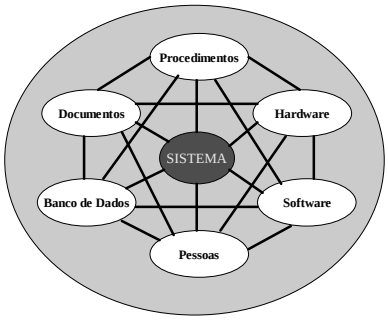
Análise de Requisitos de Software e de Sistemas

Profa Luciana Romani

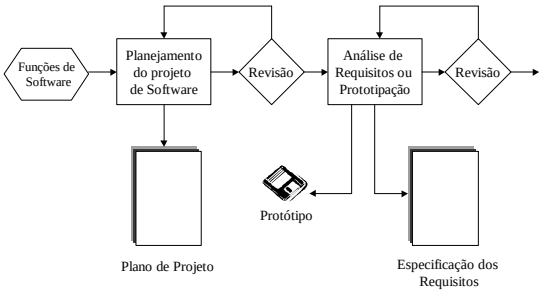
1

Elementos de um sistema

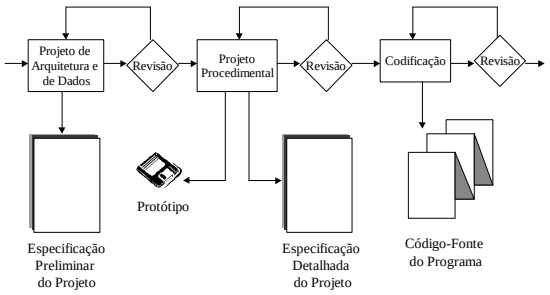


2

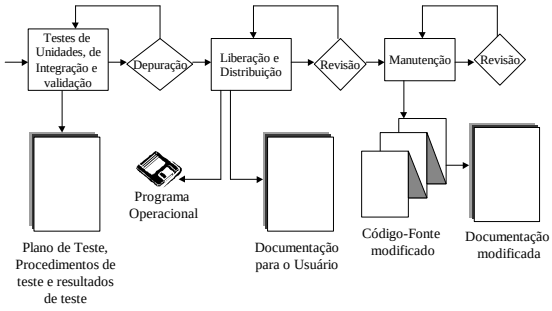
Engenharia de Software: Fase de Definição



Engenharia de Software: Fase de Desenvolvimento



Engenharia de Software: Fase de Verificação, Liberação e Manutenção



Fase de Análise de Requisitos

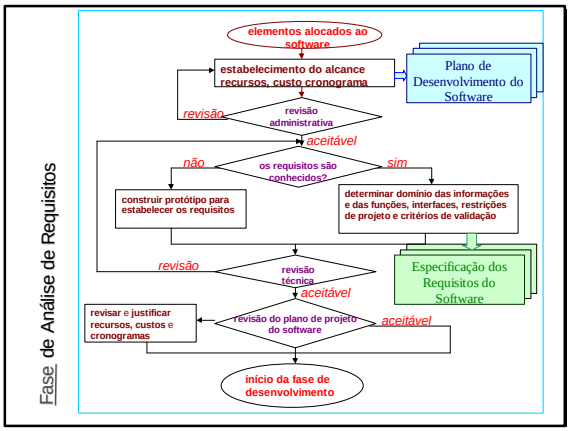
- processo de descoberta e refinamento
  - ATORES: cliente e desenvolvedor
  - PROBLEMA: grande propensão a mal entendidos
- "atividade aparentemente simples torna-se complexa"

6

Atividades de Análise:

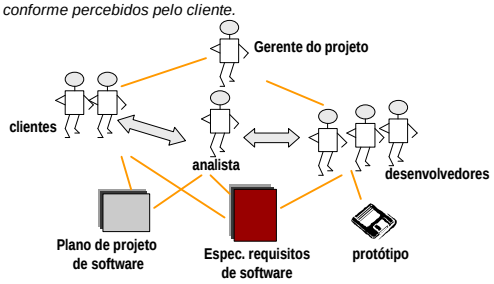
- reconhecimento do problema
- avaliação do problema
- síntese da solução (modelagem)
- especificação dos requisitos do software
- revisão

7



Atividade 1:
Reconhecimento do Problema

A meta é o reconhecimento dos elementos básicos do problema, conforme percebidos pelo cliente.



Atividade 2:
Avaliação do Problema e Síntese da Solução

- Avaliar os problemas na situação atual
- Para o novo sistema:
  - definir e elaborar todas as funções do sistema
  - identificar dados que o sistema produz e consome
  - entender o comportamento do sistema
  - estabelecer características de interface
  - descobrir restrições do projeto

10

Atividade 2:
Avaliação do Problema e Síntese da Solução

- Sintetizar uma ou mais soluções (dentro do alcance delineado no Plano de Projeto do Software)
  - O processo de avaliação e síntese continua até que o analista e o cliente concordem que o software pode ser adequadamente especificado.
  - É a maior área de esforço
- Modelagem
  - Durante a atividade de avaliação e síntese devem ser criados **modelos do sistema** para se compreender melhor o fluxo de dados e de controle, o processamento funcional e a operação comportamental, além do conteúdo da informação.
  - O modelo serve como fundamento para o projeto de software e como base para a criação de sua especificação

