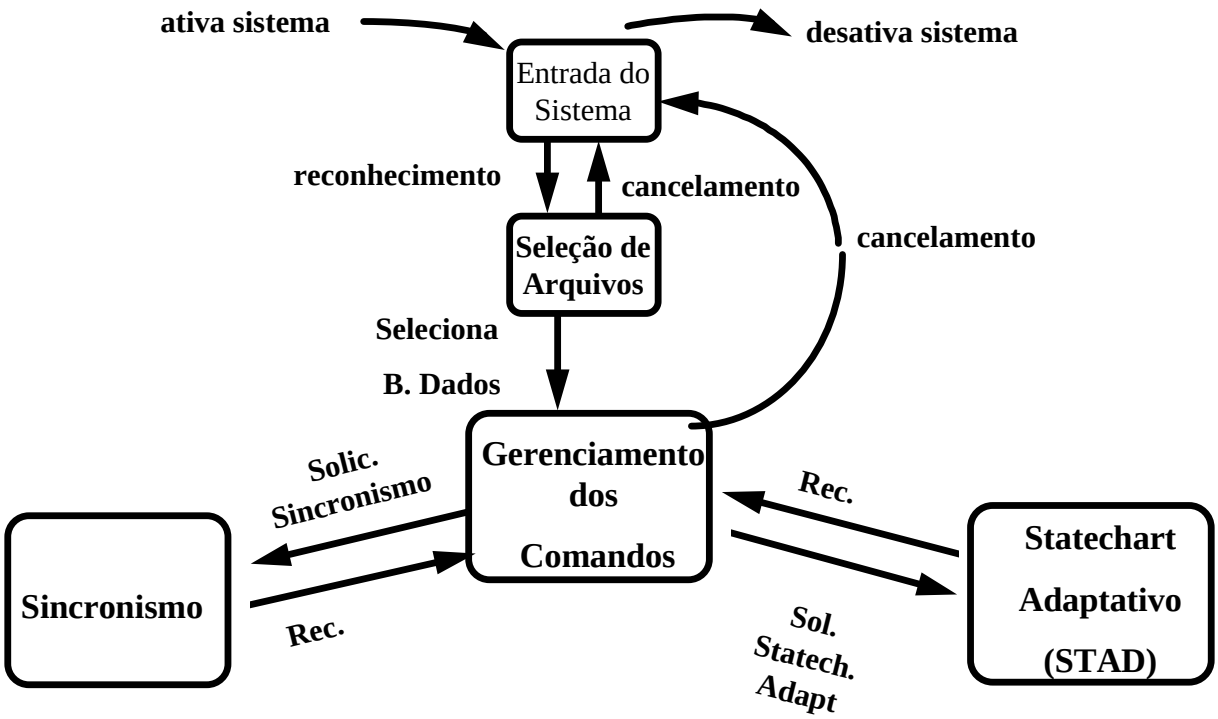


Fig 3.6 - Estrutura Principal do Software SAS

Quando selecionado a opção Statechart Adaptativo o sistema pode ser representado pelos módulos conforme a figura 3.7. O primeiro módulo, Abertura de Projetos, permite abrir um projeto existente ou criar um novo projeto. Após a seleção do Projeto (Statechart), pode-se realizar vários comandos através do módulo Gerenciamento de Comandos do STAD. Os módulos disponíveis são: Tratamento de Projetos, Tratamento de Funções Adaptativas, Tratamento de Funções Auxiliares, Tratamento de Simulação, Edição de Bolhas e Edição de Ligações. Lembrando que esses módulos são parte do software STAD, maiores detalhes desses módulos poderão ser encontrados em (Rady, 1995).

Quando selecionado a opção Sincronismo, o sistema pode ser representado pelos módulos conforme a figura 3.8. O primeiro módulo (Gerenciamento das Opções de Sincronismo) permite escolher entre os módulos: Cadastro e Opção Simulação. Após a escolha do módulo Cadastro , pode-se criar um novo sincronismo ou abrir um sincronismo existente através do módulo Abertura de Sincronismo. Após selecionado o sincronismo (novo ou existente) pode-se escolher um Statechart, através do módulo Escolha de Statechart. Após a escolha do Statechart deve-se escolher a bolha do Statechart que fará parte do sincronismo a partir do módulo Escolha de Bolha. Após a escolha da bolha pode-se continuar o processo para a escolha de outro Statechart/Bolha ou finalizar o processo chamando o módulo Término do Cadastro. Quando for selecionado o módulo Opção Simulação deve-se escolher o conjunto de Statecharts que serão simulados através do módulo Seleção de Statecharts. Após escolhidos os Statecharts pode-se simulá-los através do módulo Execução da Simulação.

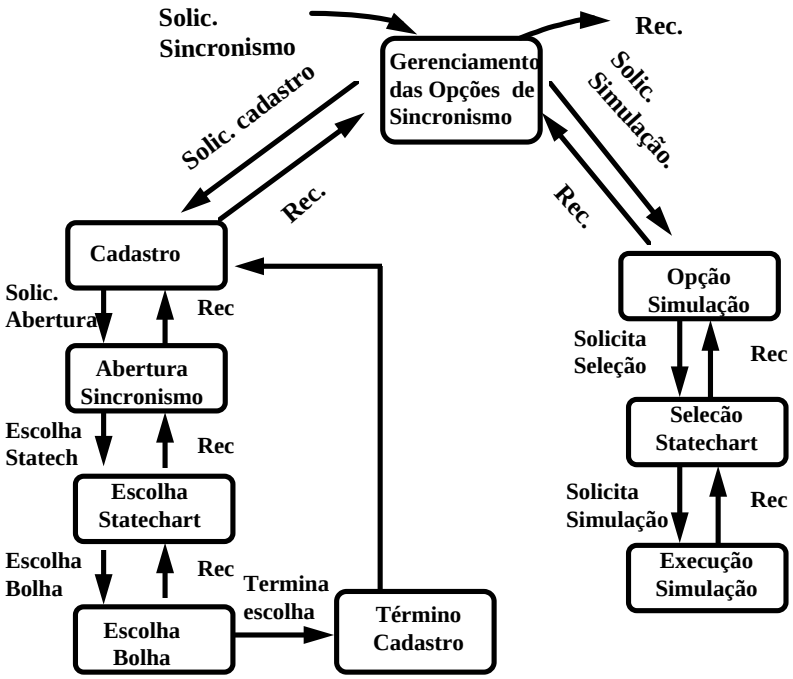


Fig 3.7 - Estrutura do módulo Statechart Adaptativo

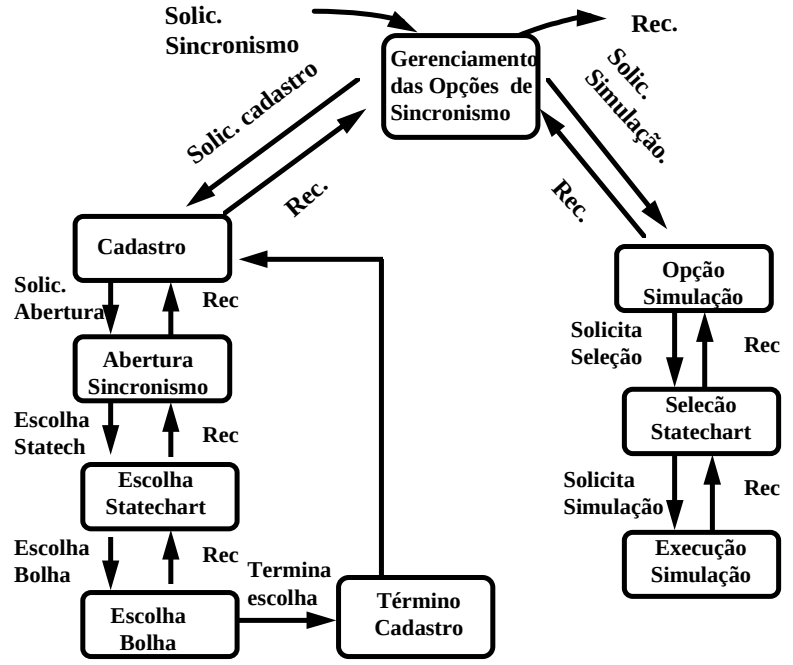


Fig 3.8 - Estrutura do módulo Sincronismo

3.3 Detalhamento das Informações Armazenadas

As informações dos Projetos (Statecharts) e dos sincronismos são armazenadas no Banco de Dados (arquivo ***tese1.mdb*** do *software Microsoft Access*), composto das seguintes tabelas:

Tabela Bolhas

Contém as informações relativas às bolhas dos Projetos, que são:

Projeto ao qual a bolha pertence

Nível de detalhamento

Tipo da Bolha - se a bolha é do tipo OR, AND ou Primitiva. Quando for uma bolha OR informa ainda se ela é Default, History ou simples.

Nome - identifica a bolha dentro de um projeto (único)

Localização física da bolha dentro do Statechart

Tamanho

Nome da bolha Pai, quando se trata de um detalhamento

Nome da função Adaptativa da qual faz parte, quando for o caso.

Estado - indicando se é uma bolha ativa ou não durante a simulação.

Tabela de Ligações

Contém as seguintes informações relativas às ligações dos projetos:

Projeto ao qual a ligação faz parte

Nível de detalhamento

Tipo: Dependente, Independente

Evento, Condição, Disparo, Atraso, Tempo de Disparo

Bolhas de destino e de origem

Nome da bolha par quando se tratar de detalhamento

Nome da função Adaptativa da qual faz parte, quando for o caso.

Tabela de Setas

Contém as seguintes informações relativas às representações das setas que compõem cada ligação dos Projetos:

Projeto ao qual a seta faz parte

Nível de detalhamento

Localização física da Seta dentro do Statechart

Nome da bolha pai (bolha ascendente) quando se tratar de detalhamento

Nome da função adaptativa da qual faz parte, quando for o caso.

Tabela Adaptativa

Contém as seguiontes informações relativas às funções Adaptativas e das Ações Adaptativas (instâncias das funções):
Projeto ao qual a função faz parte
Tipo de ação (inclusão ou exclusão de elemento)
Tipo do elemento (bolha ou ligação)
Bolhas de origem e destino, quando se tratar de ligação

Tabela Sincronismos
Contém as seguintes informações dos sincronismos:
Nome do sincronismo
Identificação se está ativo ou não durante a simulação.

Tabela Sincr_Bolhas
Contém as seguintes informações:
Sincronismo ao qual pertence
Nome do Statechart
Nome da Bolha do Statechart que é sincronizada
Tipo: se é uma bolha de entrada ou de saída do sincronismo
Qual janela da simulação contém o Statechart
Estado: Indica se o processo do sincronismo está Ativo ou não

3.4 Exemplos

A seguir vamos apresentar alguns exemplos para ilustrar a descrição do software SAS e o seu uso.

3.4.1 Exemplo 1

Neste primeiro exemplo, vamos considerar um sincronismo típico entre dois processos do tipo cliente-servidor (*Client-Server*). Esses processos podem representar, por exemplo, o pedido para gravação de informação em um Banco de Dados e a sua autorização. O processo cliente (o que faz o pedido para gravação) solicita um recurso (que representa um dispositivo, por exemplo um processador) ao processo servidor (aquele que disponibiliza a informação, representado pelo processo que faz a autorização). O processo que deve ser sincronizado é a gravação da informação em um

Banco de Dados, sendo só executado quando for solicitado e o recurso estiver disponível.

Nosso primeiro sistema é composto de várias operações em um sistema como consultas e relatórios além de um processo de gravação de informações. O sistema permanece no estado de "*Consultas/Relatórios*" enquanto não ocorrer o evento "*novo dado*". Quando ocorrer esse evento o sistema passará ao estado " *Entrada de Informações*" e depois ao estado "*Solicita Gravação*" Nesse último estado o sistema necessita de uma interação com o outro sistema, podendo passar ao próximo estado ("Gravação efetuada") somente após a sincronização e a consequente realização do processo "*Gravação da Informação*".

O nosso segundo sistema é composto de um estado "*Recurso Disponível*" que só passará ao estado "*Recurso Liberado*" somente após a execução do processo de sincronização ("*Gravação da Informação*").

A figura 3.9 representa graficamente os dois Statecharts (descritos acima) conforme o formalismo ***STAD-Sinc***.

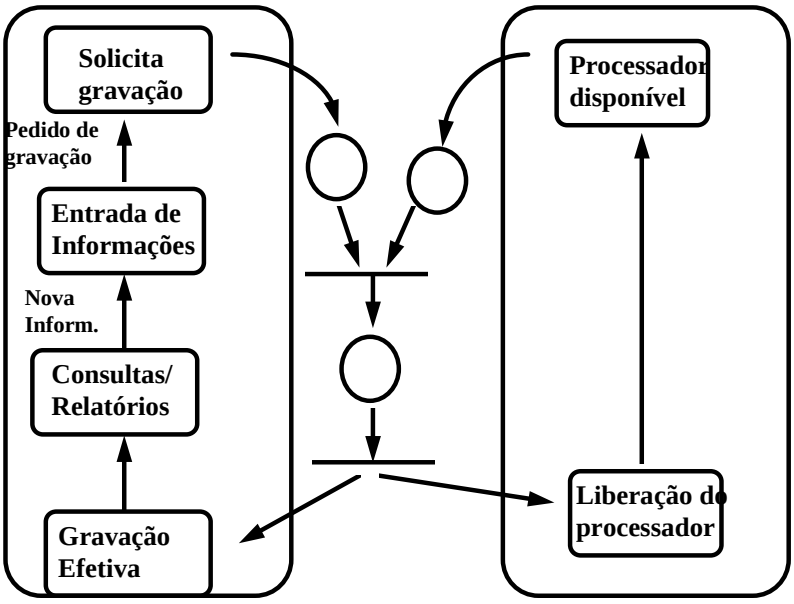


Fig. 3.9 - Representação do Exemplo 1, conforme Stad-Sinc.

A seguir vamos apresentar a forma como é realizada a simulação, para o sincronismo acima descrito, através da ferramenta ***SAS***. Primeiro temos na figura 3.10 a representação do Statechart simbolizando o processo cliente, e a figura 3.11 o processo servidor.

