MODELAGEM VISUAL DE SIMULAÇÃO A EVENTOS DISCRETOS BASEADO NO DCA

Eduardo Saliby

Coppead/UFRJ

Ilha do Fundão, Rio de Janeiro - RJ

C. Postal 68514 – CEP 21949-900

saliby@coppead.ufrj.br

Eugênio Merhi

Universidade Federal de São João Del Rei

Praça Frei Orlando, 170

Cep 36300-000 São João Del Rei - MG

merhi@funrei.br

Ricardo Miyashita

Departamento de Engenharia Industrial/UERJ

R. S. Francisco Xavier 524 Pav. João Lyra Fº/ Bloco A Sala 5030

Cep 20.550-013 Rio de Janeiro RJ

miya@uerj.br

Resumo

Este trabalho descreve a elaboração de dois novos ambientes de modelagem visual e interativo para a análise e solução de problemas de simulação a eventos discretos. O objetivo principal é possibilitar a criação de modelos de simulação utilizando o Diagrama de Ciclo de Atividades (DCA). Através destes ambientes o usuário pode criar o diagrama lógico do problema a ser estudado diretamente na tela do computador e uma vez montado o modelo é possível rodar a simulação a partir do modelo lógico. Ambos utilizam o Método das Três Fases como algoritmo de simulação e possuem uma programação orientada a componentes. Os resultados da simulação são mostrados na forma de relatórios estatísticos e histogramas. Há a possibilidade de adaptar o código dos programas para problemas que necessitem de ajustes especiais para a sua modelagem. O uso dos ambientes mostra que eles possuem características inovadoras e fornecem resultados precisos.

Palavras-chave: simulação, modelagem, DCA

Abstract

This paper describes the development of two new visual modeling interactive environments for analysis and solving of discrete-event simulation problems. The main purpose of the researches is to offer some tools for creation of simulation models through the Activity Cycle Diagram (ACD). Using these environments the user can create the logic diagram of the problem directly in the computer screen and run the simulation immediately just using this diagram. Both of them use the Three-Phase Method as the simulation algorithm and use the component-oriented paradigm. The simulation results are shown by statistics reports and histograms. There is also the possibility to customize the programming code in problems that needs special adjusts in the modeling phase. The practical use of these environments shows that they have innovative characteristics and that they give correct results.

Key words: simulation, modeling, ACD

1. Introdução

A Pesquisa Operacional (PO) tem avançado tem avançado muito nos últimos anos no que diz respeito aos métodos estocásticos de modelagem, em especial da Simulação. Isto se deve a uma questão de prática de que muitos problemas da vida real não possuem soluções analíticas adequadas (Banks et al., 1999). Outro fator que explica este avanço é a melhoria do desempenho dos computadores atuais que diminuiu o tempo de processamento dos programas. Um terceiro fator de importância está ligado à engenharia de software que criou avançadas ferramentas, dentre as quais se destaca a programação orientada a componentes, que permite o desenvolvimento de programas com alta complexidade de um modo integrado e robusto (Pidd et al., 1999, Chen e Szymanski, 2001).

A técnica da PO tratada especificamente neste trabalho é a chamada Simulação a Eventos Discretos. Para a solução de problemas típicos desta área de estudo, costuma-se separar duas etapas, uma de modelagem e outra de processamento. Para a modelagem, a literatura cita várias técnicas, dentre as quais citar como as mais importantes o Diagrama de Processos (Banks et al., 1999), e o Diagrama de Ciclo de Atividades ou DCA (Pidd, 1998). Sobre o algoritmo para o processamento da simulação, as técnicas mais utilizadas são a Interação de Processos e Método das Três Fases (Martinez e Ioannou, 1999). Observando os programas de simulação atualmente existentes (Hlupic, 2000, Banks et al., 1999) verifica-se que, embora se possa utilizar várias composições de filosofias de modelagem com algoritmos de processamento, são duas as combinações mais comuns. Muitos programas de simulação fazem a modelagem através de Diagramas de Processos e processamento através do algoritmo de Interação de Processos, formando o que se denomina de abordagem por processos (Pidd, 1998). Uma outra abordagem é a da escola inglesa de simulação, que costuma utilizar o Diagrama de Ciclo de Atividades (DCA) em conjunto com o Método das Três Fases (Martinez, 2001, Saliby, 1995).

Verifica-se que a escola inglesa, embora seja seguida por muitos pesquisadores, possui uma gama de programas disponíveis muito menor do que a da abordagem por processos (Pidd, 1998). E se analisarmos os poucos programas existentes, observamos ainda que carecem de uma interface de modelagem amigável (Martinez, 2001), tornando a tarefa de criação do modelo algo demorado, trabalhoso e sujeito a erros. Já os modernos programas de abordagem por processos possuem estas interfaces integradas ao simulador em suas versões mais modernas, eliminando desta forma os problemas citados.

Tendo em vista esta constatação, identificamos uma oportunidade para o desenvolvimento de novas ferramentas de modelagem visual interativa para a abordagem DCA-Três Fases, que consideramos ser de grande utilidade para os profissionais da área de simulação. O presente trabalho busca ir ao encontro desta necessidade.

