REPRESENTAÇÃO GRÁFICA PARA LOTOS

Maria Teresa Silva de Moura
Paulo Roberto Freire Cunha

Departamento de Informática
Universidade Federal de Pernambuco
50739, Recife, PE

RESUMO

Este trabalho apresenta uma representação gráfica para LOTOS (Language of Temporal Ordering Specification), uma técnica de descrição formal para a especificação de sistemas distribuídos. A principal característica desta representação gráfica é sua simplicidade, onde os principais aspectos da especificação são facilmente visualizados, facilitando o desenvolvimento e aumentando a clareza, uma característica importante quando se trata de especificações formais.

ABSTRACT

This work presents a graphical representation to LOTOS (Language of Temporal Ordering Specification), a formal description technique for the specification of distributed systems. The main feature of this graphical representation is its simplicity, where the principal aspects of the specification are easily visualized, facilitating the development and enhancing the clarity, an important characteristic in the area of formal specification.

1. INTRODUÇÃO

Com o surgimento de sistemas distribuídos várias técnicas para descrição destes sistemas vêm sendo criadas e a principal preocupação é prover descrições de forma clara, precisa e sem ambiguidades. Neste contexto, a linguagem LOTOS [ISO 88] é uma técnica de descrição formal para a especificação de sistemas

distribuídos, em particular protocolos e serviços definidos pelo Modelo OSI da ISO, desenvolvida por especialistas em técnicas de descrição formal sob a coordenação da ISO a partir do ano de 1981.

Em LOTOS, um sistema é descrito através de um processo que pode se constituir de vários outros de modo a formar uma hierarquia de processos que se comunicam entre si e com o ambiente. Estes processos podem ser vistos por um observador externo como uma caixa preta que se comunica com o ambiente através dos pontos de interação, de forma que o sistema é descrito por uma relação temporal entre os eventos, os quais representam a interação entre processos e o ambiente ([BoBri 87], [Souza 87]). Uma especificação LOTOS possui dois componentes:

- um componente estático para a descrição dos tipos de dados manipulados na especificação que é baseado na linguagem para especificação de tipos abstratos de dados ACT ONE [EhMha 85]. Este componente estático permite a produção de especificações estruturadas de tipos abstratos de dados na medida que permite a utilização de uma biblioteca de tipos de dados predefinidos, extensões e combinações de definições de tipos, parametrização de definições e atualizações de definições parametrizadas, e renomeação de definições.

- um componente dinâmico para a descrição do comportamento observável do processo através de suas interações e operações. Este componente é uma extensão de CCS (Calculus of Communicating Systems) [Milner 80]. Este componente dinâmico permite a representação do comportamento observável da especificação na medida que define os construtores indispensáveis à descrição do comportamento dos processos e da interação entre eles.

Neste trabalho, apresentaremos uma representação gráfica para LOTOS com a finalidade de facilitar o desenvolvimento e entedimento de especificações de sistemas distribuídos, devido ao fato de que a utilização do ferramental gráfico oferece uma interface humana mais acessível. O restante do trabalho está estruturado como segue: na seção 2 discutiremos a utilização de ferramentas gráficas dentro do processo de especificação de sistemas distribuídos; na seção 3 apresentaremos a nossa

representação gráfica para os principais construtores de LOTOS, juntamente com a sintaxe informal destes construtores; na seção 4 mostraremos a utilização da representação gráfica proposta através do desenvolvimento de uma especificação em LOTOS evidenciando através deste exemplo nossa preocupação em utilizar LOTOS para desenvolver especificações estruturadas de sistemas distribuídos de uma maneira geral, já que esta tem sido utilizada principalmente para a especificação de protocolos e serviços do Modelo OSI; e finalmente, na seção 5, concluiremos o trabalho onde procuraremos destacar a importância da representação gráfica e situar nossos futuros trabalhos nesta área.

2. UTILIZAÇÃO DE FERRAMENTAS GRÁFICAS

A utilização de representações gráficas dentro do processo de especificação de sistemas distribuídos apresenta-se como uma ferramenta poderosa visto que atribui clareza às descrições formais. Neste sentido, algumas técnicas de descrição formal, tais como Máquina de Estados Finita, Redes de Petri e SDL oferecem representações gráficas como suporte ao processo de desenvolvimento de especificações ([Peter 81], [Diaz 86]).