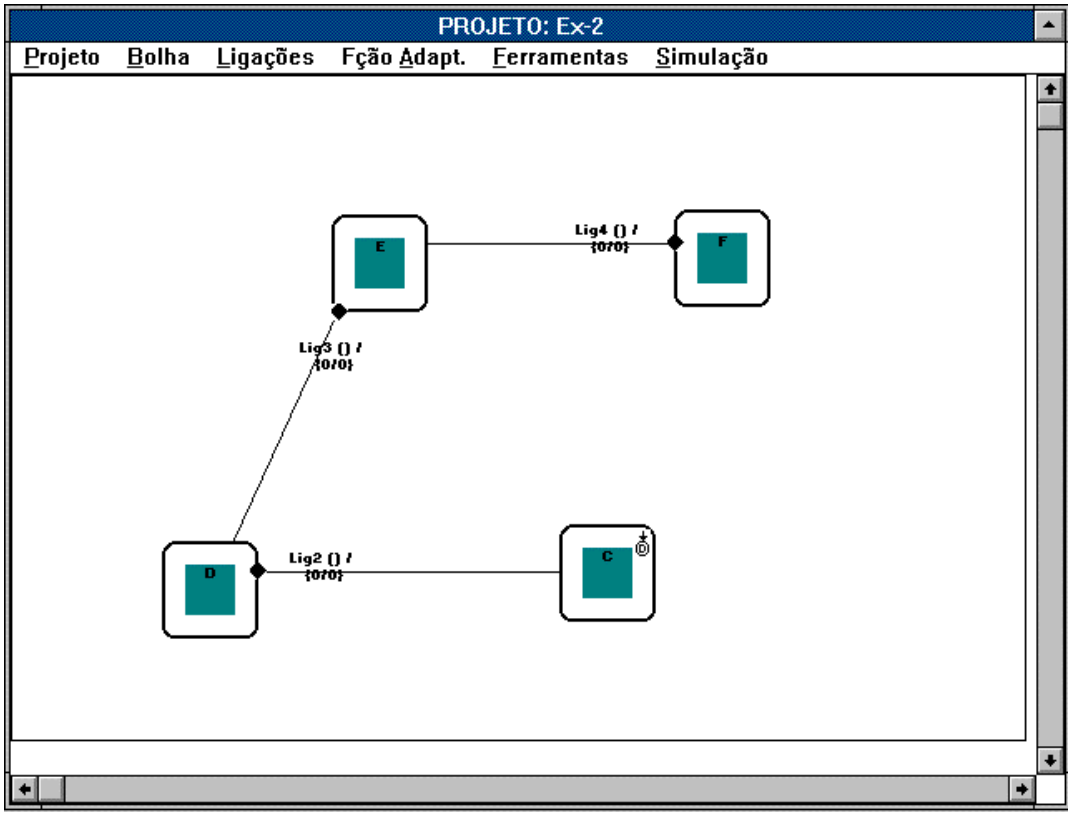


Fig. 3.10 - Representação do Statechart do Cliente no SAS

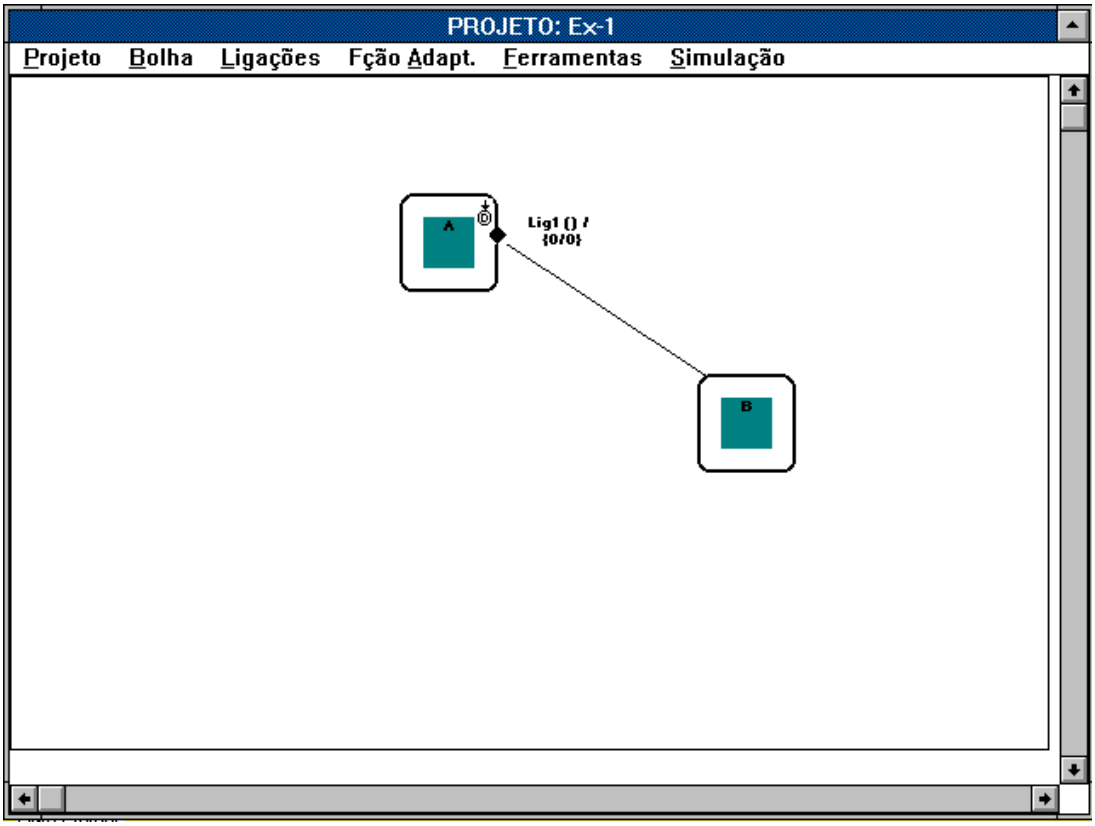


Fig. 3.11 - Representação do Processo servidor, no SAS.

Na figura 3.12 temos a tela de simulação do sincronismo. A figura 3.13 ilustra as estatísticas após 1 execução do processo de sincronismo.

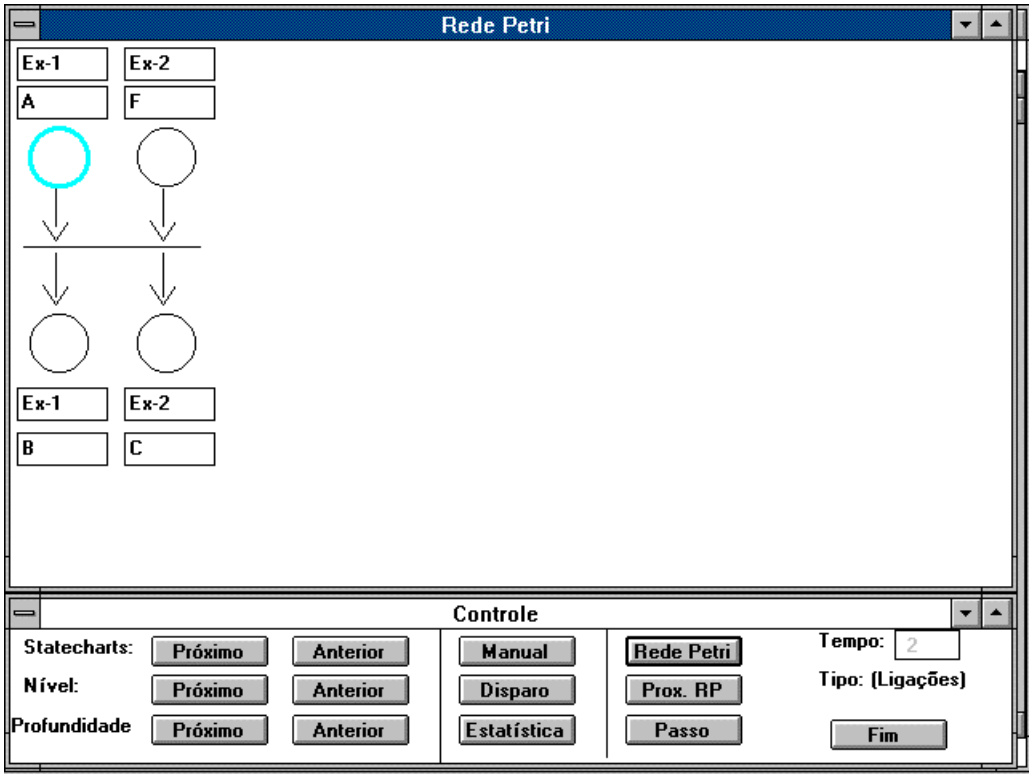


Fig. 3.12 - Representação do Processo de sincronismo, no SAS.

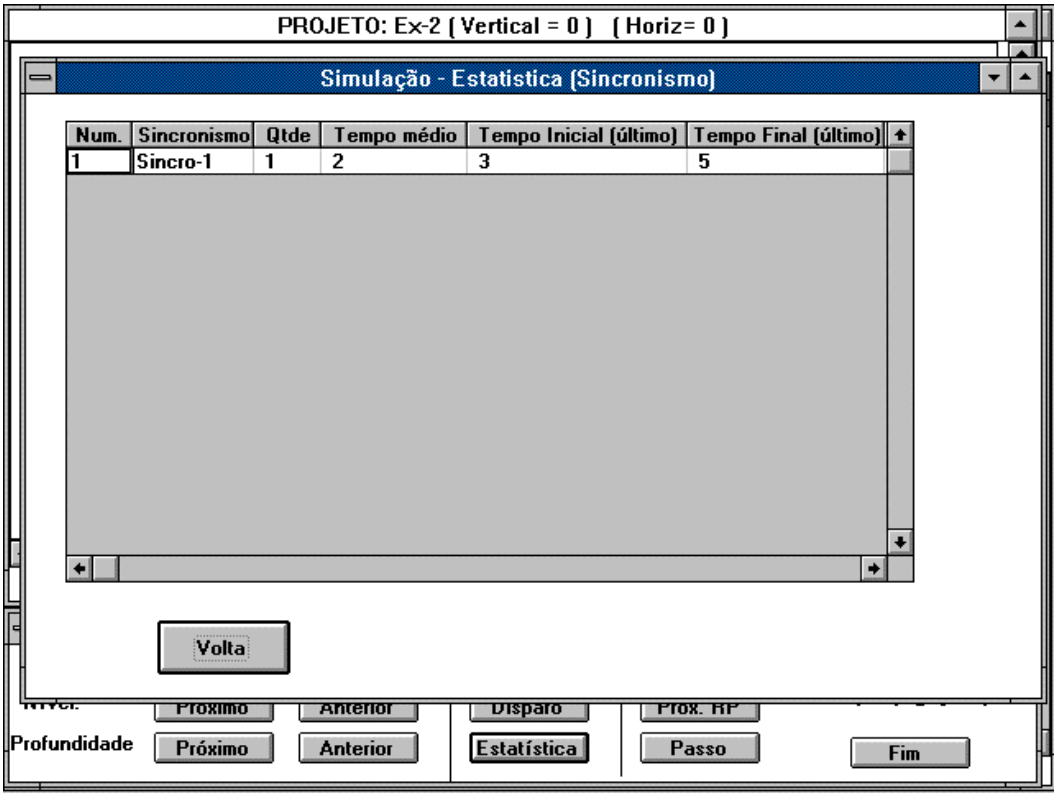


Fig. 3.13 - Tabela de Estatística de sincronismo, no SAS.

Exemplo 2

Nesse exemplo mostraremos a interação entre dois sistemas distintos. O primeiro pode ser representado por um aluno que realiza as suas lições, e, após realizá-las, submete os resultados à correção de um monitor (professor). O segundo consiste do tratamento do professor que deve atender as necessidades de correções dos exercícios quando solicitado. Na nossa simulação iremos considerar dois alunos e um professor. A figura 3.14 representa o problema dos dois alunos e um professor através do formalismo **STAD-Sinc**. A figura 3.15 mostra a tela que pede a confirmação da simulação, mostrando os Statecharts selecionados e os sincronismos envolvidos. Os Statecharts do **SAS** são: ***Ex-4B*** (simbolizando um módulo do aluno), ***Ex-4C*** (simbolizando o segundo módulo do aluno) e ***Ex-5*** (simbolizando o módulo do professor). As representações das sincronizações envolvidas estão representadas no SAS por: ***Sincr-2A*** e ***Sincr-2B***. Cada um dos sincronismos representa a interação de um dos alunos e o professor.

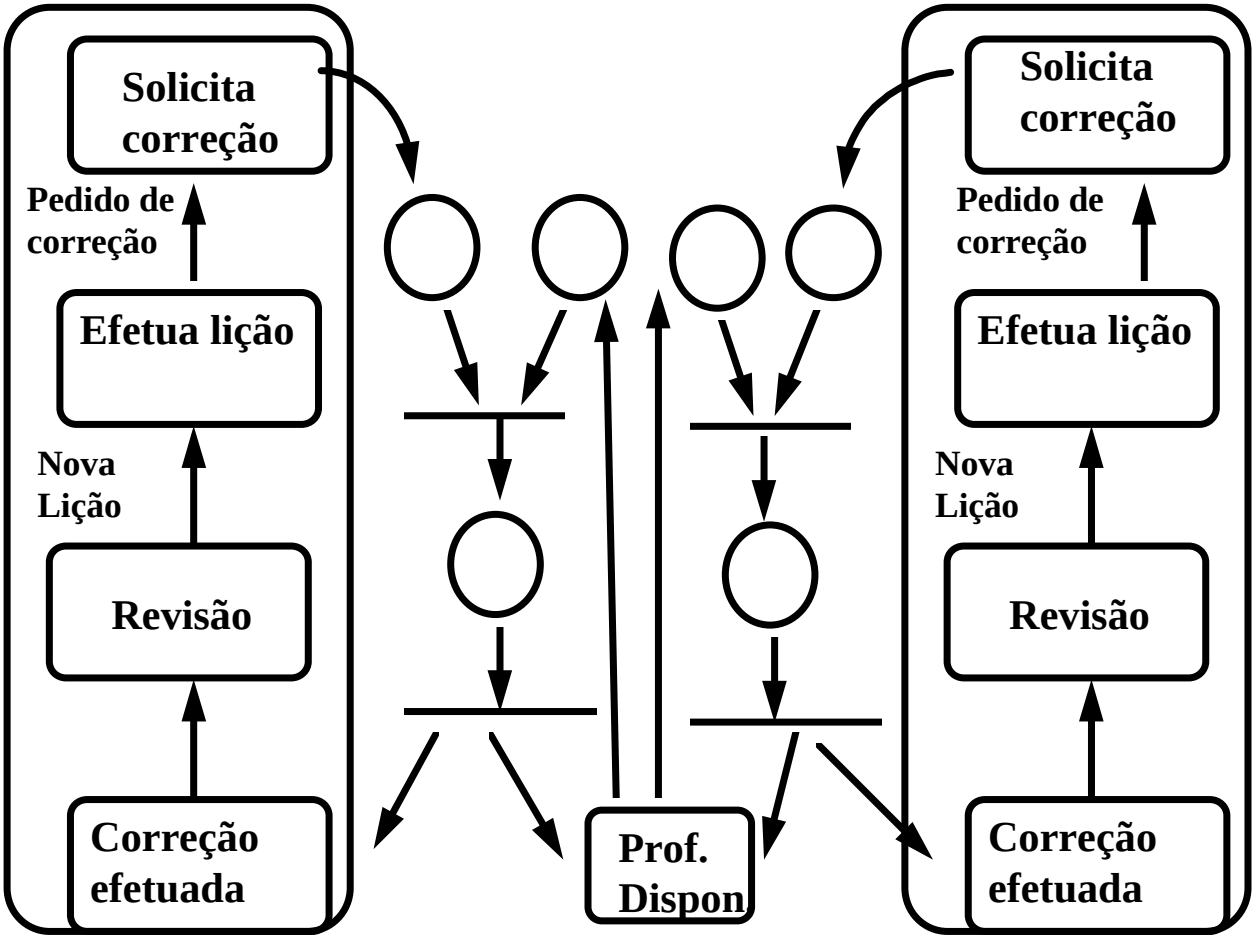


Fig. 3.14 - Representação do exemplo-2, no STAD-Sinc.

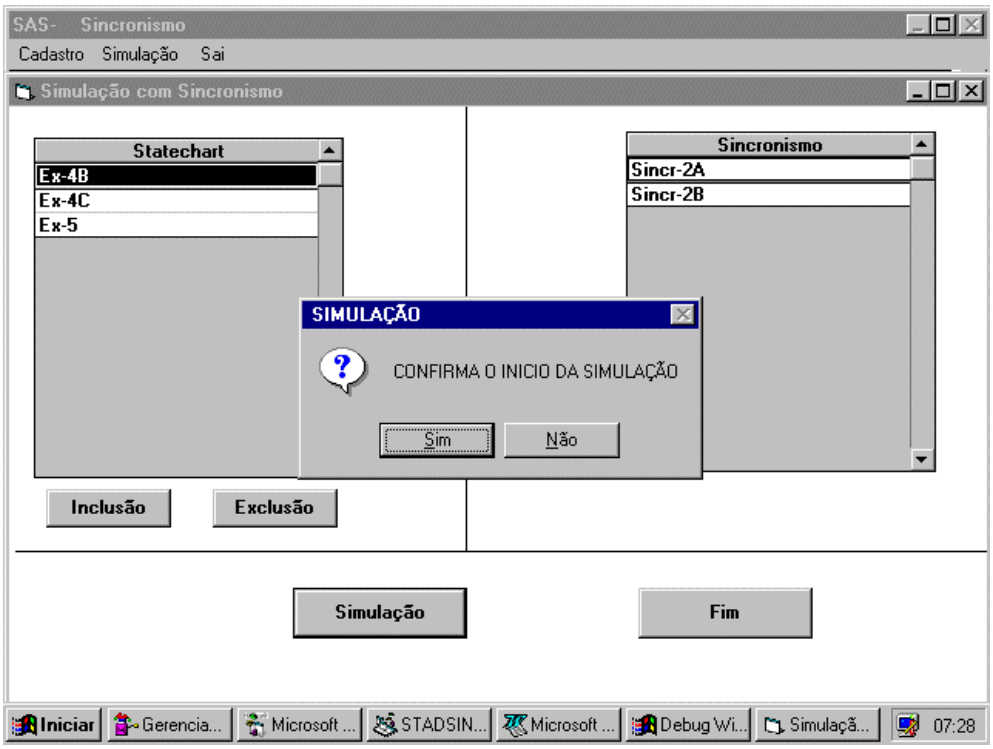


Fig. 3.15 - Tela apresentando Statecharts e sincronismos que serão simulados, no SAS

O sincronismo Sincr-2A representa o sincronismo existente entre os Statecharts Ex-4B (aluno 1) e Ex-5 (professor) enquanto que o Sincr-2B entre os Statecharts Ex-4C (aluno 2) e Ex-5 (professor).

