

ACTAS DE LA
PRIMERA
CONFERENCIA
INTERNACIONAL
EN CIENCIA DE LA
COMPUTACION

Auspician:

Banco de Chile •

Secico •

A.F.P. Santa María •

Leniz y Silva, •

Ingenieros Consultores

Cientec •

Cecinas Winter •

Revista Informativa •

004.06
C748a



PONTIFICIA UNIVERSIDAD
CATOLICA DE CHILE



UNIVERSIDAD DE CHILE

PRIMERA CONFERENCIA INTERNACIONAL
EN CIENCIA DE LA COMPUTACION

FECHA : 24 - 27 AGOSTO

LUGAR : CASA CENTRAL
PONTIFICIA UNIVERSIDAD
CATOLICA DE CHILE

UN ESTUDIO COMPARATIVO DE MÉTODOS DE ESPECIFICACIÓN DE
SISTEMAS AUTOMATIZADOS

UM ESTUDO COMPARATIVO DE MÉTODOS DE ESPECIFICAÇÃO DE
SISTEMAS AUTOMATIZADOS

A COMPARATIVE STUDY OF SPECIFICATION METHODS IN
AUTOMATED SYSTEMS

Ana Regina Cavalcanti da Rocha
Departamento de Informática
PUC - Rio de Janeiro
B r a s i l

1. INTRODUÇÃO

Este trabalho é um estudo comparativo de sete métodos de especificação de sistemas automatizados (HDM, SREM, PSL/PSA, SADT, SAFE, RDL e SMM), tendo como base certos requisitos que consideramos indispensáveis a uma boa metodologia.

É necessário, primeiramente, definir o que entendemos por comparação e distinguir comparação de avaliação. Comparação é algo objetivo que se dirige à busca de semelhanças e diferenças entre as metodologias. Avaliação é um passo a mais e implica além da comparação, um julgamento.

Nosso objetivo aqui é, portanto, comparar metodologias. Acreditamos que é possível fazê-lo de uma maneira racional e objetiva, desde que se tenha claro uma série de princípios, baseada na qual se fará a comparação.

Parte-se de uma sistematização das características que determinam a qualidade de uma especificação e de uma metodologia para criação de especificações. A partir destas características, faz-se então a comparação entre os sete métodos.

2. O QUE SÃO ESPECIFICAÇÕES

É fato conhecido que grande parte dos problemas com sistemas estão relacionados a erros, omissões e inadequações nas especificações. E o que é mais grave, estes

erros muitas vezes tardam demasiado em serem detectados . Isso se torna especialmente grave se observamos que o custo de um erro aumenta, de maneira drástica, à medida que aumenta a distância, no tempo, entre a época em que o erro é cometido e a época em que é efetivamente detectado. Como proceder então para detectar erros, omissões e inadequações o mais cedo possível ?

Qualquer produto de Software tem sua origem a partir de um conceito existente na mente de alguém. Este conceito geralmente pode ser realizado por um conjunto muito grande de programas alternativos de implementação. No entanto apenas algumas destas alternativas têm interesse prático . Esta situação é mostrada na fig. 1. Neste caso, o conceito é estabelecido informalmente, e, qualquer que seja a técnica utilizada para verificar a correção do programa, o resultado da aplicação da técnica também só pode ser estabelecido em termos informais.

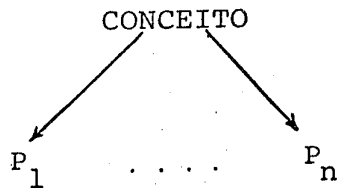


fig 1 - Um conceito e todos os programas que o implementam corretamente

Para atuar de maneira mais formal, deve-se interpor, entre o conceito e os programas, uma especificação. A especificação descreve o conceito de uma maneira formal, e a correção do programa é mostrada provando-se ser ele equivalente à especificação . Neste caso o conceito poderá ser realizado por diversas especificações alternativas, e estas por sua vez poderão ser satisfeitas por uma classe de programas (fig 2) [LISKOV 77].

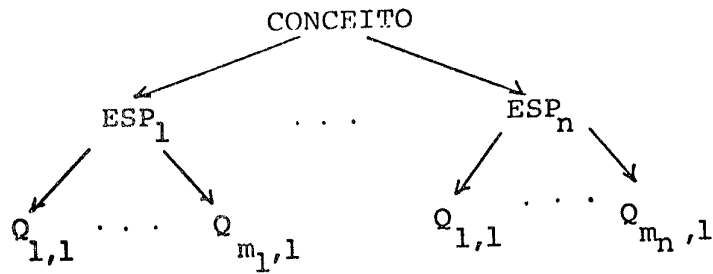


fig 2 - Um conceito suas possíveis especificações formais e todos os programas que podem ser provados equivalentes e estas especificações

Assim sendo uma especificação descreve o que faz a execução do programa ou do sistema, sem porê[m] fixar como o faz.