O objetivo deste trabalho é, portanto a descrição de dois novos ambientes interativos para a criação de modelos baseados no *Diagrama de Ciclo de Atividades* (DCA). Os ambientes são o wSimul e o SimVisio, e são resultados de pesquisas realizadas na Coppe/UFRJ. Em ambos sistemas, a modelagem do problema estudado é feita diretamente na tela do computador e o processamento do algoritmo de simulação é realizado segundo *o Método das Três Fases.* A abordagem de programação segue as modernas tendências da engenharia de software, sendo *orientada a componentes*. Pretende-se também dar a oportunidade de *customização* do programa através de programação manual para problemas que fujam à modelagem padrão.

2. Diagramas de Ciclos de Atividade (DCA)

A análise de um problema através de simulação envolve uma modelagem lógica do problema estudado. Dentre as diversas metodologias de modelagem que poderiam ser escolhidas, optamos para os nossos ambientes um tipo que é também bastante utilizado por diversos outros profissionais de simulação e que é especialmente difundido no Brasil e que se chama **Diagrama de Ciclos de Atividades** ou **DCA**, idealizado inicialmente por Tocher (1963) e seguido pela chamada escola "inglesa" de simulação (Saliby, 1995). A descrição de como é montado um diagrama DCA pode ser encontrada nos textos de autores que o utilizam (Martinez, 2001; Pinto, 1999; Pidd, 1998; Odhabi, 1998 e Tavares *et al.*, 1996). Sobre os elementos que fazem parte do diagrama há pequenas variações que dependem do autor, mas o núcleo desta metodologia está bem definido na literatura.

Podemos utilizar os DCA para descrever um problema através dos estados das entidades em cada momento. Em sua versão mais simples, os elementos do DCA são três: **entidades, filas e atividades**. Estes elementos devem ser colocados no diagrama dentro de regras definidas. A primeira regra diz que deve haver alternância entre atividades e filas dentro de um ciclo. A segunda regra diz que uma determinada fila pode conter somente um tipo de entidade. Uma terceira regra que diz que as entidades devem percorrer ciclos fechados. Há outras regras de construção, mas as três acima citadas são as principais.

O resultado final de um trabalho de modelagem segundo o DCA são vários ciclos de atividades interligados, que representam o comportamento do sistema de um ponto de vista lógico. A tarefa principal do analista do problema é abstrair da situação estudada quais são as atividades e filas que melhor representam o problema, bem como as ligações que formam os diversos ciclos. Um exemplo de diagrama DCA é mostrado na figura 1. Trata-se do problema do atendente de teatro apresentado por Pidd (1998). Um funcionário que trabalha na bilheteria de um teatro e recebe tanto as pessoas que querem comprar um ingresso quanto as ligações de cliente solicitando informações. Duas fontes representam a origem de entidades no sistema: uma fonte de pessoas e uma fonte de telefonemas. Há no exemplo quatro atividades: Chegada de novas ligações, Atendimento destas ligações, Chegada de pessoas e Atendimento de pessoas. E são três as filas existentes: Aguarda Atendimento pessoal, Aguarda Atendimento telefônico e Atendente Ocioso.

Note que há três ciclos neste sistema, um para a entidade Telefonema (à esquerda), um para o Atendente (no centro, com setas mais grossas) e um para a entidade Pessoa (à direita, com setas tracejadas). Estes três ciclos se cruzam naquelas atividades em que é necessária a cooperação de duas entidades para sua execução; é o caso da atividade Atende Pessoa, que para sua execução necessita que o atendente esteja ocioso e que haja alguma pessoa na fila de atendimento. Fica claro neste exemplo que há uma alternância entre filas e atividades dentro de cada ciclo.

Os DCA's se destacam por sua facilidade de compreensão, pois são construídos a partir de elementos simples, e pela flexibilidade em representar problemas de diversas naturezas, podendo modelar várias situações como, por exemplo, filas de atendimento, sistemas de produção, sondas de perfuração de petróleo até navios em um porto. Os diagramas DCA facilitam a criação de programas de simulação, pois a partir deles é fácil visualizar os Eventos, que são peças importantes para

descrever o comportamento do modelo. No caso do exemplo do teatro os eventos são seis, um para cada atividade de chegada e dois para cada uma das outras atividades (um correspondente ao seu início e outro ao seu fim).

Ao longo do seu desenvolvimento, que envolveu uma série de pesquisadores, o DCA foi demonstrando muitas qualidades e algumas limitações. Com o fim de contornar estas limitações, foram desenvolvidas algumas variações sobre o DCA (ou ACD, como é chamado em inglês), que são o DCA expandido (Pinto, 1999), o X-ACD (Pooley e Hugues, 1991), o SH-ACD (Odhabi *et al.*, 1998a) e o H-ACD (Kienbaum e Paul, 1994). O próprio exemplo que foi apresentado, o do atendente de teatro possui um elemento que não consta nas primeiras versões do DCA, que é elemento fonte. A fonte é importante para representar a origem de novas entidades do sistema, e está presente em todas as implementações mais modernas do DCA que foram citadas acima.

