1ª Frequência de MDS

Tópicos

Desenvolvimento profissional de software

O que é software?

São programas de computador e documentação associada. Porodutos de software podem ser desenvolvidos para um cliente em especifico ou para o mercado em geral.

Quais são as principais atividades da engenharia de software?

- Especificação de software, em que clientes e engenheiros definem o software a ser produzido e as restrições de sua operação.
- Desenvolvimento de software, em que o software é projetado e programado.
- Validação de software, em que o software é verificado para garantir que é o que o cliente quer.
- Evolução de software, em que o software é modificado para refletir a mudança de requisitos do cliente e do mercado.

Quais os principais desafios da engenharia de software?

- Lidar com o aumento de diversidade, tempos de entrega cada vez menores e desenvolvimento de software de confiança.

Quais diferenças foram feitas pela internet na engenharia de software?

A internet tornou serviços de software disponíveis e possibilitou o desenvolvimento de sistemas altamente distribuídos baseados em serviços. O desenvolvimento de sistemas baseados em Web gerou importantes avanços nas linguagens de programação e reúso de software.

Produtos genéricos e customizados

Engenheiros de software preocupam-se em desenvolver produtos de software (ou seja, software que pode ser vendido para um cliente). Existem dois tipos de produtos de software:

1. **Produtos genéricos**. Produzidos por uma organização de desenvolvimento e vendidos no mercado para qualquer cliente que esteja interessado em comprá-los.
2. **Produtos customizados**. Estes são os sistemas encomendados por um cliente em particular.

Uma diferença importante entre esses tipos de software é que, em softwares genéricos, a organização que o desenvolve controla a sua especificação. Para produtos sob encomenda, a especificação é

normalmente desenvolvida e controlada pela empresa que está adquirindo o software.

Atributos essenciais de um bom software:

- Manutenção fácil. O software deve ser escrito de forma que possa evoluir para atender às necessidades dos clientes.
- Confiança e proteção. A confiança do software inclui uma série de características como confiabilidade, proteção e segurança.
- Eficiência. O software não deve desperdiçar os recursos do sistema.
- Aceitação. Deve ser compreensível, usável e compatível com outros sistemas usados por ele.

Engenharia de software e a Web

Serviços web permitem que as aplicações possam ser acedidas através da web.
O Cloud computing é uma abordagem para provisionar serviços de computação. Os utilizadores não compram software, pagam de acordo com a sua utilização e as aplicações são executadas remotamente na “cloud”.
O surgimento da Internet trouxe mudanças radicais na organização de software e obviamente causou mudanças na maneira como os sistemas Web são projetados. Por exemplo:

- A reutilização de software é a abordagem dominante na construção de sistemas Web.
- Sistemas web devem ser desenvolvidos e entregues de forma incremental.
- Interfaces dos utilizadores estão limitados às capacidades dos browsers.

Processos de Software

- Conjunto de atividades usado para desenvolver software.

Não existe um método ou uma técnica universal de engenharia de software que se aplique a todos. No entanto, há quatro aspetos gerais que afetam vários tipos diferentes de software:

1. Heterogeneidade. Cada vez mais se requer dos sistemas que operem como sistemas distribuídos através das redes que incluem diferentes tipos de computadores e dispositivos móveis.
2. Alterações sociais e das regras de negócio.
3. Segurança e confiança.
4. Escala.

Processos baseados em planos e processos ágeis

Os processos de software, às vezes, são categorizados como baseados em planos ou processos ágeis. **Processos baseados em planos** são aqueles em que todas as atividades são planeadas com antecedência, e o progresso é avaliado por comparação com o planeamento inicial. Em **processos ágeis**, o planeamento é incremental, e é mais fácil alterar o processo de maneira a refletir as necessidades de mudança dos clientes.

