COPPEAD/UFRJ

RELATÓRIO COPPEAD Nº 255

SIMUL: UM SISTEMA COMPUTACIONAL PARA A SIMULAÇÃO A EVENTOS DISCRETOS EM TURBO PASCAL

Eduardo Saliby (*)
Milton Pimentel (**)

Agosto de 1991

(*) Professor e pesquisador da COPPEAD - Instituto de Pós-Graduação e Pesquisa em Administração da UFRJ. PhD em Pesquisa Operacional (University of Lancaster, UK) Mestre em Engenharia de Produção (COPPE/UFRJ).

(**) Professor do IME - Instituto Militar de Engenharia. Mestre em Engenharia de Sistemas e Computação - Pesquisa Operacional (IME). Major QEM.

RESUMO

Este trabalho descreve o SIMUL: um sistema computacional para a simulação a eventos discretos em TURBO-PASCAL. O SIMUL possibilita o rápido desenvolvimento de programas de simulação, com uma estrutura padrão bem definida, segundo a abordagem de modelagem do diagrama do ciclo de atividades e do método das três fases. Inicialmente, é apresentado um resumo destes conceitos básicos, descrevendo-se em seguida os principais módulos do SIMUL, a estrutura padrão do programa de simulação e sua forma de utilização. Seu uso é ilustrado por um exemplo simples, porém elucidativo, descrevendo-se os principais passos no desenvolvimento do programa que é aqui apresentado, juntamente com os resultados de uma corrida de simulação.

1. INTRODUÇÃO

Este trabalho apresenta uma descrição do SIMUL: um sistema computacional para a simulação a eventos discretos desenvolvido em TURBO-PASCAL. O SIMUL, em sua versão inicial, foi o principal resultado da tese de mestrado de Milton Pimentel (1989), do Instituto Militar de Engenharia (IME/RJ).

Considerando-se a dificuldade inerente à programação de simulações a eventos discretos, notadamente para sistemas envolvendo filas, o SIMUL foi desenvolvido com o objetivo de simplificar e agilizar este processo. Além disso, o uso do SIMUL leva a programas bem estruturados, seguindo uma forma padronizada baseada no método das três fases (Pidd, 1984). A importância desta estrutura padrão não pode ser subestimada. Ela facilita em muito a compreensão dos programas de simulação, melhorando a comunicação entre especialistas e usuários finais do sistema, aumentando assim as chances de sucesso de um estudo prático. Outro importante benefício desta padronização do programa reside na sua maior modularidade, o que permite aplicar os princípios da modelagem evolutiva em estudos de simulação (Saliby, 1989).

O SIMUL destina-se ao uso em computadores que disponham de implementações da linguagem TURBO-PASCAL, versões 4.0 ou acima, em particular microcomputadores da familia IBM-PC. O uso do SIMUL requer que o usuário tenha conhecimentos básicos de TURBO-PASCAL, exigência esta compensada pelos ganhos de produtividade obtidos no desenvolvimento de um programa de simulação. Cabe mencionar, no entanto, que trabalhos subseqüentes feitos a partir do SIMUL levaram-nos ao desenvolvimento de uma interface de modelagem e de um gerador automático de programas (Saliby e Oliveira, 1989), dispensando o usuário da tarefa de programar em TURBO PASCAL. Este sistema mais completo, que também inclui um módulo para a simulação visual, foi denominado SIMUL-PLUS.

O presente trabalho restringe-se ao sistema SIMUL em sua versão programável. Em essência, o SIMUL compreende dois módulos programados em TURBO-PASCAL:

(a) uma biblioteca de rotinas pré-definidas para a simulação a eventos discretos, e

(b) um programa executivo estruturado segundo o método das três fases.

O trabalho do usuário, ao utilizar este sistema, consiste no preenchimento do módulo executivo levando em conta as peculiaridades do problema em estudo.

O SIMUL não é fruto de uma pesquisa isolada, mas o resultado de uma evolução natural dos trabalhos iniciados em Lancaster 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 produziu o 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, posteriormente, o VS6 (Syspack Ltd, 1988), que serviu como referência básica para o SIMUL.

O SIMUL tem sido exaustivamente testado por alunos de cursos de simulação ministrados em diferentes instituições, tais como o IME/RJ, a COPPE/UFRJ, a COPPEAD/UFRJ, o ITA e as Escolas de Engenharia da UFRJ e da UNESP de Guaratinguetá. Foi também testado, com sucesso, no âmbito empresarial em aplicações na área siderúrgica e portuária. Desta forma, os principais objetivos do seu desenvolvimento foram atingidos: o de servir como instrumento de ensino, de pesquisa e, ainda, aumentar a produtividade no desenvolvimento de programas de simulação.

O SIMUL é também o primeiro sistema computacional de simulação a incorporar o uso da amostragem descritiva (Saliby, 1989) como opção. Confirma-se assim que esta nova abordagem pode ser facilmente adaptada a um programa de simulação previamente desenvolvido com o uso da amostragem aleatória simples, a abordagem tradicional.

O SIMUL baseia-se na abordagem de modelagem de simulações através do diagrama do ciclo de atividades (DCA) e na estruturação de programas pelo método das três fases. Assim, para facilitar sua utilização, é inicialmente feita uma revisão destes dois conceitos da simulação a eventos discretos. A seguir, descrevemos os módulos do SIMUL e como utilizá-lo. Ilustramos o seu uso através de um exemplo simples de aplicação, o problema da manutenção de sondas de petróleo, para o qual apresentamos o programa e os respectivos resultados obtidos. Concluímos o trabalho com uma descrição da experiência acumulada até o momento, novos desenvolvimentos e possíveis extensões.

2. CONCEITOS BÁSICOS

Simulação

No contexto da pesquisa operacional, a simulação pode ser entendida como a experimentação numérica com modelos lógico-matemáticos. Esta experimentação tem por objetivo estimar parâmetros que descrevam o comportamento do sistema representado pelo modelo. Uma introdução à simulação, sua classificação e razões de sua crescente utilização encontram-se em Saliby (1989).

O presente trabalho refere-se à simulação probabilística a

eventos discretos, também conhecida como simulação por Monte Carlo. Os eventos discretos correspondem às mudanças de estado do sistema, que ocorrem em pontos bem determinados ao longo do tempo, como por exemplo quando da chegada de um avião a um aeroporto ou da quebra de um equipamento. Além de restritos à simulação a eventos discretos, nossa atenção focaliza-se no estudo do comportamento estacionário de sistemas que envolvam a formação de algum tipo de fila. Segundo a classificação de alguns autores (Exemplo: Pidd, 1984), estes casos enquadram-se na categoria dos sistemas não-terminantes.

De acordo com a definição acima, a simulação compreende duas etapas: a modelagem do problema e a simulação computacional propriamente dita, que corresponde à etapa experimental. O uso do computador é inerente à simulação e decorre do grande volume de cálculos e operações envolvidas, mesmo nos mais simples estudos.

Modelagem para a simulação

Um modelo é uma representação da realidade, em princípio uma sua simplificação. Um modelo de simulação é geralmente definido por um programa de computador que descreve um conjunto de regras lógico-matemáticas que regem o comportamento de um sistema. Há várias abordagens para a modelagem de simulações. Dentre elas, adotamos como referência o diagrama do ciclo de atividades e o método das três fases, que corresponde à abordagem da "escola inglesa". A razão desta escolha reside no binômio simplicidade-modularidade do programa de simulação produzido, como argumentam Pidd (1984) e Paul (1988).

Diagrama do ciclo de atividades

Em sua forma mais simples, o diagrama do ciclo de atividades (DCA) faz uso de três elementos para a descrição do comportamento

dinâmico de um sistema: entidade, atividade e fila.

Entidade é o elemento básico que corresponde às entidades físicas presentes no sistema. Além disso, dependendo do problema, poderão ser definidas entidades de natureza lógica, sem qualquer contrapartida física. Este é o caso, por exemplo, dos mecanismos de controle de chegada, conhecidos como porta ou entrada.

Numa simulação, uma entidade preserva sua identidade durante todo o período simulado. Em geral, as entidades de um mesmo tipo ou classe são descritas de forma conjunta num DCA, resultando numa única representação gráfica para elas. Por exemplo, ao descrever o processo de um cliente num sistema de filas, nós o fazemos de forma única para todos os clientes de um mesmo tipo, independentemente da sua particular identidade. As entidades são ainda classificadas em permanentes, quando "existem" ao longo de todo o período simulado, ou em temporárias, quando sua "existência" é restrita a parte do período simulado. Finalmente, uma entidade pode ter um ou mais atributos a ela associados.