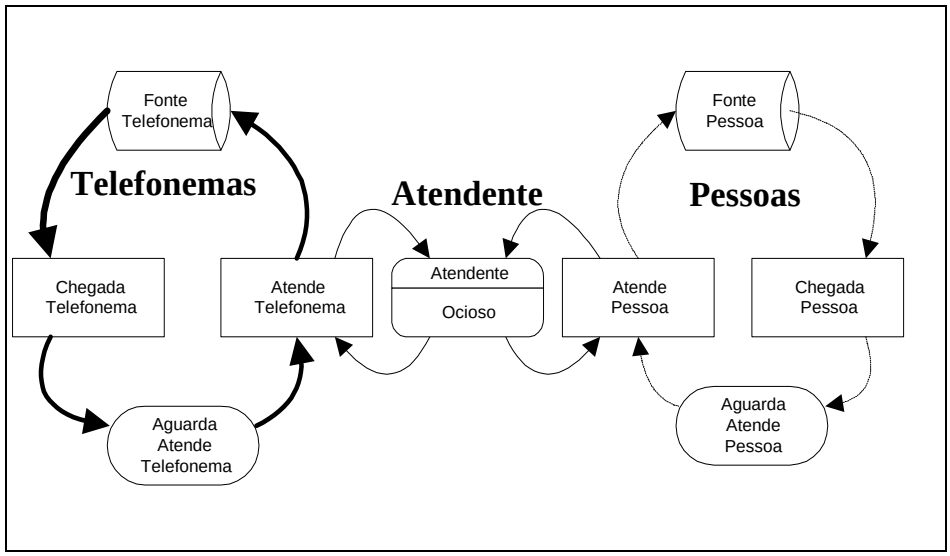


Figura 1. Diagrama lógico do tipo DCA para o problema do atendente de teatro, baseado em Pidd (1998).

De uma forma geral, o DCA tem se mostrado extremamente útil, pelas seguintes razões:

1) Possibilita a criação de modelos com uma lógica bem definida;

2) Mostra os ciclos das entidades de modo bastante claro;

3) Reflete de uma maneira completa as informações necessárias para a construção de programas de computador para a execução da simulação.

A principal alternativa ao DCA para a modelagem lógica é a modelagem por processos, que é utilizada por grande parte dos programas de simulação comerciais. O DCA apresenta vantagens em relação à metodologia por processos em duas situações básicas. Em primeiro lugar, quando há um número grande de entidades interagindo nas atividades do modelo e também quando as regras de definição de início das atividades são complexas. Devido a isto, muitos pesquisadores prefiram utilizar o DCA (Martinez e Ioannou, 1999; Pidd, 1998; Shi, 1997). Um fator negativo para o DCA em relação à abordagem por processos é o de que no DCA o usuário deve raciocinar dentro de uma lógica que lhe é bastante peculiar, pensando em termos de filas e atividades e às vezes tendo que criar artifícios especiais para adaptar estes elementos para modelar o problema estudado. Já abordagem por processos tende a ser mais intuitiva, principalmente para os usuários iniciantes.

3. Ambientes de Modelagem Visual baseados no DCA

Apresentamos a seguir os dois ambientes desenvolvidos para a modelagem visual interativa.

3.1 O ambiente SimVisio

O **SimVisio** é um ambiente integrado de construção e processamento de modelos de simulação que utiliza a metodologia do Diagrama de Ciclo de Atividades. Ele é constituído por um conjunto de Blocos ou *Shapes* de simulação preparados para o uso no programa Visio e por um programa, o SimVisio.exe que lê as informações do diagrama criado e processa a simulação utilizando as suas informações. O trabalho de criação do Simvisio é uma evolução de dois sistemas de simulação anteriores: o Simin (Pinto, 1999) e Simul (Saliby, 1995). Historicamente todos estes trabalhos fazem parte de uma família de simuladores que teve início em centros de pesquisa da Inglaterra.

Para criar um modelo de simulação basta abrir uma nova página do Visio em branco e ir arrastando os elementos desejados a partir do painel do DCA para a página em branco. Para mudar as propriedades de cada elemento basta acioná-lo e editar uma janela que abre sobre o mesmo. Para ilustrar o uso do SimVisio iniciaremos com um exemplo simples, um sistema que simula um caixa de **Banco** (fig. 2.1). Neste sistema há uma fila única e um único caixa. Utilizando a notação da Teoria das Filas classificamos este caso como um típico sistema M/M/1.