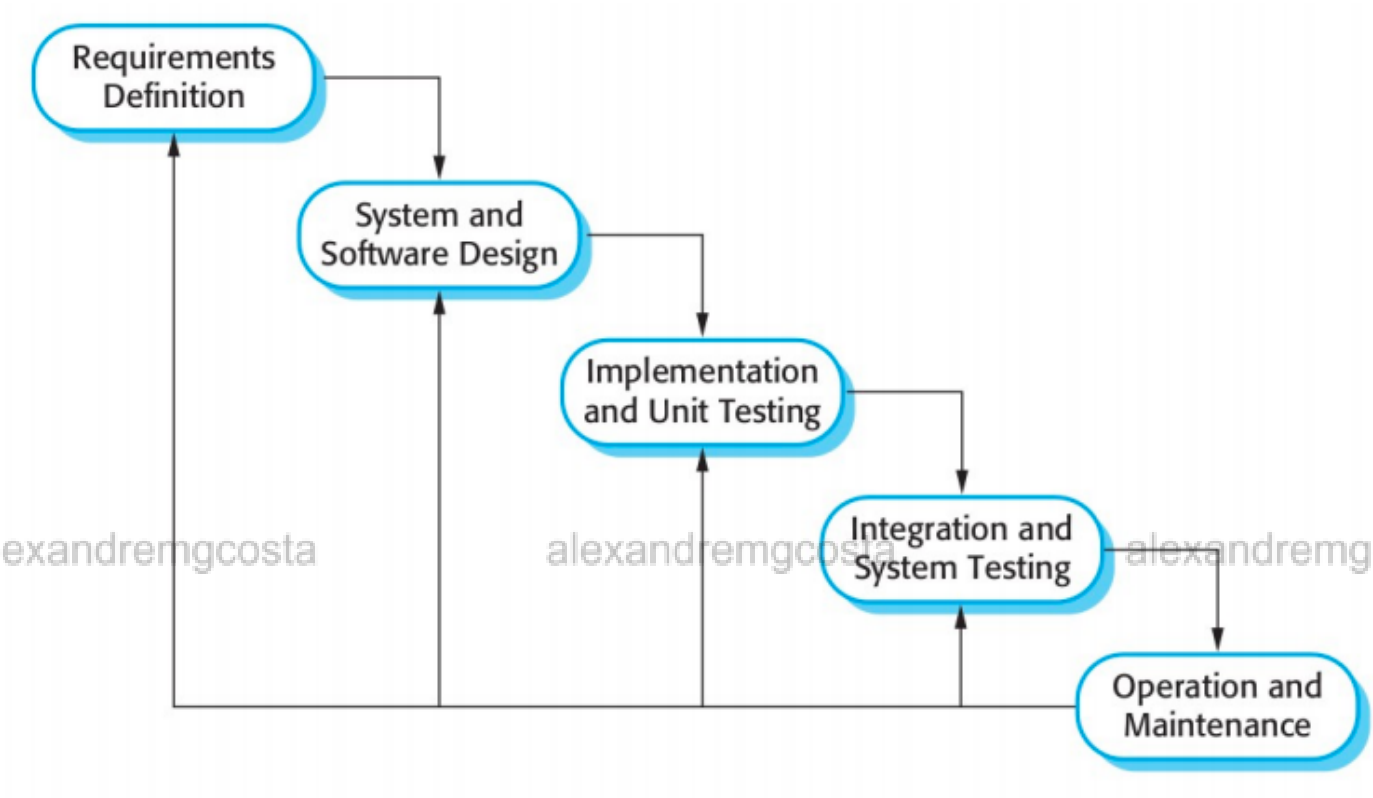
Modelos de Processos de Software

- Modelo Waterfall (cascata)
- Modelo Incremental
- Integração e configuração

Modelo Waterfall (Cascata)

```
^       para cima       ^
```

- Baseado em planos.



->Etapas Waterfall

- Análise e especificações de requísitos;
- Desenho do software;
- Implementação e testes unitários;
- Integração e testes de Sistema;
- Operação e manutenção.

->Desvantagens Waterfall

- Cada etapa só inicia quando a anterior termina;
- Divisão inflexivél do projeto;
- Dificuldade em incluir alterações depois de dar início ao seu desenvolvimento.

->Problemas Waterfall

- Divisão inflexível do projeto em diferentes etapas;
- Dificulta a resposta a alteração de requisitos;

- Os requisitos devem ser bem conhecidos;
- Poucos sistemas tem requisitos estáveis.