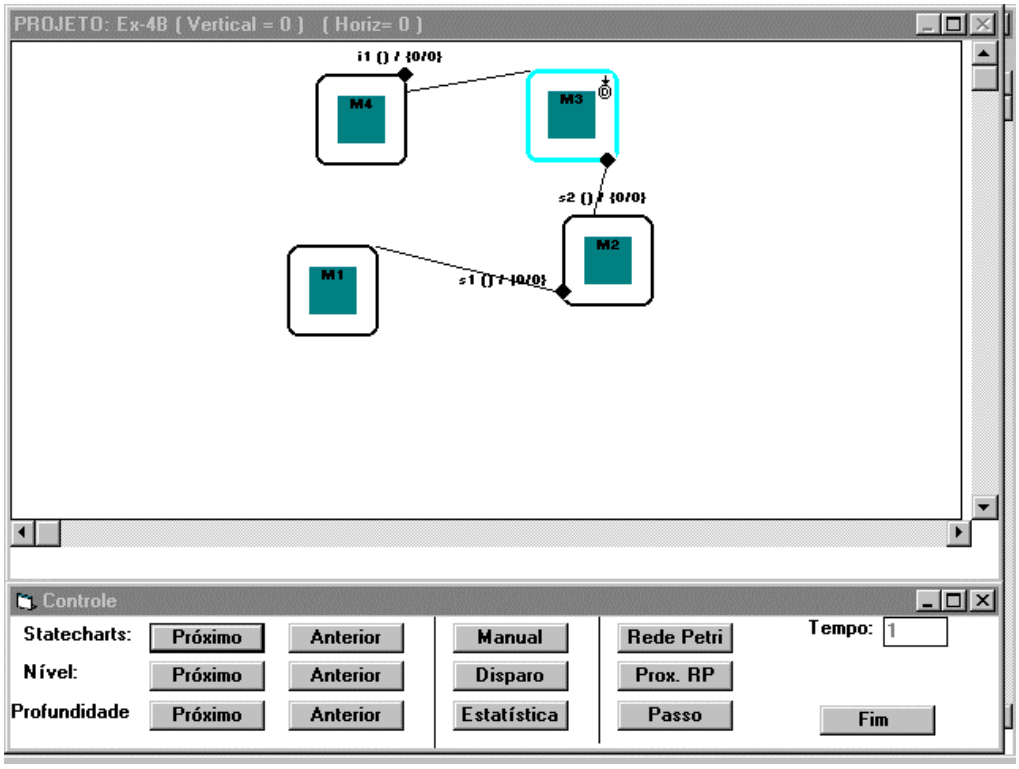


Fig. 3.16 - Representação do Statechart do módulo aluno (aluno 1), no SAS.

Para simplificar a visualização dos diagramas, foram utilizados nomes com apenas uma letra para definirem as bolhas e ligações. No módulo do aluno 1 (figura 3.16) temos:

- Bolha M1: representa "Correção efetuada"
- Bolha M2: representa "Revisão"
- Bolha M3: representa "Efetua lição"
- Bolha M4: representa "Solicita Correção"
- Ligação s2: representa "Nova lição"
- Ligação i1: representa "Pedido de Correção"

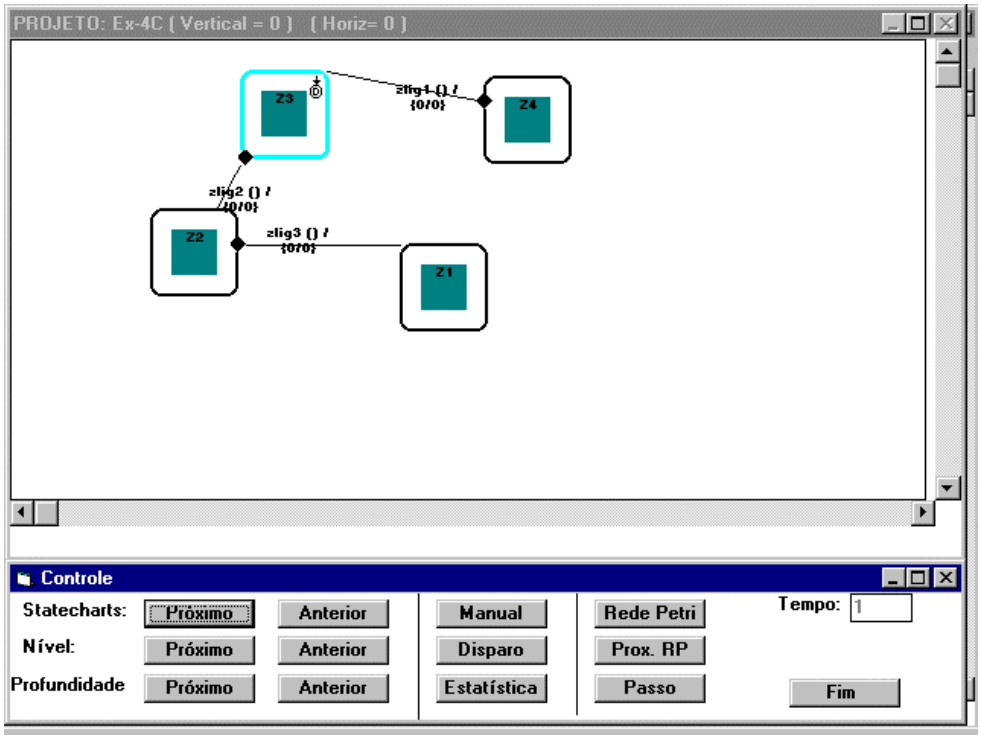


Fig. 3.17 - Representação do Statechart do módulo aluno (aluno 2), no SAS.

No módulo do aluno 2 (figura 3.17) temos:

- Bolha Z1: representa "Correção efetuada"
- Bolha Z2: representa "Revisão"
- Bolha Z3: representa "Efetua lição"
- Bolha Z4: representa "Solicita Correção"
- Ligação zlig2: representa "Nova lição"
- Ligação zlig1: representa "Pedido de Correção"

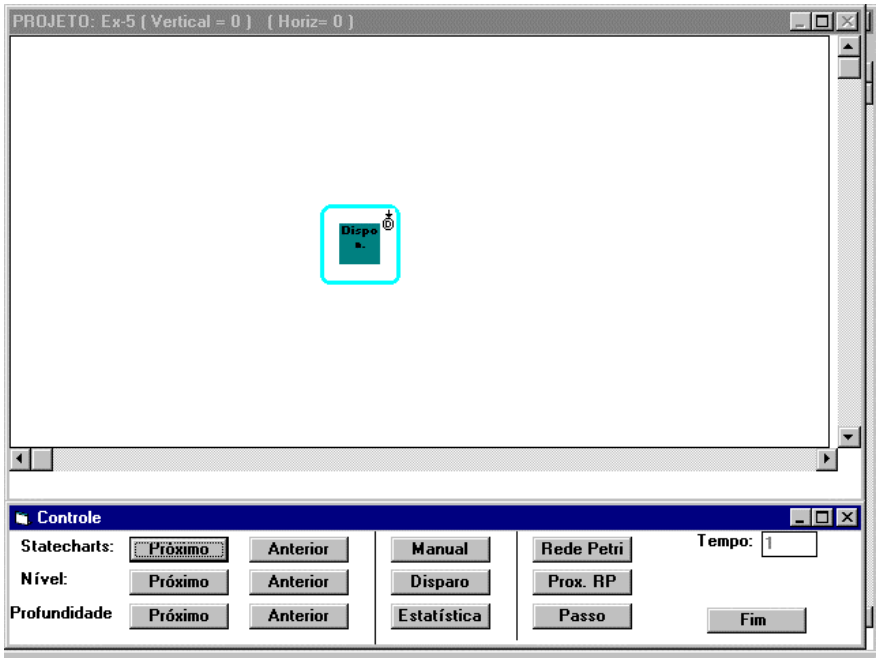


Fig. 3.18 - Representação do Statechart do módulo professor, no SAS.

No módulo do professor (figura 3.18) temos:

- Bolha Dispon.: representa "Professor disponível".

No início da simulação temos o professor disponível, mas os alunos estão efetuando as lições, representadas pelas bolhas *Z3* e *M3*. A figura 3.19 ilustra a representação do sincronismo *Sincr-2A*, envolvendo as bolhas *M4* ("Solicita correção" do módulo aluno) e *Dispon*. ("Disponível " do módulo professor)

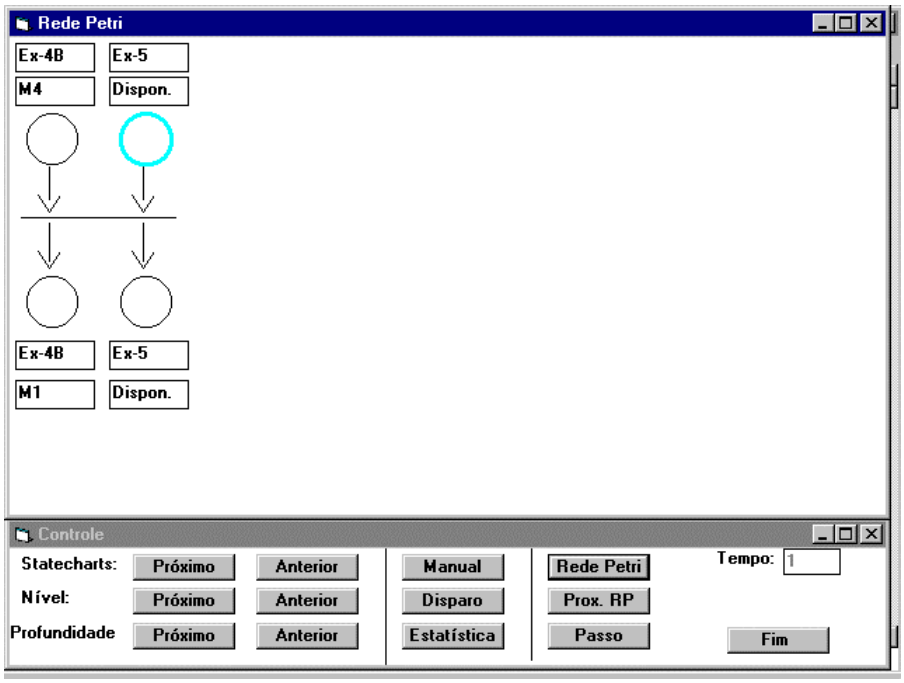


Fig. 3.19 - Representação do Sincronismo Sincr-2A, no SAS.

A figura 3.19 ilustra a representação do sincronismo *Sincr-2A*, envolvendo as bolhas *M4* ("Solicita correção" do módulo aluno 1) e *Dispon*. ("Disponível " do módulo professor). A figura 3.20 ilustra a representação do sincronismo *Sincr-2B*, envolvendo as bolhas *Z4* ("Solicita correção" do módulo aluno 2) e *Dispon*. ("Disponível " do módulo professor). Pode-se verificar que no início o lugar (estado da Rede de Petri) que representa a disponibilidade do professor está ativa nos dois sincronismos (na figura está representado com uma borda mais espessa, e no *software* **SAS** é apresentada com a cor azul).

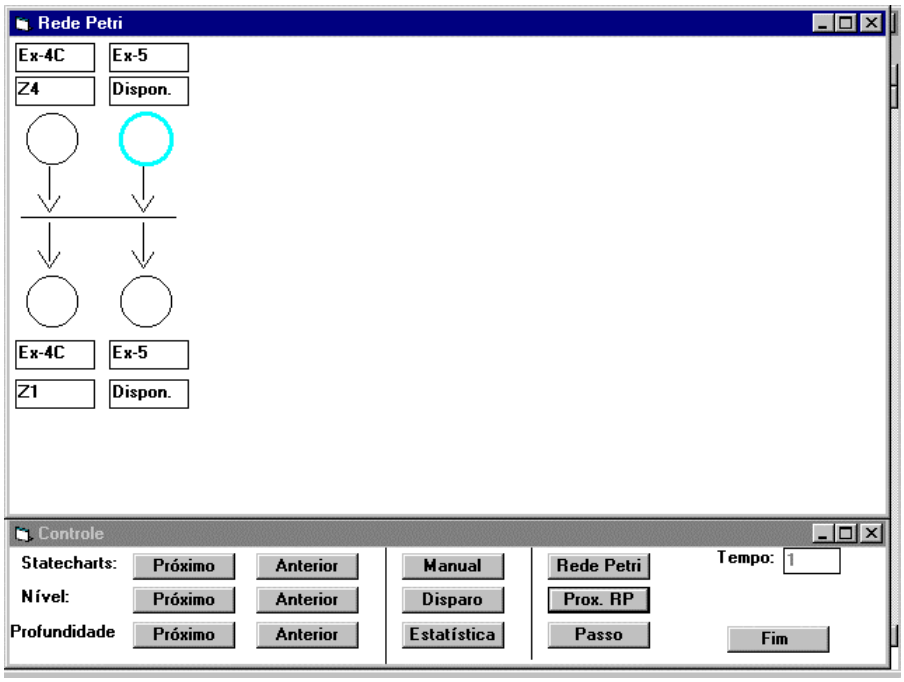


Fig. 3.20 - Representação do Sincronismo Sincr-2B, no SAS.

Vamos analisar a simulação através da tabela representada na figura 3.21.

Unidade Tempo	Aluno - 1	Aluno - 2	Professor	Sincr-2A Aluno1/Prof	Sincr-2B Aluno2/Prof
1	M3	Z3	Dispon.	Dispon.	Dispon.
2	M4	Z4	Dispon.	Dispon./M4	Dispon./Z4
3		Z4		Sincronização	Z4
4	M1	Z4	Dispon.	Dispon	Dispon/Z4
5	M2				Sincronização

Fig. 3.21 - Tabela ilustrando bolhas ativas em cada instante da simulação

No tempo 1, temos as bolhas M3 (do aluno 1), Z3 (aluno 2) e Dispon. (professor) ativas nos Statehcarts. Os dois sincronismos apresentam apenas uma bolha ativa cada um, a bolha Dispon. (professor). No instante 2, temos as bolhas M4 (do aluno 1), Z4 (do aluno 2) e Dispon. (professor) ativas. Nesse instante, os dois sincronismos contem os seus dois processos ativos. Esse instante está ilustrado na figura 3.22 e 3.23.

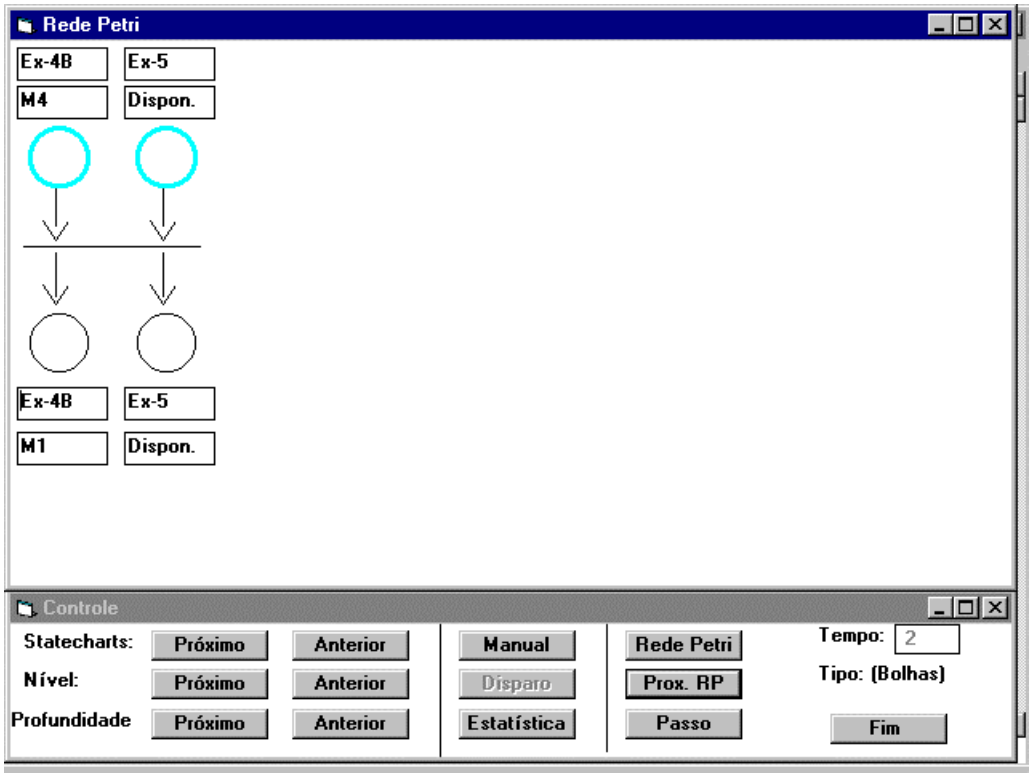


Fig 3.22 - Sincronismo Sincr-2A, com os dois processos ativos.