O passo inicial na construção de um DCA consiste na identificação das entidades a serem representadas. A seguir, descreve-se o ciclo de vida de cada tipo de entidade, ciclo este que traduz as várias etapas percorridas por esta entidade. Num DCA, este ciclo é descrito por uma seqüência alternada de atividades e filas.

Atividade define um estado ativo durante o qual uma ou mais entidades estão engajadas por um período pré-determinado de tempo. A duração de uma atividade pode ser constante; no entanto, ela é geralmente definida como uma variável aleatória descrita por uma distribuição de probabilidades supostamente conhecida. A duração específica de uma particular atividade é determinada por um processo de amostragem, antes do início desta atividade. Durante a execução de uma atividade, as entidades nela envolvidas ficam temporariamente

indisponíveis.

Num DCA, uma atividade é representada por um retângulo. Cada entidade que participa de uma atividade define uma entrada e uma saída do retângulo. A figura 1 mostra a representação de uma atividade denominada "MANUTENÇÃO", da qual participam duas entidades: "Sonda" e "Equipe de manutenção".

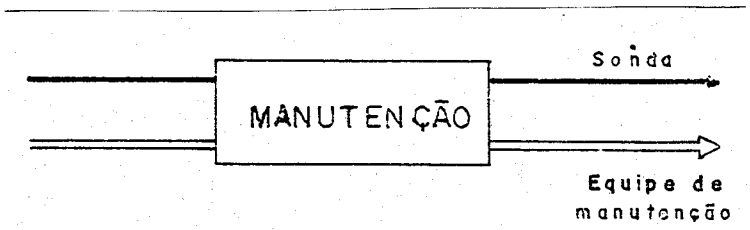

Figura 1. Representação da atividade "MANUTENÇÃO", envolvendo as entidades "Sonda" e "Equipe de manutenção".

Fila representa o estado passivo, ou de espera, de uma entidade enquanto aguarda condições favoráveis para participar de uma atividade. Uma importante restrição do DCA é que uma fila é exclusiva de uma classe de entidades, não sendo permitida a presença de diferentes tipos de entidade numa mesma fila. Outra importante diferença em relação a uma atividade é que o tempo de permanência numa fila não é pré-determinado; somente com a execução da simulação é que este tempo ficará determinado. Num DCA, é obrigatória a alternância entre atividades e filas. Há casos em que esta regra torna necessária a definição de filas artificiais ou **"dummy"**, nas quais a espera de fato não ocorre.

Num DCA, uma fila é representada por um círculo. A figura 2

mostra a representação de uma fila denominada "ESP_MNT" (Espera Manutenção), exclusiva da entidade "Sonda".

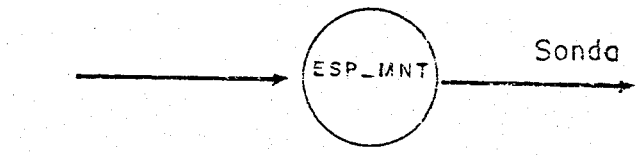

Figura 2. Representação da fila "ESP_MNT" (Espera Manutenção), exclusiva da entidade "Sonda".

Exemplo de DCA

A figura 3 mostra o DCA do problema da manutenção de sondas, cujo enunciado se segue:

Uma empresa opera 7 sondas de perfuração num campo petrolífero. As sondas estão em operação contínua, interrompendo seu funcionamento apenas para reparos de manutenção corretiva. O tempo entre falhas é descrito por uma distribuição normal com média de 7 dias e desvio padrão de 1 dia. Por outro lado, os reparos são feitos por uma única equipe de manutenção e têm duração exponencialmente distribuída com média de 1 dia. Deseja-se simular este problema para avaliar o tempo que as sondas ficam paradas por falta de manutenção, assim como estimar a ocupação média da equipe de manutenção.

O DCA da figura 3 é composto do ciclo de vida das duas entidades

presentes no sistema: "Sonda" e "Equipe de Manutenção".

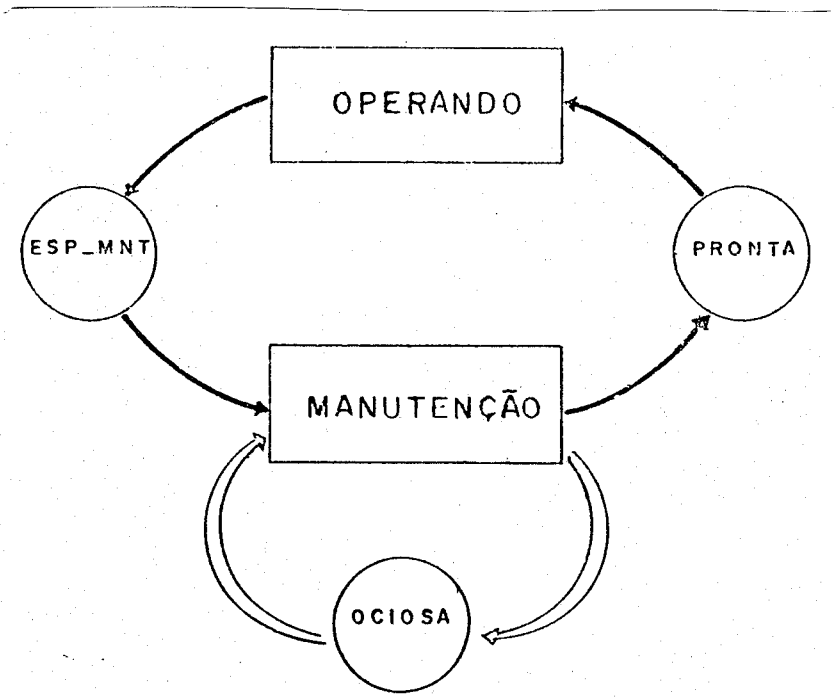

Figura 3. Diagrama do Ciclo de Atividades para o problema da manutenção de sondas de perfuração.

Ciclo de vida da sonda

Vamos supor que uma sonda encontra-se inicialmente em operação (atividade "OPERANDO"). Ao quebrar, esta sonda necessitará de manutenção. Ela aguardará na fila "ESP_MNT" até que sua manutenção possa ser iniciada. Segue-se a atividade "MANUTENÇÃO" que envolve, conjuntamente, a sonda e a equipe de manutenção. Concluída a manutenção, a sonda voltará imediatamente a operar. No entanto, como

o DCA exige que atividades e filas se alternem, incluímos a fila fictícia "PRONTA" entre as atividades "MANUTENÇÃO" e "OPERANDO". Observe, no entanto, que nenhuma sonda permanecerá parada nesta fila, passando imediatamnte à atividade "OPERANDO".

Ciclo de vida da equipe de manutenção

O ciclo de vida da equipe de manutenção é mais simples. Até que uma sonda quebre, a equipe permanecerá na fila "OCIOSA". A atividade "MANUTENÇÃO" é iniciada sempre que houver pelo menos uma sonda quebrada (na fila "ESP_MNT") e, concomitantemente, a equipe estiver ociosa. Ao término da manutenção, a equipe retornará à fila "OCIOSA", de onde sairá imediatamente caso haja, neste momento, outra sonda quebrada.

Além de mostrar os ciclos de vida de cada entidade do sistema, o DCA também explicita as condições de início das atividades. Por exemplo, a atividade "MANUTENÇÃO" somente será iniciada quando houver pelo menos uma sonda na fila "ESP_MNT" e quando a equipe estiver na fila "OCIOSA".

Outros elementos do DCA

Objetivando uma descrição mais abrangente, particularmente no caso de problemas mais complexos, outros elementos são também utilizados num DCA. São eles: as fontes/sumidouros, os desvios condicionais, as atribuições, as disciplinas das filas, as durações e prioridades das atividades, e as estatísticas a serem coletadas durante a simulação. Maiores detalhes a este respeito são encontrados em Oliveira (1989).

