UMA LINGUAGEM PARA ESPECIFICAÇÃO DE REQUISITOS DE SOFTWARE DE TEMPO REAL

CLÁUDIA KILLER

SUELI CAVALCANTE

UFRJ - COPPE

Caixa Postal 68511

CEP 21945 - RIO DE JANEIRO - RJ

I. Introdução

Com o aumento da demanda de software, gerentes de projetos se deparam com desafios crescentes no sentido de concluir projetos dentro dos prazos e custos estimados.

Estima-se que 75% dos projetos de software encomendados não chegam a ser implantados. Em sua maioria nunca são concluídos e mesmo quando entregues, muitas vêzes não são usados (01).

A principal causa desse problema é clara. Produtos de software, muitas vêzes não atingem os requisitos que motivaram sua construção, não satisfazendo assim, as necessidades de seus usuários. Estudos tem mostrado (02) que aproximadamente 2/3 dos erros encontrados em produto de software, são causados por requisitos incorretos ou mau interpretados.

Torna-se, então, evidente que a qualidade de um produto de software depende da qualidade de sua especificação. Especificações são representações que descrevem o

comportamento esperado e os requisitos de um produto a ser construído. Tem como objetivo guiar o processo de desenvolvimento, estabelecer a comunicação entre as pessoas envolvidas no desenvolvimento (desenvolvedores, usuários, gerente, etc.), permitir o controle de qualidade dos produtos e auxiliar no entendimento do problema a ser resolvido (03).

II. Técnicas de Modelagem para Geração de Especificações

Cada requisito de software é a afirmação de uma característica do sistema a ser desenvolvido. Essas características devem ser apresentadas em dois níveis: Externo e Interno. O nível externo descreve a interação do software com seu ambiente e o nível interno descreve como se processa essa interação internamente.

Teoricamente, o nível externo deve ser definido em primeiro lugar. Porém na prática torna-se difícil conseguir uma separação clara, principalmente no início do processo de desenvolvimento.

Especificações de Requisitos são, muitas vêzes escritas informalmente (linguagem natural), o que dificulta sua compreensão e verificação. O uso de técnicas formais reduz a ambiguidade e permite verificação mesmo de sistemas complexos. Entretanto, esses métodos tem grandes restrições quanto à aplicabilidade, principalmente, enquanto carecerem de apoio automatizado.

Nos últimos anos, vários métodos tem sido propostos com o objetivo de solucionar os problemas e dificuldades encontradas ao gerar especificações de requisitos. Estes métodos estão baseados em técnicas de

modelagem de requisitos, que podem ser divididas em (04) (05) (06):

1. Especificação de E/S, que expressa o comportamento requerido pelo software como uma sequência de entradas e saídas. Como exemplo, temos:

   a) Sistemas de Informações (ênfase nas saídas necessárias);

   b) Aplicações Interativas (ênfase nas entradas correntes, necessárias à geração das saídas);

   c) Sistemas de Controle de Processos (além das entradas correntes necessita conhecer entradas anteriores para gerar as saídas correspondentes, que respondem a estímulos).

2. Uso de Conjunto de Exemplos Representativos, que expressa o comportamento requerido através de exemplos representativos desse comportamento. Nem sempre é possível exemplificar todos os casos, como também especificar completamente o comportamento do sistema, mas provêem uma boa idéia de E/S necessárias.

3. Especificação de Modelos, que expressa os requisistos sob forma de um modelo. Um modelo representa uma visão particular do problema ignorando intencionalmente alguns de seus aspectos. Muitas vêzes, se faz necessário a aplicação de vários modelos para expressar os requisitos do software. Os principais modelos usados são:

a) Modelo Funcional, dá ênfase as funções que devem ser executadas pelo software. A partir desse ponto de vista, descreve o comportamento do software. Essa descrição pode ser feita de três maneiras:

i) Operações/Eventos - descreve cada operação a ser executada e os eventos que a causaram, sem considerar sua sequência.

ii) Fluxo - descreve um conjunto de caminhos de operações, considerando sequência, concorrência, comunicação, etc.

iii) Hierarquia - descreve uma hierarquia de funções, que muitas vêzes inclui aspectos de projeto.

b) Modelo de Dados, dá ênfase a organizacao dos dados usados pelo software. Como exemplo, temos, ERA - modelo entidade, relação e atributo.

c) Modelo de Estados, dá ênfase a sequência de estados que caracterizam o comportamento do software em um determinado intervalo de tempo. Como exemplo, temos, Máquinas de Estados Finitos e Redes de Petri.

d) Modelo de Interação Homem-Máquina, dá ênfase as interfaces software x usuário. Como exemplo, temos, USE (User Software Engineering).

e) Modelo Matemático, usa relações matemáticas para descrever o comportamento do software.