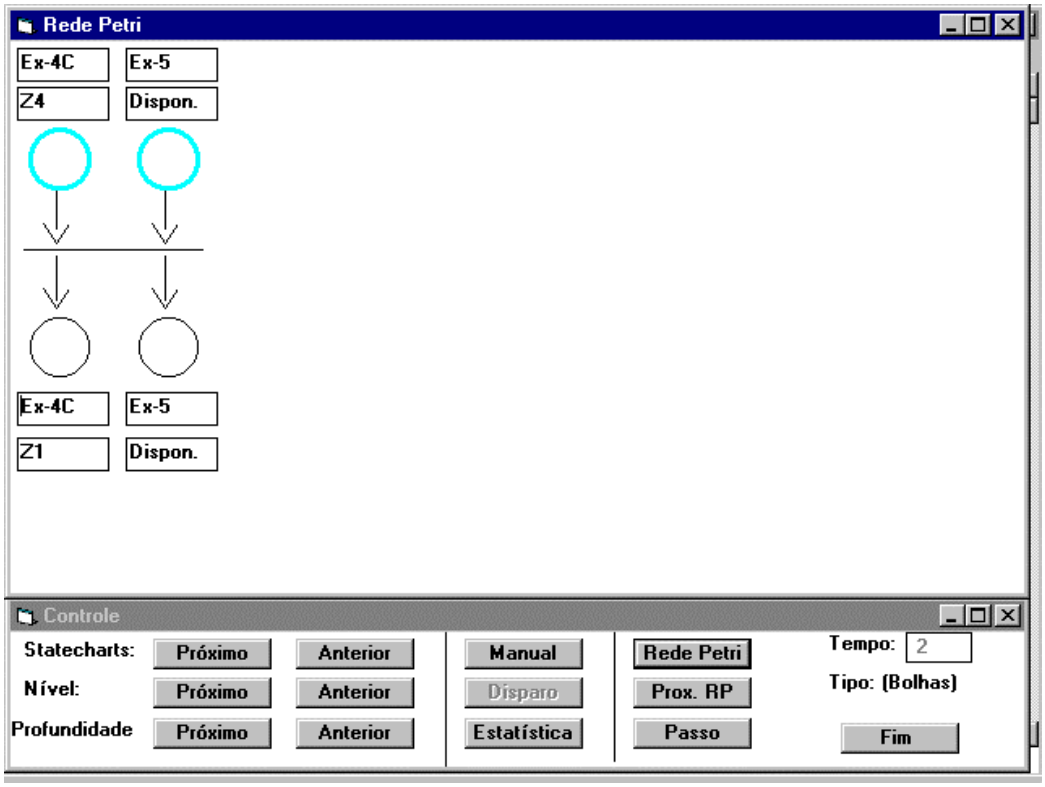


Fig 3.23 - Sincronismo Sincr-2B, com os dois processos ativos.

Mas como os dois sincronismos usam um mesmo recurso, simbolizado pela bolha "Dispon." do módulo professor, apenas um deles será ativado no instante 3. A tabela da figura 3.21 mostra que no instante 3, o processo Sincr-2A é sincronizado, as bolhas M4 e Dispon. deixam de ser ativas nos Statecharts. A figura 3.24 ilustra como é visualizado a sincronização do Sincr-2A no SAS no instante 3. O processo Sincr-2B identifica apenas o processo Z4 como disponível no instante 3, esperando ainda pela sincronização, como ilustra a figura 3.25.

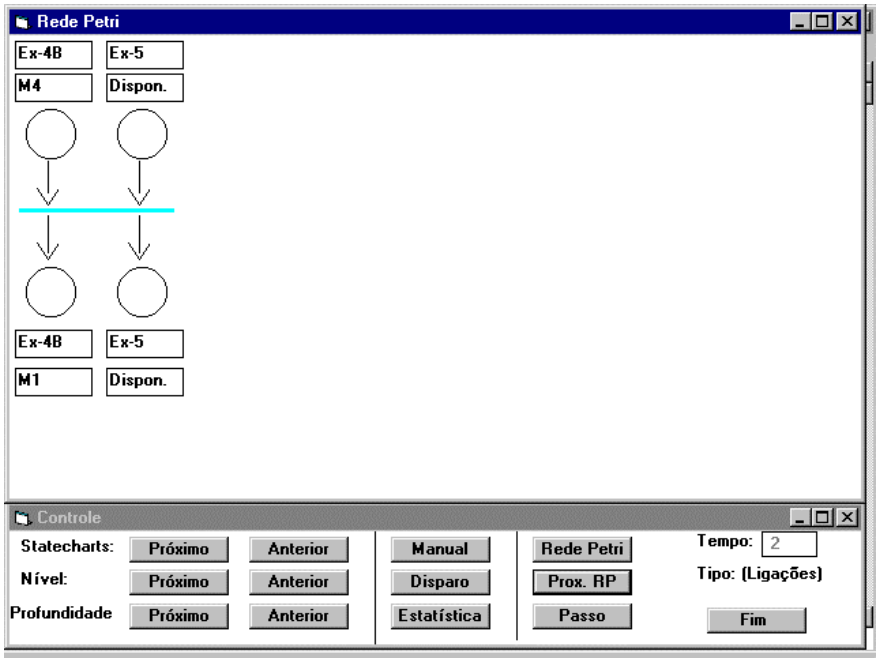


Fig 3.24 - Processo Sincr-2A sincronizado, no instante 3

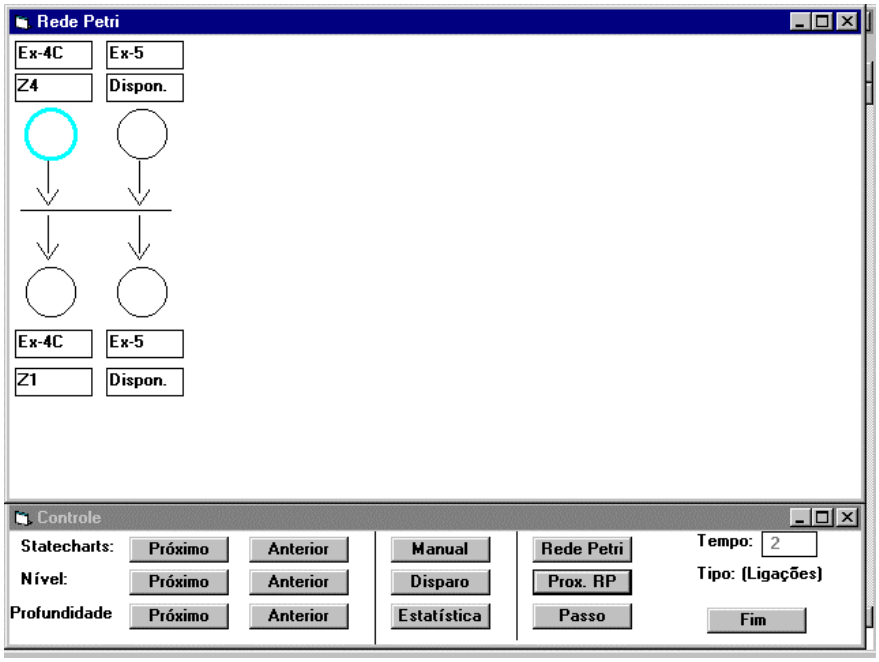


Fig. 3.25 - Sincronismo Sincr-2B, no instante 3 da simulação

O processo Sincr-2B só terá suas duas bolhas disponíveis no instante 4, sendo então sincronizado no instante 5. A figura 3.26 ilustra a tabela de estatísticas dos sincronismos que o SAS oferece. Através dessa tabela podemos verificar que o sincronismo Sincr-2B teve um processo disponivel no instante 1, mas só terminou a execução no tempo 7, levando 6 intervalos de tempo para finalizar o processo. Já o sincronismo Sincr-2A teve o primeiro processo disponivel no instante 1 e terminou a execução no instante 4, levando portanto apenas 3 intervalos para a sua execução. Essa

tabela acumula para cada sincronismo a quantidade que já foi efetuada e os seus tempos médios de duração. Com isso pode-se ter uma idéia da demora de cada sincronismo, e avaliar o seu desempenho e a demanda/recurso envolvidos no processo.

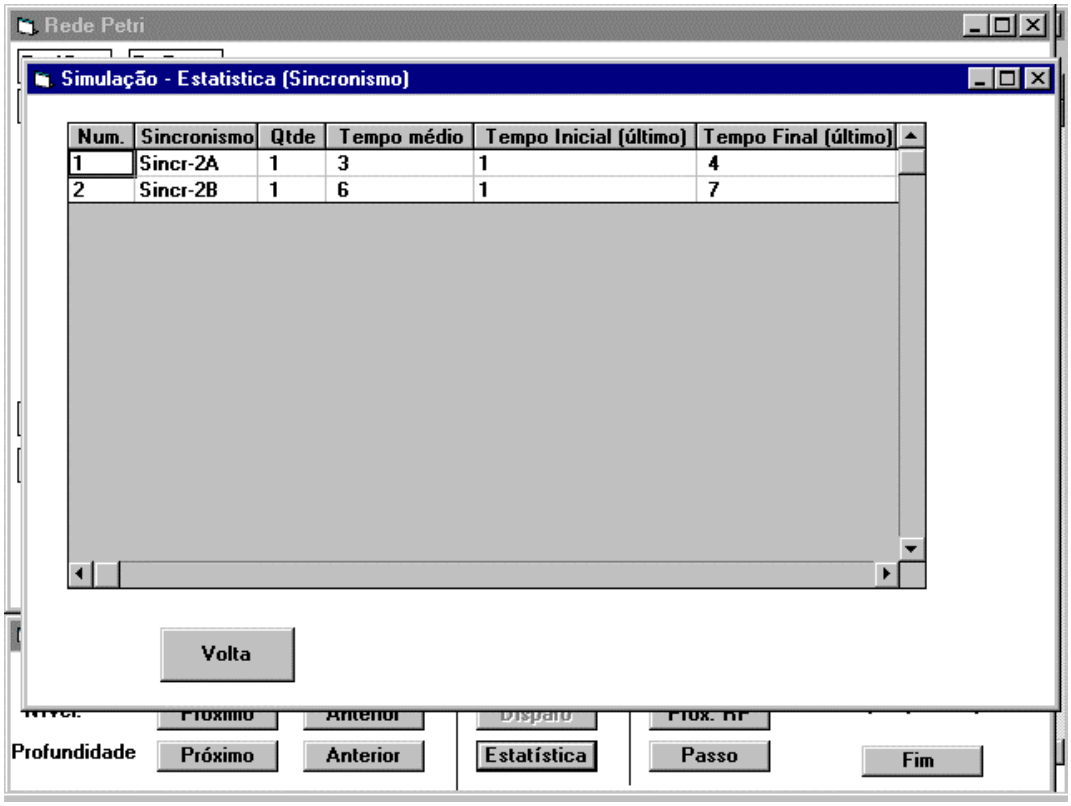


Fig. 3.26 - Tabela de estatística dos sincronismo do SAS

Exemplo 3

Nesse exemplo vamos ilustrar a aplicação de uma função adaptativa (statechart Adaptativo). É semelhante ao exemplo 2, com o acréscimo que teremos uma bolha do Statechart do aluno que cria a próxima lição a cada execução de uma correção. Assim a cada instante o número de lições (cada uma representada com uma bolha) é exatamente o necessário para o aluno. Cada aluno então teria então um conjunto diferente de lições.

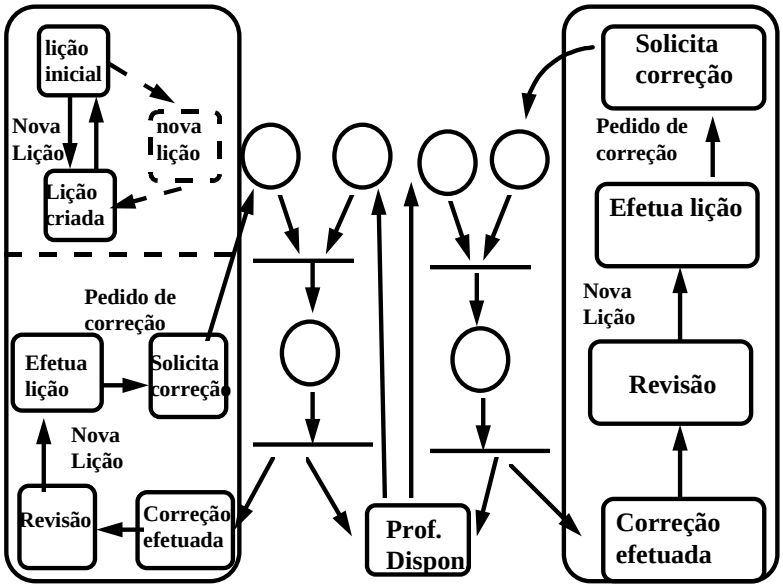


Fig. 3.14 - Representação da configuração inicial do exemplo-3, no SAS.

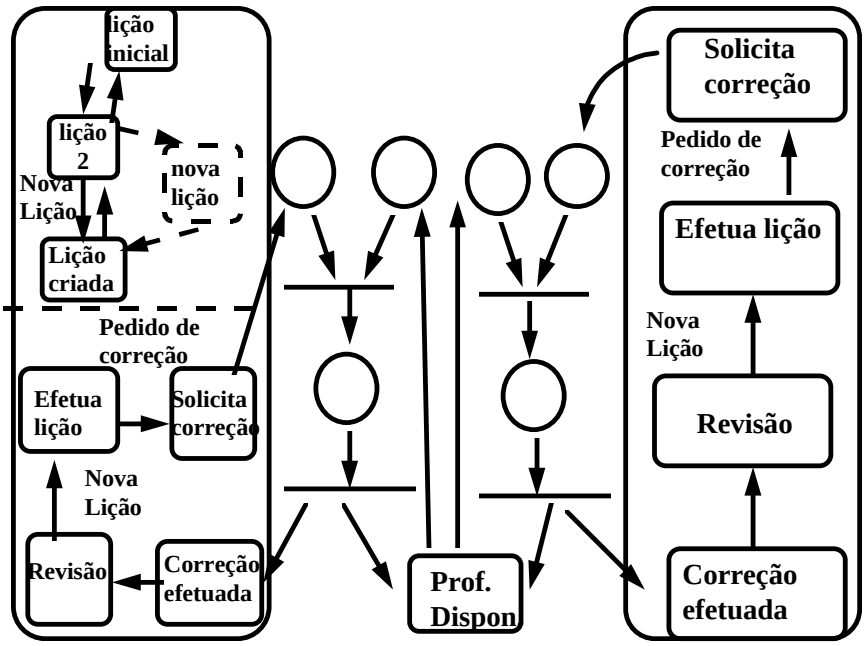


Fig. 3.14 - Representação do exemplo-3 após 1 transição adaptativa, no SAS.

CAPÍTULO 4 - CONCLUSÕES

Neste capítulo serão apresentados os resultados obtidos com o formalismo proposto (***STAD-Sinc***) e a ferramenta construída (software ***SAS***), as sugestões para aperfeiçoar a ferramenta, e, finalizando, as conclusões dessa pesquisa.

4.1 Análise dos resultados

Analisando a necessidade natural de comunicação entre sistemas distintos e as possibilidades de representações através dos formalismos tradicionais, esse trabalho apresenta a proposta de um formalismo com essa finalidade, tendo como sua principal aplicação a interação de sincronismos entre diversos sistemas. O formalismo proposto teve como principal requisito a facilidade de compreensão e visualização dos fenômenos que descrevem a sincronização na sua representação visual. Dessa forma o ***STAD-Sinc*** procura contribuir para simplificar a especificação de sistemas contendo sincronismos, propiciando diagramas de fácil compreensão, adicionando aos Statecharts (eventualmente Adaptativos) um mecanismo para a representação de sincronismos.

O formalismo teórico apresentado foi baseado nos modelos disponíveis dos Statecharts, dos Autômatos Adaptativos, dos Statecharts Adaptativos e das Redes de Petri. Como é conhecida a praticidade de se representar sincronismos através da Rede de Petri e representar sistemas reativos através de Statecharts, esse formalismo reuniu essas duas formas de representação em um mesmo formalismo, incluindo ainda a capacidade de aprendizagem, inspirada nos Autômatos Adaptativos, adotando para isso o modelo adaptativo dos Statecharts. Cada componente (Statechart Adaptativo e Rede de Petri) pode ser aplicado para representar a parte correspondente do sistema, adicionando os campos de aplicação com a utilização simultânea de ambas: os Statecharts Adaptativos são utilizados na especificação dos sistemas (processos) e a

Rede de Petri é utilizada preferencialmente para representar os sincronismos entre seus processos. Deve-se ressaltar que, como todo formalismo, a sua aplicação só traz resultados positivos quando for bem utilizado, ou seja, quando os sistemas a serem representados forem devidamente decompostos em seus elementos conforme acima descrito, usando-se a parte do formalismo estritamente adequada para a representação de cada um.

A ferramenta ***SAS*** possibilitou a representação de sincronizações verificando-se facilmente a sua simplicidade através de alguns exemplos. A representação adotada durante a simulação de vários Statecharts Adaptativos e seus sincronismos, em que cada Statechart e cada sincronismo é representado em uma tela separada, se mostrou bastante apropriada. Pode-se visualizar cada nível do Statechart e cada representação de uma sincronização em separado, podendo na tela da sincronização visualizar cada um dos processos envolvidos, verificando quais estão ativos. Mas, por outro lado, o *software* ***SAS*** não possibilita uma representação conjunta dos Statecharts Adaptativos e de seus sincronismos em um único diagrama, como o formalismo ***STAD-Sinc*** possibilita. A ferramenta tem oferecido uma certa facilidade de utilização devido à interface gráfica e à manipulação da maioria de suas opções através do *mouse*.

Na tela de simulação temos as opções disponíveis agrupadas em um painel, facilitando assim a sua utilização. A implementação da simulação de diversos Statecharts e diversos sincronismos foi trabalhosa pois pode-se visualizar qualquer sub-estado de algum estado ativo e isso é constantemente variado durante a simulação. Em um dado instante pode-se visualizar um número arbritário de diagramas (detalhamento de estado). A visualização do resumo das sincronizações durante a simulação se mostrou bastante útil pois dá subsídio para análise do comportamento dos processos dos sistemas nas sincronizações.