É importante ressaltar o fato de que a utilização do ferramental gráfico facilita o entendimento e manutenção de sistemas complexos, bem como o processo de documentação do sistema especificado, visto que através do diagrama podemos visualizar graficamente o sistema como um todo.

Najm e Queiroz em [NajQue 88] definem formalmente GLOTOS, uma linguagem gráfica que tem o propósito de apresentar símbolos para os elementos de LOTOS e combinar os símbolos elementares para formar representações gráficas mais elaboradas ou seja representações gráficas para as expressões de comportamento da linguagem LOTOS. O nosso objetivo é poder modelar, através de uma representação gráfica, a estrutura geral da especificação, facilitando o seu entendimento e validação, tal que esta possibilite a construção de diagramas simples que expressem os aspectos principais da especificação. As informações mais detalhadas, a nosso ver, devem ser descritas apenas na

representação textual com a finalidade de não tornar o gráfico muito extenso e detalhado, dificultando a clareza. Acreditamos que uma boa interação entre a representação gráfica e a representação textual, onde a primeira oferece uma interface humana mais amigável e a segunda apresenta as informações detalhadas, expressando com exatidão estruturas mais complexas, oferece ao mesmo tempo a clareza e precisão necessárias a uma boa técnica de descrição formal.

Segundo [Kramer 89] a utilização de diagramas em conjunto com as descrições textuais é uma prática já utilizada, no entanto estes diagramas eram geralmente gerados manualmente e utilizados informalmente. Neste contexto, ele apresenta um sistema gráfico que integra a informação textual e gráfica para a programação de configuração de sistemas distribuídos e concorrentes em Conic. Delisle e Schwartz em [DelSch 86] apresentam um ambiente interativo para o desenvolvimento e execução de programas em CSP, o qual utiliza representações gráficas para ilustrar as interações dos processos em execução. Neste sentido, acreditamos que a utilização de um ferramental gráfico pode ser embutida em uma interface amigável, com a finalidade de se definir um ambiente de desenvolvimento que auxilie no processo de construção de especificações formais.

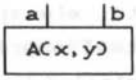
3. REPRESENTACÃO GRÁFICA PARA LOTOS

Conforme dito anteriormente, o componente dinâmico de LOTOS contem os construtores que definem o comportamento da especificação, enquanto que o componente estático apresenta-se como uma ferramenta para a definição de estruturas de dados, sendo baseado numa álgebra para a especificação de tipos abstratos de dados. Para a definição da nossa representação gráfica nos restringiremos ao componente dinâmico, visto que a nossa intenção é modelar, através do gráfico, o comportamento geral da especificação.

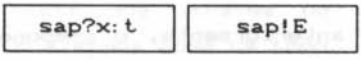
Um processo em LOTOS tem o seguinte formato:

```
process <nome-proc> [<portas>] (<param>) :=
        <comportamento>
endproc
```

onde <nome-proc> indica o nome do processo; <portas> identifica os pontos de sincronização; <param> identifica os parâmetros caso existam; e <comportamento> representa o comportamento do processo. Na representação gráfica, cada processo é descrito por um retângulo que contém no seu interior o nome do processo e os seus parâmetros, caso estes existam. Os pontos de sincronização (eventos) do processo são representados por arcos, os quais podem ser colocados em qualquer um dos quatro lados do retângulo, de forma que o nome do evento é etiquetado próximo ao arco. Desta forma, um processo A com os parâmetros x e y e com os eventos a e b é visualizado graficamente como segue:



A interação entre processos ocorre quando dois processos oferecem o mesmo evento. Uma oferta de eventos se dá através da associção de valores aos pontos de interação, de forma que sap?x:t indica a aceitação de um valor x do tipo t no ponto de interação sap e sap!E indica oferecimento do valor E no ponto de interação sap. Na representação gráfica, uma oferta de evento é representada por um retângulo, sendo que o interior deste retângulo deverá conter o nome do ponto de interação seguido dos atributos (valores ou variáveis), da seginte forma:

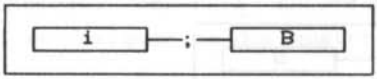


O comportamento de um processo LOTOS é descrito por expressões de comportamento, as quais são obtidas através de interações e operações. A seguir, apresentaremos informalmente os principais operadores de LOTOS, juntamente com a nossa representação gráfica para cada um deles. Neste ponto em que descrevemos as operações que definem o comportamento dos processos é importante ressaltar que caso o comportamento de um dado processo apresente aspectos importantes da especificação, estes deverão ser representados graficamente, de forma que o interior do retângulo que representa este processo deverá conter, além do nome do processo e parâmetros, a representação gráfica de

seu comportamento. Caso contrário, o interior do retângulo conterá apenas o nome do processo e parâmetros, estando o seu comportamento definido apenas na representação textual.