Limitações do DCA

Antes de concluir o estudo do DCA, cabe mencionar que, apesar da sua utilidade como linguagem simbólica para a descrição do comportamento dinâmico de um sistema, um DCA nem sempre é capaz de descrevê-lo por completo. Nestes casos, espera-se que o DCA proporcione uma descrição aproximada do sistema, servindo como um ponto de partida para o seu estudo. Assim, ficará a cargo do usuário incluir no modelo os detalhes do sistema não captados pelo DCA.

Método das três fases

Dentre as possíveis abordagens para a estruturação de programas de simulação (Pidd, 1984), o método das três fases é provavelmente aquele que conduz a programas mais modulares. Proposto por Tocher (1963), este método baseia-se no fato óbvio de que toda atividade é delimitada por dois eventos: o seu início e o seu término. Além disso, todo início de atividade é visto como um evento condicional. Já o término de uma atividade ocorrerá numa data pré-programada, uma vez que a duração de uma atividade será sempre conhecida tão logo ela se inicie.

De acordo com esta classificação, definem-se dois tipos de eventos associados a uma atividade: eventos B e eventos C.

Eventos B

Correspondem ao término das atividades e, portanto, têm sua data de ocorrência pré-determinada. Para efeito de programação com o SIMUL, será definido um evento B para cada entidade que participa de uma atividade. Por exemplo, a atividade "MANUTENÇÃO", da qual participam duas entidades, origina dois eventos B: o término da manutenção para a sonda e o término da manutenção para a equipe. Já a atividade "OPERANDO" origina um único evento B, correspondente ao término desta atividade para a única entidade que dela participa: a

sonda.

Eventos C

Correspondem ao início das atividades e, como tal, têm sua ocorrência condicionada à disponibilidade de entidades nas filas que precedem esta atividade. Para efeito de programação com o SIMUL, será definido um único evento C para cada atividade, ou seja, o evento C é comum para todas as entidades que participam de uma atividade. Por exemplo, o início da atividade "MANUTENÇÃO" origina um evento C envolvendo as entidades "Sonda" e "Equipe". Por outro lado, o início da atividade "OPERANDO" origina um evento C envolvendo unicamente a entidade "Sonda".

Executadas nesta mesma ordem, sob controle do programa executivo, as três fases que caracterizam o método são:

Fase A - Avanço do tempo. Consulta, na lista de eventos B, as datas de término das atividades em andamento. Determina o próximo evento B (o de menor data) e avança o relógio para esta data.

Fase B - Execução dos eventos B programados para a data em que o relógio foi avançado. Neste momento, a atividade correspondente é concluída, liberando as entidades nela envolvidas que são então transferidas para as filas subseqüentes.

Fase C - Verificação, para cada uma das atividades, se as condições para o seu início são satisfeitas, ou seja, teste para a execução dos eventos tipo C. Caso passe no teste, a atividade é iniciada: novos eventos B são programados e as entidades que dela participam deixam a fila em que se encontram. Para cada atividade, o teste para seu início é repetido indefinidamente, até que não seja mais possível iniciá-la. A ordem em que o início de cada atividade é testado tem relevância. É esta ordem que define a

prioridade entre atividades que "disputam" uma mesma entidade para o seu início.

Não havendo mais atividades que possam ser iniciadas num particular momento, a fase C é concluída. Em seguida, caso a execução do programa não seja interrompida e desde que a corrida não chegue ao seu final, retorna-se à fase A do processo, repetindo-se indefinidamente o ciclo das 3 fases.

A figura 4 ilustra a estrutura de um programa construído segundo o método das três fases, juntamente com outros elementos da simulação, como a Definição do Modelo, a Inicialização e a Emissão de Relatório. Observe-se que, antes de passar ao controle do executivo, é necessário que haja pelo menos uma entrada na lista de

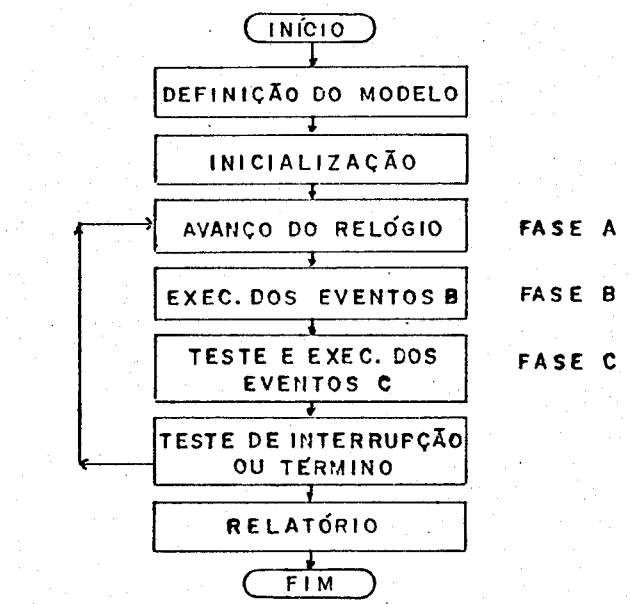

Figura 4. Estrutura de um programa de simulação utilizando um executivo de três fases.

eventos B (ou seja, pelo menos uma atividade em andamento), caso contrário o relógio não teria para onde avançar, levando o programa a uma condição de erro. A maneira mais simples e elegante para se resolver este problema consiste em iniciar a simulação com todas as entidades permanentes em filas e, ao término da etapa de inicialização, incluir uma execução da fase C. Com isso, as atividades iniciais seriam deflagradas.

No caso particular do SIMUL, dois eventos especiais são também programados durante a etapa de inicialização: um que define o término da corrida; o outro, opcional, define o término do período de aquecimento ("**Warm-up period**"), removendo assim possíveis efeitos das condições iniciais no resultado da simulação (Pidd, 1984).

O método das três fases e o diagrama do ciclo de atividades constituem um "feliz casamento" de duas metodologias que se integram perfeitamente. Pode-se dizer que a estruturação das três fases mantém uma correspondência quase perfeita com o DCA. Os términos de atividade correspondem à fase B, enquanto que os inícios à fase C. Com isso, uma vez construído um DCA, tem-se praticamente estruturado o programa de simulação correspondente. Esta propriedade leva a uma agilização do processo de modelagem e programação de simulações a eventos discretos. Ela é explorada pelo SIMUL, sistema que descrevemos a seguir.

3. O SISTEMA SIMUL

O Sistema SIMUL, em sua versão programável, compreende basicamente dois módulos ou arquivos em TURBO-PASCAL:

(a) a biblioteca de rotinas de simulação, definida pelos arquivos SIMUL_A.PAS e SIMUL_B.PAS;

(b) o programa executivo, identificado como o arquivo EMBRANCO.PAS. Este arquivo serve de base para todos os programas desenvolvidos com o SIMUL. Sua listagem é apresentada no Apêndice A.

A utilização do SIMUL, esquematizada na figura 5, inicia-se com o usuário construindo um DCA para o problema em estudo. A seguir, com base neste DCA, e observando as regras do SIMUL, o usuário editará o arquivo EMBRANCO.PAS. Esta edição do programa, bem como todo o trabalho computacional com o sistema SIMUL, é feita dentro do ambiente de trabalho do TURBO-PASCAL. Como resultado da edição do EMBRANCO.PAS, ter-se-á um programa de simulação denominado, de forma genérica, MODELO.PAS. No caso, em lugar de MODELO, o usuário definirá um nome que melhor se ajustar ao seu particular problema. A etapa seguinte consiste na compilação e execução do programa MODELO.PAS, utilizando-se para isso das rotinas pré-definidas na biblioteca SIMUL (SIMUL_A.PAS e SIMUL_B.PAS).

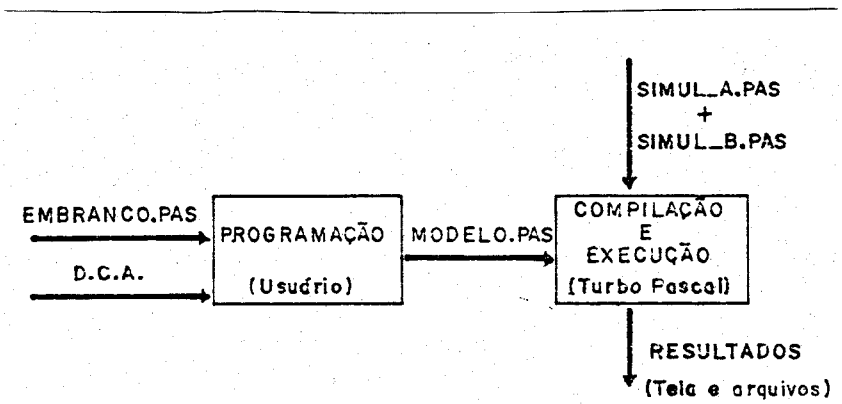

Figura 5. Diagrama ilustrando a utilização do Sistema SIMUL.

