INFORMÁTICA 82
SUCESU

INFORMÁTICA 82

XV CONGRESSO NACIONAL DE INFORMÁTICA
II FEIRA INTERNACIONAL DE INFORMÁTICA

ANAIS

INFORMÁTICA 82

XV CONGRESSO NACIONAL DE INFORMÁTICA
II FEIRA INTERNACIONAL DE INFORMÁTICA

SUCESU

ANAIS

XV CONGRESSO NACIONAL DE INFORMÁTICA
SUCESU RIO DE JANEIRO – OUTUBRO 1982

UM MODELO PARA AVALIAÇÃO DE METODOLOGIAS E LINGUAGENS DE ESPECIFICAÇÃO

Ana Regina Cavalcanti da Rocha e Arndt von Staa
Departamento de Informática da Pontifícia Universidade Católica
Rio de Janeiro, RJ

1. INTRODUÇÃO

É fato conhecido que grande parte das inadequações observadas em sistemas automatizados, está relacionada a erros, omissões e inadequações nas especificações. O que é mais grave, erros nas especificações detectados em estágios avançados do desenvolvimento, ou mesmo durante a produção, exercem uma forte pressão sobre os custos. O desenvolvimento de software de boa qualidade depende, pois, da qualidade de suas especificações. É evidente que especificações de baixa qualidade impedirão a construção de sistemas automatizados de boa qualidade, independentemente do esmero na construção, documentação e gerência desta construção.

Procuramos identificar as propriedades que especificações devem possuir para que consigamos desenvolver sistemas automatizados de elevada qualidade. Estas propriedades, no entanto, não se atingem com a mera intenção por parte de seus desenvolvedores. A qualidade das especificações está diretamente relacionada à qualidade da engenharia utilizada em sua construção (metodologias e linguagens de especificação).

Neste trabalho propomos um modelo para avaliação da qualidade de metodologias e linguagens de especificação. Para sua definição partimos da definição dos objetivos de qualidade de software, relacionando-os com os objetivos de qualidade de especificações. Para que estes objetivos sejam atingidos, especificações devem possuir certos fatores de qualidade que, por sua vez, só serão atingidos se a engenharia empregada em sua produção possuir fatores de qualidade que o favoreçam.

O modelo de qualidade utilizado possui a mesma estrutura do apresentado por Mc Call (MC CALL 78). Diferencia-se, no entanto, no que se refere aos objetivos e fatores específicos utilizados para avaliar a qualidade de especificações.

Como exemplo de utilização do modelo para avaliação de metodologias e linguagens de especificação mostramos uma avaliação das metodologias SADT e SSA feita a partir do modelo de qualidade proposto. Para os não familiarizados com as metodologias fornecemos um breve resumo acompanhado de referências bibliográficas.

2. QUALIDADE DE SOFTWARE

Qualquer ramo de engenharia preocupa-se em desenvolver de forma econômica produtos com qualidade preditível e controlável. Assim sendo, também em engenharia de software têm-se buscado intensamente melhores conhecimentos com relação à qualidade de software. Para podermos identificar os atributos de qualidade de especificações e consequentemente que fatores de qualidade da engenharia podem favorecer seu alcance precisamos saber o que vem a ser qualidade de software, já que o nosso objetivo é construir software de qualidade preditível e controlada.

Para sistematizar qualidade de software utilizamos um modelo de predição e avaliação da qualidade. Este modelo é hierárquico e baseia-se em quatro conceitos fundamentais:

– *Objetivos de Qualidade* – que são as propriedades gerais que o produto (software) deve possuir,

– *Fatores de Qualidade do Produto* – que são condições e características determinantes da qualidade do ponto de vista do usuário,

– *Fatores de qualidade da Engenharia* – que são condições e características determinantes da qualidade da engenharia e da construção do sistema, e,

– *Métricas* – que são medidas quantificáveis de propriedades do software.

Quando nos referimos à qualidade de software, temos em conta os seguintes objetivos:

1. *Útil* – o software produz resultados oportunos e úteis. Os principais fatores de qualidade relacionados a este objetivo são:
   – satisfação das necessidades e desejos do usuário;
   – oportunidade;
   – confiabilidade.
2. *Utilizável* – o software é fácil de usar e projetado para operar em ambientes reais. Os principais fatores de qualidade relacionados a este objetivo são:
   – facilidade para operar;
   – ser robusto e seguro;
   – facilidade de preparar os dados e entender os resultados;

– proteção com relação a erros ou acidentes;
– amenidade ao usuário.