Especificações são produzidas ao longo do ciclo de vida do sistema automatizado, através de refinamentos sucessivos, partindo da visão mais abrangente para a mais detalhada. Especificações surgem, então, em duas modalidades:

- a) especificação de definição (especificação de requisitos e funcional) onde é definido o problema;
- b) especificação de projeto (especificação lógica e física) onde são especificados a forma de resolver o problema e os componentes do sistema "solução" [ROCHA 81]

3. COMO PRODUZIR ESPECIFICAÇÕES

3.1. Qualidades de uma especificação

Embora seja reconhecida sua importância' o simples fato de que existam especificações

não é suficiente para que elas sejam realmente úteis. Para que sejam efetivamente uma base útil e segura para a construção do sistema, as especificações devem possuir certas qualidades. Diferentes autores fazem referência a diferentes atributos. Com base em Alford [ALFORD 76] fazemos aqui uma união dos atributos mais comumente referenciados:

- Comunicabilidade- é a habilidade de uma especificação comunicar requisitos e restrições de uma maneira explícita, inteligível e não ambígua às diversas classes de pessoas que as utilizarão, requerendo destas pouco esforço na interpretação dos documentos apresentados.

- modularidade- a possibilidade de poder-se alterar parte da especificação sem com isto afetar a validade e consistência do restante.

- estrutura- a possibilidade de poder-se determinar completa e claramente as porções antecedentes e consequentes de determinados itens chave em qualquer nível de detalhe da especificação.

- Capacidade de "trace", a possibilidade de localizar todas as referências a determinado assunto ou item chave, nos diversos níveis de especificação.

- Consistência, a propriedade de não conter contradições entre quaisquer itens especificados.

- Completeza, a propriedade de definir tudo o que seja relevante no nível de detalhe correspondente à fase atual de elaboração da especificação.

- Não condicionante, a propriedade de não antecipar-se impondo condições cujo tratamento seria posterior ao nível de detalhe atual na elaboração da especificação.

- Verificabilidade - a propriedade de poder-se determinar para cada item-chave, requisito e/ou restrição, em qualquer nível de detalhe da especificação, ter o produto final satisfeito o especificado. A verificação poderá ser conduzida por testes e/ou provas formais.

- Correção, a propriedade do produto satisfazer a intenção de seus projetistas, caso satisfaça todos os itens chave, requisitos e/ou restrições especificadas.

- Necessidade, a propriedade de cada requisito especificado contribuir para que o requisito originário seja atingido. Ou seja, se qualquer parágrafo de especificação subsequente for eliminado ou não for atingido pelo produto, a especificação originária ou a intenção dos projetistas também não será satisfeita.

- Viabilidade, a propriedade de que exista pelo menos um "design" para o produto que satisfaça as especificações. Esta existência porém não implica em que saiba determiná-lo.

3.2. Métodos de especificação. Requisitos

As dificuldades encontradas na produção de especificações deram lugar a um razoável número de pesquisas nesta área.

Segundo Liskov e Zilles [LISKOV 77] um método de especificação além de permitir a produção de boas especificações deve satisfazer os seguintes requisitos:

- Formalidade - Um método de especificação deve ser formal, escrito numa notação que permita a verificação formal da especificação antes e após o desenvolvimento do produto.

- Constructibilidade - Deve ser possível construir especificações sem demasiada dificuldade desde que o especificador conheça a técnica e entenda o conceito a ser especificado.

- Compreensividade - Uma pessoa treinada na notação usada deve ser capaz de ler a especificação e em seguida com um mínimo de dificuldade reconstruir (abstrair) o conceito que a especificação pretende descrever.

- Minimalidade - Deve ser possível, usando o método de especificação, construir especificações que descrevam as propriedades do conceito que interessam e nada mais.

- Ampla aplicabilidade - Quanto maior seja a classe de conceitos que possam ser facilmente descritos pela técnica maior sua utilidade.

- Extensibilidade - É desejável que uma pequena alteração em um conceito resulte em uma alteração similarmente pequena nas especificações.

- Detalhabilidade - É desejável que a família de linguagens de especificação possua membros adequados a cada nível de detalhe e que a evolução de um nível de detalhe para outro se dê de forma harmônica.

- Automatizabilidade - É a propriedade do processo de especificar/desenvolver poder receber auxílio automatizado para a sua execução. Um certo grau de automatismo é necessário para que se possa eliminar classes de erros atribuídos a falhas humanas e/ou para poder verificar de modo mais formal o acerto das especificações e/ou terem estas sido satisfeitas pelo produto.

4. DESCRIÇÃO DOS MÉTODOS DE ESPECIFICAÇÃO

4.1. HDM

HDM ("Hierarchical Development Methodology") desenvolvida pelo "Stanford Research Institute" é uma metodologia que foi concebida para ajudar no desenvolvimento de sistemas de Software grandes e complexos, onde é necessário um alto grau de integridade.

Como características da metodologia assinalamos:

1. O uso de decomposição hierárquica para resolver problemas complexos
2. O uso de métodos de abstração, isolam

do as propriedades do objeto apropriadas ao entendimento e descrição do mesmo, no nível de detalhe sob consideração

3. O uso de especificação formal, através de uma linguagem semi-matemática
4. Modularidade
5. A possibilidade de, a cada passo, verificar formalmente tendo como base as especificações anteriores.

HDM usa três linguagens de especificação:

1. SPECIAL, usada para especificar e representar módulos
2. HSL, usada para descrever níveis e hierarquia de níveis
3. ILPL, que é uma linguagem de programação de nível intermediário usada para descrever programas abstratos [EDP ANALYZER DEZ 79].

4.2. SREM

SREM ("Software Requirements Engineering Methodology"), desenvolvida pela TRW, tem como objetivo aumentar a qualidade das especificações de requisitos, fornecendo um método para seu desenvolvimento, junto com um sistema auxiliado por computador, para determinar sua completeza e consistência.