A seguir, descreveremos as unidades que compõem a biblioteca SIMUL, assim como alguns dos principais procedimentos.

Posteriormente, mostraremos como preencher o arquivo EMBRANCO.PAS.

Unidades do SIMUL

Com mais de duas mil e quinhentas linhas de instruções, e um tamanho aproximado de 70 Kbytes, o SIMUL é o coração do sistema. Composto de duas unidades de programa (SIMUL_A.PAS e SIMUL_B.PAS), o SIMUL compreende 6 unidades lógicas. São elas:

(a) Entidades;
(b) Filas;
(c) Atividades;
(d) Amostragem;
(e) Estatíticas;
(f) Relatórios e telas.

Descrevemos, a seguir, as funções de cada unidade e algumas de suas rotinas. Maiores detalhes sobre as rotinas disponíveis, seus parâmetros e uso são fornecidos no Dicionário do SIMUL (Pimentel e Saliby, 1990), uma publicação de grande utilidade para usuários do SIMUL.

Entidades

Define e controla as entidades e seus atributos. Faz uso, transparente para o usuário, de um conjunto complexo e sofisticado de ponteiros e listas encadeadas. Exemplos de procedimentos desta unidade são:

DEFINE_ENTIDADE: para definição de entidades do modelo.

PREPARA_B: associa, por um tempo pré-determinado, uma entidade a uma atividade.

DEFINE_ATRIBUTO: para a definição de atributos das entidades.

Filas

Provê facilidades para a organização e manipulação de filas. Inclui a definição e uso de fontes para entidades temporárias. Somente as disciplinas de atendimento FIFO e LIFO estão disponíveis, mas outras disciplinas poderão ser também implementadas pelo usuário. Alguns dos seus procedimentos são:

DEFINE_FILA: para definição de uma fila do modelo.

COLOCA_NA_FILA: transfere uma entidade para uma fila. Usada quando da execução de eventos B, ao término de uma atividade.

TIRA_DA_FILA: função que retira uma entidade de uma fila, para, em seguida, ocupá-la numa atividade. Usada quando da execução de eventos C, ao início de uma atividade.

ENCHE_FILA: coloca grupos de entidades permanentes numa fila. Usada na etapa de inicialização.

TAM_FILA: função que fornece o número de entidades presentes numa fila. Usada, principalmente, nos eventos C para teste de início de atividades.

DEFINE_FONTE: para definição de uma fonte. Uma fonte é uma fila especial, de tamanho potencialmente "infinito", utilizada para a criação de entidades temporárias. Uma fonte serve como origem para as entidades que, do mundo exterior, chegam ao sistema.

Atividades

Controla a execução das atividades e, em conjunto com o programa executivo, é responsável pelo avanço do relógio da simulação. Alguns dos seus procedimentos são:

DEFINE_ATIVIDADE: define uma atividade do modelo.

PROGRAMA_ATV: programa o término de uma atividade, gerando uma nova entrada na lista de eventos. Esta lista de eventos futuros é organizada na forma de uma árvore binária.

PROXIMO_TEMPO: função que retorna a data do próximo evento tipo B (término de atividade).

Amostragem

Provê facilidades para a geração de valores amostrais para diferentes distribuições de probabilidades, inclusive distribuições empiricamente definidas. Consiste de um conjunto de rotinas para a amostragem aleatória simples e outro para a amostragem descritiva.

O SIMUL tem um gerador próprio de números aleatórios utilizando o método congruencial linear. Ao todo, dispõe-se de 20 diferentes seqüências aleatórias, definidas pela escolha de uma particular semente. As sementes são numeradas de 1 a 20.

Alguns procedimentos e funções desta unidade são:

Amostragem aleatória simples

RND: função que retorna um "número aleatório" do tipo real, definido no intervalo unitário.

CONTINUA e GERA_CONTINUA: definição de distribuição empírica contínua e geração de valores.

DISCRETA e GERA_DISCRETA: definição de distribuição empírica discreta e geração de valores.

NORMAL, NEGEXP (exponencial), POISSON, ERLANG, LOGNORMAL, TRIANGULAR, UNIFORME, INTEIRA UNIFORME, BERNOULLI e WEIBULL: para a geração de valores aleatórios para estas distribuições.

Amostragem descritiva

INICIA_AD: reinicializa o processo de permutação aleatória.

VALOR_AD: função que fornece um valor descritivo para a distribuição especificada pelo seu parâmetro de chamada.

DEF_SET_CONT_EMPIRICAL: define um conjunto descritivo para uma distribuição empírica contínua.

DEF_SET_DISC_EMPIRICAL: define um conjunto descritivo para uma distribuição empírica discreta.

DEF_SET_NORMAL: define um conjunto descritivo normal.

DEF_SET_NEGEXP: define um conjunto descritivo exponencial.

DEF_SET_UNIF: define um conjunto descritivo uniforme.

DEF_SET_BERNOULLI: define um conjunto descritivo Bernoulli.

Estatísticas

Provê facilidades para a coleta de dados ao longo da simulação e apresentação de sumários estatísticos.

São disponíveis dois tipos de registro para os valores observados: individualmente ou integrados ao longo do tempo. Alguns procedimentos desta unidade são:

DEFINE_HIST: para definição de um histograma associado a uma variável de interesse. Usada na definição do modelo.

ACUMULA_DADOS: para o registro de um valor observado num particular histograma. Para estatísticas de tamanho de fila (do tipo integrado), este registro deve ser feito imediatamente antes de se alterar a fila. Assim, nestes casos, o ACUMULA DADOS é utilizado antes de se tirar e antes de se colocar uma entidade na fila em observação. Para estatísticas do tempo de espera de uma entidade numa fila (do tipo simples), é também necessária a definição de um atributo para esta entidade. Utiliza-se este atributo para se anotar o instante em que a entidade entra na fila. O tempo de espera é determinado e registrado no respectivo histograma quando a entidade deixa a fila. Ele é dado pela diferença entre o relógio da simulação e o valor deste atributo.

Relatórios e telas

Esta unidade destina-se à apresentação de resultados na tela e à produção de relatórios de saída que são também gravados na forma de um arquivo texto. As telas, que servem como interface sistema-usuário, são também parte desta unidade. Alguns de seus procedimentos são:

TITULO: Define um título para o relatório da simulação.

ESTATISTICAS_E_HISTOGRAMAS: apresenta na tela e também grava arquivo com sumários estatísticos e histogramas das variáveis de interesse. Usada ao término da corrida.

DEFINE_DADOS: define a duração da corrida, o período de aquecimento e o tipo de acompanhamento da simulação pelo usuário. Usada ao início de cada corrida.

Gerenciamento da simulação

Embora não caracterize uma unidade à parte do SIMUL, há ainda um conjunto de procedimentos voltados para o avanço do tempo e a manutenção da lista de eventos. Sua utilização é automática pelo SIMUL, não requerendo nenhuma atenção da parte do usuário.

4. ESTRUTURA DO PROGRAMA DE SIMULAÇÃO

A estrutura de um programa de simulação utilizando o SIMUL é definida pelo módulo EMBRANCO.PAS. Este arquivo, como já citamos, é preenchido pelo usuário com os dados específicos do problema em estudo. O EMBRANCO.PAS (ver Apêndice A) compõe-se de diferentes blocos de procedimentos, muitos dos quais preenchidos pelo usuário. Este preenchimento é feito através do editor de textos do TURBO-PASCAL, tendo sido nosso objetivo torná-lo o mais simples possível.

A figura 6 apresenta os blocos do EMBRANCO.PAS, assinalando com um asterisco aqueles que o usuário irá alterar. Os demais, deverão ser mantidos como tal, sem qualquer modificação. O seu

preenchimento, para o exemplo das Sondas, é apresentado na seção seguinte.