III. LER: Automatização da Especificação de Requisitos

LER (Linguagem de Especificação de Requisitos) é uma linguagem gráfica para especificar requisitos de sistemas de tempo real. Está sendo desenvolvida na UFRJ-COPPE SISTEMAS como parte integrante de um trabalho que pretende criar um ambiente automatizado para desenvolvimento de sistemas de

tempo real.

LER é baseado na metodologia SREM (09), (10), (11) e na teoria de Sistemas de Balzer e Neil (12). Tem como objetivo prover uma ferramenta automatizada para auxiliar na elaboração de especificação de requisitos de forma a evitar ambiguidade, facilitar a comunicação entre cliente e o pessoal envolvido no projeto, garantir uma maior confiabilidade, permitir a especificação de requisitos nao funcionais, etc.

Para elaborar uma Especificação de Requisitos de um Sistema em LER, é necessário considerar a interação dos objetos que limitam seu domínio (ambiente). (Figura 01)

O sistema a ser especificado pode ser visto como um conjunto de objetos interativos. Existem dois tipos de objetos: ativos e passivos. Objetos ativos ou agentes são aqueles que, quando provocados por estímulos externos geram uma resposta. Objetos passivos recebem estímulos sem que ocorra uma resposta. Dentre os agentes do sistema o software merece uma atenção especial, mas é fundamental que se obtenha também o entendimento dos demais agentes e de seu ambiente. Analogamente, o ambiente do sistema ou domínio, pode ser visto como um conjunto de objetos interativos (ativos ou passivos), de forma que o sistema a ser especificado é um agente neste domínio.

As principais características de LER são:

1. Resposta a Estímulos - A variação do relacionamento dos objetos ao longo do tempo gera um estímulo. Esses estímulos quando respondidos por agentes podem causar novos estímulos. Os objetivos do sistema só serão alcançados através dessa ação coordenada entre seus agentes.

