O PROCESSO DE INSTANCIAÇÃO DO SISTEMA SEM PARA CRIAÇÃO DE UM AMBIENTE DE APOIO A UM METODO DE DESENVOLVIMENTO DE SISTEMAS

Paulo Cesar Masiero [1]

SUMARIO:
E' feita uma breve descrição do sistema SEM - System Encyclopedia Manager - e em seguida mostra-se como esse sistema pode ser instânciado para criar um ambiente de apoio por computador a um método de desenvolvimento de sistema. Utiliza-se como exemplo os métodos de Análise e Projeto Estruturados, mas o processo de instanciação pode ser seguido para qualquer outro método que se deseja utilizar.

ABSTRACT:
The System Encyclopedia Manager system (SEM) is briefly described. It is shown the customization of SEM in order to create a computer aided enviroment to support the use of a software development method. The method described here is general but an example is used, based on Structured Analysis and Structured Design.

[1] Licenciado em Matemática (Unesp, 1975), Mestre em Ciências de Computação (USP, 1979), Doutor em Administração (USP, 1984, sub-área Processamento de Dados). Esp. Sistemas de Informação, Engenharia de Software, Ambientes de Desenvolvimento de Sistemas, Bancos de Dados; ICMSC-USP, Cx.Postal 668, São Carlos, SP.

Este trabalho contou com apoio financeiro da FAPESP

1. INTRODUÇAO

Neste trabalho mostra-se como o sistema SEM (System Encyclopedia Manager) pode ser adaptado para criar um ambiente automatizado de apoio a métodos para análise e projeto de software. Em particular será utilizado um exemplo de instanciação de SEM para os métodos de Análise e Projeto Estruturados.

O termo "ambiente" refere-se à coleção de ferramentas de hardware e software utilizadas por um projetista de sistemas para a construção de sistemas de software [DA87]. O têrmo "ambiente de programação" refere-se ao suporte às atividades de programaçao. O termo "ambiente de análise" será utilizado neste trabalho para referir-se ao suporte às atividades de análise e projeto de um sistema, excluindo atividades administrativas e de controle, como gerenciamento de versões, por exemplo.

Projetos em andamento, como o ETHOS [TA88], visam ao desenvolvimento de um meta-ambiente que permita ao projetista de ambientes instanciar um conjunto de ferramentas para apoio a um método qualquer escolhido, dentro de um certo dominio de aplicação. O sistema SEM é um dos poucos sistemas em fase operacional que possui essa facilidade, daí julgarmos oportuno ilustrar essa caracteristica de SEM.

Na seção 2 faz-se uma breve introdução ao sistema SEM e seus principais componentes. Na seção 3 mostra-se como funciona o processo de instanciação de SEM, aplicando-o para os métodos de Análise e Projeto Estruturado, criando-se um ambiente de apoio denominado SASD. O componente principal do ambiente é uma linguagem de definição de problemas qua também será referida pelo mesmo nome. O processo de instanciação pode ser generalizado para outros métodos. A seção 4 contém as conclusões do trabalho.

2. VISAO GERAL DO SISTEMA SEM

O sistema System Encyclopedia Manager - SEM - e' um conjunto integrado de software, métodos, exemplos e manuais de uso. SEM baseia-se no Modelo Entidade-Relacionamento (M/ER) [CH76,TE80]. A Figura 1 mostra diagramaticamente todos os componentes de SEM [IS81], que serão descritos a seguir.

Para instanciar SEM para dar apoio a um determinado método, usa-se a Linguagem de Descrição de Sistemas de Informação (ISDL). Esta linguagem permite a definição formal de um modelo ER e, fazendo-se analogia com os Sistemas Gerenciadores de Bases de Dados, tem o mesmo papel da Linguagem de Descrição de Dados dos SGBDs.

O processador da ISDL gera relatórios dos erros sintáticos encontrados na descrição do modelo e quando esses não existem, cria uma base de dados contendo as regras sintáticas da linguagem a ser criada, chamada de meta base de dados ou base de dados de regras. Um componente desse processador permite a criação do manual de referência da linguagem, com várias opções de detalhes.

Ultrapassada essa etapa, podem ser criadas bases de dados para descrever sistemas aplicativos, de acordo com o modelo implícito na linguagem criada. Essas bases de dados são chamadas de base de dados de fatos a respeito do sistema. SEM possui duas formas padronizadas para entrada de dados: cadeia de caracteres (uma linguagem de definição de problemas) e gráficos (versão diagramática da linguagem de definição de problemas).