4.2 Conclusões

A ferramenta SAS se mostrou prática para se aplicar a especificação do aspecto comportamental de sistemas reativos complexos com sincronismo. A seguir vamos apresentar algumas sugestões de prosseguimento do desenvolvimento da ferramenta, para que ela venha a fornecer mais recursos ao usuário.

- Criar uma opção que auxilie a utilização do software *on-line*.
- Permitir que se visualize em uma mesma tela o detalhamento da bolha.

- Permitir que se visualize em uma tela geral todos os Statecharts e as representações das sincronizações durante a simulação, e escolher até qual nível de detalhamento a ser apresentado (apenas nível 0, nivel 1, etc.); conforme o formalismo **STAD-Sinc**
- Permitir a edição dos elementos do sincronismo, durante a criação dos elementos do Statechart.

Permitir que se crie uma opção de simulação remota, na qual se determina em uma tabela a ocorrência de estímulos externos em cada instante, podendo-se visualizar o resumo das sicnronizações após o término da simulação, para análise.

Concluindo esse trabalho podemos dizer que o Statechart Adaptativo é um formalismo de grande utilização para a especificação do aspecto comportamental de sistemas reativos complexos, por possuir uma representação de fácil interpretação (em relação aos outros formalismos) e ter uma visualização gráfica. Com a adição de propriedades para a representação de sincronismos (***STAD-Sinc***) aumentamos a sua aplicação, podendo representar de um modo formal e claro diversos sistemas distintos entre si e seus sincronismos. A ferramenta desenvolvida (***SAS***) pode ser utilizada, para a especificação de sistemas com sincronismos, bem como para a criação de exercícios práticos durante o aprendizado dos conceitos de Statecharts, Statecharts Adaptativos e de sincronismos, auxiliando o ensino desses conceitos.

ANEXO A - MANUAL DE OPERAÇÃO DO SAS

Neste anexo é apresentado o manual de operação do Software desenvolvido, denominado ***SAS*** (Statecharts Adaptativos com Sincronismos). Serão apresentados as descrições dos elementos que compõem as telas do sistema e depois as telas utilizadas para se cadastrar e simular os sincronismos com as explicações de cada um de seus elementos. Convém ressaltar que não serão descritas as funções que tratam da criação dos Statecharts, porém podem ser consultadas em (Rady, 1995).

A.1 Composição das Telas

Os elementos que compõem as telas do SAS são:

- **Botão:** Permite executar uma função do SAS, que dependerá do contexto onde está inserido. Como padrão em softwares desenvolvidos em ambiente Windows, esse botão pode ser selecionado através do mouse (presionando o botão da direita do mouse quando sua seta indicativa de sua posição na tela estiver sobre o botão), ou através da tecla “Enter” quando o foco estiver sobre o botão (o foco indica qual a opção está selecionada, podendo ser alternada entre os diversos controles da tela através da tecla “Tab”)
- **Menus**: Apresenta uma série de comandos agrupados que quando selecionados executam uma função dos sistema (que depende do contexto). Pode ser selecionado através do mouse ou pelas teclas “Alt” seguida da tecla identificada pela letra grifada na opção do menu.
- **Campos**: São representados por uma caixa, onde deverá ser fornecido a informação referente ao campo.
- **Listas**: São representadas por uma caixa contendo vários nomes, podendo ser utilizada para escolha de uma opção ou apenas para informar um conjunto de

informações.

A.2 Abertura do Sistema

Ao se iniciar o sistema será apresentado a tela representada na figura A.1.

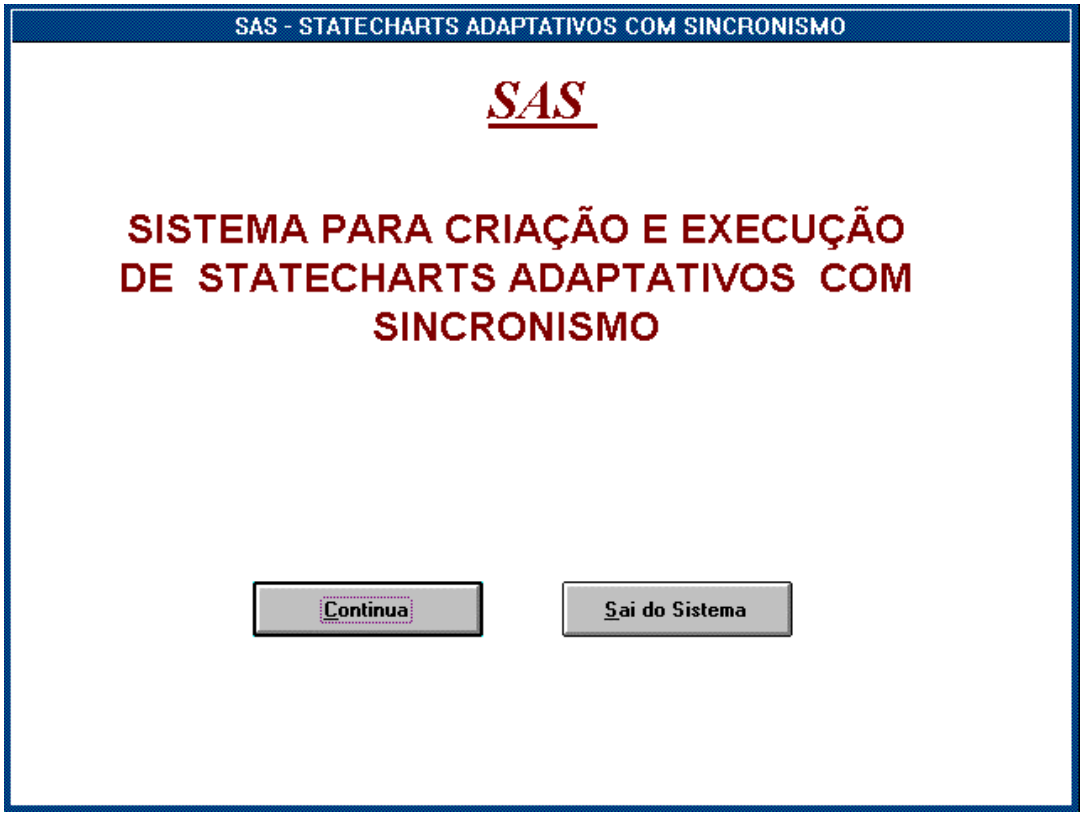


Figura A.1 - Tela inicial de apresentação do SAS

⇒ **Controles da figura A.1**

- **Botão**

  Continua: Prossegue o sistema, apresentando a próxima tela (representada na figura A.2).

  Sai do Sistema: termina a execução do sistema

Quando for selecionado o botão *Continua* o sistema apresentará a tela, representada na figura A.2, onde deverá ser selecionado o arquivo do Banco de Dados *Access*. Nesse arquivo são armazenadas as informações dos Statecharts e Sincronismos . É apresentado uma lista com os nomes dos arquivos encontrados, de acordo com as opções selecionadas sobre sua localização: drive, diretório e tipo do arquivo (extensão).

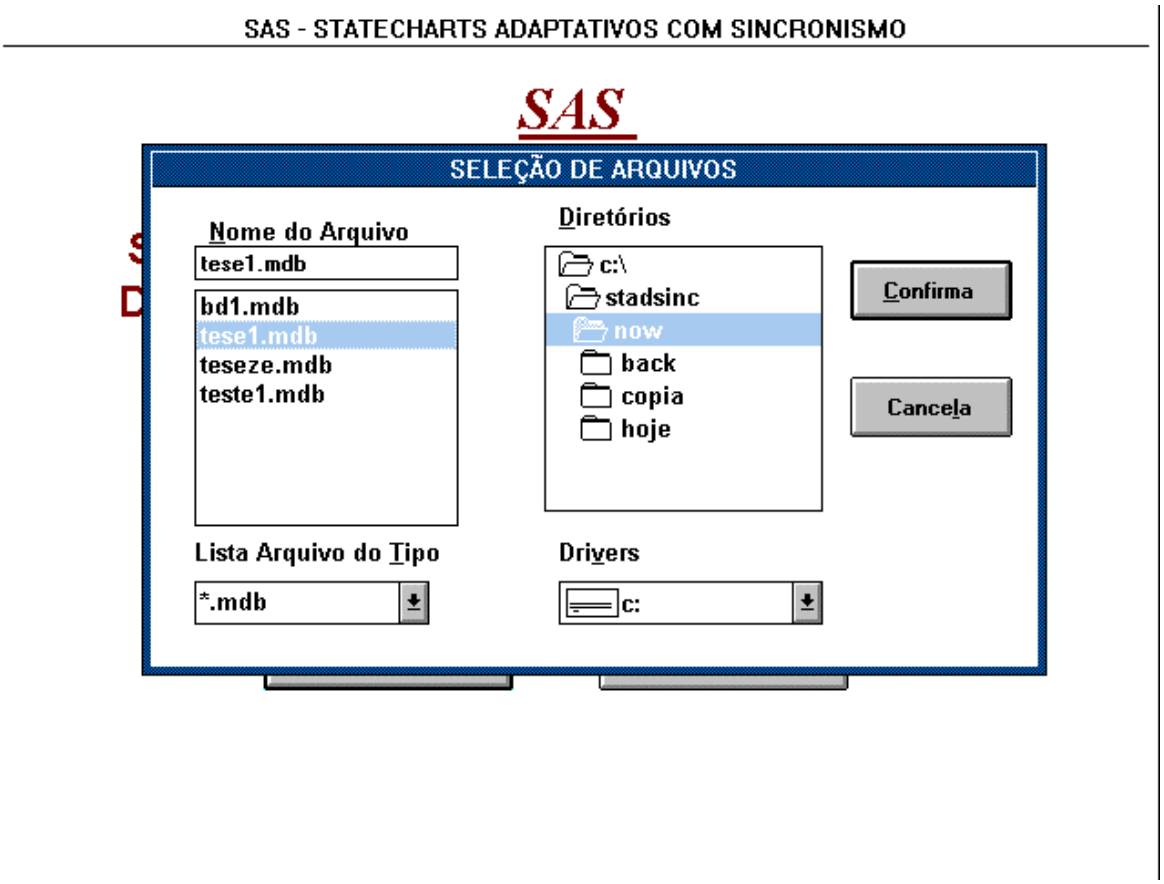


Figura A.2 - Tela de abertura do arquivo do Banco de Dados

⇒ **Controles da figura A.2**

- **Campos**

  Nome do Arquivo: Aqui deve ser selecionado o nome do arquivo do Banco de Dados Access, através do mouse escolhendo-o na lista apresentada (que satisfazem as condições do drive, diretório e tipos selecionados).

- **Listas**

  Diretórios: Apresenta os diretórios do drive selecionado.para escolha

  Drivers: Apresenta a relação de *"drivers"* (unidades de disco) para escolha

  Lista Arquivo do Tipo: Apresenta os tipos de arquivos que devem ser apresentados.

- **Botão**

  Confirma: Abre o arquivo selecionado

  Cancela: Volta à tela anterior

  Quando for selecionado o botão *confirma* será apresentada a tela principal do

sistema, representada na figura A.3.

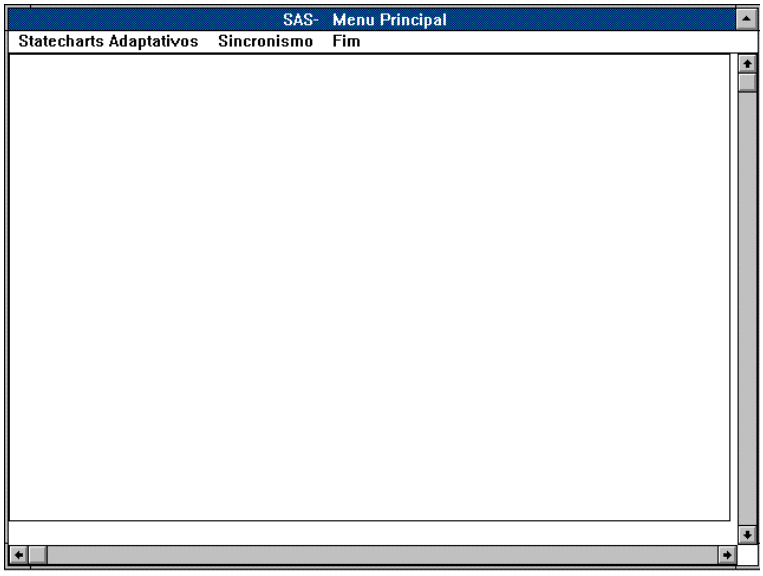


Figura A.3 - Tela com as opção do menu principal do SAS

⇒ **Controles da figura A.3**

- **Menu**

  Statecharts Adaptativos: quando selecionado, apresentará a tela representada na figura A.4 , que contém as opções para se criar Statecharts Adaptativos.

  Sincronismo: quando selecionado, essa opção apresentará a tela representada na figura A.5 que contém as opções para se criar Sincronismos e realizar simulações contendo vários Statecharts Adaptativos e sincronismos.

  Fim: essa opção permite voltar à tela inicial (figura A.1) de onde poderá ser encerrado o sistema ou iniciado novamente.

A.I - MÓDULO STATECHART

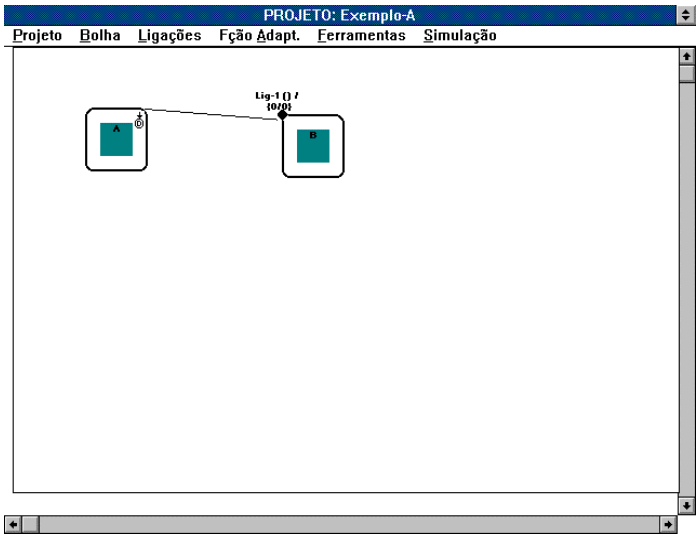


Figura A.4 - Tela com as opções para edição de Statechart Adaptativo

A.II - MÓDULO SINCRONISMO

A figura A.5 ilustra o menu da opção de sincronismo. Esse módulo permite que se criem sincronismos e se simulem vários Statecharts e sincronismos simultaneamente.

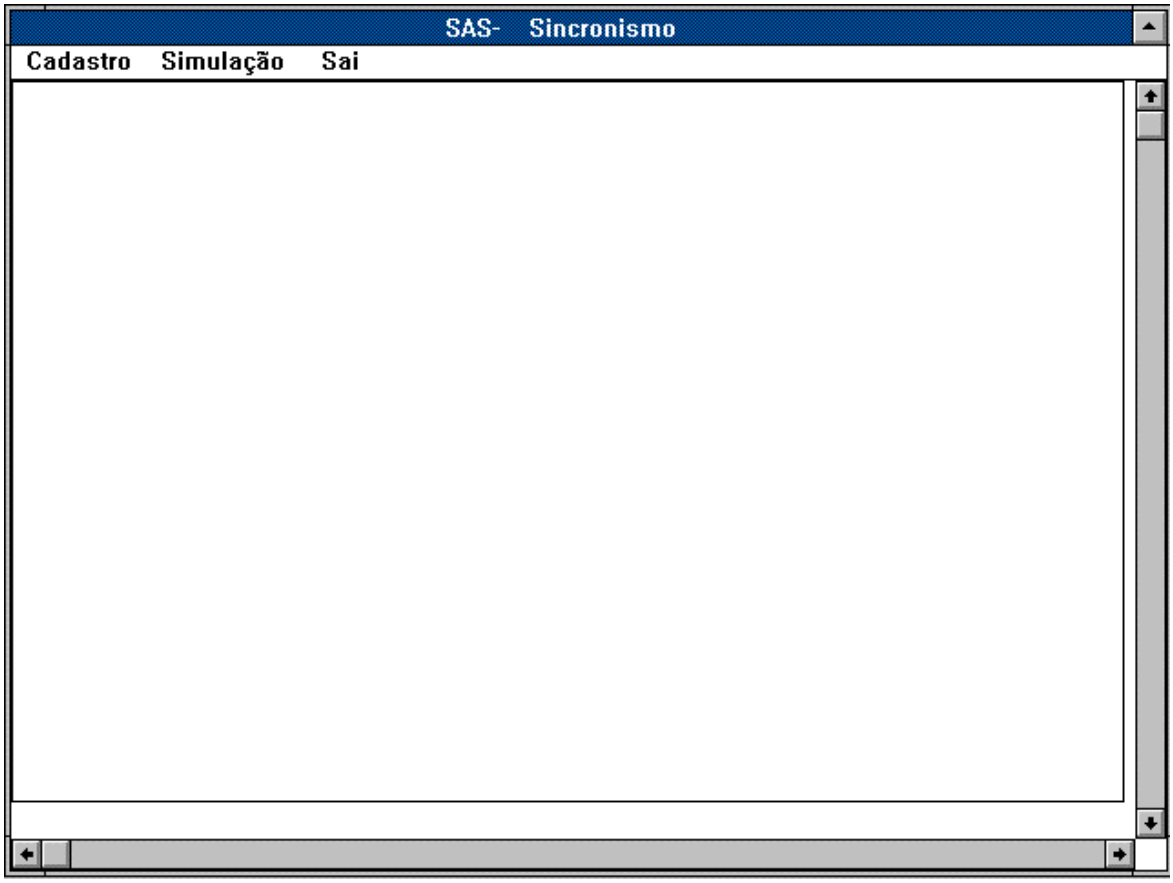


Figura A.5 - Tela com as opções do menu de Sincronismo

⇒ **Controles da figura A.5**

- **Menu**

Cadastro: Quando selecionada esta opção apresentará a tela representada na figura A.6, onde se poderá incluir, alterar ou excluir os sincronismos.