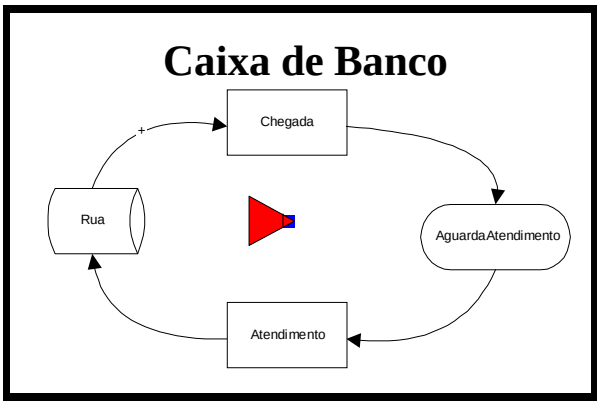


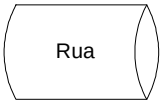
Figura 2.1 Modelo DCA feito no SimVisio do problema do banco (M/M/1)

Figura 2.2 Shapes de simulação do Simvisio colocados no Painel DCA do Visio.

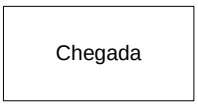
Para criar modelos de simulação no Simvisio devemos primeiramente carregar no Visio com o Painel de simulação de nome DCA (fig.2.2) e utilizar os Shapes de simulação disponíveis para criar o modelo.

3.1.1 Elementos básicos de modelagem do SimVisio

O trabalho do usuário do sistema se restringe à criação do modelo na interface do Visio, levando em conta as peculiaridades do problema em estudo. A seguir descrevemos o uso de cada um dos blocos que servem para a montagem do DCA através do Simvisio.

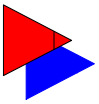


O bloco **Fonte** representa a porta de entrada de entidades temporárias no sistema. As entidades temporárias são aquelas que, após o atendimento, deixam o sistema. Por exemplo, no modelo representativo do caixa de banco, pode-se criar uma fonte de nome Rua para representar a chegada de clientes ao banco. Neste caso, os clientes são entidades temporárias, pois deixam o sistema após o atendimento. Esta denominação foi criada para diferenciar de entidades permanentes, que são aquelas que estão ativas no sistema durante todo o período de simulação.

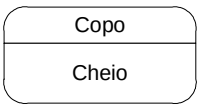


O bloco **Atividade** representa as atividades do sistema. Várias entidades podem participar da atividade. Uma atividade somente inicia quando houver entidades disponíveis nas filas ou fontes que a precedem. A duração da atividade possui uma distribuição de probabilidade que deve ser indicada quando a atividade é criada. Ao final da atividade podem ser realizados cálculos com atributos das entidades envolvidas. No exemplo do banco há um caixa que realiza a tarefa de atendimento segundo uma distribuição de probabilidade Exponencial Negativa e capacidade igual a 1. Para a atividade de chegada, o intervalo entre chegadas tem distribuição Exponencial Negativa e capacidade Infinita (por definição as atividades de chegada devem ter capacidade infinita).

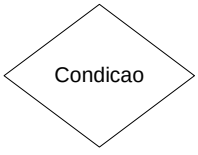
O bloco **Fila** representa as filas do sistema. As filas estão localizadas antes de cada atividade, sendo que deve ser indicada uma fila para cada tipo de entidade que entra nas atividades. As filas podem ter capacidade infinita ou limitada. Quando se atinge o limite de uma fila, podem ocorrer dois tipos de reação: ou se eliminam as novas entidades que tentam entrar na fila, ou se bloqueia a atividade anterior para que ela não envie novas entidades para esta fila. Não é necessário indicar a entidade que irá permanecer na fila; a identificação é feita automaticamente pelo programa quando é analisado o diagrama. No caso do banco, devemos colocar uma fila antes da atividade de atendimento. Note que o nome da fila foi colocado sem espaço em branco entre “Aguarda” e “Atendimento” por restrições do programa, que não aceita este tipo de caracteres.



O bloco **Simulação** determina os parâmetros da execução da simulação, como tamanho e número de corridas e tamanho do período de aquecimento. Define-se o nome do problema e as informações que se desejam ver durante a simulação, como por exemplo, a atividade que está sendo processada no momento.No caso do Banco, o título do problema é “CaixaDeBanco”, a duração da corrida, o tamanho do período de aquecimento e o número de corridas podem ser indicados segundo a conveniência do usuário. Podem ser solicitadas informações para visualizar o andamento da simulação.



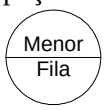
Um bloco **FilaRec** se assemelha a um bloco Fila, mas com algumas propriedades adicionais. Quando se coloca um bloco deste tipo no modelo, o simulador se encarrega de preencher a fila com entidades permanentes no início da simulação. O nome da entidade permanente, sua quantidade e seus atributos são definidos na própria janela de propriedades da FilaRec. Depois de iniciada a simulação, o bloco FilaRec se comporta como um bloco Fila simples, segundo as suas características de capacidade, disciplina de retirada e histogramas. Pode-se dizer, portanto, que um bloco FilaRec é como uma combinação de um bloco Fila com um bloco Fonte, mas com a diferença que o FilaRec cria entidades permanentes e o Fonte entidades temporárias.



O bloco **Condição** estabelece desvios condicionais para as entidades que saem de um bloco Atividade. É determinado um teste para a entidade que termina a atividade, envolvendo um atributo desta entidade, o operador matemático para o teste (<,>, <=, >= ou =) e o valor limite para o teste. Caso o atributo em questão esteja de acordo com a condição de teste, a entidade é encaminhada para a fila para a qual o conector “Sim” indicar; caso contrário, encaminha-se para onde o conector “Não” estiver apontando.