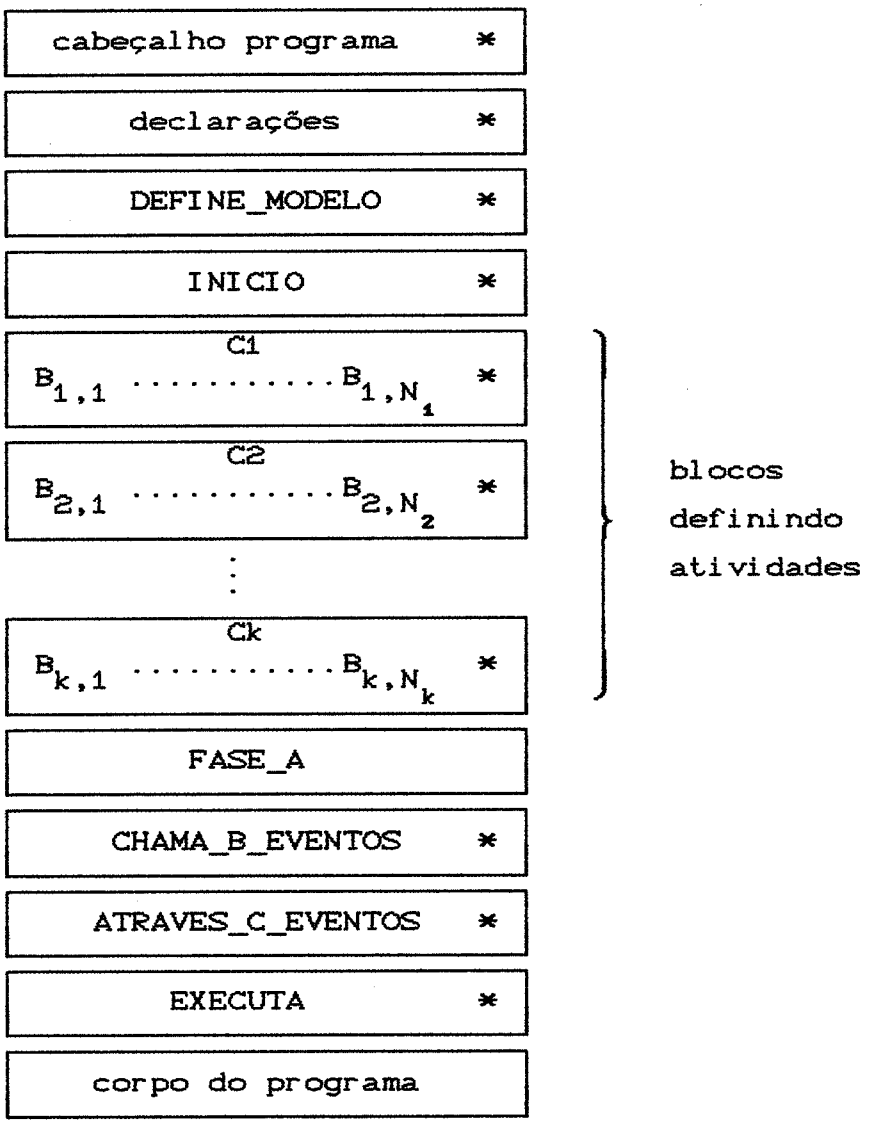

Figura 6. Blocos do EMBRANCO.PAS. Um (*) indica os blocos a serem preenchidos pelo usuário, com os dados específicos do seu problema.

Em essência, o trabalho de preenchimento do programa concentrar-se-á nas declarações, e nas rotinas de definição do

modelo e de especificação dos eventos B e C.

Quanto à estrutura geral do programa, ele pode ser dividido em:

(a) uma etapa de definições. Compreende o cabeçalho do programa, as declarações específicas do problema e o procedimento DEFINE MODELO.

(b) uma etapa de inicialização onde são inicializadas as estatísticas e sementes do gerador aleatório, definidas as condições iniciais de operação, a duração da corrida e o tipo de acompanhamento da simulação pelo usuário. Fazem parte desta etapa, os blocos DEFINE MODELO, INICIO e EXECUTA.

(c) o executivo da simulação, que controla o seu andamento de acordo com o método das três fases. Define os eventos B e C, assim como o mecanismo de controle. Fazem parte do executivo, os blocos FASE_A, CHAMA_B_EVENTOS, ATRAVES_C_EVENTOS e os procedimentos que definem os eventos C e B, associados a cada uma das atividades.

(d) uma etapa para apresentação de resultados. Além de uma saída padrão com um resumo geral da corrida, são também apresentados os histogramas pedidos pelo usuário. O procedimento RELATORIO é chamado no bloco EXECUTA.

5 EXEMPLO DE PROGRAMAÇÃO COM O SIMUL

O problema das Sondas, anteriormente descrito, ilustrará o uso do SIMUL. De acordo com o DCA da figura 3, o modelo compreende duas entidades (SONDA e EQUIPE DE MANUTENCAO) e duas atividades (OPERANDO e MANUTENCAO).

Assim sendo, lembrando que é definido um evento B para cada entidade que participa de uma atividade, os eventos C e B do modelo são:

C_OPERANDO	:	Início da operação (somente entidade SONDA)
B1	:	Término da operação para a entidade SONDA
C_MANUTENCAO	:	Início da manutenção (para todas entidades)
B2	:	Término da manutenção para EQUIPE DE MANUTENCAO
B3	:	Término da manutenção para SONDA

As informações do DCA são utilizadas para o preenchimento de EMBRANCO.PAS, resultando no programa SONDAS.PAS cuja listagem é apresentada no apêndice B.

A seguir, descreveremos o preenchimento de cada um dos blocos do programa.

(1) **Definir o nome do programa.**

```
program SONDAS;
```

(2) **Declarar Variáveis do Programa.** Todos os elementos do DCA (entidades, filas e atividades) e os histogramas são identificados. A identificação dos atributos é feita internamente, não sendo explicitada no programa.

```
var
  SISTEMA,EQP_MNT,SONDA                : Entidade;
  OCIOSA,PRONTA,ESPERA_MNT             : Fila;
  OPERANDO,MANUTENCAO                  : Atividade;
  TEMPO_FILA_HIST,TAM_FILA_HIST,
  OPERACAO_HIST,MANUTENCAO_HIST        : Histograma;
```

(3) **Preencher a rotina DEFINE_MODELO.** Este é um módulo que deve ser

preenchido com muito cuidado pelo usuário, pois nele são definidos todos os elementos que compõem o modelo. Neste ponto, as entidades e seus atributos, filas e fontes, atividades e histogramas têm suas estruturas de dados criadas.

Cada elemento aqui definido (exceção feita aos atributos), está associado a um identificador previamente declarado no programa (item 5.2). Além deste identificador, um nome, definido por uma "**string**", é também associado a cada elemento. Embora não seja necessário, é aconselhável que este nome e o identificador coincidam. Assim procedendo, resulta num programa de mais fácil compreensão.

```
procedure Define_Modelo;
begin
  Define_Nome_Problema('SONDAS');
  Define_Entidade(SISTEMA,'SISTEMA');
  Define_Entidade(EQP_MNT,'EQP_MNT');
  Define_Entidade(SONDA,'SONDA');
  Define_Fila(EQP_MNT,OCIOSA,'OCIOSA');
  Define_Fila(SONDA,PRONTA,'PRONTA');
  Define_Fila(SONDA,ESPERA_MNT,'ESPERA_MNT');
  Define_Hist(TEMPO_FILA_HIST,'T_ESP_MNT',Simples,0.5,0.5);
  Define_Hist(TAM_FILA_HIST,'F_ESP_MNT',Integrado,1,0.5);
  Define_Hist(OPERACAO_HIST,'Temp_Opera',Simples,0.5,4);
  Define_Hist(MANUTENCAO_HIST,'Temp_Manut',Simples,0.3,0);
  Define_Atributo(SONDA,'data_avaria');
  Define_Atividade(OPERANDO,'OPERANDO');
  Define_Atividade(MANUTENCAO,'MANUTENCAO');
end;
```

A rotina DEFINE_HIST é a única que necessita de alguma explicação. Os dois parâmetros iniciais se destinam ao encadeamento dos histogramas e à sua identificação no relatório; o próximo define o tipo de histograma e os dois últimos estabelecem, respectivamente, a largura das células e o valor a partir do qual o histograma será construído.