SEM fundamenta-se em quatro conceitos básicos: entidades (também referido pelo termo "objeto"), relações (entre entidades), textos e propriedades. Os dois últimos são tipos particulares do conceito mais geral de atributo.

Se, por exemplo, SEM for usado num ambiente de processamento de dados, PROCESSO e ELEMENTO-DE-DADO podem ser considerados entidades relevantes nesse ambiente. Essas entidades têm relações entre si. Uma relação possível seria o estabelecimento do fato de que um determinado processo usa um certo elemento-de-dado. A declaração desse fato em SEM poderia ser:

```
PROCESSO        Processo-P1;
    USA         Dado-D1;

ELEMENTO-DE-DADO  Dado-d1;
    USADO POR     Processo-P1;
```

Nesse exemplo, Processo-P1 e Dado-d1 são instâncias das entidades PROCESSO e ELEMENTO-DE-DADO.

Da mesma forma, a linguagem na qual o processo é codificado, constitui-se num atributo da entidade PROCESSO. SEM designa como uma propriedade os atributos de tamanho fixo e tipos definidos, como por exemplo: inteiros, reais e constantes alfanuméricas de até no máximo 30 caracteres. Por outro lado, a pessoa ou departamento responsável pela definição do processo também pode ser um atributo do processo, não tendo, portanto, tamanho e tipo definidos. SEM trata esse atributo como um texto que pode ter um número variável de caracteres. Abaixo mostra-se como uma propriedade ou um texto podem ser definidos em SEM:

```
PROCESSO Processo-P2;
  LINGUAGEM "COBOL";
  RESPONSAVEL ;
       Analista Senior e Lider do Projeto ;
```

SEM possui três tipos de comandos : comandos de modificação, de relatórios e de controle. O primeiro tipo permite a entrada de dados na base de dados de fatos e sua modificação quando necessário. Os comandos de relatório permitem a emissão de relatórios padronizados de SEM. Os relatórios mostram os dados na base de dados de fatos sob vários ângulos e em formato de gráficos, textos ou matrizes. Os comandos de controle permitem especificar os valores dos vários parâmetros que regulam o