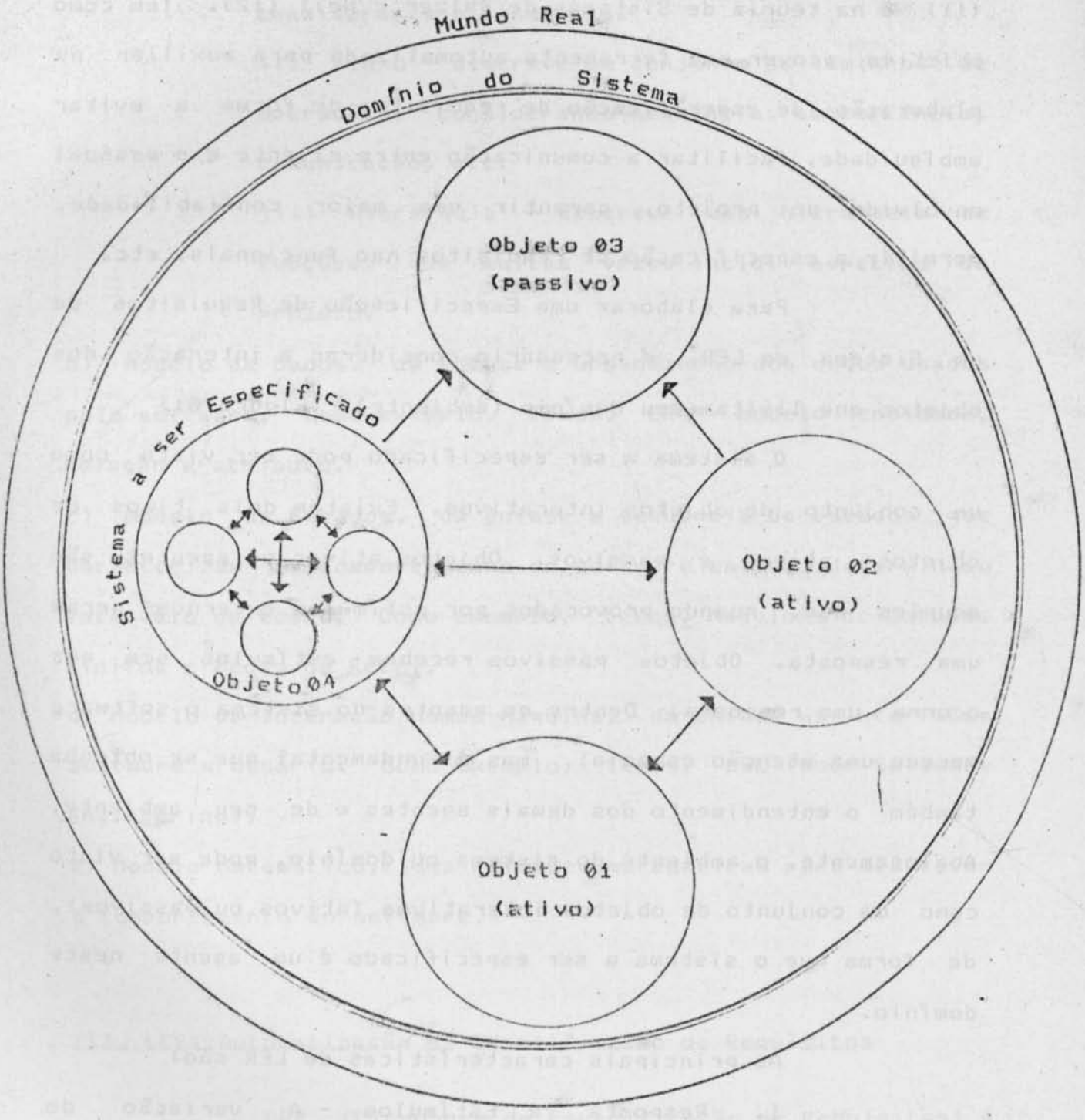


Figura 41 - Modelo Adotado na LER - Objetos

2. Composição/Decomposição - Diferentes aspectos do mundo real são capturados separadamente (determinação dos objetos), definindo-se o interrelacionamento entre seus objetos (estímulos/respostas) compondo assim, o sistema e seu ambiente. Posteriormente, é feito uma descrição das ações provocadas pelos estímulos identificados (decomposição).

3. Fluxo de Ações - A especificação de requisitos é feita através da construção de uma rede (REDE-R) a qual mostra o fluxo de ações necessárias a geração de uma resposta ao estímulo.

O processo de especificação (Figura 02), inicia com a construção de um Diagrama de Contexto, a partir da identificação dos componentes do domínio do sistema e dos estímulos/resposta que os interrelacionam, apresentando ainda, seus requisitos gerais. Para cada um dos estímulos identificados no Diagrama de Contexto, o usuário deve elaborar uma REDE-R, especificando as respostas funcionais com um mínimo de ações genéricas, que dão uma idéia geral do comportamento. Posteriormente, as ações podem ser expandidas em termos de SUB-REDES-R. Este processo pode ser repetido até que o nível de detalhe desejado seja alcançado.

Na tabela I, temos uma relação dos principais elementos de LER (Ação, Estímulo, Seleção, Concorrência, Requisito, Dados Retidos, Objeto Externo), com seus respectivos objetivos.

Em Sistemas de Tempo Real, se faz necessário mostrar a existência de objetos repetidos associados a ação. Como exemplo, em um sistema de Monitoramento de Pacientes, existem 'N' dispositivos iguais para aquisição de sinais. Cada