Para se identificar uma particular característica de uma entidade ou do sistema, em DEFINE_MODELO é criado um atributo

através da rotina DEFINE_ATRIBUTO.

(4) **Preencher a rotina INICIO.** Após chamar SET_SIST_ENT, são incluídas as chamadas à rotina ENCHE_FILA, para que todas as entidades permanentes, como condição inicial, sejam colocadas em filas . É desta forma, pois, que as entidades permanentes do modelo têm sua quantidade determinada

```
procedure Inicio;
begin
  InicioVars;
  Set_Sist_Ent(SISTEMA);
  Enche_Fila(OCIOSA,1,1);
  Enche_Fila(PRONTA,1,7);
  Prepara_Termino(999,DURACAO);
  Cria_Registros;
end;
```

Nota: Nos programas gerados automaticamente, o preenchimento desta rotina é outro. Neste caso, os dados iniciais das entidades são lidos de um arquivo do tipo *.FIL (ex.: SONDAS.FIL), e a rotina CHAMA_ENCHE_FILA, no início da execução da simulação, coloca as entidades nas respectivas filas.

(5) **Rotinas C.** Para cada atividade, define-se a rotina C que caracteriza seu início. Uma atividade somente pode começar quando todas as entidades nela envolvidas estiverem disponíveis nas filas precedentes; portanto, as rotinas C possuem a restrição inicial de que as filas (não as fontes) precedentes tenham pelo menos uma entidade. É atribuída a duração da atividade à variável global TEMPO_ATV. A programação da atividade é feita através da rotina PROGRAMA_ATV, seguida pela chamada da rotina PREPARA_B, tantas vezes quantas forem as entidades envolvidas nesta atividade, e, por último, pela rotina

FIM_PROGRAMA_ATV.

Por ex.:
```
procedure C_MANUTENCAO;
begin
  while ( Tam_Fila(OCIOSA) >= 1)
    and ( Tam_Fila(ESPERA_MNT) >= 1)
  do
  begin
    Tempo_Atv:=NEGEXP(1.0,11);
    Acumula_Dados(MANUTENCAO_HIST,Tempo_Atv);
    Acumula_Dados(TAM_FILA_HIST,Tam_Fila(ESPERA_MNT));
    Programa_Atv(MANUTENCAO);
      Prepara_B(2,Tira_da_Fila(frente,OCIOSA));
      Prepara_B(3,Tira_da_Fila(frente,ESPERA_MNT));
    Fim_Programa_Atv;
  end;
end;
```

(6) **Rotinas B.** Para cada tipo de entidade participante de uma atividade, ou seja, para cada PREPARA_B corresponde uma rotina B. É nesta rotina que, ao término da atividade, uma entidade (ENT_CORRENTE) é liberada e colocada na fila apropriada, de acordo com o seu ciclo de vida.

Nota: Para uma maior clareza e modularidade do programa é aconselhável que cada rotina C seja seguida pelas rotinas B correspondentes à atividade em questão. O programa listado no apêndice B, e também os programas gerados automaticamente, seguem este padrão.

Por ex.:
```
procedure B2_Fim_MANUTENCAO_EQP_MNT;
begin
  Coloca_na_Fila(Ent_corrente,Atras,OCIOSA);
end;

procedure B3_Fim_MANUTENCAO_SONDA;
begin
  Coloca_na_Fila(Ent_corrente,Atras,PRONTA);
end;
```

(7) **Preencher a rotina CHAMA_B_EVENTOS.** Listar todos os procedimentos B, fazendo-se a correspondência do valor do CASE com o parâmetro de PREPARA_B.

```
procedure Chama_B_Eventos;
begin
  while ( Tempo_Corrente(TEMP) ) do
  begin
    while (Prox_B_Evento) do
    begin
      case Nr_ProxB of
          1 : B1_Fim_OPERANDO_SONDA;
          2 : B2_Fim_MANUTENCAO_EQP_MNT;
          3 : B3_Fim_MANUTENCAO_SONDA;
        998 : Fim_Aquec;
        999 : Fim_Corrida;
      end;
    end;
  end;
end;
```

(8) **Preencher a rotina ATRAVES_C_EVENTOS.** São listados todos os procedimentos C. É a ordem nesta lista que estabelece a prioridade de início das atividades.

```
procedure Atraves_C_Eventos;
begin
  C_OPERANDO;
  C_MANUTENCAO;
end;
```

(9) **Preencher a rotina EXECUTA.** Opcionalmente pode-se adicionar um título para o relatório da simulação através da rotina TITULO.

(10) **Coleta de Dados Estatísticos.** Havendo histogramas, surge o problema de como e quando coletar os dados estatísticos. São apresentadas as seguintes regras:

(a) Se o histograma é do tipo "simples", para dados que possuem valor definido ao longo da simulação, a coleta realiza-se numa rotina C ou B. Ela é feita pela rotina ACUMULA_DADOS, tão logo o valor esteja disponível. Histogramas de atributos e histogramas de tempo de espera enquadram-se neste caso.

(b) Se o histograma é do tipo "integrado", destinado à análise de comprimento de fila, a rotina ACUMULA_DADOS deve ser chamada imediatamente antes de qualquer alteração do tamanho da fila, seja pela retirada ou pela inclusão de uma entidade, isto é, antes da chamada das rotinas TIRA_DA_FILA ou COLOCA_NA_FILA.

(11) Ao ser executado um programa de simulação, uma tela inicial é apresentada para que as opções de saída sejam definidas. Sendo solicitados resultados em vídeo e/ou arquivo, todos os passos da simulação serão apresentados, implicando em maior tempo computacional. Na tela seguinte são definidos a duração, o tempo de aquecimento e a velocidade da simulação.

(12) No Apêndice C são apresentados resultados completos de uma corrida de simulação, com duração de 10 anos (3650 dias) e sem período de aquecimento. Inicialmente é fornecido um sumário geral da corrida, seguindo-se as estatísticas e histogramas das variáveis de interesse.

(13) No Apêndice D são apresentados resultados parciais de uma corrida de simulação, considerando a existência de duas equipes de manutenção. O programa com duas equipes é obtido simplesmente pela alteração de uma linha do programa original. Na rotina INICIO, ao se definir o número de entidades, a fila

OCIOSA passa a receber duas equipes:
Enche_Fila(OCIOSA,1,2) .

6. CONCLUSÕES E CONSIDERAÇÕES FINAIS

Não se espera, naturalmente, que esta breve descrição do SIMUL seja suficiente para que o leitor tenha condições de modelar e executar programas de simulação de natureza mais complexa, sem consultar os demais materiais de referência do SIMUL. No entanto, acreditamos que o trabalho permitiu chegar próximo a este ponto. Por razões de clareza e, também, de espaço, restringimo-nos a um exemplo simples que não ilustra por completo os recursos disponíveis no SIMUL.

Assim, deixamos para outros trabalhos, tópicos adicionais como o uso de fontes, atributos e desvios condicionais, e a utilização automática da amostragem descritiva em lugar da amostragem aleatória simples.

Embora o conhecimento de PASCAL seja um pré-requisito para uso do SIMUL, o programa de simulação é de tal forma estruturado e claro que a "leitura" do programa do Apêndice B é fácil, mesmo para os não familiarizados com esta linguagem de programação. Cabe repetir que a modularidade do programa de simulação é uma das principais vantagens que o SIMUL tem a oferecer. Esta modularidade, em conjunto com a clareza do código fonte, simplifica a manutenção do programa. Isto possibilita que o modelo de simulação seja também construído de maneira modular, utilizando-se assim uma abordagem de modelagem evolutiva, que é sempre recomendável.

A experiência de uso do SIMUL, seja no ambiente acadêmico e de pesquisa, seja no meio empresarial, tem sido extremamente

gratificante. O sistema já foi exaustivamente testado. Apesar deste fato não garantir a inexistência de "bugs" no sistema, ele se apresenta extremamente confiável. Cópias dos programas e do material de referência são fornecidos pelos autores do trabalho, sendo cobrado um preço meramente simbólico no caso de instituições públicas de ensino e/ou pesquisa.

A experiência acumulada com o SIMUL levou-nos a promover novos desenvolvimentos, aperfeiçoando o sistema. Merecem destaque a interface de modelagem, o gerador automático de programas e a interface gráfica para animação da simulação. Estes módulos, juntamente com outros ainda em desenvolvimento, serão oportunamente divulgados.