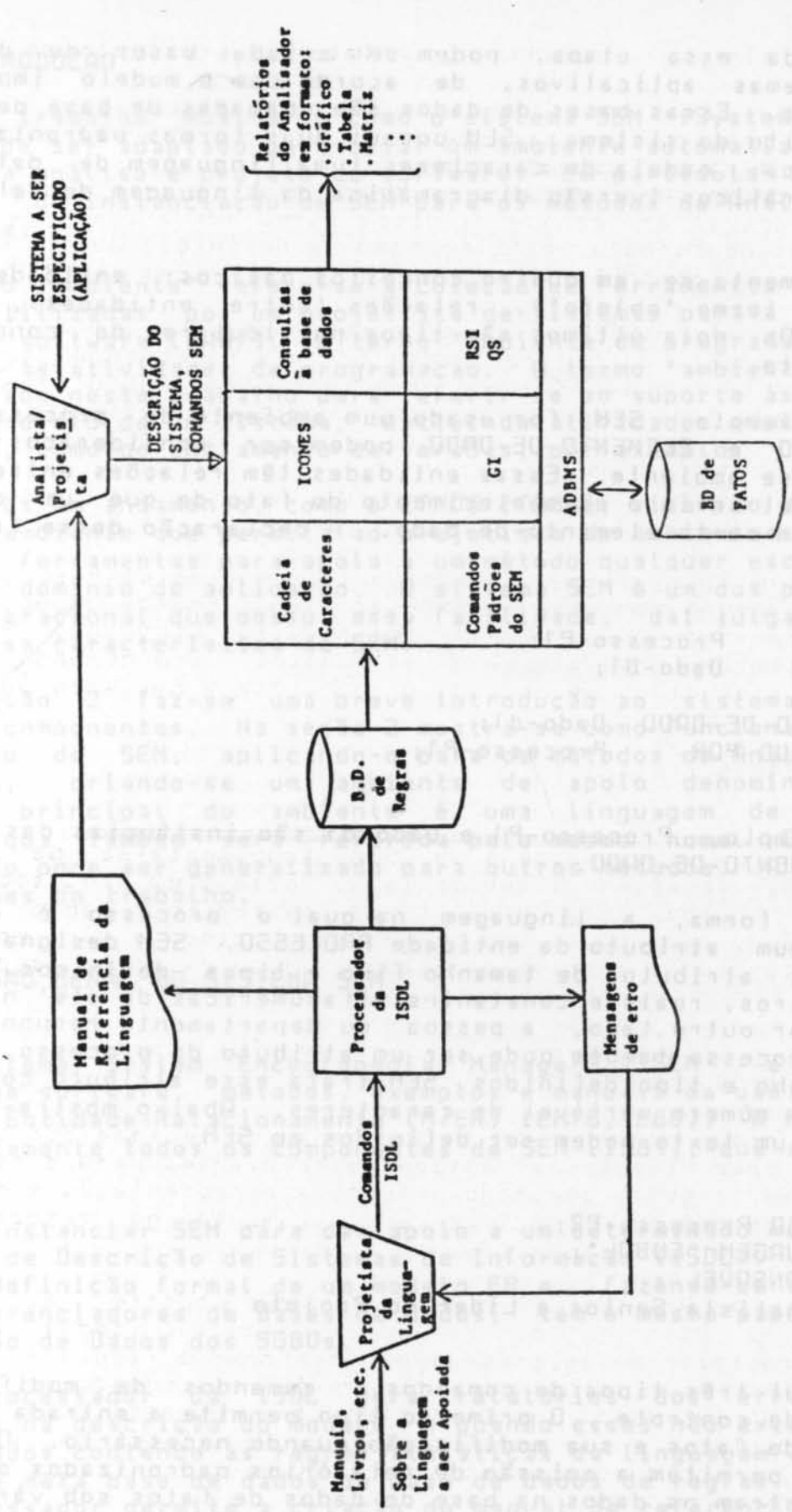


Fig. 1 - Componentes do SYSTEM ENCYCLOPEDIA MANAGER

comportamento do sistema: tamanho dos relatórios, conjunto de caracteres utilizados, etc.

A interface gráfica (GI) permite que o usuário crie ou modifique uma base de dados através de um terminal gráfico. Os diagramas são armazenados e podem ser impressos em traçadores de gráficos e a parte textual do diagrama e' armazenada da mesma forma que o seria se a entrada tivesse ocorrido no formato de linguagem de definição de problemas. A partir dai, todos os comandos de relatório de SEM podem ser utilizados.

A linguagem de consultas (QS) e' o componente de SEM que permite a extração de informações não padronizadas (ou não facilmente previsíveis) da base de dados de fatos. Ela processa consultas formuladas de forma não procedimental e recupera a informação desejada da base de dados. A QS tem capacidade para executar operações lógicas e operações de conjuntos. Uma de suas aplicações mais importantes e' a execução de testes de consistência e completeza sobre as informações armazenadas na base de dados de fatos.

Como recurso final para extração de informações da base de dados e sua manipulação, assim como sua apresentação em formatos específicos, ha' a linguagem denominada RSI - Report Specification Language. RSI e' uma linguagem algoritmica (procedural) de alto nIvel, com comandos embutidos de navegação na base de dados de SEM, o que a torna bastante poderosa.

3. O PROCESSO DE INSTANCIAÇAO

O processo de instanciação de SEM fundamenta-se na modelagem, através do M/ER, do método a ser apoiado. Esse é um processo que exige experiência da pessoa que desenvolverá o modelo, tanto no uso e restrições de SEM, quanto no conhecimento do método a ser modelado. O processo de instanciação será apresentado em sete passos, a seguir:

Passo 1: Identificar as entidades relevantes para o método a ser modelado.

A primeira etapa no processo de adaptação do sistema SEM a um método particular e' a determinação das entidades relevantes para o método, isto e', as entidades as quais o método permite a representação e manipulação ou mesmo de conceitos definidos abstratamente no método e sem representação gráfica explicita.

No caso da Análise Estruturada, são quatro as entidades representadas no Diagrama de Fluxo de Dados e todas elas foram selecionadas para fazer parte do modelo, conforme a Tabela 1 abaixo. O Projeto Estruturado é um pouco mais complexo, mas chegou-se a um modelo com nove tipos de entidades, necessárias para representação do Diagrama de Estrutura Modular. Para representação do modelo de dados, optou-se também por quatro entidades, mostradas na Tabela 1, formando um total geral de dezessete entidades.