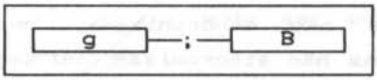
i) ação: indica sequencialidade entre uma oferta de evento e uma expressão de comportamento. A expressão a;B indica que a expressão de comportamento B pode ser executada após a ocorrência do evento a. Esta oferta de evento pode se dar através de um evento observável ou através de um evento interno, da seguinte forma:

- não observável (interno): i;B

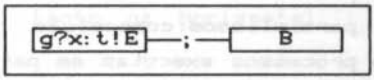


- observável:

a) g;B

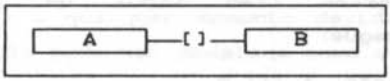


b) g?x:t!E;B



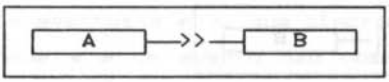
ii) escolha: indica uma escolha não determinística entre expressões de comportamento de acordo com o evento oferecido pelo ambiente. Uma operação de escolha entre os processos A e B é representada da seguinte forma:

A [] B



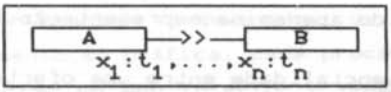
iii) composição sequencial: expressa a sequencialidade entre processos, onde a terminação com sucesso de um processo habilita a execução do processo seguinte. A composição sequencial entre os processos A e B é representada da seguinte forma:

A >> B



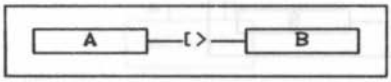
Esta operação pode ser generalizada para permitir a passagem de informações entre os processos, da seguinte forma:

A >> accept $x_1 : t_1, \ldots, x_n : t_n$ in B



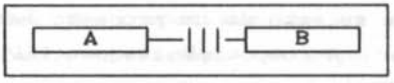
iv) desabilitação: expressa a situação onde um processo pode interromper a execução de outro. Esta desabilitação acontece se o evento inicial do processo desabilitador ocorrer antes da terminação com sucesso do outro processo. Um processo A que pode ser interrompido por um processo B é representado como segue:

A [> B

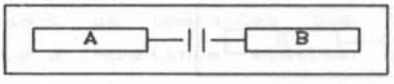


v) composição paralela: expressa a execução concorrente de processos. Em LOTOS existem três tipos de paralelismos: composição paralela não sincronizada, onde os processos executam em paralelo mas não sincronizam, ou seja, os processos não se comunicam entre si; composição paralela com plena sincronização, onde os processos executam em paralelo e sincronizam com relação a todos os eventos em comum; e finalmente a composição paralela geral, onde os processos executam em paralelo e sincronizam com relação a uma lista de eventos para sincronização. Estes três tipos de paralelismos são representados como segue:

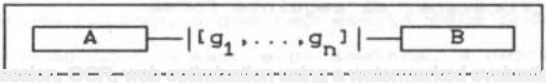
- sem sincronização: A ||| B



- com plena sincronização: A || B

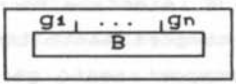


- caso geral: A $|[g_1, \ldots, g_n]|$ B



vi) **hide:** no caso de utilização da composição paralela, pode-se ter a presença de eventos que serão usados apenas para a comunicação interna entre os processos. Estes eventos não devem portanto ser acessíveis pelo ambiente. A situação em que os eventos $g_1,\ldots,g_n$ do processo B são escondidos do observador externo é representada da seguinte forma:

hide $g_1,\ldots,g_n$ **in** B



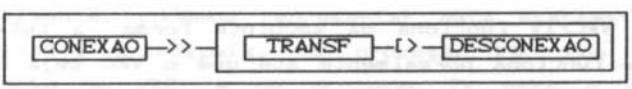
Neste caso, como na representação gráfica dos eventos observáveis de um processo descrita anteriormente, os arcos podem ser colocados em qualquer um dos quatro lados do retângulo.