O bloco **Inspeção** deve ser colocado sempre após um bloco de atividade, e serve para direcionar a entidade ao final de sua execução. Define-se a porcentagem de

entidades a serem aprovadas, havendo várias porcentagens de aprovação. As restantes serão, portanto, rejeitadas, somando entre aprovadas e rejeitadas o total de 100%. As rejeitadas irão para a fila indicada pelo conector “Sim” (sem defeito) e as aceitas para a fila indicada pelo conector “Não” (com defeito). As entidades rejeitadas podem voltar para uma parte anterior do ciclo, para serem reprocessadas. Para exemplificar o uso do bloco inspeção, tomemos o exemplo de uma máquina em operação em uma fábrica, onde há 10% de probabilidade de produzir uma peça defeituosa. Caso a peça tenha defeito, é processada novamente na máquina. Caso contrário, segue na linha de produção.



O bloco **Auto** deve ser colocado logo após o término de uma atividade e realiza a busca automática da fila de menor tamanho dentre aquelas que estão disponíveis. Um exemplo de aplicação de um bloco auto é o de um pequeno supermercado, onde os clientes chegam em intervalos exponenciais. Os clientes pegam a mercadoria em um carrinho e depois escolhem a menor fila dentre as várias existentes, cada uma correspondente a uma determinada caixa do mercado.

Uma vez montados os blocos do SimVisio no modelo, é possível rodar a simulação, chamando o programa que contém as rotinas de leitura do modelo e execução da simulação denominado Simvisio.exe (fig. 2.3). Ao rodarmos o modelo do Banco, aparece em primeiro lugar uma janela onde são mostradas as etapas de análise do modelo. Nesta etapa o Simvisio lê as propriedades do diagrama DCA e cria os componentes de simulação internos, para que ele possa rodar o algoritmo de simulação sobre estes componentes. É feita uma verificação dos erros de sintaxe do modelo e por fim aparece a janela de execução da simulação.

Figura 2.3 Tela Principal do SimVisio.

3.1.3 Recursos de saídas de dados, programação manual e analisador de sintaxe.

Ao final da simulação é possível ver os **relatórios** contendo estatísticas da simulação. Há dois tipos: o Relatório de Atividades e Entidades e o Relatório de Histogramas. O primeiro contém, além dos dados de atividades e entidades, os dados referentes ao número de entidades que passaram em cada fila bem como as estatísticas dos blocos de Inspeção e Condição.

De modo semelhante aos relatórios podem ser vistos os **histogramas** de Tempo de Espera e de tamanho das filas do problema. Uma outra funcionalidade muito importante que foi colocada à disposição do usuário foi a possibilidade de **exportação dos dados** de cada histograma para arquivos texto. Estes arquivos são solicitados pelo usuário quando é definida a propriedade Histograma dos blocos Fila, FilaRec e Atividade. Para cada histograma para o qual foram pedidos os seus dados são criados vários arquivos contendo o nome do histograma e o número da corrida. Com base nos dados contidos nestes arquivos é possível fazer uma série de análises através de pacotes estatísticos.

O SimVisio possui um módulo programável que permite a adaptação do sistema à peculiaridades de problemas em que não é possível modelar somente com os blocos lógicos do DCA. A **programação manual** é feita com a ajuda do compilador Delphi, e gerando um novo programa que interage com o modelo lógico criado no ambiente SimVisio. Para possibilitar a identificação dos ciclos de atividades, deve-se utilizar o bloco “Prog” na construção do modelo lógico. Este bloco tem por fim indicar ao programa quais as possíveis alternativas de filas que podem se seguir a uma atividade e se comporta durante o período de identificação dos ciclos de modo semelhante ao bloco “Auto”.

O sistema conta também com um **verificador de sintaxe** para erros de modelagem, bem como um verificador de erros de preenchimento das propriedades. Ao mesmo tempo em que são processadas as informaçõe do diagrama, são informados ao usuário os eventuais erros que possam ter sido cometidos, tanto no preenchimento das informações quanto na montagem dos ciclos. Um bom processamento da simulação depende do correto preenchimento das propriedades de cada shape do modelo. Algumas propriedades podem ficar em branco, mas outras devem necessariamente ser fornecidas com valores de determinado tipo. Para cada propriedade importante é feita uma verificação do valor fornecido pelo usuário, e mensagens de erro são emitidas pelo programa, indicando o tipo de erro cometido, em que parte do modelo ocorreu e se dá possibilidade ao usuário para que o corrija, de modo a que depois o programa possa rodar corretamente.