Tabela 1 - Entidades Escolhidas

ANALISE ESTRUTURADA	PROJETO ESTRUTURADO	MODELAGEM DE DADOS
PROCESSO	MODULO	ENTIDADE
FLUXO-DE-DADOS	MODULO-INCLUSO	RELACIONAMENTO
DEPOSITO	MODULO-DE-BIBLIOTECA	GRUPO-DE-DADOS
ENTIDADE-EXTERNA	AREA-DE-DADOS	ELEMENTO-DE-DADOS
	AREA-DE-DADOS-GLOBAL	
	PACOTE	
	AGREGADO	
	CHAMADA-CONDICIONAL	
	CHAMADA-ITERATIVA	

Cabe alguma discussão quanto às decisões tomadas na escolha das entidades. O principio geral aplicado foi o de que se a entidade tem um representação (simbolo) própria, ela será então considerada como uma entidade independente. Isso deve-se à restrição imposta por SEM, que não permite o conceito de generalização, senão, poderiamos ter, por exemplo, uma entidade MODULO, com subtipos: incluso, de biblioteca e normal.

Passo 2 - Identificar os atributos das entidades definidas no passo 1.

Cada entidade pode ter atributos que permitem sua melhor caracterização. As Tabelas 2 e 3 abaixo mostram os atributos separados pelo tipo : texto ou propriedade. As propriedades SINONIMO e AUTOR e os textos NOME-COMPLETO e DESCRIÇAO são atribuidos a todas as entidades. A entidade PROCESSO, por exemplo, tem as propriedades TIPO-DO-PROCESSO, FREQUENCIA e IDENTIFICAÇAO e o texto MINI-SPEC. Outros atributos poderiam ser facilmente adicionados ao modelo.

Tabela 2 - Propriedades das Entidades

Propriedades	Entidades
SINONIMO	Todos os tipos
AUTOR	de entidades
TAMANHO	
FORMATO	ELEMENTO-DE-DADO
TIPO	
DISPOSITIVO-IO	FLUXO-DE-DADOS
TIPO-PROCESSAMENTO	
FREQUENCIA	PROCESSO
IDENTIFICAÇAO	
IDENTIFICAÇAO	MODULO (os 3 tipos)

Tabela 3 - Textos associados a cada Entidade

Textos	Entidades
DESCRIÇAO NOME-COMPLETO	Todas aos tipos de entidades
MINI-SPEC	PROCESSO
FORMATO	ENTIDADE GRUPO-DE-DADO FLUXO-DE-DADO
VOLUME	FLUXO-DE-DADO DEPOSITO
ESPECIFICAÇAO	MODULO MODULO-INCLUSO

Abaixo mostram-se, como exemplo, as definições de uma entidade, uma propriedade e um texto em ISDL:

```
OBJECT PROCESSO;
        SYNONYMS PROC;
        DOCUMENTATION;
        Conjunto de instruções que transformam o(s)
        fluxos de dados de entrada no(s) fluxo(s) de
        dado(s) de saida;
        CODE OBIPRC 20;

PROPERTY AUTOR;
        SYNONYMS AUT;
        DOCUMENTATION;
        Indica quem é o responsável pelas informações na
        base de dados;
        APPLIES ALL;
        VALUES STRING;
        CODE PYTAUT 93;

TEXT DESCRIÇAO;
        SYNONYMS DESC;
        DOCUMENTATION;
        É um texto livre que pode ser asssociado a
        qualquer entidade para melhor descrevê-la;
        APPLIES ALL;
        CODE TXDSC 90;
```

A sintaxe da ISDL pode ser percebida do exemplo acima e algumas cláusulas são auto explicativas, como por exemplo, SYNONYMS e APPLIES. A cláusula DOCUMENTATION permite a entrada de uma descrição da entidade ou

atributo e esta será, posteriormente, usada na criação do manual de referência da linguagem. CODE e' uma exigência fisica do processador da ISDL e poderá ser utilizado por programas que navegam na base de dados de SEM. VALUES indica o tipo dos valores a serem aceitos para a propriedade, estabelecendo um exemplo de checagem sintática que SEM efeturá' posteriormente. Alguns dos valores pré-definidos são : cadeias, números, constantes, etc..