->Utilizacao Waterfall

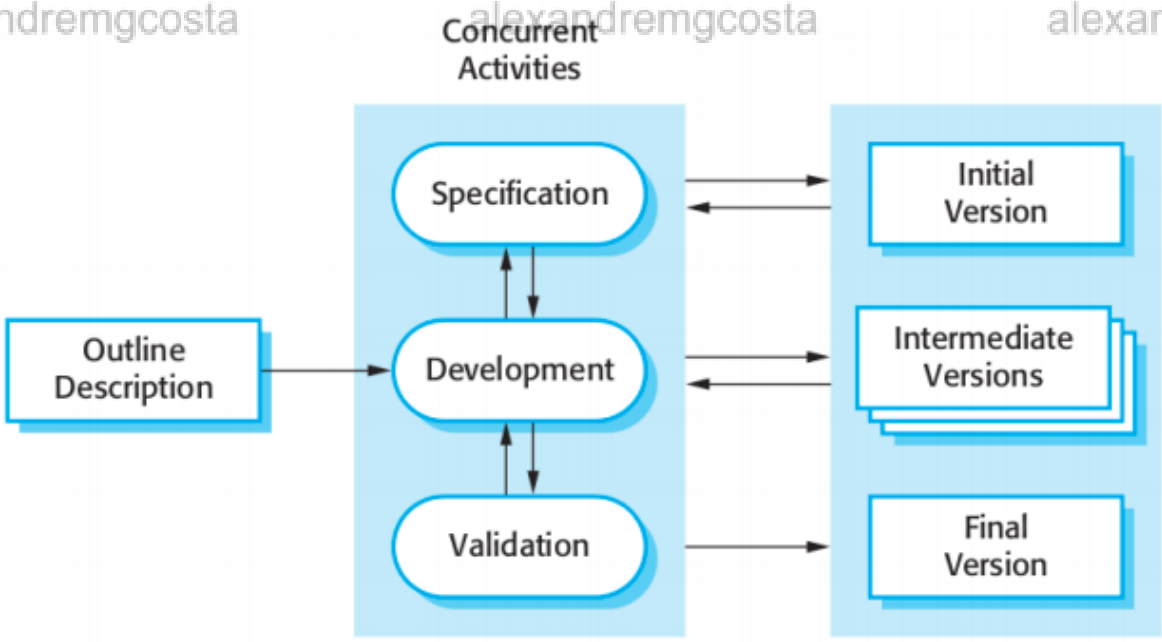
- Para sistemas grandes;
- Desenvolvido por equipas grandes para grandes sistemas.

Modelo Incremental

```
^        para cima        ^
```

O modelo incremental é baseado na ideia de desenvolver uma implementação inicial, expô-la aos comentários dos clientes e continuar por meio da criação de várias versões até que um sistema adequado seja desenvolvido.

Desenvolvimento incremental de software, que é uma parte fundamental das abordagens ágeis, é melhor do que uma abordagem em waterfall para a maioria dos sistemas de negócios, e-commerce e sistemas pessoais. Ao desenvolver um software de forma incremental, é mais barato e mais fácil fazer mudanças no software durante o seu desenvolvimento. O cliente pode avaliar o sistema num estágio relativamente inicial do desenvolvimento para ver se ele oferece o que foi requisitado. Em caso negativo, só o incremento que estiver em desenvolvimento no momento precisará de ser alterado e, possivelmente, novas funcionalidades deverão de ser definidas para incrementos posteriores.



->Vantagens Incremental

- Custos para incluir alterações de requisitos é reduzido;
  - Menos análise e documentação.
- Mais fácil obter feedback do cliente relativo ao desenvolvimento que está a ser feito
  - Os clientes podem analisar demos do software e perceber o que já se encontra implementado.
- Entrega e deployment mais rápido de software utilizável.

->Problemas Incremental

- Processo não é visível
  - Difícil medir o progresso.
    - Software desenvolvido de forma rápida.
- Estrutura do software degrada-se a cada incremento
  - Incorporar novas funcionalidades torna-se cada vez mais difícil