REFERÊNCIAS

ANGULO, I.E. New facilities for combined simulation modelling. Lancaster: Universidade de Lancaster, 1983. Tese de Doutorado.

BALMER, D.; PAUL, R.J. CASM - The right environment for simulation. Journal of the Operational Research Society, v. 37, n. 5, p. 443-452, May 1986.

CARVALHO, R.S. Cellular simulation. Lancaster: Universidade de Lancaster, 1975. Tese de Doutorado.

CROOKES, J.G. Simulation using Turbo Pascal. Lancaster: Universidade de Lancaster, 1985 (Working Paper).

———————— et al. A three-phase simulation system written in Pascal. Journal of the Operational Research Society, v. 37, n. 6, p. 603-618, June 1986.

DAVIES, R.; O'KEEFE, R. Simulation modelling with Pascal. Hemel Hempstead: Prentice Hall, 1989.

OLIVEIRA, G.C. Modelagem interativa e geração automática de programas de simulação a eventos discretos. Rio de Janeiro: IME/RJ, 1989. Tese de Mestrado.

PAUL, R.J. Simulation modelling: the CASM project. London: London School of Economics, Department of Information Systems, 1988. (Working Paper, 18)

PIDD, M. Computer simulation in management science. Chichester: J. Wiley, 1984.

PIMENTEL, M. Sistema computacional para simulação discreta com opção de uso da amostragem descritiva. Rio de Janeiro: IME/RJ, 1989.

Tese de Mestrado.

PIMENTEL, M.; SALIBY, E. Dicionário do SIMUL. Rio de Janeiro: COPPEAD/UFRJ, 1990. (Em preparação)

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

———————— ; OLIVEIRA, G.C. GERSIMUL: um gerador automático de programas de simulação em TURBO PASCAL. Rio de Janeiro: COPPEAD/UFRJ, 1989. (Relatório Técnico, 131)

SYSPACK LTD. The VS6 user's guide. London: Syspack, 1988.

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

APÊNDICE A

ARQUIVO EMBRANCO.PAS

```
(*====  EMBRANCO.PAS/SIMUL em TURBO Pascal para IBM-PC, Versao 2.0 ====*)

PROGRAM NOMEPRG;        (* escolha o nome *)


Uses
  Crt,
  Simul_A,
  Simul_B;

(* defina as entidades, filas, fontes, atividades e histogramas *)

VAR
   SISTEMA    : ENTIDADE;
(* NOMEENTS   : ENTIDADE;
   NOMEFILAS  : FILA;
   NOMEFONTES: FONTE;
   NOMEATVS   : ATIVIDADE;
   NOMEHISTS  : HISTOGRAMA; *)


PROCEDURE DEFINE_MODELO;
BEGIN
  DEFINE_NOME_PROBLEMA('NOME_PROB');
  DEFINE_ENTIDADE(SISTEMA,'SISTEMA');
  {Definir entidades, atributos, filas, fontes, atividades e histogramas}
END;


PROCEDURE INICIO;
BEGIN
  INICIOVARS;
  SET_SIST_ENT(SISTEMA);
(*ENCHE_FILA(FILA_1,PRIMEIRO,ULTIMO)
  ENCHE_FILA(FILA_2,PRIMEIRO,ULTIMO)   *)
  PREPARA_TERMINO(999,DURACAO);
  CRIA_REGISTROS;
END;


PROCEDURE C1;
BEGIN
END;


PROCEDURE C2;
BEGIN
END;
```

```
(* O limite para o numero de B eventos depende da memoria disponivel;
   Sao B eventos reservados:
   998, usado no PERIODO DE AQUECIMENTO (procedure FIM_AQUEC)
   999, usado no TERMINO DA CORRIDA (procedure FIM_CORRIDA)

   ATENCAO: Ajustar a declaracao CASE na  procedure CHAMA_B_EVENTOS *)


PROCEDURE B1;
BEGIN
END;

PROCEDURE B2;
BEGIN
END;

PROCEDURE B3;
BEGIN
END;



FUNCTION FASE_A:BOOLEAN;
BEGIN
   AUMENTA_TEMPO;
   FASE_A := (TEMP <= DURACAO);
END;



PROCEDURE CHAMA_B_EVENTOS;
BEGIN
  WHILE (TEMPO_CORRENTE(TEMP)) DO
  BEGIN
    WHILE (PROX_B_EVENTO) DO
    BEGIN
      CASE NR_PROXB OF
          1 : B1;
          2 : B2;
          3 : B3;
        998 : FIM_AQUEC;
        999 : FIM_CORRIDA;
      END;
    END;
  END;
END;



PROCEDURE ATRAVES_C_EVENTOS;
BEGIN
   C1;
   C2;
END;
```

```
PROCEDURE EXECUTA;
BEGIN
  OPEN_FILE(NOME_PROB + '.RST');
  INICIO;
  ATRAVES_C_EVENTOS;
  IF (DUR_AQUECIMENTO <> 0) THEN PREPARA_REINICIO(998,DUR_AQUECIMENTO);
  WRITELN;
  WHILE FASE_A DO
  BEGIN
     CHAMA_B_EVENTOS;
     ATRAVES_C_EVENTOS;
  END;
(*TITULO('QUALQUER TITULO COM ATE 50 CHAR');*)
  RELATORIO;
  CLOSE_FILE(NOME_PROB + '.RST');
END;


BEGIN          (* programa principal *)
  INICIALIZA;
  DEFINE_MODELO;
  REPEAT
    DEFINE_DADOS;
    IF (DURACAO > 0) THEN EXECUTA;
  UNTIL (DURACAO <= 0);
  CLRSCR;
END.
```