3.2. O Ambiente wSimul

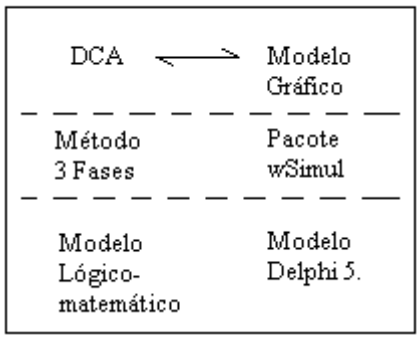
3.2.1. Apresentação do wSimul

O **wSimul** constitui-se num ambiente para modelagem e execução de Simulação a Eventos Discretos composto da união de três ferramentas básicas: (1) uma interface de modelagem visual/gráfica; (2) um pacote de sub-rotinas e (3) a linguagem de programação Delphi 5.

A **Interface de modelagem visual** permite a criação de DCA's através de operações Drag-and-Drop típicas do sistema operacional Windows. As informações adicionais dos modelos, tais como intervalo entre chegadas de entidades, tempo de duração das atividades ou disciplina de filas, são inseridas no modelo através de caixas de diálogos acessadas a partir do respectivo elemento gráfico utilizado no DCA.

Uma vez construído o DCA representativo do modelo em estudo, a ferramenta de interface gera um programa estruturado segundo o Método das 3 Fases utilizando as sub-rotinas do **pacote wSimul**. Este programa gerado poderá, então, ser editado (quando necessário) e traduzido pelo compilador **Delphi**, gerando um programa/modelo executável. A execução deste programa/modelo cria uma animação da simulação e, ao término desta, os seus tradicionais relatórios .

Fig. 3.1. Estrutura do wSimul



3.2.2. A Interface de Modelagem wSimul

A **Interface de modelagem do wSimul** é composta de uma tela principal com uma barra de ferramentas e um formulário para criação do modelo DCA. A fig. 3.2 mostra sua aparência.

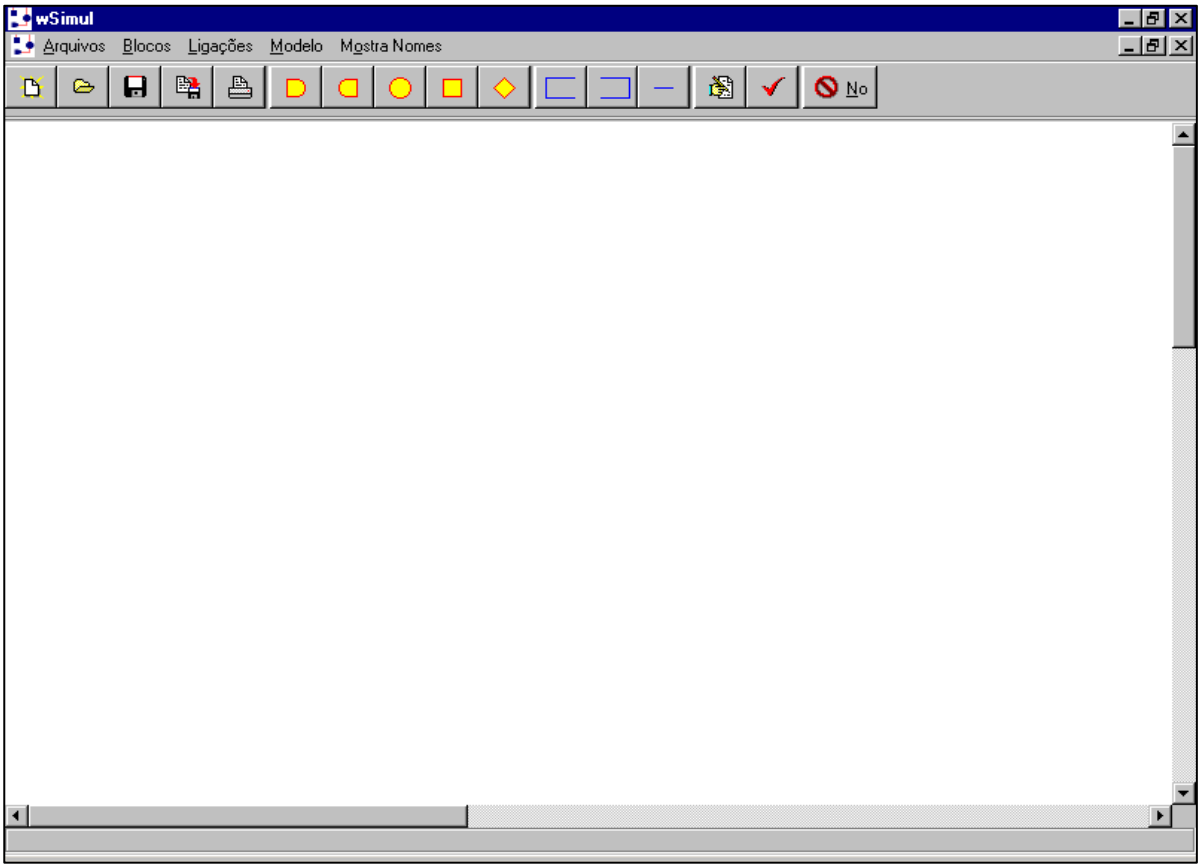


Fig. 3.2. Tela Principal da Interface

Nesta tela pode ser observada a existência de cinco grupos de botões, a saber:

- O primeiro grupo, composto de 5 botões, permite operar com os arquivos/modelos. Cada um destes botões realiza as operações de iniciar um novo modelo, abrir um modelo existente, salvar o modelo atual, salvar como, e imprimir modelo atual.
- segundo grupo, composto de 5 botões, representa os elementos do DCA que deverão ser utilizados na criação do modelo. Cada click em cada um destes botões cria um novo elemento DCA no modelo.
- O terceiro grupo, composto de 3 botões, permite unir cada um dos elementos do DCA, criando o ciclo de atividades propriamente dito.
- O quarto grupo é composto por apenas 2 botões: o primeiro realiza uma verificação no modelo com respeito à ocorrência de erros de modelagem; o segundo, uma vez que o modelo não contenha erros de modelagem, cria o programa/modelo wSimul/Delphi.
- Por fim, o quinto e último grupo, composto de apenas um botão, cuja função é a de mostrar ou esconder os nomes dos elementos DCA.

Os modelos são criados através da colagem de elementos e a criação de seus relacionamentos, que são determinados pelos links (setas) que os unem. As informações adicionais do modelo, tais como intervalo entre chegadas de entidades, duração de atividades, etc., podem ser inseridas nos próprios elementos do modelo, em suas respectivas caixas de diálogo que se abrem com um duplo

click. As informações sobre disciplinas de filas podem ser inseridas através dos links, cujas caixas de diálogo também são abertas através de um duplo click.

Uma vez criado o modelo gráfico e wSimul/Delphi, o usuário poderá - e em muitos casos, deverá - editá-lo no ambiente Delphi, bem como comandar a sua compilação. Vencidas estas etapas, um novo programa executável é gerado, consistindo no modelo criado pelo usuário. Com a execução deste último modelo, a simulação poderá ser realizada gerando os seus respectivos relatórios.

3.2.3. O Pacote de Sub-Rotinas wSimul

O **pacote wSimul** foi inspirado em outro pacote de sub-rotinas denominado Simul. O Simul: sistema computacional para a simulação a eventos discretos é o resultado de uma evolução natural de trabalhos iniciados em Lancaster (Inglaterra) por Spinelli de Carvalho (1975), que desenvolveu o sistema XLSIM. Ainda em Lancaster, seguiram-se os trabalhos de Irma Angulo (1983) e de John Crookes (1985) que resultou no TURBOSIM, a primeira versão do sistema de simulação usando o TURBO PASCAL. O trabalho de Crookes foi continuado em duas frentes: uma pelo grupo CASM da London School of Economics (Balmer e Paul, 1986) e outra por Ruth Davies e Robert O'Keefe (1989) da Universidade de Southampton. Do trabalho do grupo CASM resultou inicialmente o sistema ELSE (Crookes et Alii, 1986) e, posteriormante, o VS6 (Syspack Ltd, 1989), que serviu como referência básica para o SIMUL. O SIMUL, em sua versão original, foi o principal resultado da tese de mestrado de Milton Pimentel (1989), do Instituto Militar de Engenharia (IME/RJ). Desde então ele vem sendo testado e utilizado em diversas aplicações: do ensino em cursos de graduação e pós-graduação até as aplicações em problemas reais em empresas. Souza (1994) realizou diversas melhorias no pacote originando a atual versão do Simul.

O pacote wSimul é uma coleção de sub-rotinas voltadas para a criação de programas/modelos baseados no DCA e estruturadas segundo o Método das 3 Fases. Os programas criados com base nestas duas metodologias têm uma correspondência direta com o DCA criado e são bem fáceis de serem alterados e entendidos.

O wSimul, em sua implementação atual, procurou manter a estrutura básica de seu predecessor. Não somente a forma de estruturação foi parcialmente seguida. Os nomes e parâmetros das sub-rotinas também foram mantidos, embora de maneira parcial. Esta nova implementação foi necessária por 3 motivos principais: (1) o Simul ainda era um sistema projetado para o sistema operacional DOS; (2) uma pequena mudança na forma de estruturação dos modelos foi efetuada e (3) um novo recurso de modelagem foi acrescentado, a saber : a possibilidade de existência de atividades preemptivas.

3.2.4. Resultado da Integração

Apesar de oferecer poucos recursos gráficos de modelagem, a ferramenta como um todo se mostra muito útil para aplicações de um modo geral. A grande capacidade do DCA de representar modelos, aliada a uma adequada estruturação do programa fonte, resultado da utilização do Método das 3 Fases e do pacote wSimul, fez desta **integração** uma metodologia de grande poder de modelagem. Decorre desta integração uma interessante característica do wSimul: **dada as metodologias utilizadas, há uma passagem natural e coerente entre os diversos níveis de modelagem**.