SREM consiste em:

1. Uma metodologia para compor requisitos;
2. Uma linguagem processável para estabelecer requisitos (RSL - " Requirements Statement Language ");
3. Um conjunto integrado de ferramentas para auxiliar no desenvolvimento dos requisitos em RSL (REVS - " Requirements Engineering and Validation Systems ").

Um conceito unificador na metodologia é a especificação de requisitos em termos de fluxos.

A linguagem RSL foi concebida para ser um meio de se estabelecerem requisitos, de uma maneira natural, embora mantendo um nível de rigor suficiente para poder ser interpretado por máquina. RSL aborda o problema de um ponto de vista integrado. Os requisitos são estabelecidos como sequências de processos, que devem ser executados como respostas a estímulos. Estas sequências, são chamadas R- Nets (" Requirements Networks ")

REVS consiste de três segmentos:

1. um tradutor para RSL;
2. um banco de dados centralizado (ASSM - " Abstract System Semantic Model ");
3. um conjunto de ferramentas automatizadas para processar a informação no banco de dados.

A figura 3. mostra um diagrama do sistema. As descrições do sistema em RSL podem ser entradas iterativamente ou em batch e é verificada sua consistência e completeza.

O significado das declarações em RSL é então abstraído e armazenado no banco de dados. Estes dados podem ser utilizados pelas ferramentas. O conjunto básico de ferramentas em REVS contem auxílio para a validação, para verificar a completeza, a consistência e a correção dos requisitos e ferramentas orientadas para a análise de fluxo, que fornece entrada e saída gráfica, realiza análise estática e cria simuladores. A arquitetura flexível do REVS torna ainda possível extensões em dois sentidos: adições de novas ferramentas e criação de relatórios especiais [ALFORD 76], [ALFORD 77], [BELL 76], [BELL 77], [DAVIS 77].

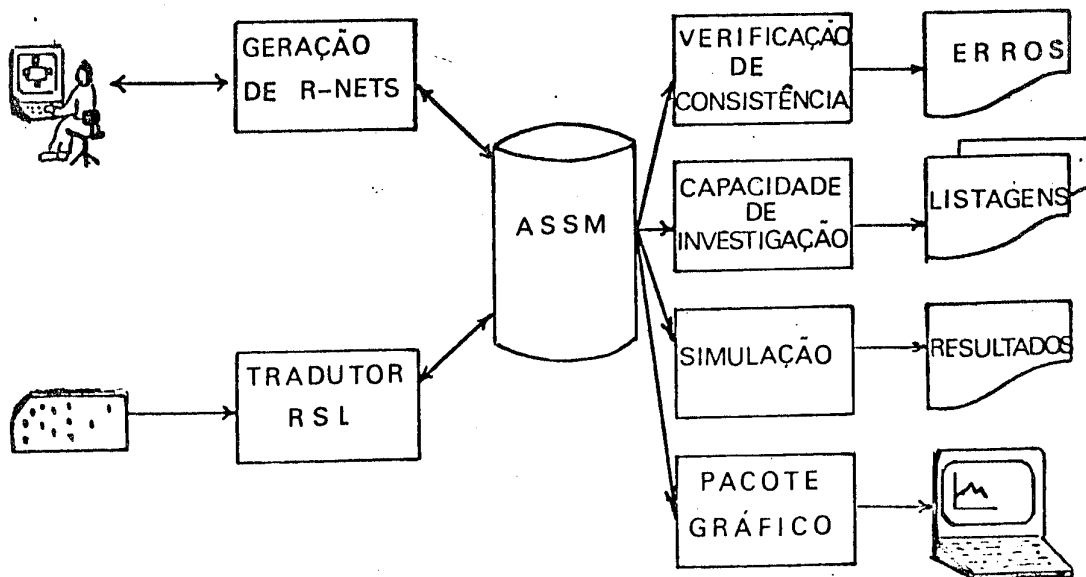


fig 3 - Metodologia SREM

4.3. PSL / PSA

PSL/PSA é uma técnica para documentação auxiliada por computador. Foi desenvolvida pela Universidade de Michigan (Projeto ISDOS) para auxiliar na análise, documentação de requisitos e preparação de especificações funcionais para sistemas de informação.

Como características da metodologia podemos assinalar:

1. Os resultados de cada uma das atividades no processo de desenvolvimento do sistema são armazenadas em forma processável por computador à medida que são produzidas.
2. Um banco de dados é usado para manter todos os dados básicos sobre o sistema
3. O computador é usado na produção de cópias da documentação

Os requisitos do sistema devem estar expressos numa linguagem não ambígua e possível de ser processada. Para atingir este objetivo foi desenvolvida a linguagem PSL (" Problem Statement Language "), uma linguagem para descrever sistemas.

À medida que as informações são obtidas, são expressas em PSL e armazenadas no banco de dados usando o " Problem Statement Analyzer " (fig 4). A partir daí a qualquer momento podem ser produzidas saídas e relatórios padronizados.