3. *Monitorável* – o software e seus resultados podem ser continuamente avaliados em relação ao consumo de recursos e à qualidade dos resultados. Os principais fatores de qualidade relacionados a este objetivo são:

– possibilidade de detectar erros decorrentes de uso inadequado, falhas, acidentes, fraudes, sabotagens, etc.;
– possibilidade de continuamente avaliar o grau de alcance dos objetivos;
– possibilidade de realizar auditoria.

4. *Evolutível* – o software pode ser mantido, expandido e reutilizado. Os principais fatores de qualidade relacionados a este objetivo são:

– manutenibilidade;
– flexibilidade, portabilidade;
– reutilizabilidade.

5. *Rentável* – o software consome apenas o necessário para atingir seus objetivos. Os principais fatores de qualidade relacionados a este objetivo são:

– compatibilidade entre custos e benefícios (financeiros e sociais);
– eficiência no uso de recursos.

Estes objetivos de qualidade, no entanto, só podem ser alcançados se seus desenvolvedores se empenharem para isto desde o início do processo de desenvolvimento, pondo-o sob controle. Para tal deverá ser especificado e validado o modelo de predição e avaliação da qualidade do software a ser desenvolvido. Ou seja, deverão ser definidos os fatores e métricas que caracterizam a qualidade do software, bem como devem ser definidas as tolerâncias destes fatores e métricas para o software em apreço.

3. QUALIDADE DE ESPECIFICAÇÕES

O desenvolvimento de um sistema automatizado se dá através de uma seqüência de etapas durante as quais são produzidos especificações e projetos, culminando em um projeto que permita a construção do sistema com um mínimo de risco. Estas especificações e projetos são produzidos através de refinamentos sucessivos, partindo de uma visão abrangente para uma visão detalhada.

Tendo em vista a inexistência de um modelo para avaliação da qualidade de especificações, aplicamos um método de aprendizado para identificar os fatores de qualidade de especificações. Ao aplicarmos este método tomamos por base as referências (ALFORD 76), (LISKOV 77), (WASSERMAN 82).

Quando falamos em qualidade de especificações temos em vista os seguintes objetivos de qualidade:

1. Evolutibilidade – que se refere à capacidade das especificações evoluirem em detalhe à medida que se avança no ciclo de vida;

2. Mensurabilidade – que se refere à capacidade de avaliar as propriedades inerentes às especificações;

3. Utilizabilidade – que se refere à capacidade de especificações servirem para predizer o comportamento e, após o desenvolvimento, avaliar o produto especificado;

4. Modificabilidade – que se refere à capacidade de modificar especificações sem que isto afete negativamente sua qualidade.

Para que estes objetivos sejam atingidos, devem ser realizados através dos seguintes fatores de qualidade de especificações:

1. *Detalhabilidade* – que se refere à facilidade que especificações devem ter para que sejam agregados detalhes a itens já parcialmente especificados;

2. Estrutura – que se refere à facilidade de percorrer-se o conjunto de especificações desde a visão mais geral até a mais detalhada e vice-versa;

3. Rastreabilidade – que se refere à facilidade de percorrer a seqüência de agregação de detalhes a um determinado aspecto desde sua visão mais parcial até a mais detalhada e vice-versa;

4. *Ser não-condicionante* – que se refere à inexistência de restrições com relação à escolha de alternativas em passos posteriores do processo de desenvolvimento;

5. Comunicabilidade – que se refere à facilidade de entendimento do conteúdo da especificação;

6. Modularidade de comunicação – que se refere à facilidade de compreensão dos elementos da especificação relacionados com determinado aspecto ou assunto. O usuário interessado em um determinado aspecto deverá poser concentrar-se exclusivamente neste aspecto e, ainda assim, deverá obter um perfeito entendimento de tudo relacionado com este aspecto;

7. Correção – todos os aspectos especificados estão formalmente corretos e/ou correspondem perfeitamente à percepção que o especificador tem do problema a ser resolvido;

8. Completeza – todos os aspectos que deveriam constar da especificação, efetivamente constam dela;

9. Necessidade – todos os aspectos tratados são imprescindíveis;

10. Uniformidade de detalhe – o grau de detalhe da abordagem dos diversos itens especificados é uniforme;

11. Viabilidade – a construção do software é possível dos pontos de vista econômico, técnico e administrativo;

12. Formalidade – que se refere ao rigor e precisão com que são definidos os aspectos especificados;

13. Verificabilidade – que se refere à facilidade de examinar-se a especificação com relação a normas e padrões, bem como com relação a especificações anteriores, caso existam;

14. Validabilidade – que se refere à facilidade de examinar-se a especificação com relação às necessidades e expectativas do usuário.