Passo 3 - Identificar os relacionamentos entre as entidades definidas no passo 1.

A etapa seguinte no processo de particularização de SEM a um determinado método e' a especificação dos relacionamentos existentes entre as entidades escolhidas. SEM aceita a definição de relacionamentos binários, ternários e quaternários.

A Figura 2 mostra um exemplo de como identificar e modelar os relacionamentos. Para cada relação ou grupo de relações afins existem quatro pontos importantes. Primeiramente, na parte (1) dessa figura, mostra-se um exemplo de como a relação aparece no método original. No caso da Figura 2 tem-se um diagrama de contexto. Esse diagrama mostra relações entre fluxos de dados que têm origem ou se destinam a entidades externas e são processados pelo processo de contexto, que representa o sistema como um todo. Ha' duas versões para esse diagrama: alguns autores representam as entidades externas e outros não julgam necessário fazê-lo [GA79,De78].

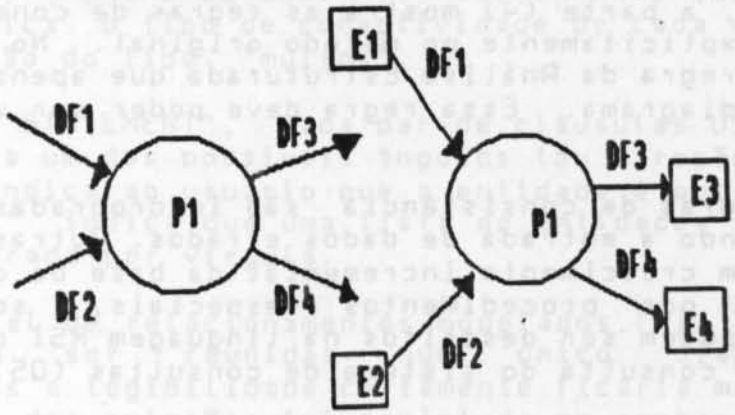
Na parte (2) da Figura 2 aparece o diagrama ER que modela os relacionamentos existentes no diagrama de contexto. Nesse caso foram necessárias duas relações, cada uma delas contando com a participação de três entidades (relações ternárias, portanto). Deve-se notar que a linha tracejada que liga a entidade ENTIDADE-EXTERNA indica que essa e' uma entidade opcional na relação, modelando-se, dessa forma, as duas versões do diagrama de contexto.

A indicação de que a conectividade de uma relação é do tipo "muitos" ou "um", no caso de uma relação ternária, e' interpretada da seguinte forma: fixam-se duas das entidades e estuda-se o comportamento da outra. Por exemplo, no diagrama ER da Figura 2, ENTIDADE-EXTERNA tem conectividade "muitos" porque uma instância especifica de processo pode receber (ou gerar) o mesmo fluxo de dados de e para várias entidades externas ao mesmo tempo. O mesmo raciocinio se aplica para a conectividade das demais entidades.

A parte (3) da Figura 2 mostra como SEM deverá aceitar a sintaxe das relações. As declarações mostram (parcialmente) como o diagrama de contexto ilustrado na parte (1) seria descrito usando a sintaxe da linguagem SASD. Deve-se notar que cada relação ternária pode ser vista de três ângulos diferentes: o ângulo de cada uma das entidades que participam do relacionamento.

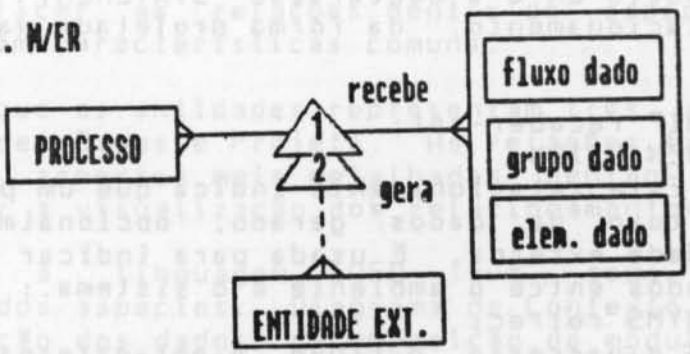
Isso significa que dizer " PROCESSO P1 RECEBE DF1 DE E1" é o mesmo que dizer "FLUXO-DE-DADO DF1 RECEBIDO POR P1 DE E1", que por sua vez tem o mesmo significado que "ENTIDADE-EXTERNA E1 ORIGEM DE DF1 RECEBIDO POR