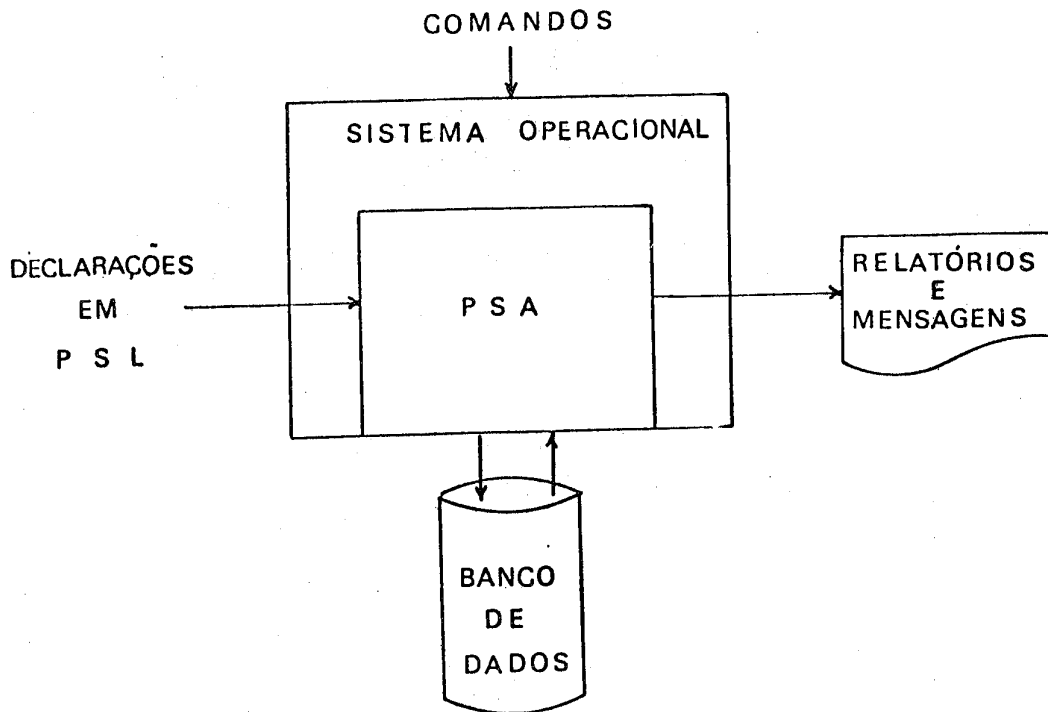


fig 4 - " Problem Statement Analyzer "
 fonte: [TEICHROEW 77]

O Analisador foi projetado para operar em ambiente iterativo usando as facilidades do sistema operacional hospedeiro. Pretende ser o mais independente possível do sistema. Para isto o Software, incluindo o DBMS, está escrito quase que totalmente em Fortran IV e pode ser instalado em qualquer ambiente que possua computador Fortran IV e memória suficiente [TEICHROEW 76], [TEICHROEW 77] .

4.4. SADT

A metodologia SADT (" Structured Analysis and Design Technique") desenvolvida pela SoftTech é uma técnica para especificar requisitos de sistemas utilizando diagramas.

SADT fornece meios para:

1. Pensar de maneira estruturada sobre problemas grandes e complexos.
2. Trabalhar em equipe
3. Comunicar resultados com uma notação clara e precisa
4. documentar resultados de maneira tal que forneça auditoria da história do projeto
5. Controlar exatidão, completeza e qualidade através de constantes revisões e procedimentos de aprovação.
6. Planejar e gerenciar o progresso

SADT é uma representação gráfica da estrutura hierárquica de um sistema, e o modelo é estruturado de modo a aumentar gradualmente o nível de detalhe (fig 5)

A linguagem gráfica SADT fornece um número limitado de primitivos a partir dos quais é possível compor estruturas. A notação consiste apenas de caixas e flechas. As caixas representam partes de um todo, e as flechas representam interfaces entre as partes. Assim sendo, as flechas não representam fluxo de controle ou sequência e sim mostram condições impostas às caixas. Os diagramas representam o todo e são compostos de caixas, flechas e nomes em linguagem natural. Um modelo SADT é uma sequência de diagramas [ROSS 77 a], [ROSS 77 b], [DAVIS 77].

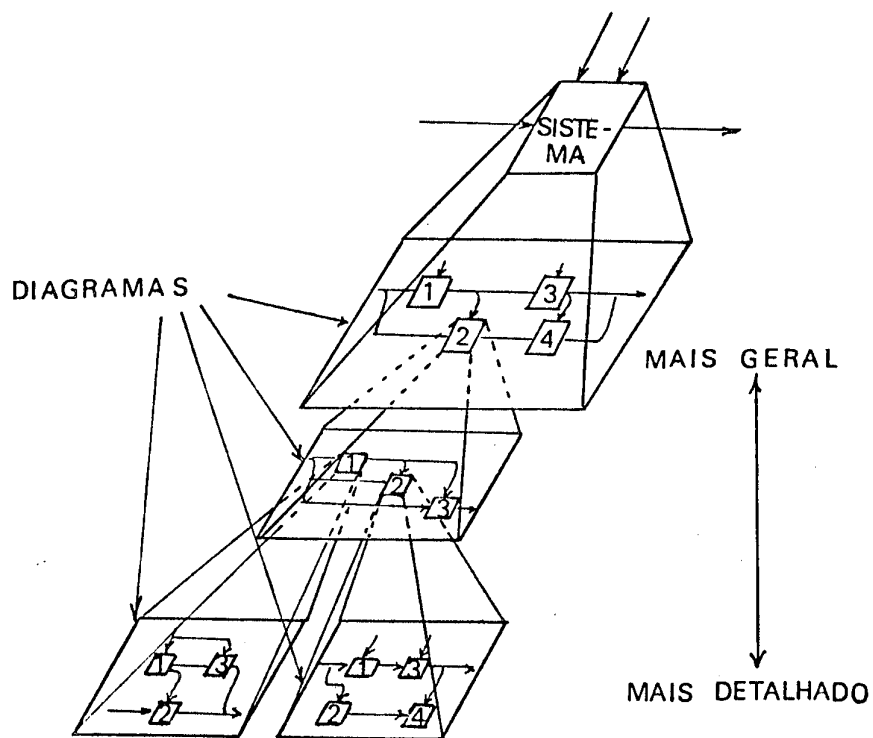


fig 5 - Decomposição estruturada

Fonte: [ROSS 77 b]