Uma consideração importante a fazer é o fato de que todas as representações gráficas definidas valem tanto da esquerda para a direita, conforme foram apresentadas, quanto de cima para baixo, ou seja, o diagrama pode ser construído tanto na horizontal quanto na vertical.

A título de ilustração, apresentaremos dois exemplos de expressões de comportamento LOTOS, acompanhadas da representação gráfica:

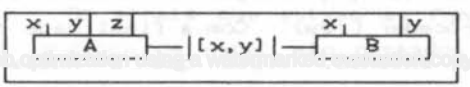
a) Num serviço de transporte a fase de transferência de dados só ocorre após a conexão ter sido estabelecida e a transferência pode ser interrompida a qualquer momento devido a um aborto de conexão. Esta situação pode ser modelada como segue:

CONEXÃO >> (TRANSF [> DESCONEXÃO)



b) Um processo A, com eventos x, y e z, composto em paralelo com um processo B, com eventos x e y, que sincronizam com relação aos eventos x e y, onde x é um evento não observável, pode ser representado como segue:

hide x in (A[x,y,z] |[x,y]| B[x,y])



Com a definição desta representação gráfica estamos desenvolvendo uma interface amigável, com o intuito de simplificar o processo de construção de especificações em LOTOS. A idéia é embutir esta representação gráfica na interface de forma que o usuário vai fornecendo as informações sobre o sistema sendo especificado e a interface apresenta o diagrama, tal que o usuário pode visualizar graficamente as principais características do sistema. A interface fornece também a geração automática de expressões de comportamento LOTOS para o diagrama, de forma que é fornecido o comportamento geral da especificação, restando ao usuário completar a especificação do sistema, através de informações mais detalhadas.

4. EXEMPLO: GRAVADOR DE VÍDEO/TV

Nesta seção, apresentaremos um exemplo de uma especificação para um sistema que controla o funcionamento da interação gravador de vídeo(VRC)/TV. Para tal, utilizaremos a metodologia de desenvolvimento de especificações em LOTOS definida em [MoCun 89], metodologia esta que estende os princípios de estruturação e modularidade da programação sequencial para o ambiente de especificações de sistemas distribuídos e define que a descrição de um sistema seja feita através de uma estrutura hierárquica de módulos independentes, estrutura esta definida através de refinamentos sucessivos. A representação gráfica, dentro desta metodologia, é uma ferramenta poderosa visto que a cada passo do refinamento as informações são representadas graficamente facilitando o entendimento e a documentação do sistema.

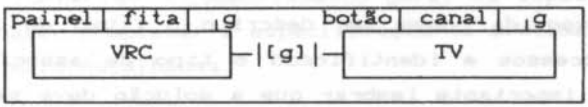
O sistema VRC/TV funciona da seguinte forma: a televisão, quando ligada, funciona normalmente até que o VRC seja ligado. Neste caso, se canal da televisão estiver setado para o canal que recebe os sinais do VRC, o controle passa para o VRC. O VRC pode funcionar como tv, setando o canal para alguma das estações disponíveis, ou como VRC propriamente, permitindo a execução de fitas, sendo que neste caso temos as funções iniciar (*play*), pausa (*pause*), parar (*stop*), imagem em câmara lenta (*slow*), adiantar (*ff*) e retroceder (*rew*). Com a finalidade de não tornar este exemplo muito extenso, visto que a nossa intenção é

demonstrar a utilização da representação gráfica, simplificaremos este sistema assumindo que: i) o canal do VRC não deverá obrigatoriamente estar setado para um canal especial para que a imagem gravada em fita passe para a tv; ii) a função *pause* só poderá ser desativada teclando-se *pause* novamente; iii) a função *play* será automaticamente ativada se uma fita for introduzida no VRC; iv) não consideraremos a função gravar (*rec*) do VRC.