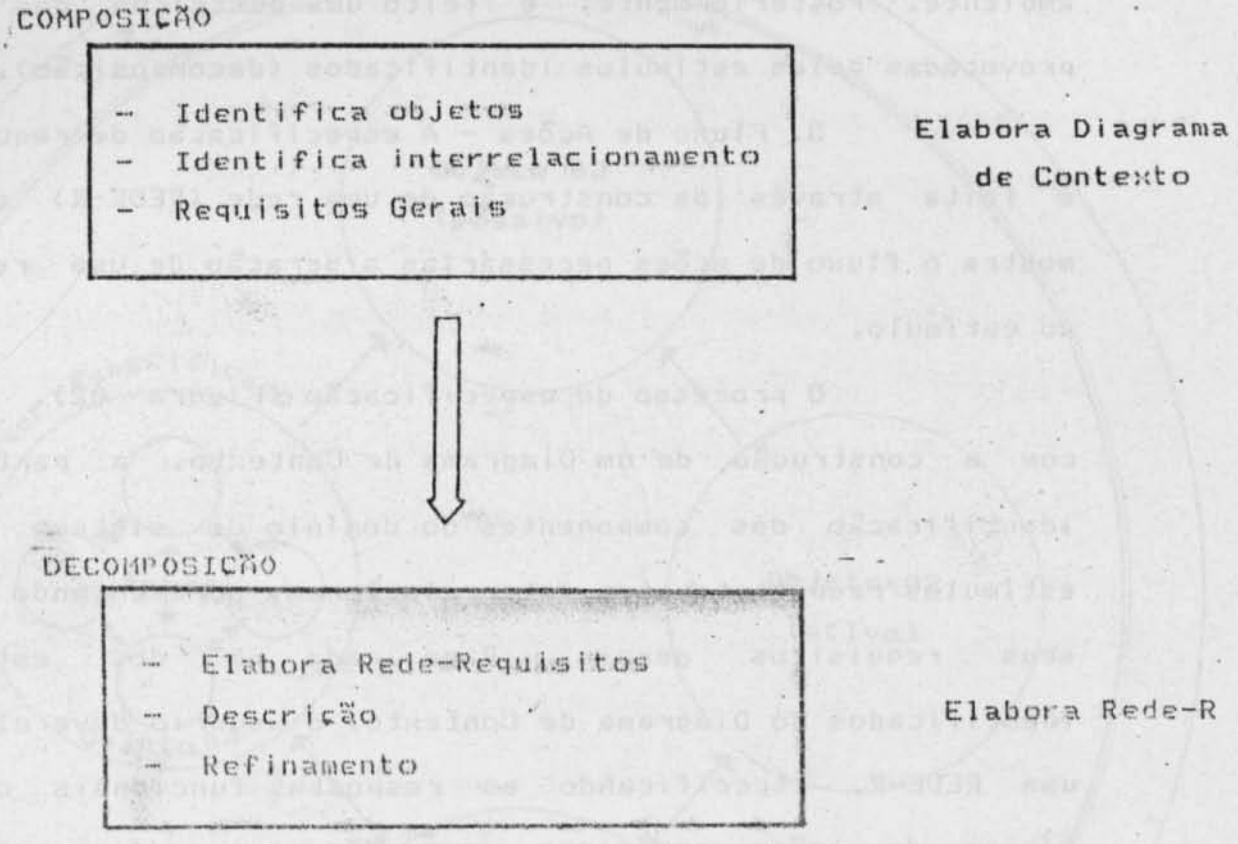


Figura 02. Atividades do Processo de Especificação em LER

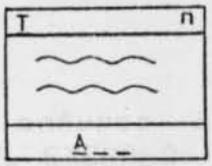
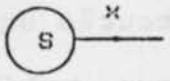
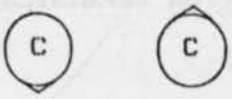
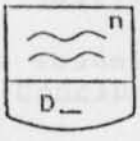
	AÇÃO - Representa uma ação executada a partir da ocorrência de um estímulo.
	ESTÍMULO - Habilita a execução de uma ação.
	SELEÇÃO - Seleciona a execução de um caminho único no fluxo de ações de acordo com uma variável seletora 'x'.
	CONCORRÊNCIA - Indica que as ações apresentadas a seguir podem ser executadas em paralelo.
	REQUISITO - Usado para identificar requisitos não funcionais no fluxo de ações. Nesses pontos podem ser extraídas informações que permitem a verificação deste requisito.
	DADOS RETIDOS - Indica que a informação deve ser retida pelo sistema. A informação está associada a ação.
	OBJETO EXTERNO - Objeto com o qual o agente descrito pela Rede-R interage.
	TERMINAL - Início e Fim de um fluxo.

Tabela I - Principais Elementos da LER

um dos dispositivos deve ser tratado por uma mesma sequência de ações. LER é capaz de exprimir ocorrências múltiplas de ações e objetos externos.

Ações estão sempre associadas a estímulos. Existem cinco tipos de associações:

a) Simples - Um estímulo aleatório habilita a execução de uma ação que precisa ser completada antes que um novo estímulo ocorra;

b) Periódica - Um estímulo aleatório habilita a execução periódica de uma ação;

c) Única - Estímulo único que habilita a execução de uma ação apenas uma vez durante a operação do sistema;

d) Contínua - Um estímulo aleatório inicia a execucao contínua de uma ação até que outro estímulo provoque o seu término;

e) Interrupção - Um estímulo aleatório habilita a execução da ação, e outro estímulo pode ocorrer mesmo antes que a resposta tenha sido gerada.