4.5. O Projeto SAFE

O projeto SAFE (" Specification Acquisition from Experts "), desenvolvido pela University of Southern California, tem como objetivo a tradução automática de especificações informais em especificações formais.

O sistema tem três fases (Linguística , Planejamento e Meta - Avaliação), que são chamadas sequencialmente para processar a especificação informal, cada fase usando os resultados das fases anteriores.

Como o objetivo do sistema é criar especificações formais a partir de informais , isto significa que ele deve completar cada uma das descrições parciais de entrada para produzir a saída. Geralmente cada descrição parcial pode ser completada de diferentes maneiras e torna-se necessária uma decisão separada para selecionar em cada caso a maneira correta de completar.

Baseado na descrição parcial e no contexto no qual ocorre, é criado um conjunto de possíveis conclusões para cada descrição parcial. Uma decisão, no entanto, não pode ser feita isoladamente de outras para que a especificação final tenha sentido como um todo. Para isto, após cada decisão o programa é testado para ver se obedece aos critérios. Caso isto se dê, continua até que todas as decisões tenham sido tomadas e o resultado aceito.

Este processo termina ou por se ter encontrado uma solução (especificação formal) ou porque se verificou que não é possível encontrá-la. O objeto resultante (programa) é uma solução aceitável (especificação formal) para o problema (especificação informal)[BALZER 78].

4.6. RDL

RDL (" Requirements and Development Language "); desenvolvida pela SPERRY UNIVAC, é uma linguagem não - procedural que permite ao usuário descrever todos os aspectos do desenvolvimento de Software desde a especificação de requisitos até a manutenção.

A linguagem, que sofreu influência da linguagem PSL (Projeto ISDOS), é usada incrementalmente para construir um banco de dados com todos os dados pertinentes ao projeto. O banco de dados é a principal fonte de informação para todos os grupos envolvidos no projeto e a partir dele são gerados relatórios e produzida a documentação [HEACOX 79].

4.7. SAMM / SAMDF

SAMM (" Systematic Activity Modeling Method ") é uma técnica de modelagem para descrição de requisitos que utiliza comunicação gráfico - linguística baseada em Matemática. Com esta propriedade pretende satisfazer as necessidades do técnico e do

cliente. O esquema de representação SAMP foi automatizado e sua automatização chama-se SAMMDF.

SAMP foi desenvolvida pela Boeing Company e sofreu influência do uso da Metodologia SADT. Baseia-se na necessidade de um método analítico de modo que o comportamento do sistema possa ser simulado, medido e avaliado.

O projeto da metodologia SAMP teve em conta três conceitos:

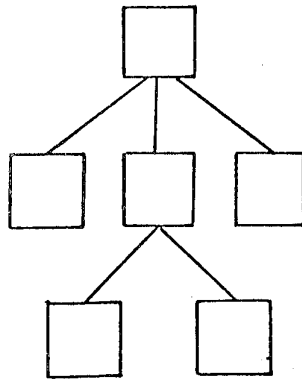
1. Refinamento semântico, que é um método de abstração hierárquica de significado
2. Racionalidade limitada, que se refere à limitação da capacidade humana e impõe restrições à quantidade de informação que pessoas podem receber, processar e lembrar-se
3. Teoria dos grafos

O objetivo da metodologia é modelar um sistema através de uma estrutura ordenada de atividades e fluxo de dados. Sua representação gráfica compreende três elementos (fig 6)

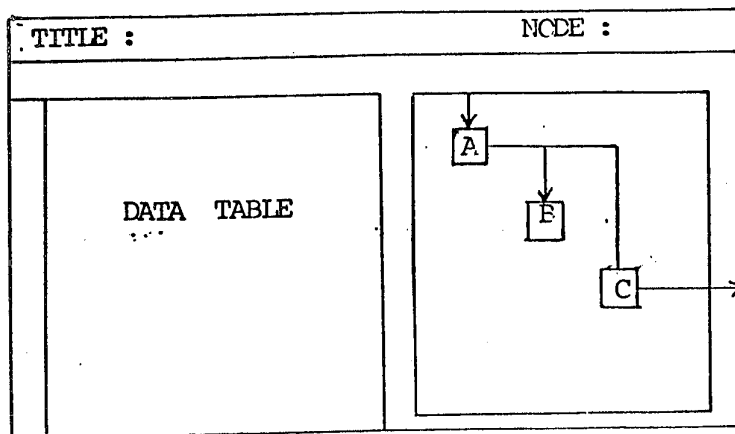
1. Uma estrutura de árvore que descreve o contexto do diagrama no sistema
2. Um "Activity diagram" que descreve o fluxo atividade-dados

3. Um " condition chart " , cujos objetivos são indicar quais as entradas necessárias para cada saída do diagrama de atividade e documentar o comportamento funcional do diagrama.