Simulação: quando selecionada, esta opção apresentará a tela representada na figura A., onde será feita a simulação de um conjunto de Statechart e seus sincronismos.

Sai: quando selecionada, esta opção retorna a tela representada na figuraA.3.

A.II.I. MÓDULO SINCRONISMO-CADASTRO

Essa opção permite se criar os sincronismos, assim como alterá-lo e excluí-los.

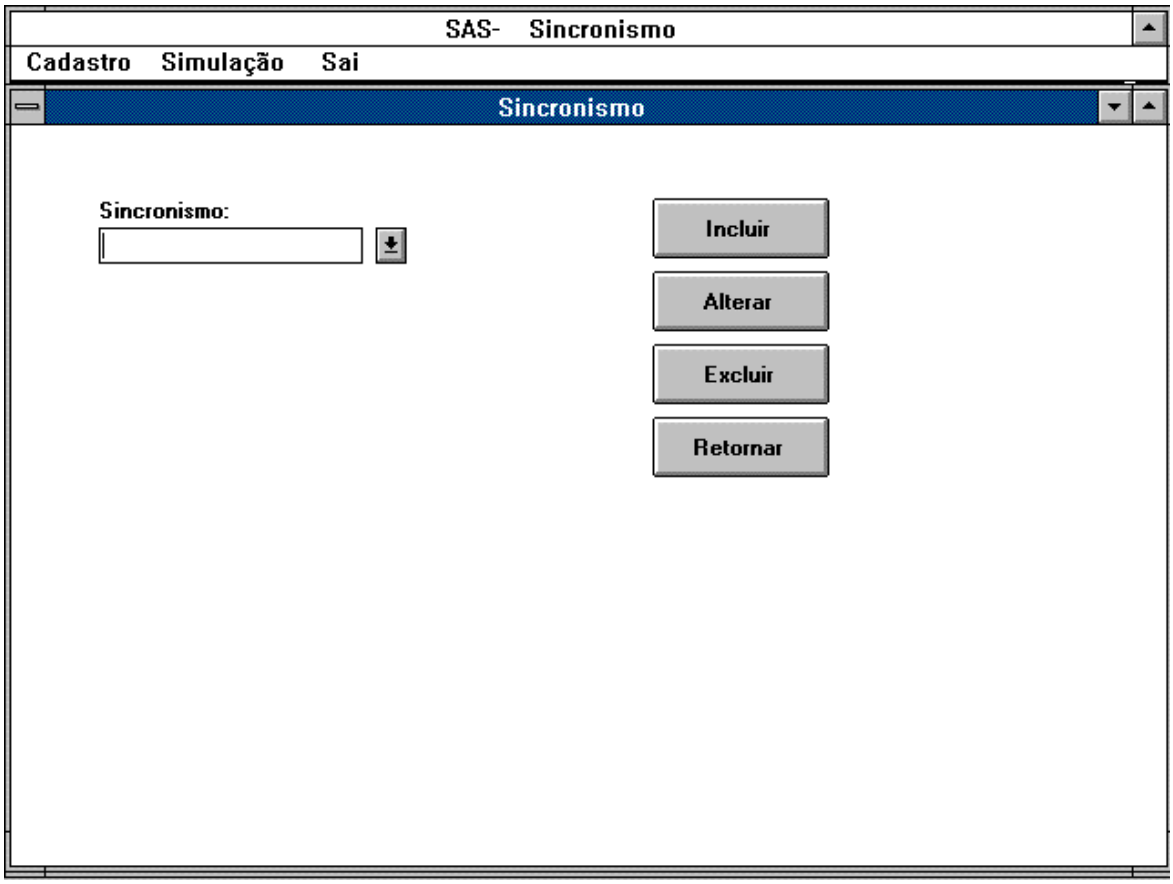


Figura A.6 - Tela com opções do cadastro de Sincronismo

⇒ **Controles da figura A.6**

- **Lista**

  Sincronismo: Essa lista apresenta todos os sincronismos cadastrados, que podem ser selecionados para alteração ou exclusão.

- **Botão:**

Incluir: Esse botão apresenta a tela representada na fig A.7 para que sejam escolhidos todos os Statecharts/Bolhas que compõe o sincronismo

Alterar: Apresenta uma tela idêntica à representada na figura A.7. A única diferença é que o botão *confirma* irá atualizar os dados do sincronismo, enquanto que na inclusão ele cria um sincronismo novo.

Excluir: Esse botão exclui o sincronismo selecionado na lista.

Retorna: Esse botão quando selecionado retorna a tela principal do módulo de sincronismo, representado pela figura A.5

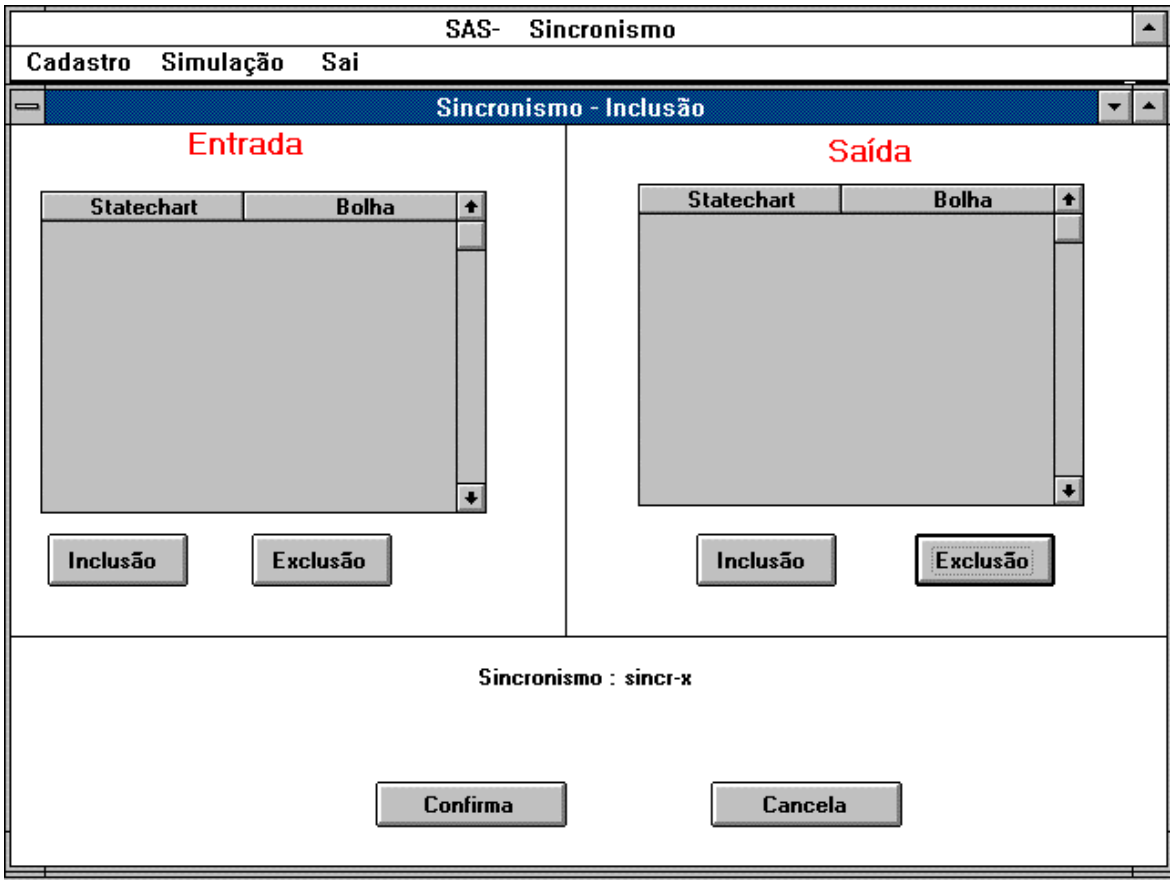


Figura A.7 - Tela de inclusão/Alteração de sincronismo

⇒ **Controles da figura A.7**

- **Botão**:

  Entrada-Inclusão: apresenta uma tela representada na figura A.8 contendo uma lista dos Statecharts cadastrados, para a escolha através do mouse.

  Entrada-Exclusão:Exclui o Statechart selecionado pela barra, da lista de entrada.

  Saída-Inclusão: apresenta uma tela representado na figura A.8. contendo uma lista dos Statecharts cadastrados, para a escolha através do mouse.

Saída-Exclusão: Exclui o Statechart selecionado pela barra da lista de saída.

Confirma: Efetua a criação (quando for inclusão de sincronismo) ou modificação (quando for alteração de sincronismo) com os Statecharts/Bolhas apresentados na lista de entrada e de saída.

Cancela: Cancela a inclusão (ou alteração), voltando à tela com as opções de cadastro, representado na figura A.6.

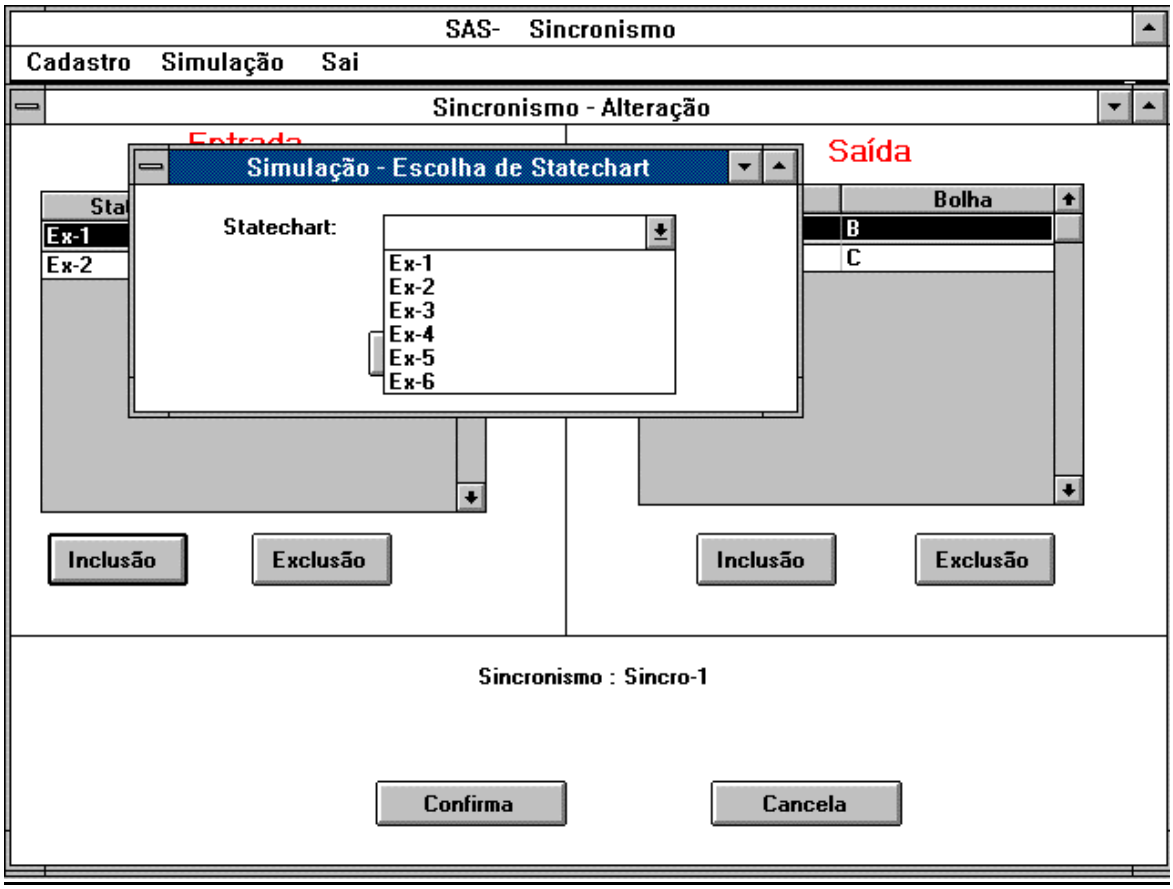


Figura A.8- Escolha do Statechart para inclusão na Lista de Entrada ou Saída

⇒ **Controles da figura A.8**

- **Lista:**

  Statechart: Essa lista apresenta todos os Statecharts cadastrados, para escolha. Quando for selecionado um Statechart (através do *mouse*) será apresentada a tela com o seu diagrama, para prosseguimento do cadastro (escolha de cadastro).

- **Botão**

  Volta: Esse botão retorna à tela de inclusão/alteração de sincronismo, representado na tela A.7, sem efetuar a inclusão do Statechart.

Após a seleção de um Statechart, é apresentado o seu diagrama para que se escolha qual a bolha que irá fazer parte do sincronismo. Essa escolha é feita pressionando o botão direito do mouse quando sua posição indicativa na tela estiver sobre a bolha desejada. Após a escolha, a bolha fica identificada com um traço no canto superior direito. Quando for uma bolha de entrada o traço é identificado com a cor vermelha, e quando de saída com a cor azul. Na figura A.9 temos um exemplo de uma seleção de bolha do Statechart.

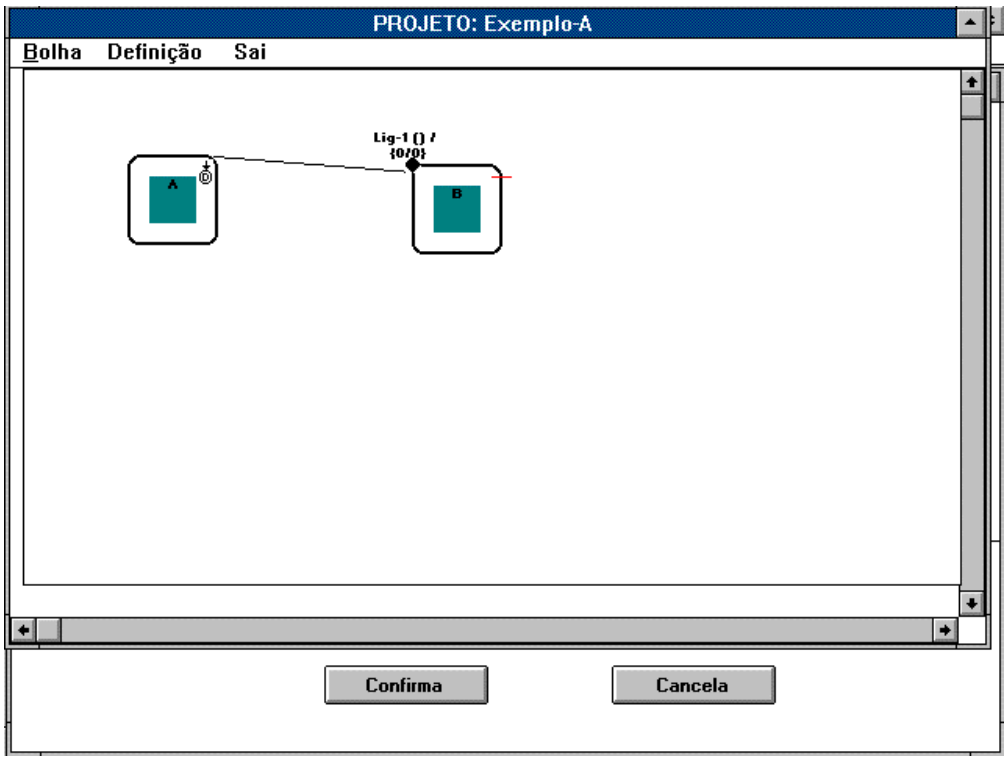


Figura A.9 - Statechart com uma bolha selecionada para o sincronismo

Após a escolha de um Statechart/Bolha o sistema volta à tela representada na figura A.7, para que se repita o processo de inclusão de Statecharts/Bolha para todas as bolhas envolvidas no sincronismo. A cada inclusão de Statechart/Bolha o sistema cria uma linha na lista identificando-o, como mostra a tela representada na figura A.10.