1. DIAGRAMA DE CONTEXTO



versão 1 versão 2

2. M/ER



3. SINTAXE DA LINGUAGEM (exemplo)

```
PROCESSO P1;
  RECEBE DF1 [DE E1 ];
  GERA DF3 [ PARA E3 ];

ENTIDADE-EXTERNA E1;
  ORIGEM DE DF1 RECEBIDO POR P1;

FLUXO-DE-DADOS DF1;
  RECEBIDO POR P1 [ DE E1 ];
```

4. REGRAS DE CONSISTÊNCIA

DEVE HAVER UM ÚNICO PROCESSO NO DIAGRAMA DE CONTEXTO

FIG. 2 - RELAÇÕES "RECEBE" E "GERA"

P1". Para que a relação em SEM fique estabelecida, basta declarar uma das formas anteriores.

Finalmente, a parte (4) mostra as regras de consistência que existem implícita ou explicitamente no método original. No caso do diagrama de contexto, e' regra da Análise Estruturada que apenas um processo exista nesse tipo de diagrama. Essa regra deve poder ser verificada de alguma forma por SEM.

Certas regras de consistência são incorporadas naturalmente pelo modelo, impedindo a entrada de dados errados. Outras regras, no entanto, para permitir um crescimento incremental da base de dados de fatos, devem ser checadas por procedimentos especiais, sob demanda. Esses procedimentos podem ser descritos na linguagem RSI ou então codificados na linguagem de consulta do sistema de consultas (QS).

Mostra-se abaixo a codificação em ISDL da relação que indica quais processos recebem fluxos de dados de entidades externas. O nome dado a essa relação foi "receber-rel". A declaração "RELATIONSHIP" permite a descrição do modelo ER e a declaração "STATEMENT" permite a descrição da sintaxe do relacionamento, da forma projetada na parte (3) da Figura 2.

```
RELATIONSHIP receber-rel;
     DOCUMENTATION;
          Este relacionamento indica que um processo recebe
     um fluxo de dados gerado, opcionalmente, por uma
     entidade externa. E usada para indicar troca de fluxos
     de dados entre o ambiente e o sistema.;
     SYNONYMS relrec;
     PARTS p-processo, p-fluxo, p-ent-externa;
     COMBINATION p processo PROCESSO
               WITH p-fluxo FLUXO-DE-DADOS,
                            GRUPO-DE-DADOS,
                            ELEMENTO-DE-DADOS
               WITH p-ent-externa ENTIDADE-EXTERNA;
     CONNECTIVITY MANY p-processo, p-fluxo, p-ent-externa;


STATEMENT receber-sint;
     DOCUMENTATION;
          Esta   declaração   descreve   a   sintaxe   do
          relacionamento receber-rel;
     USED p-processo receber-rel;
     FORM RECEBE (p-fluxo : , )
          [DE p-ent-externa];
     USED p-fluxo receber-rel;
     FORM RECEBIDO POR p-processo
          [DE p-ent-externa];
     USED p-ent-externa receber-rel;
     FORM ORIGEM DE (p-fluxo : , )
          RECEBIDO POR p-processo;
```

A cláusula "PARTS" dá nomes genéricos a cada uma das "partes" de uma

relação. Como esta é uma relação ternária, tem-se 3 partes. A cláusula "COMBINATION" indica que tipos de entidades podem participar de cada uma das partes da relação. Note que no caso dessa relação, "p-fluxo" pode ser do tipo "ENTIDADE", "GRUPO-DE-DADO" ou "ELEMENTO-DE-DADO". A cláusula "CONNECTIVITY" indica o tipo de conectividade de cada uma das partes. Neste caso, todas são do tipo "muitos".

Na declaração "STATEMENT", cada par de cláusulas USED e FORM indica a sintaxe para cada um dos possíveis ângulos (ou direção) da relação. A notação "[...]" indica ao usuário que a entidade é opcional e a notação "(.... : ,)" indica que uma lista de entidades pode ser dada, cada uma delas separada por virgula.

O número total de relacionamentos modelados foi 22. Todas essas relações poderiam ser reunidas num único diagrama Entidade-Relacionamento, mas a legibilidade certamente ficaria muito prejudicada, por isso, cada uma das relações foi projetada em separado e todas podem ser encontradas no documento [MA87a].