A metodologia fornece meios para a verificação do modelo através de teste de sintaxe e consistência , análise de conectividade do grafo e emissão de relatórios para avaliação [STEPHENS 78],[PETERS 78] .



Estrutura de árvore



"Activity Diagram"

TITLE:			NCDE:
OUTP	INP RQ.	CC	CONDITION DESCRIPTION

"Condition Chart"

fig 6 - Representação gráfica da metodologia SAMM

Fonte: [STEPHENS 78]

5. COMPARAÇÃO DAS METODOLOGIAS

5.1. Dimensão da comparação

Nosso objetivo é comparar as metodologias descritas tendo como base certos princípios e características que consideramos fundamentais. Partimos do objetivo básico de cada metodologia, de sua aplicação no ciclo de desenvolvimento do sistema e das características que lhes são essenciais. A partir daí fazemos uma comparação com relação aos requisitos que qualquer método de especificação deve satisfazer (seção 3.2).

Assim sendo analisamos cada metodologia quanto:

1. Objetivo básico
2. Aplicação no ciclo de desenvolvimento do sistema
3. Atributos Gerais
 - Características fundamentais
 - Linguagens de especificação disponíveis
 - Formalidade
 - Constructibilidade
 - Compreensividade
 - Minimalidade
 - Amplo campo de Aplicabilidade
 - Extensibilidade
 - Detalhabilidade
 - Automatizabilidade
 - Suporte à documentação

A comparação foi feita a partir desta análise. Para algumas destas metodologias possuímos referência de depoimentos de usuários [BAIL 79], [EDP ANALYZER DEZ 79]. Estes depoimentos foram levados em consideração no momento de ser feita a comparação, principalmente no que se refere aos atributos de constructibilidade, compreensividade e minimalidade. Também fazemos referência aos aspectos positivos e às dificuldades encontradas por estes usuários.

5.2. Comparação entre as metodologias

A análise das metodologias sobre os aspectos propostos está sintetizada no quadro da fig 7. Faremos agora algumas considerações sobre cada item comparado.

5.2.1. Objetivo básico das metodologias

Os objetivos indicam a própria natureza das metodologias. Pretendem ajudar no desenvolvimento de todo o sistema (HDM, RDL) ou no desenvolvimento da especificação de definição (SREM, PSL/PSA, SADT, SAFE, SAMM).

5.2.2. Aplicação ao ciclo de desenvolvimento

HDM e RDL são utilizadas durante todo o ciclo de desenvolvimento. SREM, PSL/PSA, SADT, SAFE e SAMM são utilizadas durante a fase de definição.

TECNOLOGIA	HDM	SPEM	PSL / PSA	SADT	SAFE	PDL	SPAM
OBJETIVO BASICO	Ajudar no desenvolvimento de sistemas grandes, complexos e onde é necessário alto grau de intergradação	Fornecer um método para especificação de requisitos e um sistema automatizado para verificar as especificações	Fornecer uma técnica auxiliadora para o computador para documentação de sistemas de informação	Facilitar a comunicação durante a especificação de requisitos	Tradução automática de especificações informais em especificações formais	Descrever todos os aspectos do desenvolvimento de software	Utilizar comunicação gráfica-linguística baseada em NLP para ajudar na definição de requisitos
APLICACAO	Em todo o ciclo	Fase de definição	Fase de definição	Fase de definição	Fase de definição	Em todo o ciclo	Fase de definição
DESENVOLVEDOR	Stanford Research Institute (SRI)	TRP	Universidade de Michigan (projeto ISX)	SYNTECH	Universidade de Southern California	STERN UNIVAC	STERN COMPANY
CAPACIDADES	<ul style="list-style-type: none"> Usa: -Decomposição hierárquica -Abstração de dados -Esp. formal -Modularidade -Verificabilidade (ferramentas de auxílio) 	<ul style="list-style-type: none"> -Especificação de requisitos em termos de fluxos -Possui conjunto de ferramentas (REVS, tradutor NPL, banco de dados) 	<ul style="list-style-type: none"> -Ênfase nos detalhes de documentação -Banco de dados com os dados básicos sobre o sistema 	-técnica gráfica - linguística	-Tradução automática	<ul style="list-style-type: none"> Usa: -técnica "top-down" -Esp. formal -Banco de dados com todos os dados pertinentes ao projeto -Influência NPL 	<ul style="list-style-type: none"> -Técnica gráfica - linguística baseada em NLP -Influência SADT
LINGUAGENS	SPECIAL HSL ILFL	NPL	NPL	- natural e gráfica	-natural	PDL	-natural e gráfica
PROBLEMAS	SEM	SEM	SEM	SEM	-arcs tradução	SEM	SEM
CONSTRUCOES DE	SEM	SEM	SEM	SEM	SEM	SEM	SEM
COMPRENSIVIDADE	SEM	SEM	SEM	SEM	SEM	SEM	SEM
INTERPRETACAO	SEM	SEM	SEM	SEM	SEM	SEM	SEM
ALGOS COMO DE	SEM	SEM	SEM	SEM	SEM	SEM	SEM
ALICABILIDADE	SEM	SEM	SEM	SEM	SEM	SEM	SEM
DETERMINABILIDADE	SEM	SEM	SEM	SEM	SEM	SEM	SEM
ADAPTABILIDADE	SEM	SEM	SEM	SEM	SEM	SEM	SEM
SUPORTE A DIVERSIFICACAO	SEM	SEM	SEM	SEM	SEM	SEM	SEM
ASPECTOS POSITIVOS	-Minimalidade	-PDL foi facilmente adaptada às necessidades do projeto -Maior entendimento das especificações	-Assegurou disciplina	F um veículo de comunicação -Ajuda na coerência -Facilita trabalho em equipe -Assigura disciplina -Minimiza erros	-Disciplina -Auxílio à gestão -Banco de dados com todos os dados do projeto -Aumenta a produtividade -Aumenta a qualidade	-Facilitou a produção e verificação de especificações	
DIFICULDADES ENCONTRADAS	-Linguagem SPECIAL (difícil produzir e ler especificações) -Documentação da Metodologia -Treinamento	-Treinamento -Portabilidade					