APÊNDICE B

ARQUIVO SONDAS.PAS

```
(*****   Programa de Simulacao em Pascal 5.0 para uso do SIMUL.PAS   *****)

program SONDAS;

Uses
  Crt,
  Simul_A,
  Simul_B;


var
  SISTEMA,EQP_MNT,SONDA : Entidade;
  OCIOSA,PRONTA,ESPERA_MNT : Fila;
  OPERANDO,MANUTENCAO : Atividade;
  TEMPO_FILA_HIST,TAM_FILA_HIST,OPERACAO_HIST,MANUTENCAO_HIST : Histograma;



procedure Define_Modelo;
begin
  Define_Nome_Problema('SONDAS');
  Define_Entidade(SISTEMA,'SISTEMA');
  Define_Entidade(EQP_MNT,'EQP_MNT');
  Define_Entidade(SONDA,'SONDA');
  Define_Fila(EQP_MNT,OCIOSA,'OCIOSA');
  Define_Fila(SONDA,PRONTA,'PRONTA');
  Define_Fila(SONDA,ESPERA_MNT,'ESPERA_MNT');
  Define_Hist(TEMPO_FILA_HIST,'T_ESP_MNT',Simples,0.5,0.5);
  Define_Hist(TAM_FILA_HIST,'F_ESP_MNT',Integrado,1,0.5);
  Define_Hist(OPERACAO_HIST,'Temp_Opera',Simples,0.5,4);
  Define_Hist(MANUTENCAO_HIST,'Temp_Manut',Simples,0.3,0);
  Define_Atributo(SONDA,'data_avaria');
  Define_Atividade(OPERANDO,'OPERANDO');
  Define_Atividade(MANUTENCAO,'MANUTENCAO');
end;



procedure Inicio;
begin
  InicioVars;
  Set_Sist_Ent(SISTEMA);
  Enche_Fila(OCIOSA,1,1);
  Enche_Fila(PRONTA,1,7);
  Prepara_Termino(999,DURACAO);
  Cria_Registros;
end;
```

```
procedure C_OPERANDO;
begin
  while ( Tam_Fila(PRONTA) >= 1)
  do
  begin
    Tempo_Atv:=NORMAL(7,1,6);
    Acumula_Dados(OPERACAO_HIST,Tempo_Atv);
    Programa_Atv(OPERANDO);
      Prepara_BC1,Tira_da_Fila(frente,PRONTA));
    Fim_Programa_Atv;
  end;
end;

procedure B1_Fim_OPERANDO_SONDA;
begin
  Acumula_Dados(TAM_FILA_HIST,Tam_Fila(ESPERA_MNT));
  Def_Val_Atrib(Ent_Corrente,'data_avaria',TEMP);
  Coloca_na_Fila(Ent_corrente,Atras,ESPERA_MNT);
end;



procedure C_MANUTENCAO;
begin
  while ( Tam_Fila(OCIOSA) >= 1)
    and ( Tam_Fila(ESPERA_MNT) >= 1)
  do
  begin
    Tempo_Atv:=NEGEXP(1.0,11);
    Acumula_Dados(MANUTENCAO_HIST,Tempo_Atv);
    Acumula_Dados(TAM_FILA_HIST,Tam_Fila(ESPERA_MNT));
    Acumula_Dados(TEMPO_FILA_HIST,
                  TEMP - Avalia_Atrib(Prim_da_Fila(ESPERA_MNT),'data_avaria'));
    Programa_Atv(MANUTENCAO);
      Prepara_BC2,Tira_da_Fila(frente,OCIOSA));
      Prepara_BC3,Tira_da_Fila(frente,ESPERA_MNT));
    Fim_Programa_Atv;
  end;
end;

procedure B2_Fim_MANUTENCAO_EQP_MNT;
begin
  Coloca_na_Fila(Ent_corrente,Atras,OCIOSA);
end;

procedure B3_Fim_MANUTENCAO_SONDA;
begin
  Coloca_na_Fila(Ent_corrente,Atras,PRONTA);
end;



function Fase_A:boolean;
begin
  Aumenta_Tempo;
  Fase_A := (TEMP <= Duracao);
end;
```

```
procedure Chama_B_Eventos;
begin
  while ( Tempo_Corrente(TEMP) ) do
  begin
    while (Prox_B_Evento) do
    begin
      case Nr_ProxB of
          1 : B1_Fim_OPERANDO_SONDA;
          2 : B2_Fim_MANUTENCAO_EQP_MNT;
          3 : B3_Fim_MANUTENCAO_SONDA;
        998 : Fim_Aquec;
        999 : Fim_Corrida;
      end;
    end;
  end;
end;

procedure Atraves_C_Eventos;
begin
  C_OPERANDO;
  C_MANUTENCAO;
end;


procedure Executa;
begin
  Open_File(NOME_PROB+'.RST');
  Inicio;
  Atraves_C_Eventos;
  if ( Dur_Aquecimento <> 0 )
    then Prepara_Reinicio(998,Dur_Aquecimento);
  while (Fase_A) do
  begin
    Chama_B_Eventos;
    Atraves_C_Eventos;
  end;
  Titulo('SIMULACAO DO PROBLEMA DAS SONDAS USANDO AAS');
  Relatorio;
  Close_File(NOME_PROB+'.RST');
end;


begin          (* programa principal *)
  Inicializa;
  Define_Modelo;
  repeat
    Define_Dados;
    if ( Duracao > 0 ) then Executa;
  until ( Duracao <= 0 );
  clrscr;
end.
```

APÊNDICE C

RESULTADOS DE UMA CORRIDA COM 1 EQUIPE E DURAÇÃO DE 10 ANOS (3650 DIAS)

****
SIMULACAO DO PROBLEMA DAS SONDAS USANDO AAS
ESTATISTICAS E HISTOGRAMAS
****

Duracao da corrida : 3650

Aquecimento : 0

ATIVIDADE	Nr de vezes	Duracao	
		Media	D. Padrao
OPERANDO	2716	7.002	1.0054
MANUTENCAO	2710	1.021	1.0373

ENTIDADE	Quantidade	Utilizacao (%)
EQP_MNT	1	0.7577
SONDA	7	0.8516

FILA	Tamanho final	Entidade
OCIOSA	0	EQP_MNT
PRONTA	0	SONDA
ESPERA_MNT	0	SONDA

====== ESTATISTICAS BASICAS DE T_ESP_MNT ======

Media	=	1.3987
DP	=	1.8610
Variancia	=	3.4634
Minimo	=	0.0000
Maximo	=	12.0756
Soma de (freq*obs)	=	3790
Nr total de frequencias	=	2710

============HISTOGRAMA DE T_ESP_MNT ============

CELULA			FREQ.	
	<	0.5	1249	##
0.5	-	1.0	285	##########
1.0	-	1.5	262	#########
1.5	-	2.0	191	#######
2.0	-	2.5	148	######
2.5	-	3.0	129	####
3.0	-	3.5	109	###
3.5	-	4.0	84	###
4.0	-	4.5	49	##
4.5	-	5.0	46	#
5.0	-	5.5	42	#
5.5	-	6.0	33	#
6.0	-	6.5	15	
6.5	-	7.0	18	#
7.0	-	7.5	12	
7.5	-	8.0	7	
	=>	8.0	31	#

====== ESTATISTICAS BASICAS DE F_ESP_MNT ======

Media	=	1.0386
DP	=	1.2954
Variancia	=	1.6780
Minimo	=	0.0000
Maximo	=	6.0000
Soma de (freq*obs)	=	3790
Nr total de frequencias	=	3650

============HISTOGRAMA DE F_ESP_MNT ============

CELULA			FREQ.	
	<	0.5	1790	##
0.5	-	1.5	778	##################
1.5	-	2.5	519	############
2.5	-	3.5	338	########
3.5	-	4.5	164	####
4.5	-	5.5	56	#
5.5	-	6.5	3	
6.5	-	7.5	0	
7.5	-	8.5	0	
8.5	-	9.5	0	
9.5	-	10.5	0	
10.5	-	11.5	0	
11.5	-	12.5	0	
12.5	-	13.5	0	
13.5	-	14.5	0	
14.5	-	15.5	0	
	=>	15.5	0	

====== ESTATISTICAS BASICAS DE TEMP_OPERA ======

Media	=	7.0024
DP	=	1.0054
Variancia	=	1.0108
Minimo	=	3.9650
Maximo	=	10.5553
Soma de (freq*obs)	=	19018
Nr total de frequencias	=	2716

============HISTOGRAMA DE TEMP_OPERA ============

CELULA			FREQ.	
	<	4.0	1	
4.0	-	4.5	6	
4.5	-	5.0	45	###
5.0	-	5.5	133	##########
5.5	-	6.0	241	###################
6.0	-	6.5	432	##################################
6.5	-	7.0	535	##
7.0	-	7.5	487	######################################
7.5	-	8.0	381	##############################
8.0	-	8.5	252	####################
8.5	-	9.0	137	##########
9.0	-	9.5	44	###
9.5	-	10.0	18	#
10.0	-	10.5	3	
10.5	-	11.0	1	
11.0	-	11.5	0	
	=>	11.5	0	

====== ESTATISTICAS BASICAS DE TEMP_MANUT ======

Media	=	1.0207
DP	=	1.0373
Variancia	=	1.0759
Minimo	=	0.0005
Maximo	=	7.8322
Soma de (freq*obs)	=	2766
Nr total de frequencias	=	2710

============HISTOGRAMA DE TEMP_MANUT ============

CELULA			FREQ.	
	<	0.0	0	
0.0	-	0.3	684	##
0.3	-	0.6	521	##################################
0.6	-	0.9	384	#########################
0.9	-	1.2	289	###################
1.2	-	1.5	230	###############
1.5	-	1.8	144	#########
1.8	-	2.1	119	########
2.1	-	2.4	90	######
2.4	-	2.7	61	####
2.7	-	3.0	44	###
3.0	-	3.3	23	#
3.3	-	3.6	29	##
3.6	-	3.9	28	##
3.9	-	4.2	13	#
4.2	-	4.5	13	#
	=>	4.5	38	##

Termino do relatorio da simulacao .

APÊNDICE D

RESULTADOS PARCIAIS DE UMA CORRIDA COM 2 EQUIPES E DURAÇÃO DE 10 ANOS (3650 DIAS)

****
SIMULACAO DO PROBLEMA DAS SONDAS USANDO AAS
ESTATISTICAS E HISTOGRAMAS
****

Duracao da corrida : 3650

Aquecimento : 0

ATIVIDADE	Nr de vezes	Duracao	
		Media	D. Padrao
OPERANDO	3141	7.000	0.9973
MANUTENCAO	3134	1.019	1.0309

ENTIDADE	Quantidade	Utilizacao (%)
EQP_MNT	2	0.4376
SONDA	7	0.9849

FILA	Tamanho final	Entidade
OCIOSA	2	EQP_MNT
PRONTA	0	SONDA
ESPERA_MNT	0	SONDA