4. Conclusões

A maior contribuição deste trabalho foi proporcionar novos ambientes de modelagem visual e processamento de simulação para problemas de simulação segundo o Diagrama de Ciclos de Atividades. Os ambientes wSimul e SimVisio demonstraram ser de fácil utilização e robustos do ponto de vista estatístico, possibilitando a modelagem de uma gama bastante grande de problemas de simulação. Suas interfaces são bastante simples e intuitivas, sendo de fácil aprendizado. Os diagramas são montados na tela com grande facilidade, utilizando elementos gráficos representativos dos blocos lógicos e de suas ligações. Com eles torna-se fácil criar diferentes cenários para a simulação, bastando para isto mudar as propriedades dos blocos do modelo. O processamento da simulação é também bastante simples. Cabe ressaltar que é possível modificar as linhas de código de ambos programas para que eles se adequem a problemas específicos que fogem à modelagem através dos elementos disponíveis no DCA padrão.

O algoritmo de simulação utilizado, denominado Método das Três Fases, mostrou-se muito adequado para ser utilizado para modelos criados através do Diagrama do Ciclo de Atividades. Os testes de aplicação a uma série de modelos mostraram que ele possui rapidez de processamento. Além disso, verificou-se que seus resultados estatísticos são precisos, quando compararados com os calculados pela Teoria das Filas e com outros programas de simulação mais tradicionais.

Como resultado final consideramos que o trabalho possui uma série de inovações no campo da simulação e esperamos que os sistemas criados sejam úteis para o uso de profissionais e pesquisadores da área.

5. Referências Bibliográficas

BANKS, J., CARSON, J.S., NELSON, B.L., 1999, *Discrete-Event System Simulation*. 2 ed. New Jersey, Prentice-Hall.

CHEN, G., SZYMANKI, B. K., 2001, "Component-oriented Simulation Architecture: Toward Interoperability and Interchageability ". In: *Proceedings of the 2001 Winter Simulation Conference*, pp. 495-501, December.

KIENBAUM, G., PAUL, R.J., 1994, "H-ACD: Hierarchical Activity Cycle Diagrams for Object-Oriented Simulation Modeling". In: *Proceedings of the 1994 Winter Simulation Conference*. December.

KIENBAUM, G. 1995. *A Framework for Automatic Simulation Modelling Using an Object-Oriented Approach*". Tese Ph.D., Brunel University, England.

MARTINEZ, J.C., 2001, "Ezstrobe – General-Purpose Simulation System based on Activity Cycle Diagrams". In: *Proceedings of the 2001 Winter Simulation Conference*, pp. 1556-1564, December.

MARTINEZ, J.C., e IOANNOU, P.G., 1999, "General-Purpose Systems for Effective Construction Simulation". *Journal of Construction Engineering and Management.* pp. 265-275 (Jul-Aug).

ODHABI, H.I., PAUL, R.J., MACREDIE, R.D., 1998a, "Developing a Graphical User Interface for Discrete Event Simulation". In: *Proceedings of the 1998 Winter Simulation Conference*, pp. 429-436, Washington, D.C., December.

ODHABI, H.I., PAUL, R.J., MACREDIE, R.D., 1998b, "Making Simulation More Accessible in Manufacturing System through a 'Four Phase' Approach". In: *Proceedings of the 1998 Winter Simulation Conference*, pp. 1069-1075, Washington, D.C., December.

PIDD, M., 1998, *Computer Simulation in Management Science*. 4 Ed. Chichester, England, John Wiley & Sons.

PIDD, M., OSES, N., BROOKS, R.J. 1999, "Component-based Simulation on the Web? ". In: *Proceedings of the 1999 Winter Simulation Conference*, pp. 1438-1444, December.

PIMENTEL, M., 1989, *Sistema computacional para simulação discreta com opção de uso da amostragem descritiva*, Dissertação de M.Sc., IME, Rio de Janeiro, RJ, Brasil.

PINTO, L. R., 1999, *Metodologia de Análise do Planejamento de Lavra de Minas a Céu Aberto Baseada em Simulação das Operações de Lavra*. Tese de D.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

POOLEY, HUGUES, 1991, "Towards a Standart for Hierarchical Process Oriented Discrete Event Simulation Diagrams. Part II: The Suggested Approach for Flat Models". *Transactions of the Society for Computer Simulation*, 8 (1): 33-41.

SALIBY, E., 1989. *Repensando a simulação: a amostragem descritiva*, São Paulo/ Rio de Janeiro, Atlas/EDUFRJ.

SALIBY, E., 1995. *Simul 3.1 Manual do Usuário*, Rio de Janeiro, Coppead/UFRJ.

SHI, J., 1997, "A Conceptual Activity Cycle-Based Simulation Modeling Method". In: *Proceedings of the 1997 Winter Simulation Conference*, pp. 1127-1133, December.

TAVARES, L.V., OLIVEIRA, R.C., THEMIDO, I.H., CORREIA, F.N. *Investigação Operacional*, Lisboa, McGraw-Hill, 1996.

TOCHER, K. D. , 1963. *The art of simulation*, London, English Universities Press.

Dados da publicação:

MIYASHITA, R., SALIBY, E., MERHI, E.
Modelagem Visual De Simulação A Eventos Discretos Baseado No DCA In: XXXIV Simpósio Brasileiro de Pesquisa Operacional, 2002, Rio de Janeiro.

Anais do XXXIV Simpósio Brasileiro de Pesquisa Operacional. , 2002. v.1.