5.2.3. Características fundamentais

No que se refere às características fundamentais das metodologias notamos que há uma certa coincidência. Decomposição hierárquica é uma característica que de uma forma ou de outra sempre está presente. Também está sempre presente uma preocupação com a verificação e a modularidade. SADT e SAMP caracterizam-se por serem técnicas gráfico-linguísticas, sendo que SAMP tem profunda influência da metodologia SADT. RDL por sua vez sofreu influência de PSL. O projeto SAFE caracteriza-se por pretender uma tradução automática de especificações informais em especificações formais. HDM, SREM, SAMP possuem um conjunto de ferramentas de auxílio a verificação. Nas metodologias SREM, PSL/PSA e RDL, é construído um banco de dados com os dados pertinentes ao projeto.

5.2.4. Linguagens

HDM, SREM, PSL/PSA e RDL possuem linguagens formais de especificação. SADT e SAMP utilizam linguagem natural e gráfica e o projeto SAFE utiliza linguagem natural e faz sua tradução automática para uma linguagem formal

5.2.5. Formalidade

A formalidade está diretamente relacionada com o uso de linguagens formais. Assim sendo HDM, SREM, PSL/PSA e RDL tem um alto grau de formalidade. No projeto SAFE após a fase de tradução têm-se também especificações formais. SADT e SADM são técnicas gráficas e não possuem linguagens formais.

5.2.6. Constructibilidade e Compreensividade

Os requisitos de Constructibilidade e Compreensividade nem sempre tem sido atingidos pelas metodologias. Os usuários que relatam suas experiências apontam uma série de dificuldades. Em um painel na "Specification of Reliable Software Conference", reuniram-se usuários e desenvolvedores de quatro metodologias: HDM, SREM, SADT e PSL/PSA [BAIL 79].

Berson, da "Ford Aerospace" usuário da metodologia HDM observou que a metodologia é difícil de ser usada. Além disso a linguagem SPECIAL fez com que as especificações consumissem muito tempo e esforço para serem escritas e ainda mais para serem lidas.

Os usuários das metodologias SREM, SADT e PSL/PSA não apontaram dificuldades especiais quanto a constructibilidade e compreensividade.

5.2.7. Minimalidade

Minimalidade é um objetivo presente em todas as metodologias.

5.2.8. Ampla Campo De Aplicabilidade

Embora várias metodologias tenham sido desenvolvidas a partir da necessidade de auxílio ao desenvolvimento de um tipo específico de Software, em princípio todas pretendem ter um amplo campo de aplicabilidade.

HDM foi utilizada em vários projetos de diferente natureza, incluindo projetos de sistemas operacionais e projetos de controle de tráfego aéreo [EDP ANALYZER DEZ 79], [BAIL 79].

SREM foi desenvolvida pela TRW, com o objetivo de fornecer técnicas para aumentar a confiabilidade de Software produzido pelo " Ballistic Missile Defense Advanced Technology Center". Bail [BAIL 79] faz referência ao uso da metodologia SREM pelo JHU Applied Physics Laboratory, que observou que a metodologia foi

facilmente adaptada às necessi
dades do projeto.

PSL/PSA já foi usada em grande número de organizações' entre elas Chase Manhattan Bank, TRW, U.S. Air Force, IBM, UNIVAC. Também é utilizada por instituições acadêmicas com fi
nalidade de ensino e pesquisa [TEICHROEW 77].

SADT também tem sido apli
cada a diferentes tipos de pro
blema. Entre seus usuários ' podem-se citar, ITT Europa , ' TRAIDEX (treinamento militar), General Motors [ROSS 77 b] , [BAIL 79] .

Para SMM e RDL, só há re
ferência de uso interno por seus desenvolvedores.

5.2.9. Extensibilidade

Extensibilidade é um ob -
jetivo sempre presente, embora sejam ainda poucos os resulta-
dos experimentais que o compro
vem.

5.2.10. Detalhabilidade

Só HDM possui uma família de linguagens de especificação .

5.2.11. Automatizabilidade

Todas as metodologias , exceto SADT que é manual, oferecem um certo grau de automatismo .

5.2.12. Suporte à documentação

Todas as metodologias oferecem suporte à documentação. Os relatórios são geralmente fixos, a não ser na metodologia SREM, onde pela flexibilidade do conjunto de ferramentas REVS, é possível produzir relatórios flexíveis.