11

Atividade 3
Especificação de Requisitos

- descrição do fluxo e estrutura da informação
- refinamento detalhado de todas as funções do software
- estabelecimento das características de interface
- identificação das restrições de projeto
- especificação dos critérios de validação

12

Atividade 4: Revisões

- Devem ser efetuadas revisões técnicas e revisões no Plano de Projeto de Software
  - as revisões são conduzidas pelo Cliente e pelo Desenvolvedor
  - a base para a revisão são os documentos produzidos na Especificação dos Requisitos
- O Plano de Projeto do Software deve ser revisto devido ao conhecimento adquirido durante a análise.

13

Características do Analista de Sistemas

1) Capacidade para compreender conceitos abstratos, reorganizar esses conceitos em divisões lógicas e sintetizar "soluções" baseado em cada divisão.
2) Capacidade de absorver fatos pertinentes a partir de fontes conflitantes ou confusas.
4) Capacidade de se comunicar bem de forma escrita e verbal.
5) Capacidade de "ver a floresta ao invés das árvores"

14

Áreas Problemas

1. Aquisição da informação
   - que informação deve ser coletada e como ela deve ser representada?
   - quem fornece as informações?
   - que técnicas e ferramentas estão disponíveis para facilitar a coleta de informações?

15

Áreas Problemas

2. Tamanho do sistema
   - como eliminar inconsistências na especificação de grandes sistemas?
   - é possível detectar omissões?
   - pode um grande sistema ser efetivamente particionado para que se torne intelectualmente administrável?

16

Áreas Problemas

3. Alterações
   - como as alterações efetuadas em outros elementos do software são coordenadas com os requisitos do software?
   - como se determina o impacto de uma alteração em outras partes do software aparentemente não relacionadas?
   - como se corrige erros na especificação para que não se gere efeitos colaterias?

17

Causas dos Problemas

- comunicação ineficiente
- técnicas e ferramentas inadequadas
- tendências de eliminar a tarefa de Especificação dos Requisitos
- falha de considerar alternativas antes que o software seja especificado

18

Princípios de Análise (4)

- **domínio de informação** do problema → representado e compreendido (para que a função possa ser entendida + completamente)
- **modelos** que descrevam a informação, a função e o comportamento do sistema → desenvolvidos (para que a informação possa ser comunicada compactamente)

19

Princípios de Análise (4)

- modelos (e o problema) → **particionados**, de maneira que revele os detalhes em forma de camadas (ou hierarquicamente) (para reduzir a complexidade)
- processo de análise → mover-se da informação **essencial para os detalhes** de implementação (para acomodar as restrições lógicas impostas por requisitos de processamento e as restrições físicas impostas por outros elementos do sistema)

20

1. Princípio - **Domínio da Informação**

- Todo software é construído para processar dados e eventos.
- Os dados e itens de controle residem no domínio de informação de um problema.
- 3 diferentes pontos de vista:
  - **Fluxo da Informação**: maneira pela qual os dados e o controle se modificam à medida que cada um se movimenta pelo sistema
  - **Conteúdo da Informação**: os dados e os itens de controle individuais que compreendem certo item de informação mais amplo.
  - **Estrutura da Informação**: a organização interna de vários itens de controle e de dados

21

2. Princípio - **Modelagem**

- O modelo deve ser capaz de modelar a informação que o software transforma, as funções (ou subfunções) que possibilitam que as transformações ocorram e o comportamento do sistema quando a transformação está se desenvolvendo.
- Os modelos concentram-se naquilo que o sistema deve fazer, não em como ele faz.
- Papéis importantes do Modelo:
  - ajuda o analista a entender a informação, a função e o comportamento de um sistema, tornando a tarefa + fácil e sistemática.
  - torna-se o ponto focal para a revisão e, portanto, a chave para a determinação da completitude, consistência e precisão da especificação.
  - torna-se a base para o projeto, fornecendo ao projetista uma representação essencial do software, a qual pode ser "mapeada" num contexto de implementação.

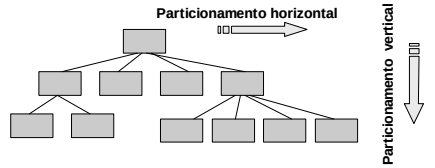
3. Princípio - **Particionamento**

- Os problemas freqüentemente são grandes demais e muito complexos para serem compreendidos como um todo.
- O particionamento divide o problema em partes mais facilmente entendidas
- Através das interfaces estabelecidas entre as partes, a função global do software pode ser executada.

23

3. Princípio - **Particionamento**

- Particionamento Horizontal: decomposição funcional do problema
- Particionamento Vertical: expõe detalhes crescentes



24

4. Princípio - **Concepções essenciais e de implementação**

- A concepção essencial dos requisitos do software apresenta as funções a serem realizadas sem tratar dos detalhes de implementação.
- Ao se concentrar atenção na essência do problema nas primeiras etapas da análise de requisitos, deixa-se as opções abertas para especificar detalhes de implementação durante as últimas etapas de especificação dos requisitos e projeto de software.
- A concepção de implementação dos requisitos de software apresenta a manifestação das funções de processamento e estruturas de informação no mundo real.
- Não deve ser interpretada como uma representação do como. Um modelo de implementação representa o modo de operação corrente, ou seja a atribuição existente ou proposta para todos os elementos do sistema.