A especificação é dinâmica, podendo ocorrer alterações (criação, projeto, manutenção, etc.) durante todo o ciclo de vida do sistema. LER permite lidar com definições incompletas ou pouco detalhadas durante a elaboração da especificação de requisitos. É possível identificar a existência de um objeto sem descrição, mas isso não impede que ele seja usado na elaboração da REDE-R.

Pode-se extender a linguagem através da redefinição de seus elementos. O usuário pode definir seus próprios elementos, o que permite introduzir novos conceitos no modelo.

IV. EXEMPLO

Para ilustrar o uso de LER apresentamos na Figura 03, uma solução ao problema de Monitoramento de Pacientes em um hospital proposto por STEVENS, MYERS e CONSTANTINE.

ALFORD (09), descreve o problema de uma maneira suscinta:

1. É necessário um programa de Monitoramento de Pacientes para um hospital.

2. Cada paciente é monitorado por um dispositivo analógico que mede fatores, tais como, pulsação, temperatura, pressão sanguínea, resistência de pele.

3. O programa lê esses fatores periodicamente (especificado para cada paciente) e armazena-os em uma base de dados.

4. Para cada paciente é especificado um intervalo permitido para cada fator (Exemplo: O intervalo de temperatura permitido ao paciente X é 95 à 97.9 graus Farenheit).

5. Se um fator ultrapassa os limites permitidos ou se um dispositivo analógico falha, a estação da enfermeira é notificada.

V. Conclusão

LER permite a elaboração de um modelo Cognitivo (mostra como os objetos cooperam no domínio do sistema) e não de Projeto. LER está atualmente em fase de implementação. Ao mesmo tempo, está sendo definida uma outra ferramenta automatizada, LEP (Linguagem para Especificação de Projeto),