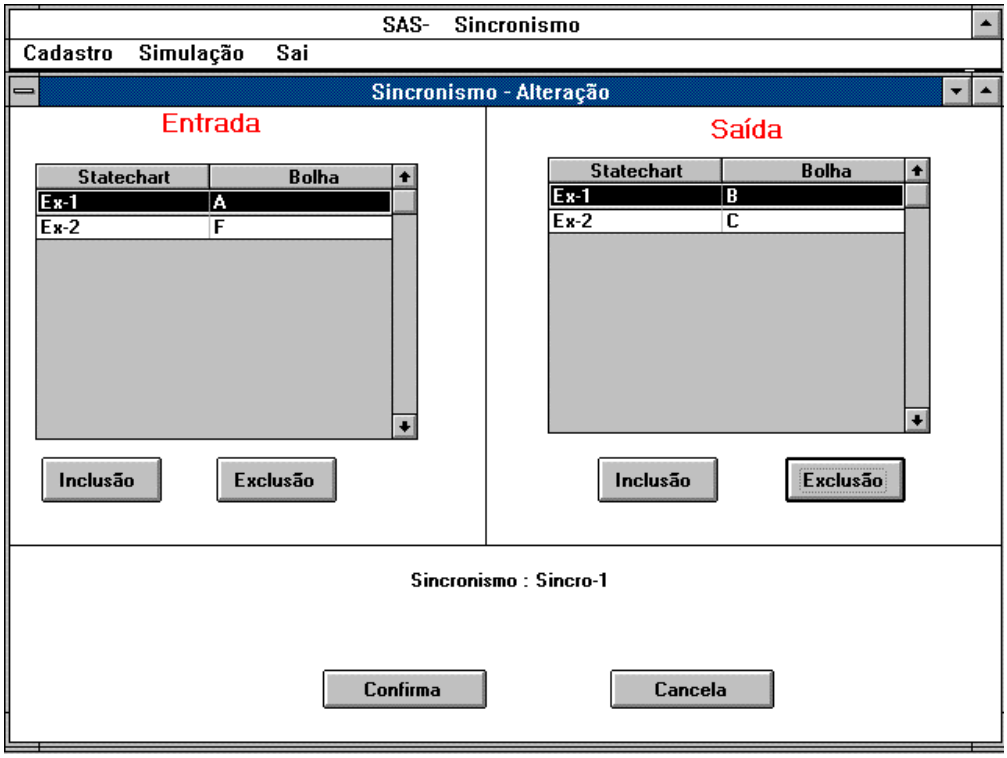


Figura A.10 - Exemplo de sincronismo com listas preenchidas.

A.II.II MÓDULO SINCRONISMO-SIMULAÇÃO

Esse módulo permite que se simule um conjunto de Statecharts e seus sincronismos. Para se realizar a simulação deve-se escolher um conjunto de Statecharts que serão simulados. Para isso o sistema apresenta a tela representada na figura A.11.

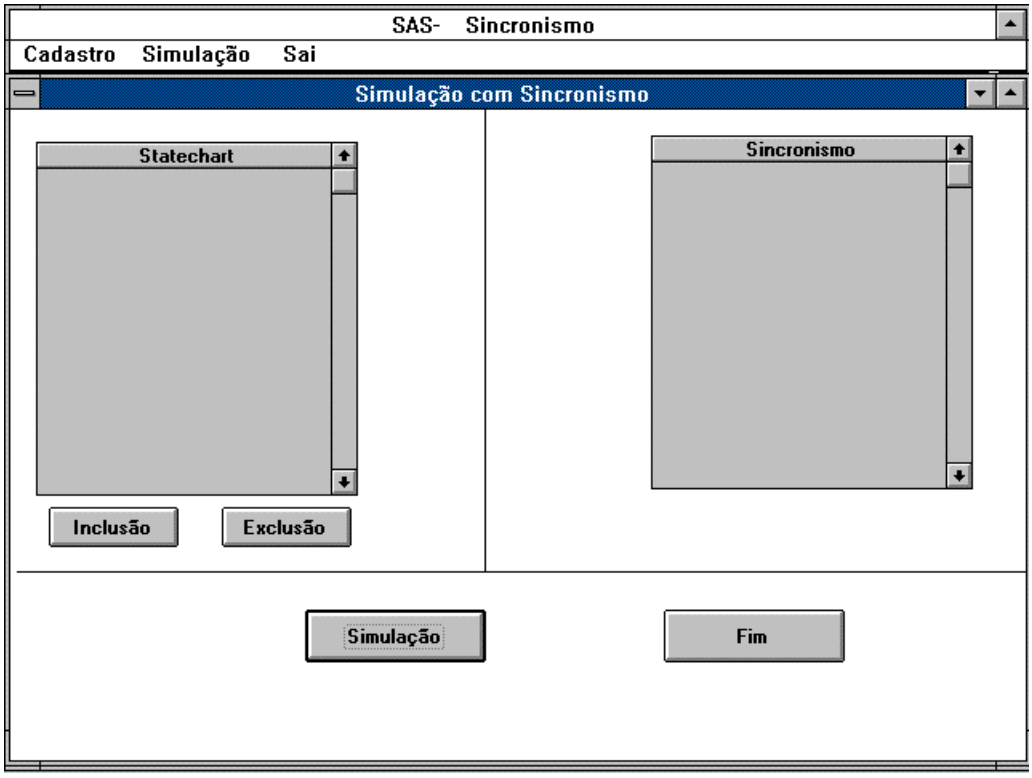


Figura A.11 - Tela com lista de Statecharts que serão simulados

⇒ **Controles da figura A.11**

- **Botão**

  Inclusão: Essa opção apresenta uma tela com a lista dos Statecharts cadastrados para escolha, representada na figura A.12.

  Exclusão: Essa opção exclui da lista o Statechart selecionado pela barra de seleção.

  Simulação: Inicia a simulação dos Statecharts selecionados.

  Fim: Retorna ao menu principal de sincronismo, representada na figura A.5.

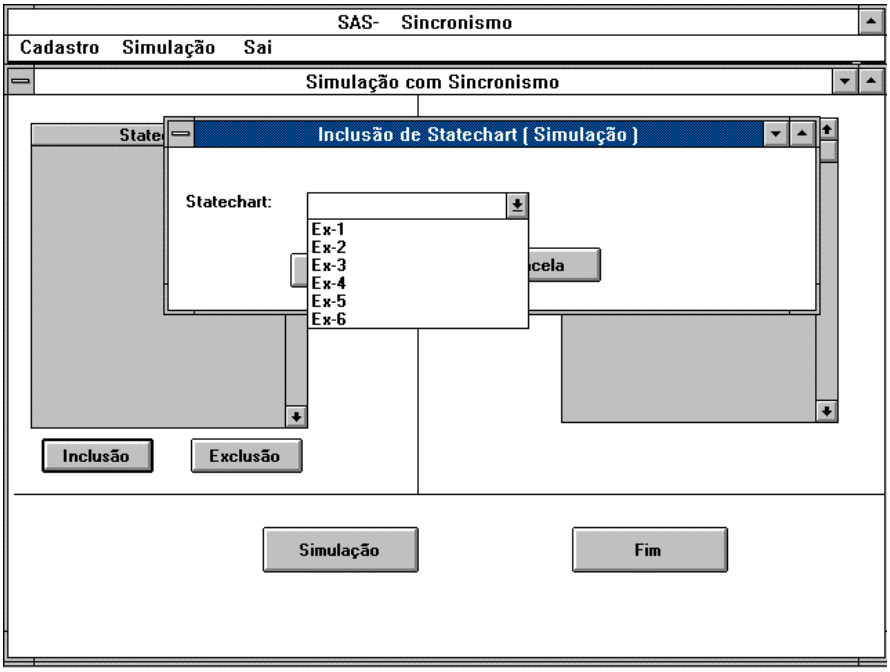


Figura A.12 - Seleção de Statecharts para simulação

Após serem feitas todas as escolhas dos Statecharts, deve-se pressionar o botão *Simulação* para início do processo de simulação. O sistema automaticamente procura todos os sincronismos cujos todos Statecharts que o formam estejam selecionados na simulação.

A tela de simulação é composta de um diagrama e um conjunto de botões como mostra a figura A.13.

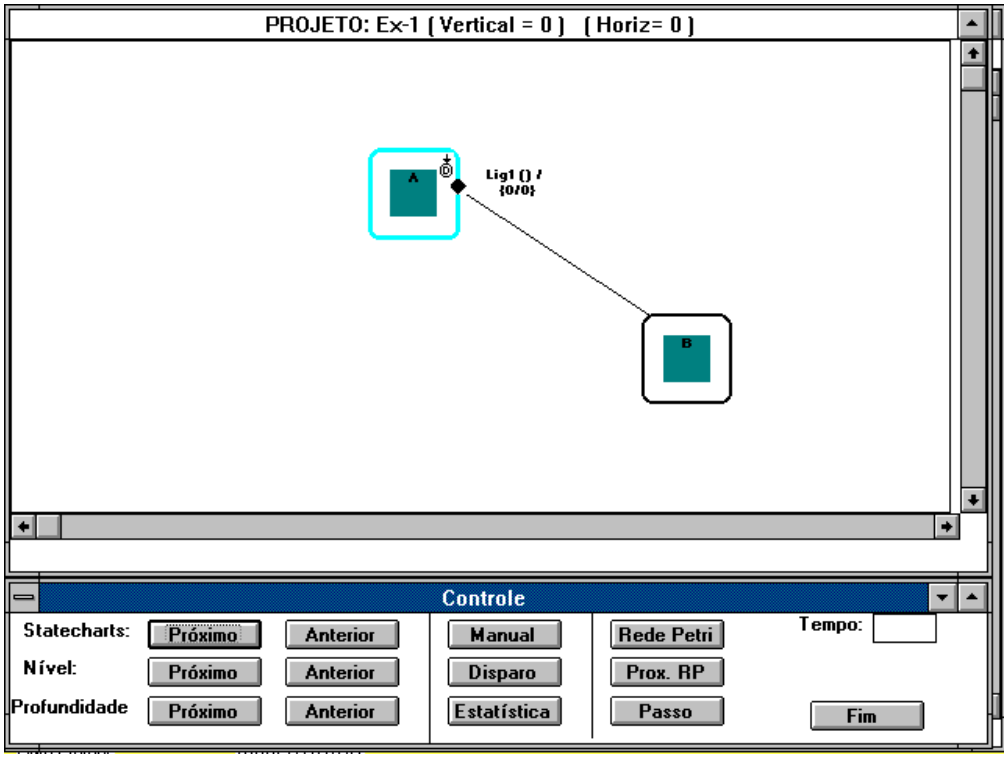


Figura A.13 - Exemplo ilustrando tela de simulação

⇒ **Controles da figura A.8**

- **Botão**

  Statecharts -Próximo: Apresenta o diagrama de outro Statechart da simulação (próximo).

  Statecharts-Anterior: Apresenta o diagrama de outro Statechart da simulação (anterior)

  Nível -Próximo: Quando um Statechart possuir um detalhamento, e a bolha pai (ancestral) for ativa, essa opção permite que se visualize o diagrama do detalhamento.

  Nível Anterior: Idem à opção Nível-Próximo, com a diferença que a ordem em que são mostradas os diagramas é inversa.

  Profundidade-Próximo: Quando uma bolha ativa do tipo AND possuir detalhamento em várias (pelo menos mais de uma) bolhas, essa opção permite que se visualize qualquer um dos diagramas das bolhas filhas. Convém notar que se uma bolha do tipo AND estiver ativa, todas as sub-bolhas filhas estão ativas.

  Profundidade-Anterior: Idem à opção Profundidade-Próximo, com a diferença que a ordem em que são mostradas os diagramas é inversa.

  Manual: Esse botão identifica o modo de simulação. Quando seu conteúdo

contém *"Manual"* (como no exemplo da figura A.13) indica que a simulação necessita de um comando manual para que se execute as transições disponíveis. Quando esse botão é pressionado seu conteúdo passa a ser *"Automático"*, indicando que as transições são disparadas a cada intervalo de tempo automaticamente. Quando pressionado, o botão alterna o conteúdo entre "*Manual*" e "*Automático*".

Disparo: Esse opção apresenta uma lista contendo as transições definidas como externa, podendo ser acionadas, simulando assim a interação com o meio externo.

Estatística: Apresenta uma tabela com as estatísticas dos sincronismos, conforma mostra a figura A.14

Rede Petri: essa opção visualiza o processo de sincronismo (Rede de Petri), conforme mostra a figura A.15

Prox RP: Essa opção permite que se visualize outro sincronismo (Rede de Petri) que esteja sendo simulado.

Passo: quando estiver sendo visualizado a opção Manual, esta opção dispara alternadamente as bolhas e as transições (que satisfazem as suas condições)

Fim: Essa opção encerra a simulação, voltando à tela inicial de sincronismo representada na figura A.5.

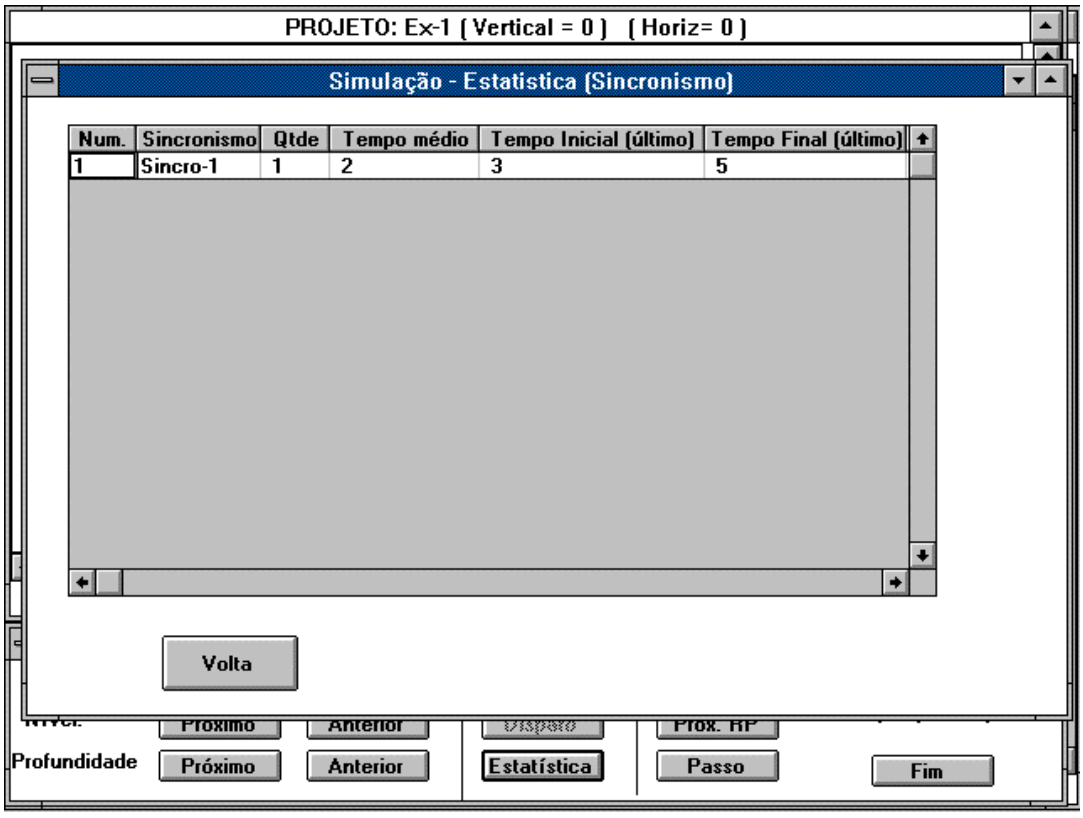


Figura A.14 - Exemplo da tabela de Estatística da simulação dos sincronismos

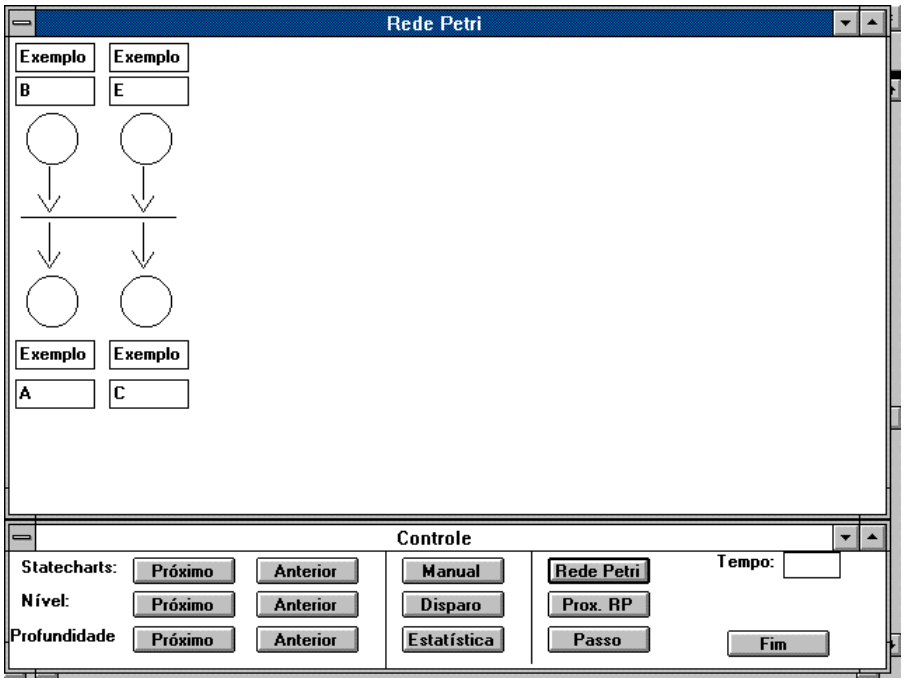


Figura A.15 - Exemplo ilustrando a visualização de um sincronismo na simulação

Quando uma bolha do statechart estiver ativa e ela for uma bolha de entrada do sincronismo, ela aparece na figura A.15 com a cor azul, identificando assim na tela de sincronismo que o recurso (simbilizado pela bolha de entrada) já está disponivel. Assim a cada instante de simulação pode-se acompanhar quais bolhas do sincronismo estão ativas.