Passo 4 - Classificar as relações dentro de aspectos que englobem caracteristicas comuns.

Já foi visto que as entidades representam três aspectos gerais: Análise, Modelagem de Dados e Projeto. As relações também podem ser agrupadas dentro de aspectos mais detalhados dentro desses aspectos gerais, para facilitar a visualização dos relacionamentos.

Como exemplo, a linguagem SASD teve seus relacionamentos classificados dentro dos aspectos: Diagrama de Contexto, decomposição de processos, estruturação dos dados, decomposição de módulos, agregação de módulos, etc. A separação dos atributos e relacionamentos dentro dos aspectos pode ser vista no exemplo mostrado no Apêndice 1, onde os relacionamentos "recebe" e "gera" aparecem dentro do aspecto " Diagrama de Contexto".

Passo 5 - Processar a descrição em ISDL

Neste passo deve-se processar a descrição do modelo da linguagem, em ISDL, da maneira como foi mostrado nos passos anteriores. Deve-se repetir esta atividade até não obter mais mensagens de erro. Como resultado, SEM fornecerá o manual de referência da linguagem, além de criar a meta base de dados.

O Apêndice 1 mostra uma página do manual de referência da linguagem SASD, contendo parte da sintaxe dos atributos e relacioamentos da entidade "Processo". Esse manual pode agora ser utilizado para orientar na modelagem de qualquer aplicação que se queira desenvolver com o apoio do ambiente SEM instanciado para a Análise e Projeto Estruturado.

Passo 6 - Testar a linguagem

De posse da primeira versão da linguagem, ela pode ser usada em

casos de teste e/ou sistemas reais, fazendo a "sintonia fina" da linguagem. Os passos 1 a 5 devem ser repetidos quando forem encontrados erros ou então se se deparar com situações não tratadas de maneira satisfatória pela linguagem.

Para ilustrar o uso da linguagem SASD escolheu-se o caso apresentado por Page-Jones em [Pa80]. Trata-se de um sistema de entrada de pedidos e é um exemplo razoavelmente completo: apresenta DFDs com algum grau de detalhamento, o modelo ER das entidades envolvidas, amostras do dicionário de dados e um diagrama de estrutura modular com alguma complexidade. Além disso, algumas entidades e relações foram acrescentadas para ilustrar outras situações importantes não cobertas no caso.

A descrição completa do caso possue 234 instâncias de entidades e por volta de 400 relacionamentos e atributos. Essa descrição foi processada por SEM e uma base de dados foi criada. O documento [MA87a] apresenta a descrição completa desse caso e os relatórios padronizados extraídos da base de dados através dos comandos de relatórios de SEM.

Entre esses relatórios estão todos os que são obtidos normalmente por um dicionário de dados, acrescido de outros que mostram as estruturas de processos, análises de consistências e completeza, etc.

Passo 7 - Construção de ferramentas específicas

O ambiente pode ser enriquecido com ferramentas construídas especialmente para tirar proveito do modelo do sistema armazenado na base de dados, e isso pode ser feito utilizando-se a linguagem RSI, que possui comandos de navegação na base de dados de SEM, de forma a facilitar enormemente a programação dessas ferramentas.

Foge ao escopo deste trabalho apresentar detalhadamente essas ferramentas, mas algumas delas serão comentadas a seguir, com referências para documentos mais detalhados:

1. A técnica conhecida como "Número de pontos por função" [BU85], utilizada para estimar os recursos a serem empregados no desenvolvimento de um sistema foi automatizada, fazendo-se um mapeamento entre as entidades da Análise Estruturada e os conceitos próprios do método. Dessa forma, utiliza-se a base de dados contendo a descrição do sistema para calcular, automaticamente, o número de pontos por função do sistema [MA87a].

2. Foram acrescentadas ao modelo extensões para modelagem de Sistemas de Tempo Real, bem como relatórios específicos para essa área. Os resultados podem ser encontrados em [Ma86].

3. Foi desenvolvido um algoritmo para fazer semi-automaticamente a transformação da análise para o projeto do sistema, isto é, do Diagrama de Fluxo de Dados para o Diagrama de Estrutura Modular, utilizando as técnicas de Análise das Transformações e Análise das Transações [Ma87b].