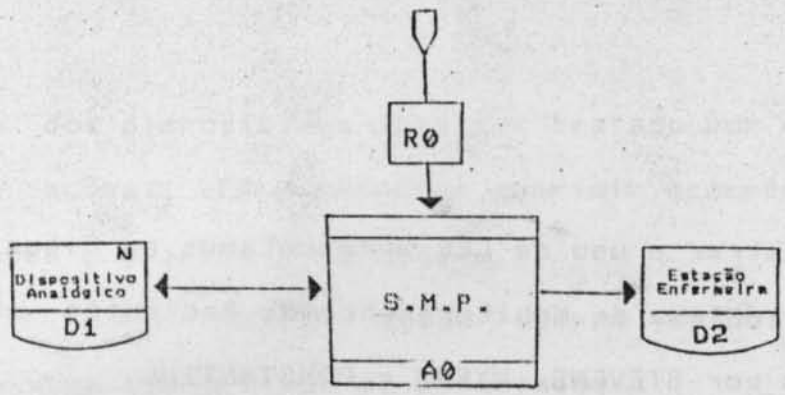


Diagrama de Contexto

S.M.P. Sistema de Monitoramento de Pacientes

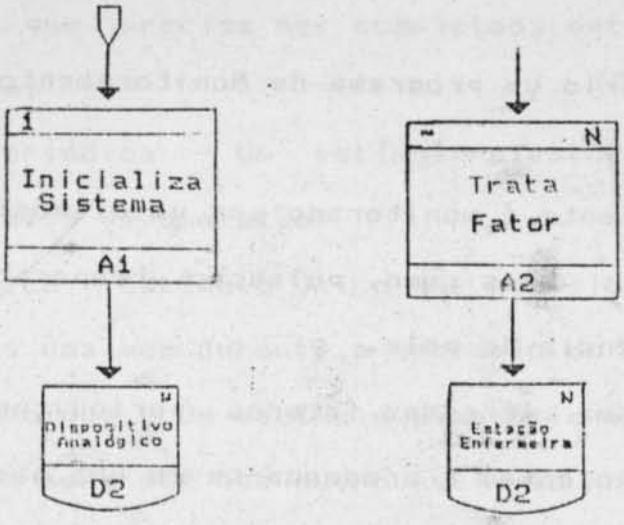
D1-A0 Dados de Fatores

A0-D1 Início de Aquisição

A0-D2 Alarme:
Falha Comunicação
Fator

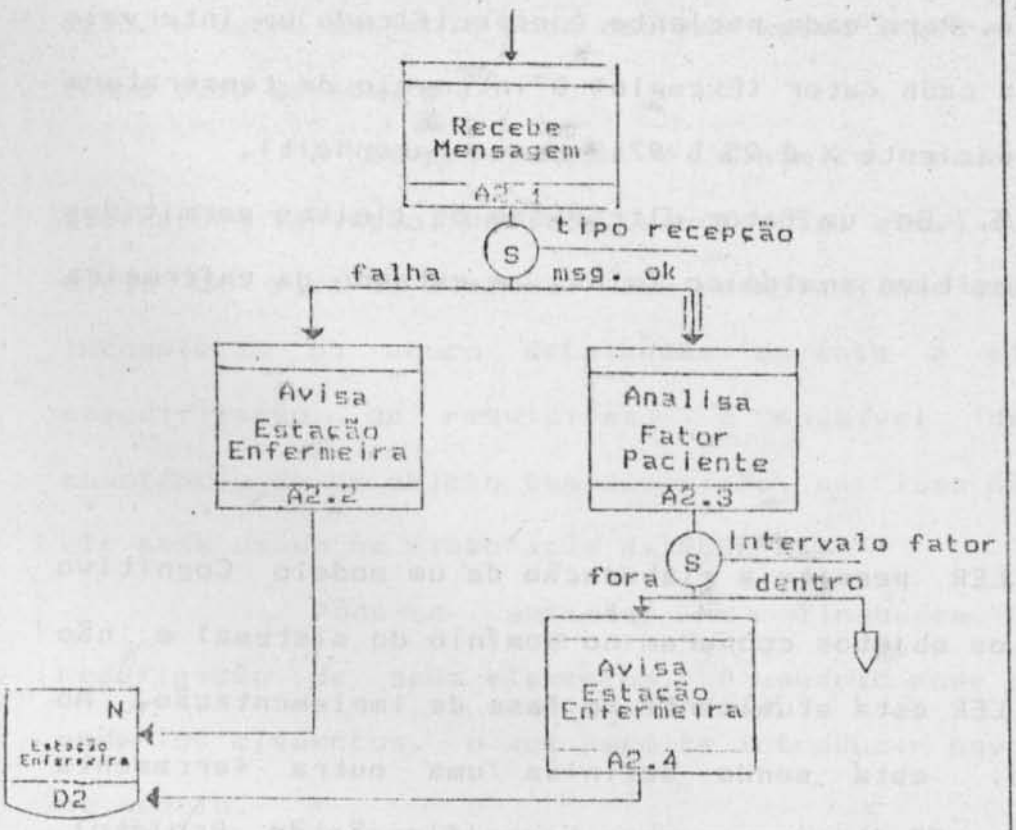
I-A0 Início Operação

R0 Requisitos Gerais



A1 : Ação única de inicialização

A2 : Ação cíclica de ocorrência múltipla que trata fatores de 'N' pacientes



Detalhamento da Ação A2

que auxilia o projetista na transformação de uma especificação de requisitos expressa em LER, numa especificação de Projeto de Arquitetura.

VI. Bibliografia

(01) Teamwork Management Overview, Cadre Inc., 1987.

(02) Communications of ACM, Janeiro/1984.

(03) ROCHA, Ana Regina, Notas de Aula da disciplina Engenharia de Software, UFRJ-COPPE, 1985.

(04) ANSI/IEEE Std 830/1944, "Guide to Software Requirements Specification".

(05) RIDDLE, William E. and WRILEDEN, Jack C., "Languages for Representing Software Specification and Design"

(06) LISKOV, Barbara et alii, "An Introduction to Formal Specifications of Data Abstractions", Current Trends in Programming Methodology, Vol.I, Prentice-Hall, Inc., Englwood Cliffs, New Jersey, 1977.

(07) MITERMEIR, Roland T. et alii, "Requirements Analysis An Integrated Approach, Techical Report #1155, University of Maryland.

(08) ASTI VERA, Armando, "Metodologia da Pesquisa Cientifica", Editora Globo 1978

(09) ALFORD, Mack W., "A Requirements Engineering Methodology

for Real-Time Processing Requirements, IEEE Transactions Software Engineering, vol. SE3, no.1, Janeiro/77.

(10) SCHEFFER, Paul et alii, "A Case Study of SREM", IEEE Transactions Software Engineering, Abril/1985.

(11) ALFORD, Mack, "SREM at the Age of Eight: Distributed Computing Design System, Computer, Abril/1985.

(12) BALZER, ROBERT and NEIL, "Principals of Good Software Specification and their Implications for Specification Language", AFTPS Conference Proceedings, vol.50, pp.393-400, 1981.

(13) CASE, Albert F., "Information Systems Development: Principals of Computer - Aided Software Engineering, Prentice-Hall, 1986.