ANEXO B - ESTRUTURA DO BANCO DE DADOS DO SAS

Este anexo apresentará uma descrição das estruturas (Tabelas) utilizadas para armazenamento das informações dos Statecharts e Sincronismos do ***SAS.***

B.1.- Campos da Tabela Bolhas

Essa tabela armazena informações sobre as bolhas dos Statecharts.

Projeto: Tipo texto, com 20 caracteres, para armazenar o nome do Projeto (Statechart)

Nível: Tipo numérico, inteiro, indica o nível do detalhamento do Statechart. O primeiro diagrama (principal) tem nível 0, enquanto que cada detalhamento tem um nível maior que o nível do pai.

Pai: Tipo texto, com 20 caracteres. Armazena o nome da bolha Pai quando for um detalhamento.

Número: Tipo numérico, inteiro. Utilizado internamente pelo sistema para identificar qual a bolha que está sendo tratada.

Filha: Tipo texto, com 20 caracteres. Indica o nome da bolha que é um detalhamento da bolha cadastrada.

Cima: Tipo numérico Inteiro. Indica a abcissa do canto superior esquerdo da bolha, em relação ao canto superior esquerdo da janela de edição.

Esquerda: Tipo numérico Inteiro. Indica a ordenada do canto superior esquerdo da bolha em relação ao canto superior esquerdo da janela de edição.

Largura: tipo numérico inteiro. Indica a largura da Bolha.

Altura: Tipo numérico Inteiro. Indica a altura da bolha

Tipo: Tipo texto, de 1 caractere. Indica o tipo da bolha (“A” = AND, “D”= Default, “H”= History ou “C”= Comum)

Estado: tipo texto, com 10 caracteres. Indica o estado em que se encontra a bolha durante a execução (“A”- Ativo ou “I”- Inativo).

History: Tipo numérico inteiro. Quando conter 1 na simulação, indica que o estado estava ativo quando a sua bolha Pai foi desativada; caso contrário indica que o estado não estava ativo.

- **Executado**: Tipo texto, 5 caracteres. Só é utilizado no caso da bolha pertencer a uma Ação Adaptativa. Se contém “SIM” indica que a bolha já foi movimentada (incluída ou excluída) pela Ação Adaptativa.

Adaptativo: Tipo texto, com 10 caracteres. Se tiver o conteúdo “SIM” indica que a bolha faz parte de uma Função Adaptativa.

Metanome: Tipo texto, com 20 caracteres. Contém o metanome da bolha, usado quando o seu nome é alterado por alguma Ação Adaptativa.

Nome_Função: Tipo texo, com 20 caracteres. Se a bolha for montada em uma ação adaptativa indica o nome desta Função Adaptativa, caso contrário o número da ligação que está acionando a Função.

Param1: tipo texto, com 20 caracteres. Indica um parâmetro utilizado pelas bolhas primitivas.

Param2: Tipo texto, 20 caracteres. Indica um segundo parâmetro, utilizado pelas bolhas primitivas.

Nome_Aux: Tipo texto, com 20 caracteres. Quando uma função adaptativa acionar outra função adptativa, indica o nome da função acionada.

B.2 - Campos da Tabela de Ligações

Projeto: tipo texto, com 20 caracteres. Indica o nome do Projeto.

Nível: Tipo numérico, inteiro, indica o nível do detalhamento do Statechart. O primeiro diagrama (principal) tem nível 0, enquanto que cada detalhamento tem um nível maior que o nível do seu pai (ancestral).

Pai: Tipo texto, com 20 caracteres. Armazena o nome da bolha Pai quando for um detalhamento.

Primeiro: Tipo texto, com 1 caractere. Quando o conteúdo for igual a “P” indica que o segmento é o primeiro segmento da ligação.

Último: Tipo texto, 1 caractere. Quando o conteúdo for igual a “U” indica que o segmento é o último segmento da ligação.

Origem: Tipo numérico inteiro. Contém o número de referência da Bolha de origem da ligação.

Destino: tipo numérico inteiro. Contém o número de referência da Bolha de destino da ligação.

Num: Tipo numérico inteiro. Contém o número de referência da ligação.

X1: Tipo numérico inteiro. Indica a abcissa da origem do segmento.

Y1: Tipo numérico inteiro. Indica a ordenada da origem do segmento.

X2: Tipo numérico inteiro. Indica a abcissa do término do segmento.

Y2: Tipo numérico inteiro. Indica a ordenada da término do segmento.

Tipo: tipo texto, com 3 caracteres. Indica qual o tipo da ligação. Se o conteúdo for “I” indica que é Interna, “EE” indica externa de entrada, “ES” indica externa de saída, “II” indica Interna Independente, “EEI” indica externa de entrada independente e “ESI” indica externa de saída independente.

Condição: Tipo texto, com 20 caracteres. Indica o nome da bolha (condição) que deverá estar ativa para que a ligação ocorra.

Disparo: Tipo Tipo texto com 20 caracteres. Indica o nome do evento que deverá ser disparado pela transição em questão.

Posição: tipo texto, 1 caractere.

Cima: Tipo numérico inteiro. Indica a abcissa da posição da tela em que deve ser escrita as informação da ligação.

Esquerda: tipo numérico inteiro. Indica a ordenada da posição da tela em que deve ser escrita a informação da ligação.

Estado: Tipo texto, com 10 caracteres. Se o conteúdo for “ATIVO” indica que a bolha está ativa. Este campo é utilizado durante a simulação.

Nome_Origem: Tipo texto, com 20 caracteres. Contém o nome da bolha de origem da ligação.

Nome-Destino: Tipo texto, com 20 caracteres. Contém o nome da bolha de destino da ligação.

Executado: Tipo texto, com 5 caracteres. No caso de bolha de ações adaptativas quando o conteúdo for “SIM” indica que a ligação já foi inserida ou excluída no Statechart.

Adaptativo: Tipo texto, com 10 caracteres. Se tiver o conteúdo “SIM” indica que a bolha faz parte de uma Função Adaptativa.

Atraso: Tipo numérico inteiro. Indica o atraso que deve ocorrer para que a transição ative a bolha destino.

T_Disparo: tipo numérico inteiro. Contém o tempo mínimo absoluto (em relação ao início da simulação) para que a bolha de destino seja ativada.

T_restante: Tipo numérico inteiro. Utilizado internamente pelo sistema para calcular o tempo que resta para que se ative a bolha de destino, considerando os campos de atraso e disparo.

Metanome: Tipo texto, com 20 caracteres. Contém o metanome do evento, usado quando o seu nome é alterado por alguma Ação Adaptativa.

Nome_Função: Tipo texo, com 20 caracteres. Se a bolha for montada em uma ação adaptativa indica o nome desta Função Adaptativa, caso contrário o número da ligação que está acionando a Função.

Metanome: Tipo texto,com 20 caracteres. Indica o metanome associado à ligação, utilizado quando o nome da ligação é alterado por alguma ação adaptativa.

Função_Usada: Tipo texto, com 20 caracteres, campo auxiliar contendo o nome da funcão Adaptativa.

Execução: Tipo texto, com 10 caracteres. Quando o conteúdo for "*Anterior*" indica que a execução da ação Adaptativa deverá ser realizada antes da execução da ligação, e se o conteúdo for "*Posterior*" indica que a Ação Adaptativa deverá ser realizada depois da execução da ligação.

Valor: Tipo texto, com 20 caracteres. Contém o valor assumido pela ligação. Campo utilizado durante a simulação.

Nome_aux: Tipo texto, com 20 caracteres. Esse campo é utilizado quando uma ligação pertence a Função Adaptativa e também aciona outra função adaptativa; servindo para armazenar o nome desta última ação adaptativa.

B.3 - Campos da Tabela Setas

Projeto: Tipo texto, com 20 caracteres. Indica o nome do Projeto

Nível: Tipo numérico inteiro. Indica qual o nível de detalhamento onde se encontra a ligação.

Pai: Tipo texto, com 20 caracteres. Armazena o nome da bolha Pai quando for um detalhamento.

Número: Tipo numérico inteiro. Contém um número de referência indicando a ligação associada.

Cima: Tipo numérico inteiro. Indica a abcissa da posição da tela em que deve ser escrita as informação da seta.

Esquerda: tipo numérico inteiro. Indica a ordenada da posição da tela em que deve ser escrita a informação da seta.

Adaptativo: Tipo texto, com 10 caracteres. Se tiver o conteúdo "SIM" indica que a seta faz parte de uma Função Adaptativa.

Executado: Tipo texto, com 5 caracteres. No caso de bolha de ações adaptativas quando o conteúdo for "SIM" indica que a seta já foi inserida ou excluída no Statechart.

Nome_Função: Tipo texo, com 20 caracteres. Se a ligação associada à seta for montada em uma ação adaptativa indica o nome desta Função Adaptativa, caso contrário o número da ligação que está acionando a Função.

Metanome: Tipo texto, com 20 caracteres. Indica o metanome associado à ligação (da qual a seta faz parte), utilizado quando o nome da ligação é alterado por alguma ação adaptativa.

Nome_aux: Tipo texto, com 20 caracteres. Esse campo é utilizado quando uma ligação pertence a Função Adaptativa e também aciona outra função adaptativa; servindo para armazenar o nome desta última ação adaptativa.

B.4 - Campos da Tabela Adaptativa.

Projeto: Tipo texto, com 20 caracteres. Indica o nome do Projeto

Nível: Tipo numérico inteiro. Indica qual o nível de detalhamento onde se encontra a função adaptativa.

Pai: Tipo texto, com 20 caracteres. Armazena o nome da bolha Pai quando for um detalhamento.

Nome: Tipo texto, com 20 caracteres. Armazena o nome da função Adpatativa.

Nome_aux: Tipo texto, com 20 caracteres. Esse campo é utilizado quando uma ligação pertence a Função Adaptativa e também aciona outra função adaptativa; servindo então para armazenar o nome desta última ação adaptativa.

Nome_Geral: tipo texto, com 20 caracteres. Indica o nome da Função Adaptativa que estiver sendo detalhada.

Adaptativo: Tipo texto, com 5 caracteres. Se tiver o conteúdo “SIM” indica que a seta faz parte de uma Função Adaptativa.

Tipo: Tipo texto, com 5 caracteres. Quando o seu conteúdo for “+” indica que é uma ação de inclusão, e se for “-” indica que é uma ação de exclusão.

Elemento: Tipo texto, com 10 caracteres. Indica o nome que está sendo tratado pela função adaptativa (nome da bolha ou da ligação).

Meta_Nomevento: Tipo texto, com 20 caracteres. Indica o metanome do elemento (bolha ou ligação) que está sendo referenciado pela Função Adaptativa.

Origem: Tipo texto, com 20 caracteres. Quando estiver sendo tratado na Função Adaptativa uma ligação indica o nome da sua bolha de origem.

- **Destino**: Tipo texto, com 20 caracteres. Quando estiver sendo tratado na Função

Adaptativa uma ligação indica o nome da sua bolha de destino

Num: Tipo numérico inteiro. Indica o número da Ligação.

Ligação: Tipo numérico inteiro. Indica o número da ligação.

Execução: Tipo texto, com 10 caracteres. Quando o conteúdo for "*Anterior*" indica que a execução da ação Adaptativa deverá ser realizada antes da execução da ligação, e se o conteúdo for "*Posterior*" indica que a Ação Adaptativa deverá ser realizada depois da execução da ligação.

P1: Tipo texto, com 5 caracteres. Indica se o nome da bolha ou ligação da função adaptativa pode ser alterado quando executado em uma ação adaptativa.

P2: Tipo texto, com 5 caracteres. Indica se a origem da ligação definida pela função Adaptativa pode ser alterada quando executado em uma ação adaptativa.

P3: Tipo texto, com 5 caracteres. Indica se o destino da ligação definida pela função Adaptativa pode ser alterada quando executado em uma ação adaptativa.

B.5 - Campos da Tabela Sincronismo

Projeto: Indica o nome do projeto ao qual pertence o sincronismo

Nome_Sincr: Indica o nome do sincronismo

- **Estado**:
- **Status**:

B.6 Campos da Tabela Sincr_Bolhas

Essa tabela contem as informações das bolhas (lugares) dos sincronismos.

Nome_Sincr: Tipo texto, com 20 caracteres. Indica o nome do sincronismo

Statechart: Tipo texto, com 20 caracteres. Indica o nome do Statechart da qual a bolha pertence.

Bolha: Tipo texto, com 20 caracteres. Indica o nome da bolha.

Tipo: Tipo texto, com 1 caractere. Quando seu conteúdo for "E" indica que a bolha é uma bolha de entrada do sincronismo, e quando for "S" indica que é uma bolha de saída do sincronismo.

Status: Tipo texto, com 10 caracteres. Quando seu conteúdo for "ATIVO" indica que a bolha (só é utilizada para bolha de entrada) está ativa.

- **Número_Proc**: Tipo numérico inteiro.

REFERÊNCIAS BIBLIOGRÁFICAS

REFERÊNCIAS BIBLIOGRÁFICAS

HAREL, D. Statecharts: A visual formalism for Complex Systems **Science of Computer Programming**, v.8, n.3, p.231-74, Aug. 1987

HUIZING, C.; ROEVER W. P. Introducing to Design choices in the semantics of Statechart. **Information Processing Letters**, v.37, p 205-13 , 1991.

COLEMAN, D.; BEAR, S. Introducing Objectchart or How to Use Statecharts in Object-Oriented Design. **IEEE Transactions on Software Enginnering,** v.18, n.1, january 1992

MASIERO, P.C.; MEIRA, C.A.A. Development and Instantation of a generic Application Generator. **Journal Systems Software**, 1993

HAREL, D.; ET AL. Statemate: A Working Enviroment For The Development Of Complex Reactive Systems **IEEE Transactions os Software Engineering**, v.16, n.4, 1990.

WALTERS, N. Using Harel Statechart to Model Object-Oriented Behavior **Software Engineering Notes** v.17, n.4 Oct 1992

Statechart Based Requirements Analysis: Deriving User Oriented Model
Microprocessing and Microprogramming 30 (1990)

Timo Jokela, Kai Lindberg

A Comparision Of Techniques For The Specification Of External System Behavior
Communications of the ACM, 31 (1988)
Alan M. Davis

Relation Grammars For Modelling Multi-Dimensional Structure
C. Crimi, A. Guercio, G. Nota, G. Pacini, G. Tortora, M. Tucci

Stochastic Statecharts And Rapid Prototype Software Architecture
Dennis B. Mulcare

Embedded Behavior Pattern Languages: A Contribution To A Taxonomy Of Case Languages
The Journal of Systems and Software, 9 (1989)
Paul T. Ward

An Integrated Framework For Software Prototyping
Jijun Cvhen
Jeremy Kuo

Petri Nets
Computing Surveys, 9 (1977)
James L. Peterson

A Comparison of Techniques for the Specification of External System Behavior
Computing Practices, 9 (1988)
Alan M. Davis

Modelling Statechart Behavior in a fully abstract way
C. Huizing, R. Gerth, W.P. de Roever

Propriedades Dinâmicas de Statechar
Dissertação de Mestrado USP - São Carlos (1992)
Inês Aparecida Gasparotto Boaventura

Um Editor Gráfico para Statechart
Dissertação de Mestrado USP - São Carlos (1991)
João do E. S. Batista Neto

Contribuição à metodologia de Construções de Compiladores
Tese de Livre-Docência USP (1993)
João José Neto

STAD - Uma Ferramenta para Representação e Simulação de Sistemas Através de Statecharts Adaptativos
Tese de Doutorado - USP (1995)
Jorge Rady de Almeida Junior

Lewis, H.R. e Papadimitrious, C.H.
Elements of the Theory of Computation
Printice hall - 1981

Casey, B. e Taylor, B.
Proceedings of the IEEE Software Engineering Standard
Workshop, IEEE Press. Wash., 1981