Os fatores de qualidade do produto, no entanto, não se atingem com a mera intenção por parte dos desenvolvedores. A qualidade do produto está diretamente relacionada à qualidade da engenharia e construção do sistema. Assim sendo, os fatores de qualidade do produto se realizam através dos fatores de qualidade da engenharia.

4. QUALIDADE DA ENGENHARIA DE ESPECIFICAÇÕES

Para auxiliar na realização dos objetivos e fatores de qualidade do produto (especificações), metodologias e linguagens de especificação devem possuir fatores de qualidade de acordo com os seguintes pontos de vista:

– a metodologia utilizada;
– a linguagem de especificação utilizada;

– o uso da linguagem de especificação.

1. Quanto à metodologia:

Para serem um auxílio eficaz à produção de especificações que assegurem a qualidade do produto especificado, metodologias devem:

1.1 – Servir de *apoio a todo o ciclo de desenvolvimento* facilitando a passagem de uma fase a outra, dado que atualmente se considera de pouca utilidade uma metodologia que apoie apenas uma parte do ciclo de vida,

1.2 – Possuir um *amplo campo de aplicabilidade,* isto é, espera-se que a metodologia seja suficientemente geral para que possa ser aplicada a diferentes classes de projetos,

1.3 – Dar *amplo suporte,* preferentemente automatizado, para:

– criação de especificações,
– verificação,
– validação,
– gerência.

2. Quanto à linguagem de especificação:

A partir dos fatores de qualidade do produto (especificações), uma linguagem de especificação deve possuir os seguintes fatores de qualidade:

2.1 – *Construtibilidade,* que se refere à facilidade para produzir a especificação desde que o especificador conheça a linguagem de especificação e entenda o problema a ser resolvido;

2.2 – *Inteligibilidade,* que se refere à facilidade do leitor entender o que foi especificado, desde que possua nível de formação e treinamento adequados;

2.3 – *Naturalidade,* que se refere à necessidade de que a linguagem de especificação seja natural à área do problema que está sendo especificado;

2.4 – *Formalidade,* que se refere à necessidade de que a linguagem de especificação seja formal com uma notação que permita a verificação formal;

2.5 – *Auto-verificação,* que se refere à facilidade oferecida pela linguagem de através de seu próprio uso auxiliar na detecção de inconsistências e ambigüidades;

2.6 – *Suporte para validação,* que se refere à facilidade oferecida pela linguagem que auxiliar na verificação e validação da especificação;

2.7 – *Capacidade de produzir índices,* que se refere à facilidade oferecida pela linguagem de auxiliar na criação de índices remissivos visando a facilitar o acesso a informações específicas contidas nos diversos documentos.

3. Quanto ao uso da linguagem de especificação:

Além dos fatores de qualidade relativos à metodologia e à linguagem de especificação é necessário, ainda, que o uso da linguagem satisfaça certos fatores de qualidade. Assim sendo, uma linguagem de especificação deve possibilitar seu uso de modo:

3.1 – *Hierárquico,* que se refere à necessidade de que a linguagem ajude o especificador a pensar de modo hierárquico;

3.2. – *Modular,* que se refere à necessidade de que a linguagem ajude o especificador a expressar seu pensamento de forma modular;

3.3 – *Conciso,* que se refere ao volume de informação conduzido por unidade de texto;

3.4 – *Consistente,* que se refere a não existência de contradições entre quaisquer aspectos especificados;

3.5 – *Não ambíguo,* que se refere à necessidade de não ser deixada margem a interpretações por parte do leitor para qualquer um dos aspectos especificados;

3.6 – *Uniforme,* que se refere ao uso de simbologia, notação e termos técnicos de modo uniforme em todos os documentos produzidos.

A figura 2 mostra a relação entre:

a. objetivos de qualidade de software e objetivos de qualidade de especificações,

b. fatores de qualidade de especificações visando o alcance dos objetivos de qualidade das especificações e satisfazendo, assim, indiretamente os objetivos de qualidade do produto,

c. fatores de qualidade da engenharia de especificações visando satisfazer os fatores de qualidade de especificações.

A figura 3 dá uma visão geral dos fatores envolvidos na realização de cada um dos objetivos de qualidade de especificações. É importante observarmos que esses fatores muitas vezes são conflitantes. Atingir um deles em maior grau pode significar diminuir o grau de alcance de outro ou outros fatores. Na realidade o que se deve procurar é um equilíbrio entre os mesmos, de acordo com as características próprias de cada projeto e com a fase do processo de desenvolvimento.