4. CONCLUSOES

Mostrou-se neste trabalho como os métodos de Análise e Projeto Estruturados foram modelados segundo o modelo Entidade-Relacionamento e processados por SEM para criar-se a linguagem denominada SASD. Com essa linguagem pode-se dar apoio por computador aos profissionais que usam esses métodos para modelagem de software, através de um conjunto de relatório padronizados, bem como de ferramentas especificas que podem ser desenvolvidas para fazer análise da informação armazenada na base de dados e para automatizar técnicas e ou procedimentos sugeridos no método original.

Acreditamos que um dos pontos fortes de SEM é a rapidez com que permite a geração de um ambiente de apoio a um certo método, permitindo que se testem idéias e ferramentas, como se fosse um protótipo de ambiente que poderá depois ser desenvolvido com mais rigor.

Como pontos fracos de SEM, podem ser citados a interface com o usuário (o projetista da linguagem), porque utiliza pesadamente os recursos do sistema operacional do computador hospedeiro, e a interface gráfica bastante pobre.

REFERENCIAS

[BU85] BURROUGHS - Function Point Analysis - Student Guide. Burroughs, 1985.

[CH76] CHEN, P. P. - The Entity-Relationship Model - toward a unified view of data. ACM Transactions on Data Base Systems, Vol. 1 (1), 1976.

[DA87] DART,S.A ; ELLISON, R.J. ; FEILER, P.H. & HABERMAN, A.N. - Software Development Environments, IEEE Trasnsactions on Software Engineering, Nov. 1987.

[De78] De MARCO, T. - Structured Analysis and Systems Specification, Yourdon Press, New York, 1978.

[GA79] GANE,C. & SARSON, T. - Structured Systems Analysis: tools and techniques. Prentice-Hall Inc., Englewood Cliffs, NJ, 1979.

[IS81] ISDOS - An Introduction to the Use of Information System Definition Language (ISDL) and System Encyclopedia Manager (SEM), ISDOS REF. 320-0, 1981.

[MA86] MASIERO,P.C. ; CHASE,T. ; PADMANABHAN,S. & TEICHROEW,D. - An Implementation of a Structured Methodology for Real-Time Systems Using the System Encyclopedia Manager System. National Conference and Workshop on Methodologies and Tools for Real-Time Systems, Washington,DCm, USA, 1986.

[MA87a] MASIERO,P.C. - Um Ambiente de Análise de Sistemas Baseado nos Métodos de Análise e Projeto Estruturado, Tese de Livre Docência,

ICMSC-USP, 1987.

[MA87b] MASIERO,P.C. ; GERMANO, F.S. & TEICHROEW, D. - Computer Aided Transition from Analysis to Design. Eight Annual Conference on Applications of Computer-Aided Systems Engineering Tools, Ann Arbor, Michigan, USA, 1987.

[Pa80] PAGE-JONES, M. - The Practical Guide to Structured Systems Design, Yourdon Press, New York, 1980.

[TA88] TAKAHASHI,T. & HAEBERER,A.M. - Proyeto ETHOS: Informe Final de la Fase Preliminar, Rio de Janeiro, 1988

[TE80] TEICHROEW,D. ; MACASOVIC, P. ; HERSHEY,E. & YAMAMOTO,Y. - Application of the Entity Relationship Approach to Information Systems Analysis and Design. In: CHEN,P.P. (Ed.) - Entity Relationship Approach to Systems Analysis and Design, North Holland, 1980.

Language Summary

Object name=PROCESS Synonym(s)=PROC

Transformation of input data flow(s) into output data flow(s).

DEFINE PROCESS (NAME : ,) ;

========== General ==========

AUTHOR <string> ;

Indicates who has entered the definitions in the database

DESCRIPTION ;
TEXT;

It is a text that can be attached to all objects to give a better idea of their contents, to place observations related to the object, etc.

FULL-NAME ;
TEXT;

This text may be used to record the complete name of objects with names more than thirty caracters long.

========== Context Diagram ==========

GENERATES ({ DATAFLOW-name | GROUP-name | ELEMENT-name } : ,)
[TO TERMINATOR-name] ;

Describes the syntax of the relationship generate-rel:
GENERATES ... TO ... /
GENERATED BY ... TO ... /
SINK OF ... GENERATED BY ... .

RECEIVES ({ DATAFLOW-name | GROUP-name | ELEMENT-name } : ,) [
FROM TERMINATOR-name] ;