5.2.13. Observações de usuários

Existem referências de observações de usuários das metodologias HDM, SREM, PSL/PSA, SADT. No caso das metodologias RDL e SMM, só possuímos referência de seu uso interno na UNIVAC e Boeing Company [BAIL 79], [ROSS 77 b], [EDP ANALYZER DEZ 79], [TEICHROEW 77], [HEACOX 79], [STEPHENS 78] .

Os aspectos positivos mais comumente referenciados pelos usuários são:

- Minimalidade (HDM)
- Assegura disciplina (PSL/PSA, SADT, RDL, SMM)

- Facilidade de adaptação às necessidades do projeto (RDL)
- Comunicabilidade (SADT, SREM)
- Auxílio à gerência (SADT, RDL)
- Existência do banco de dados com todos os dados pertinentes ao projeto (RDL)
- Aumento de produtividade (RDL)
- Aumento de qualidade de Software (RDL, SADT)
- Facilita trabalho em equipe (SADT)
- Facilita produção e verificação de especificações (SAMM)

As dificuldades encontradas são principalmente:

- Dificuldade de produzir e ler as especificações utilizando a linguagem de especificação (HDM)
- Documentação (HDM)
- Treinamento (HDM, SREM)
- Portabilidade (SREM)

6. CONCLUSÃO

Como dissemos no início, nosso objetivo foi fazer uma comparação entre metodologias de especificação, tendo como base critérios pré-estabelecidos. Não cabe, portanto, fazer uma avaliação das mesmas, emitindo julgamentos. Quizermos, apenas destacar as semelhanças e diferenças entre seus atributos, aspectos positivos e dificuldades encontradas por seus usuários.

Não se pode dizer que uma metodologia seja melhor que as demais. Todas são potencialmente aplicáveis e to-

das possuem inadequações. Os resultados experimentais de sua utilização apresentam-se, no entanto, bastante animadores e tudo faz supor que nos próximos anos surgirão resultados interessantes nesta área de pesquisa.

REFERÊNCIAS BIBLIOGRÁFICAS

- [ALFORD 76] - Alford, M.W.; Burns, I.F. " R-Nets : a graph model for real-time software requirements", Proceedings of the Symposium on Computer Software Engineering , New York 1976
- [ALFORD 77] - Alford, M.W.; " A Requirements Engineering Methodology for Real-Time Processing Requirements", IEEE Transactions on Software Engineering, Vol. SE-3, nº 1, janeiro 1977
- [BAIL 79] - Bail, William G. "User experience with specifications tools", Panel from Specifications of Reliable Software Conference, ACM SIGSOFT Software Engineering Notes, vol 4 nº 3, julho 1979
- [BALZER 78] - Balzer, Robert; Goldman, Neil; Wile, David "Informality in Program Specifications" , IEEE Transactions on Software Engineering, vol SE- 4 , nº 2 março 1978

- [BELL 76] - Bell, T.E.; Bixler D.C. "A flow -oriented requirements Statement Language" Proceeding of the Symposium on Computer Software Engineering, New York 1976
- [BELL 77] - Bell, T.E.; Bixler, D.: Dyer, M. "An extendable approach to Computer- Aided Software Requirements Engineering", IEEE Transactions on Software Engineering, vol SE- 3 , nº 1, janeiro 1977
- [DAVIS 77] - Davis, G.; Vick, Charles R. "The Software Development System", IEEE Transactions on Software Engineering, vol SE-3, nº 1, janeiro 1977
- [EDP ANALYZER 79]- "Progress Toward System Integrity" EDP ANALYZER vol 17, nº12, dezembro 1979
- [HEACOX 79]- Heacox ,H.C. "RDL: A language for Software Development", ACM SIGPLAN Notices, Vol 14, nº 12, dezembro 1979
- [LISKOV 77]- Liskov, Barbara; Zilles, Stephen " An introduction to formal Specifications' of data abstractions", in Current trends in Programming Methodology, vol I, ed ' Yeh, Raymond T., Prentice-Hall Inc, New Jersey 1977

- [PETERS 78] - Peters, L: "Relating Software Requeriments and Design" Proceedings of the Software Quality And Assurance Workshop, San Diego, California, novembro 1978
- [ROCHA 81] - Rocha, Ana Regina C; Staa, Arndt von "Especificação e Verificação de Sistemas Automatizados, uma avaliação do estado da arte", Anales de la VIII Conferencia Latinoamericana de Informática, Buenos Aires, março 1981
- [ROSS 77 a] - Ross, D.T. - " Structured Analysis for Requirements Definition", IEEE Transactions on Software Engineering , vol SE- 3, nº 1 janeiro 1977
- [ROSS 77 b] - Ross, D.T. "Structured Analysis (SA): A language for Communications Ideas" IEEE Transactions on Software Engineering, vol SE-3, nº 1, janeiro 1977.
- [STEPHENS 78] - Stephens, S.A, Tripp L.L. "Requeriments Expression and Verification Aid", Proceedings of the 3rd International Conference on Software Engineering , Atlanta, Georgia, maio 1978
- [TEICHROEW 76]- Teichroew, D; Hersley III, E.A "Computer Aided Structured Documentation and Analysis of Information Processing System Requirements", ISDOS Project University of Michigan, agosto 1976

[TEICHROEW 77] - Teichroew, D.; Hersley, E.A.III;
" PSL/PSA: A Computer Aided'
Technique for Structured '
Documentation and Analysis of
Information Processing Systems",
IEEE Transactions on Software '
Engineering, vol SE-3 n° 1 , '
janeiro 1977