5. DESCRIÇÃO DAS METODOLOGIAS SADT E SSA

5.1 SADT (ROSS 77a),(ROSS 77b), (BAIL 79)

SADT (Structured Analysis and Design Technique), desenvolvida por Douglas Ross (SoftTech) é uma técnica para especificar requisitos de sistemas utilizando diagramas de fluxo de dados ("actigram") e de estados de dados ("datagram"). SADT é útil também para a fase de projeto. Consiste de:

1. um conjunto de métodos que auxiliam a pensar de uma maneira estruturada sobre problemas grandes e complexos, utilizando refinamentos sucessivos,

2. uma linguagem gráfica para comunicar especificações de uma maneira clara e precisa, e,

3. auxílios para o planejamento e gerência do progresso.

Uma especificação utilizando SADT é uma representação gráfica da estrutura hierárquica de um sistema, sendo esta representação estruturada de modo a aumentar gradualmente o nível de detalhe (figura 4).

A linguagem gráfica SADT fornece um número limitado de primitivos a partir dos quais compõem-se estruturas. A notação consiste de caixas que representam partes de um todo e indicam:

1. uma atividade (caso se trate de um "actigram"),
ou 2. um objeto (no caso de ser um "datagram").

As flechas representam interfaces entre as partes e,

1. no caso de "actigram" descrevem os dados, controles e instrumentos necessários para cada atividade, e possivelmente gerados para serem consumidos em uma atividade posterior.

2. no caso de "datagram" descrevem os processos, controle e meios de armazenagem necessários para criar, manter e controlar os diversos objetos (dados).

A preocupação dos diversos diagramas concentra-se

no fluxo de transformação de dados, sendo inexistente a preocupação com o fluxo de controle (fluxograma tradicional).

As figuras 5 e 6 mostram exemplos de "actigram" e de "datagram".

SADT vem sendo utilizada desde 1974 por diversas organizações. Seus usuários apontam como aspectos positivos o alto grau de comunicabilidade, o aumento da qualidade do software produzido e o fato de assegurar disciplina e facilitar a gerência e o trabalho em equipe.

5.2 SSA (GANE 79), (GANE 80), (DE MARCO 78)

SSA (Structure Systems Analysis) é um conjunto de técnicas e ferramentas cujo objetivo é auxiliar na análise e definição de sistemas. Como no caso de SADT, o conceito fundamental é a construção de um modelo do sistema utilizando técnicas gráficas. A metodologia envolve a construção "top-down" do sistema por refinamentos sucessivos.

SSA consiste em (figura 7).

1. Diagramas de fluxos de dados;
2. Dicionário de dados, com detalhes sobre todos os dados utilizados no fluxo;
3. Ferramentas para descrever a organização lógica dos processos (tabelas de decisão, árvores de decisão, pseudo linguagens de programação, linguagem natural estruturada);
4. Ferramentas para descrever a organização lógica dos dados (diagrama de acesso indicando como localizar um determinado dado).

Estes documentos fornecem uma descrição do sistema, sua especificação funcional lógica, estabelecendo detalhadamente o que o sistema faz, sendo o mais independente possível de considerações físicas sobre a implementação.

A partir do modelo lógico é posteriormente produzido um projeto ("design"). A metodologia mais apropriada é a conhecida como "Structured Design", desenvolvida por Larry Constantine e posteriormente refinada por Myers (MYERS 75) e Yourdon (YOURDON 79).

Os aspectos positivos da metodologia SSA mais comumente apontados são o alto grau de comunicabilidade, a redução do tempo de desenvolvimento do sistema e o aumento da qualidade do sistema produzido.

6. AVALIAÇÃO DAS METODOLOGIAS SADT E SSA SEGUNDO O MODELO DE QUALIDADE

Nesta seção mostramos, como exemplo de utilização do modelo de qualidade para a avaliação de metodologias e linguagens de especificação, uma avaliação das metodologias SADT e SSA segundo os fatores de qualidade da engenharia (figura 8).

7. CONCLUSÃO

Neste trabalho foi mostrada a relação entre qualidade de software, qualidade de especificações e qualidade de engenharia de especificações. Foi também apresentado um modelo para avaliação da engenharia de especificações (metodologias e linguagens), definido a partir de um modelo para avaliação e predição da qualidade de software.