Princípios de uma boa especificação (Balzer e Goldman)

1. Separe funcionalidade de implementação
2. É necessária uma linguagem de especificação de sistemas orientada ao processo
3. A especificação deve abranger o sistema do qual o software é um componente
4. Uma especificação deve abranger o ambiente no qual o sistema opera
5. Uma especificação de sistema deve ser um modelo cognitivo
6. Uma especificação deve ser operacional
7. A especificação do sistema deve ser tolerante com a não completitude e ser expansível
8. Uma especificação deve ser localizada e fracamente acoplada.

26

Formato da Especificação de Requisitos

I. **Introdução** - declara as metas e os objetivos do software, descrevendo-os no contexto do sistema baseado em computador
II. **Descrição da Informação** - descrição detalhada do problema que o software deve resolver
III. **Descrição Funcional**
IV. **Descrição Comportamental**
V. **Critérios de Validação**
VI. **Bibliografia**
VII. **Apêndice**

*A Especificação pode ser acompanhada de um **PROTÓTIPO** executável (ou em papel) e/ou um **MANUAL PRELIMINAR DE USUÁRIO.***

27

Revisão da Especificação (nível macroscópico)

- Os revisores tentam garantir que a especificação seja completa, consistente e precisa. Questões:
  - Metas e objetivos do software permanecem consistentes com metas e objetivos do sistema?
  - Foram descritas as interfaces importantes para todos os elementos do sistema?
  - O fluxo e a estrutura de informação são adequadamente definidas para o domínio da informação?
  - Os diagramas são claros?

28

Revisão da Especificação (nível macroscópico)

- As funções importantes permanecem dentro do escopo e cada uma foi adequadamente descrita?
- O comportamento do software é consistente com a informação que ele deve processar e as funções que deve executar?
- As restrições de projeto são realísticas? Qual é o risco tecnológico de desenvolvimento? Requisitos de software alternativos foram considerados?
- Critérios de Validação foram declarados detalhadamente? Eles são adequados para descrever um sistema bem sucedido?
- Existem inconsistências, omissões ou redundâncias?
- O usuário revisou o Manual Preliminar ou o protótipo?
- Como as estimativas do Plano de projeto de Software foram afetadas?

29

Revisão da Especificação (nível detalhado)

- A preocupação é com o enunciado da especificação. Tenta-se descobrir problemas que possam estar ocultos no conteúdo da especificação
- Diretrizes:
  - Esteja alerta para perceber conectivos persuasivos e perguntar por que eles estão presentes.
  - Procure termos vagos e peça esclarecimento
  - Quando forem fornecidas listas que não sejam completas, certifique-se de que todos os itens sejam entendidos
  - Esteja certo de que os limites declarados não contenham pressuposições não declaradas

30

Revisão da Especificação (nível detalhado)

- Diretrizes:
  - Cuidado com verbos vagos. Há muitas maneiras de interpretá-los.
  - Cuidado com pronomes "pendentes".
  - Procure declarações que impliquem certeza e depois peça prova
  - Quando um termo for explicitamente definido num lugar, evite utilizar outras definições para o mesmo termo
  - Quando uma estrutura for descrita em palavras, verifique se há um gráfico ou uma figura para auxiliar a compreensão
  - Quando um cálculo for especificado, desenvolva pelo menos dois exemplos.

31

Ferramentas de Especificação Automatizadas

- 1ª categoria: técnicas automatizadas que nada mais são do que um método manual que foi complementado com uma ferramenta CASE
  - Possibilitam que o analista atualize informações e rastreie as conexões entre representações novas e existentes do sistema
  - Ex: DEC Design (Digital Equipment Corp.), Design Aid (Transform Logic Corp.), Excelerator (Intersolv), IEF (Texas Instruments), ADW (Knowledgeware), STP (Interative Development Environments), Teamwork (Cadre Technologies).

32

Ferramentas de Especificação Automatizadas

- 2ª categoria: técnicas automatizadas que fazem uso de uma notação especial (na maioria dos casos, essa é uma linguagem de especificação de requisitos) que foi explicitamente projetada para processamento usando-se uma ferramenta automatizada.
  - Ex: SREM (linguagem de especificação: RSL), PSL/PSA (linguagem de especificação: PSL), TAGS (linguagem de especificação: IORL)

33

Conclusão

- Logo que a Revisão for concluída, a Espec. de Requisitos de Software é "assinada" pelo cliente e pelo desenvolvedor
- A especificação torna-se um "contrato" de desenvolvimento de software.
- Mudanças solicitadas depois que a Espec. for concluída serão consideradas, porém cada mudança posterior pode aumentar o custo e/ou alongar o prazo de entrega
- Mesmo com os melhores procedimentos de revisão em andamento, uma série de problemas de especificação ainda persiste

34