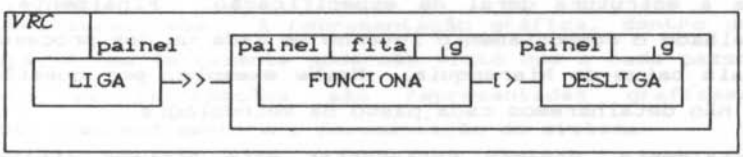
De acordo com a metodologia proposta em [MoCun 89] o desenvolvimento de uma especificação começa com a decomposição do sistema em módulos principais, os quais representam processos LOTOS. Em seguida, deve ser descrito o fluxo de informações entre os processos e identificado o tipo de associação entre eles, onde é importante lembrar que a solução deve ser descrita de modo a atingir um alto grau de paralelismo entre os processos. Todas estas informações devem ser representadas graficamente de forma que é possível efetuar uma validação, através do gráfico, com os requisitos do sistema. Em seguida, deve ser definida uma representação textual para o diagrama estruturado na fase anterior. Depois de definida esta primeira decomposição do sistema, deve ser efetuado, se possível, refinamentos em cada um dos processos definidos anteriormente, tal que os principais aspectos da especificação vão sendo identificados e a hierarquia de processos vai sendo construída sistematicamente. Assim, é definida a estrutura geral da especificação. Finalmente, deve ser detalhado o comportamento interno de cada um dos processos do nível mais baixo da hierarquia. Neste exemplo, por questões de espaço, não detalharemos cada passo da metodologia.

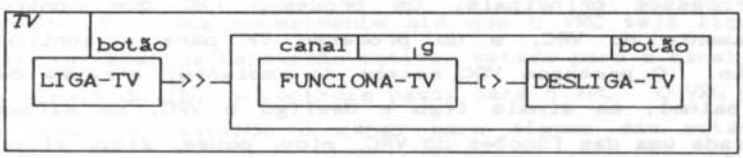
Inicialmente, podemos representar este sistema através de dois processos principais, um processo *VRC* que controla o funcionamento do VRC, e um processo *TV* para o controle da televisão. O processo VRC recebe do ambiente, através de um evento **painel**, os sinais *liga* e *desliga* o VRC, os sinais que ativam cada uma das funções do VRC: *play*, *pause*, *stop*, *slow*, *ff* e *rew*, e um número indicando mudança de canal; e através de um evento **fita** os sinais de entrada e saída da fita, *in* e *out* respectiviamente. Os sinais que ativam as funções do VRC e o número indicando mudança de canal são passados ao processo TV,

este por sua vez recebe do ambiente externo os sinais *liga* e *desliga* a tv através de um evento botão, e um número indicando mudança de canal, através de um evento canal. Os dois processos Vídeo e TV podem ser colocados em paralelo tal que sincronizam através de um evento g para passagem do número para mudança de canal e dos sinais que ativam as funções do VRC. Este evento g deve ser "escondido" do ambiente, visto que será utilizado apenas para comunicação interna entre os dois processos. A seguir, apresentamos a representação gráfica destas informações:



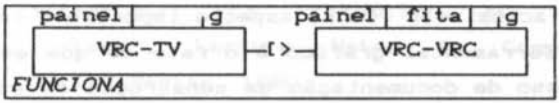
Neste exemplo, não apresentaremos a representação textual para os diagramas, visto que a nossa intenção é mostrar que a representação gráfica modela de forma eficiente o comportamento dos processos. O processo VRC pode ser refinado em três subprocessos, um processo *Liga* para ligar o VRC, um processo *Funciona* que trata o funcionamento do VRC, e um processo *Desliga* que trata o desligamento do VRC. Da mesma forma, o processo TV pode ser decomposto respectivamente em *Liga-TV*, *Funciona-TV* e *Desliga-TV*. Abaixo, apresentamos graficamente estes refinamentos:





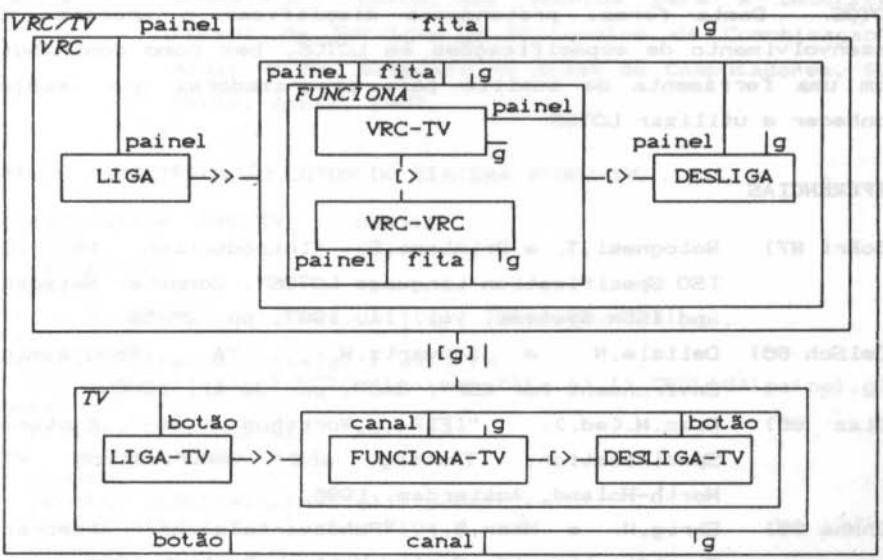
No caso do funcionamento do VRC, o processo Funciona pode ser refinado em dois subprocessos, um processo *VRC-TV* que trata o funcionamento do VRC como uma tv, no qual a única função do VRC é

funcionamento do VRC propriamente dito, que é a execução de fitas. Estes dois processos não trocam informações. O VRC, quando ligado, funciona como tv até que um fita seja introduzida, desta forma, o processo *VRC-VRC* desabilita o processo *VRC-TV*. Abaixo, apresentamos graficamente o processo Funciona:



Para finalizar a especificação do sistema deve ser descrito o comportamento interno dos processos Liga, Desliga, VRC-TV, VRC-VRC, Liga-TV, Funciona-TV e Desliga-TV. Não entraremos em detalhes com relação ao comportamento destes processos, visto que a nossa intenção aqui é apresentar a utilização da representação gráfica. Maiores detalhes quanto ao desenvolvimento da parte sequencial destes processos pode ser encontrado em [MoCun 89]. Em anexo, apresentamos a especificação LOTOS completa deste sistema.

O diagrama completo do sistema Video/TV é apresentado a seguir:



5. CONCLUSÕES

Neste trabalho apresentamos uma representação gráfica para os principais construtores da linguagem LOTOS. De uma maneira geral, a utilização de representações gráficas é uma ferramenta poderosa, visto que facilita o entendimento e a clareza com uma interface mais acessível. Outro aspecto importante com relação à utilização do ferramental gráfico é o fato de que esta facilita também o processo de documentação da construção da especificação do sistema funcionando como apoio a futuras modificações no mesmo. A importância desta representação gráfica está no fato de que esta é uma ferramenta simples embutida numa metodologia de desenvolvimento que permite a representação dos principais aspectos da especificação de forma estruturada; o que pudemos constatar com o desenvolvimento da especificação do sistema VRC/TV aqui apresentado.

Esta representação gráfica está sendo modelada em uma interface gráfica de forma que através do diagrama, o qual representa a estrutura geral da especificação, a interface fornece a geração automática de expressões de comportamento LOTOS. Desta forma, pretendemos simplificar o processo de desenvolvimento de especificações em LOTOS, bem como contribuir com uma ferramenta de auxílio para programadores que desejem conhecer e utilizar LOTOS.

REFERÊNCIAS

[BoBri 87] Bolognesi,T. e Brinksma,E.: "Introduction to the ISO Specification Language LOTOS", **Computer Networks and ISDN Systems**, vol. 14, 1987, pp. 25-59.

[DelSch 86] Delisle,N. e Schwartz,M.: "A Programming Environment for CSP", **CACM**, pp. 34-41, 1986.

[Diaz 86] Diaz,M.(ed.): "IFIP Workshop on Protocol Specification, Testing and Verification V", **North-Holand**, Amsterdam, 1986.

[EhMha 85] Ehrig,H. e Mhar,B.: "Fundamentals of Algebraic Specification 1", **Spring-Verlag**, Berlim, 1985.

[ISO 88] ISO IS 8807: "LOTOS - A Formal Description

Technique Based on the Temporal Ordering of Observational Behaviour", Maio 1988.

[KrMaNg 89] Kramer,J., Magee,J. e Ng,K.: "Graphical Support for Confuguration Programming, Research Report, Departament of Computing, Imperial College, Jan. 1989.

[Milner 80] Milner,R.: "A Calculus of Communication Systems", Lecture Notes in Computer Science, Spring-Verlag, 1980.