Este modelo foi utilizado para a avaliação das metodologias SADT e SSA. O modelo de avaliação de qualidade apresentado é certamente útil para os que necessitam comparar ou selecionar metodologias.

REFERÊNCIAS BIBLIOGRÁFICAS

(BAIL 79) – Bail, W.G. – "User Experience with Specifications Tools" – Panel from Specifications of Reliable Software Conference, ACM SIGSOFT, Software Engineering Notes, vol. SE-2, n. 3, julho 1979.

(BARROS 82) – Barros, S.C. – "Controle de Qualidade aplicado ao desenvolvimento de software". Dissertação de Mestrado, Informática, PUC-RJ, 1982.

(DE MARCO 78) – De Marco, T. – "Structured Analysis and System Specification" – Yourdon Press, N.Y. 1978.

(GANE 79) – Gane, C.; Sarson, T. – "Structured Systems Analysis: Tools and Techniques". – Prentice-Hall Inc. N.J. 1979.

(GANE 80) – Gane, C. – "Data Design in Structured Systems Analysis" – in Tutorial on Software Design Techniques, Freeman, P.; Wasserman, A.I. eds., IEEE Computer Society, 1980.

(MAZZONI 81) – Mazzoni, C.J. – "Um modelo para avaliação da qualidade de software". Tese de Mestrado, Departamento de Informática, PUC/RJ, outubro de 1981.

(MC CALL 78) – Mc Call, J. – "An Introduction to Software Quality Metrics". in software Quality Management, Cooper, J.D.,; Fisher, M.J. ed., Petrocelli Books, 1978.

(MYERS 75BD) – Myers, G.J. – "Reliable Software Trough Composite Design" – Petrocelli/Charter, N.Y. 1975.

(ROSS 77a) – Ross, D.T. – "Structured Analysis for Requirements Definition" – IEEE Transactions on Software Engineering, vol. SE-3, n. 1, jan 1977.

(ROSS 77b) – Ross, D.T. – "Structured Analysis (SA): a Language for Communicating Ideas". IEEE Transactions on Software Engineering, vol. SE-3, n. 1, jan. 1977.

(YOURDON 79) – Yourdon E.; Constantine, L.L. "Structured Design", Prentice-Hall, N.J. 1979.

(WASSERMAN 82) – Wasserman, A.I. – "Automated Tools in the Information System Development Environment" in Automated Tools for Information Systems Design, Schneider, H.J.; Wasserman, A. I., ed., North-Holland Publishing Company, IFIP 82.

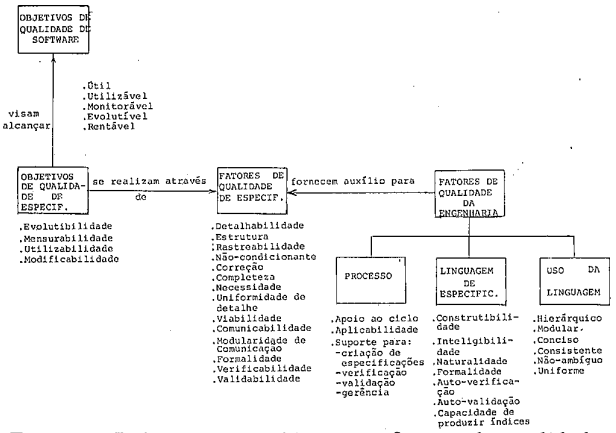

Figura 2. Relação entre objetivos e fatores de qualidade.

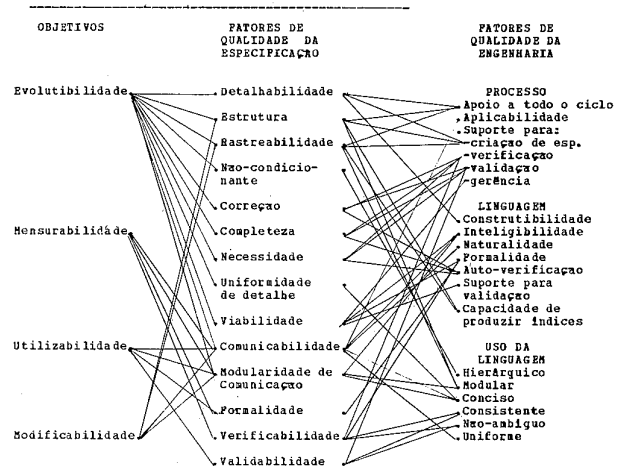

Figura 3. Qualidade de especificações.

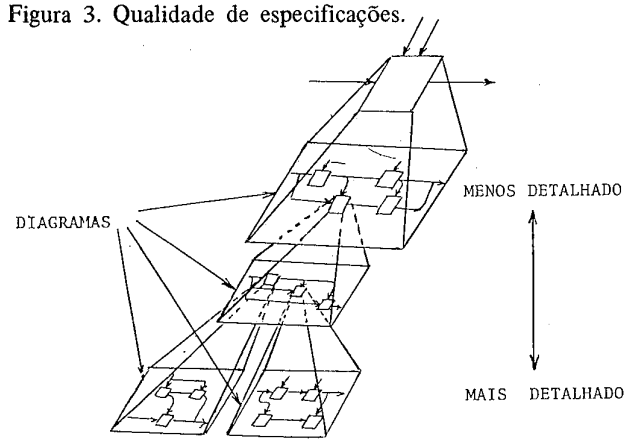

Figura 4. Estrutura hierárquica SADT (Fonte: ROSS 77b).

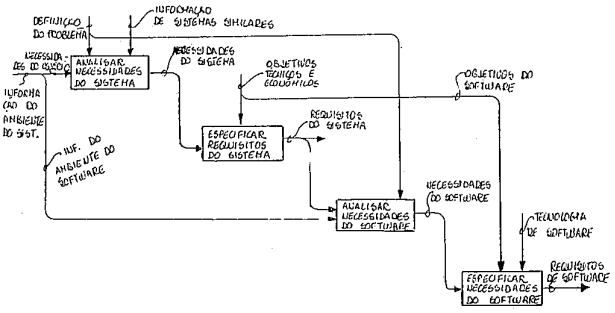

Figura 5. Exemplo de "actigram".

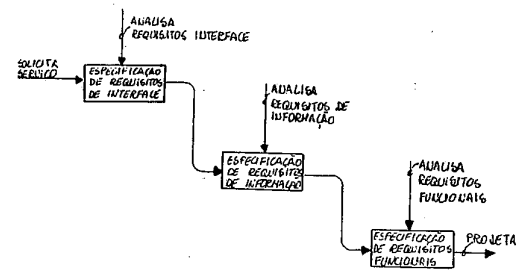

Figura 6. Exemplo de "datagram".

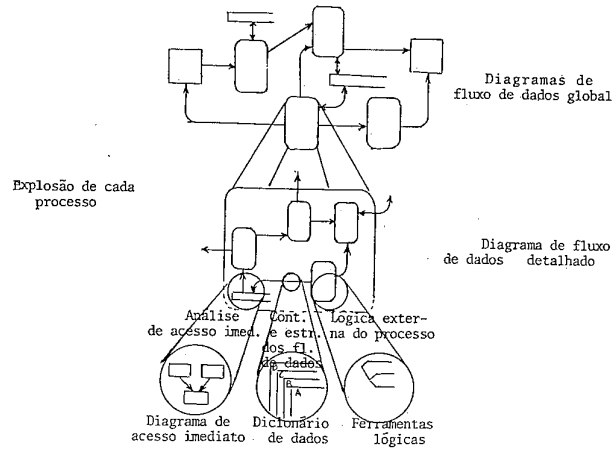

Figura 7. Structured Systems Analysis. Fonte: (GANE79)

	SADT	SSA
QUANTO AO PROCESSO		
.Apoio a todo o ciclo	N	N*
.Amplo campo de aplicabilidade	S	S
.Suporte automatizado para:		
- criação de especificações	N	N
- verificação	N	N
- validação	N	N
- gerência	N	N
QUANTO A LING. DE ESPECIFICAÇÃO		
.Construtibilidade	M	G
.Inteligibilidade	G	G
.Naturalidade	G	G
.Formalidade	M	P
.Auto-verificação	M	P
.Suporte para validação	M	M
QUANTO AO USO DA LINGUAGEM		
.Hierárquico	S	S
.Modular	S	S
.Conciso	S	S
.Consistente	S	S**
.Não ambíguo	S	S**
.Uniforme	S	S

Figura 8. Avaliação da metodologia SADT segundo os fatores de qualidade da engenharia.

Legenda: S - Sim; N - Não; G - Grande; M - Médio; P - Pequeno.

* A técnica Structured Design parte dos diagramas de fluxo de dados. A composição de SSA com Structured Design fornece pois apoio a todo o ciclo.

** Em SSA, dado o uso de um dicionário de dados separado, existe um risco maior do que em SADT de inconsistências entre o fluxo e o dicionário de dados.