[MoCun 89] Moura,M.T.S. e Cunha,P.R.F.: "Desenvolvendo Especificações de Sistemas Distribuídos na Linguagem LOTOS, Relatório Técnico, Departamento de Informática - UFPE, 1989.

[Myers 78] Myers,G.J.: "Composite/Structured Design", van Nostrand Reinhold Co., 1978.

[NajQue 88] Najm,E e Queiroz,J.: "Graphical BNF: A Simple Approach for the Definition of GLOTOS", INRIA, Relatório Técnico, Rocquencourt, França, 1988.

[Peter 81] Peterson,J.L.: "Petri Net Theory and the Modeling of Systems", Prentice-Hall Inc., New Jersey, 1981.

[Souza 87] Souza,W.L.: "LOTOS, uma Técnica para a Descrição Formal de Serviços e Protocolos de Comunicação", Anais do 5o. Simpósio de Redes de Computadores, São Paulo, Abril, 1987.

ANEXO: ESPECIFICACÃO LOTOS DO SISTEMA VÍDEO/TV.

```
specification VRC/TV : noexit
   type ...
   end of def
behavior
hide g in VRC[painel,fita,g] |[g]| TV[botão,canal,g]
where
process VRC[painel,fita,g] : exit :=
   LIGA[painel] >> (FUNCIONA[painel,fita,g] [> DESLIGA[painel,g])
where
   process LIGA[painel] : exit :=
      painel!liga ; exit
   endproc
   process DESLIGA[painel,g] : noexit :=
      painel!desliga ; g!desliga ; exit
   endproc
   process FUNCIONA[painel,fita,g] : noexit :=
      VRC-TV[painel,g] [> VRC-VRC[painel,fita,g]
```

```
    where
       process VRC-TV[painel,g] : noexit :=
          painel?canal:int ; g!canal ; VRC-TV[painel,g]
       endproc
       process VRC-VRC[painel,fita,g] : noexit :=
          fita!in ; g!play ; CONTROLE[painel,fita,g]
       endproc
       process CONTROLE[painel,fita,g] :=
          [painel!play]  → g!play ; CONTROLE[painel,fita,g]
       [] [painel!rew]   → g!rew  ; CONTROLE[painel,fita,g]
       [] [painel!ff]    → g!ff   ; CONTROLE[painel,fita,g]
       [] [painel!slow]  → g!slow ; CONTROLE[painel,fita,g]
       [] [painel!pause] → g!pause; painel!pause ; g!pause ;
                           CONTROLE[painel,fita,g]
       [] [painel!stop]  →
             g!stop ; ((VRC-TV[painel,g]
                        |||
                        CONTROLE-VRC[painel])
                       [>
                        ([painel!play] → g!play;
                                         CONTROLE[painel,fita,g]
                         [] [fita!out] → FUNCIONA[painel,fita,g]))
       where
          process CONTROLE-VRC[painel] :=
             [painel!rew]  → (ATIVA-VRC(rew)
                              >> CONTROLE-VRC[painel])
          [painel!ff]      → (ATIVA-VRC(ff)
                              >> CONTROLE-VRC[painel])
             [painel!stop] → (ATIVA-VRC(stop)
                              >> CONTROLE-VRC[painel])
          endproc
       endproc
    endproc
endproc
process TV[botão,canal,g] : noexit :=
   let x:int=4 in (LIGA-TV[botão] >> (FUNCIONA-TV[canal,g]
                                      [> DESLIGA-TV[botão]))
where
   process LIGA-TV[botão] : exit :=
      botão!liga ; exit
   endproc
   process DESLIGA-TV[botão] : exit :=
      botão!desliga ; exit
   endproc
   process FUNCIONA-TV[canal,g] :=
      [canal!x:int] → (MUDA-CANAL-TV(x) >> FUNCIONA-TV[canal,g])
   [] [g?y:sinal [y <> pause] ; x=4] → (ATIVA-VRC-TV(y)
                                        >> FUNCIONA-TV[canal,g])
   [] [g?z:int ; x=4] → (MUDA-CANAL-VRC-TV(z)
                         >> FUNCIONA-TV[canal,g])
   [] [g!pause ; x=4] → (ATIVA-VRC-TV(pause) ; g!pause ;
                         (ATIVA-VRC-TV(continue)
                          >> FUNCIONA-TV[canal,g]))
   endproc
endproc
```