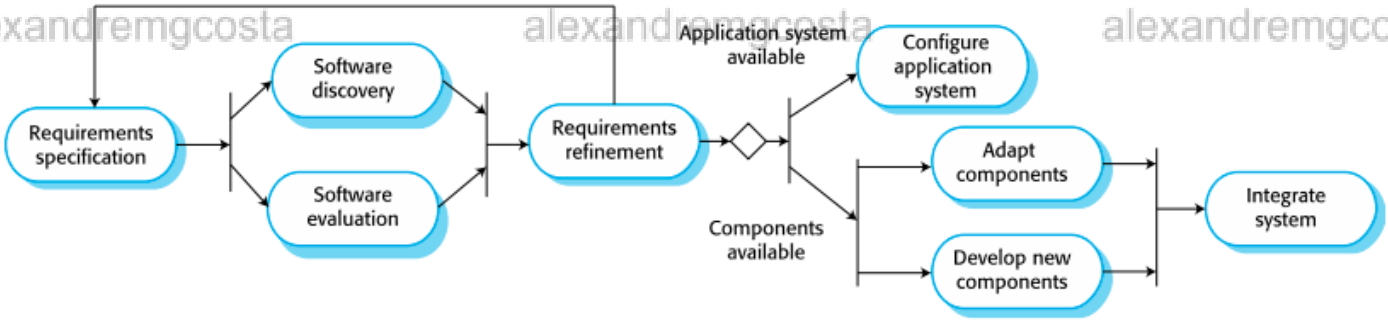
Modelo Integração e Configuração

```
^        para cima        ^
```

- Baseado na reutilização de software
  - Sistemas são construidos usando componentes ou aplicações existentes

Existem três tipos de componentes de software que podem ser usados num processo orientado ao reúso:

- Web services desenvolvidos de acordo com os padrões de serviço e que estão disponíveis para invocação remota.
- Coleções de objetos que são desenvolvidos como um pacote a ser integrado com um framework de componentes.
- Sistemas de software stand-alone configurados para uso num ambiente particular.



Integração e Configuração

->Etapas do Processo

- Especificação de requísitos;
- Pesquisa e análise de software;
- Alteração / Adaptação de requísitos;
- Configuração das aplicações do sistema;
- Adaptação e integração de componentes

->Vantagens

- Custos e riscos reduzidos;
  - Menos software é criado de raiz.
- Entregas e deployments do sistema mais rápidas.

->Desvantagens

- Compromissos com os requisitos;
  - Sistema pode não estar de acordo com as reais necessidades dos utilizadores
- Não existe controle sobre a evolução dos componentes reutilizados.

Atividades do processo

Processos reais de software são intercalados com sequencias de atividades técnicas, colaborativas e de gestão, com o intuito de especificar, projetar, implementar e testar um sistema de software.
As quatro atividades básicas do processo – especificação, desenvolvimento, validação e evolução – são organizadas de forma diferente conforme o processo de desenvolvimento. No modelo em cascata são organizadas em sequência, enquanto no desenvolvimento incremental são intercaladas.

Entrega incremental

- Sistema entregue por incrementos
  - Desenvolvimento e entrega são divididos em "incrementos"
  - Cada incremento introduz uma funcionalidade
- Prioridades
  - Requisitos do utilizador têm maior prioridade
    - Incluidos nos primeiros incrementos
- Vantagens:
  - Os clientes podem usar os incrementos iniciais como protótipos e ganhar experiência, podem ajudar a identificar requisitos para os próximos incrementos.
  - Os clientes não necessitam esperar até que todo o sistema seja entregue para obter ganhos a partir dele. O primeiro incremento satisfaz os requisitos mais críticos de maneira que eles possam usar o software imediatamente.
  - Menor risco de o projeto falhar.
  - Os serviços/funcionalidades mais importantes tendem a ser mais testados porque são implementados nas fases iniciais.
- Problemas:
  - A maioria dos sistemas exige um conjunto de recursos básicos de funcionalidades, usados por diferentes partes do sistema.
  - Especificação desenvolvida juntamente com o software. Causa conflitos com o modelo de comprar de muitas organizações, em que a especificação completa do sistema é parte do contrato de desenvolvimento do sistema.


```
^       para cima       ^
```

Engenharia de Requisitos

- O processo de descobrir, analisar, documentar e verificar esses serviços e restrições é chamado engenharia de requisitos.

```
^       para cima       ^
```

Requisitos

```
^       para cima       ^
```

O que sao requisitos?

- Os requisitos de um sistema são as descrições do que o sistema deve fazer, os serviços que oferece e as restrições a seu funcionamento.

Tipos de requisitos

- Requisitos do Utilizador
  - Requisitos de utilizador são declarações, numa linguagem natural com diagramas, de quais serviços o sistema deverá fornecer a seus utilizadores e as restrições com as quais este deve operar.

Ex: “O sistema deve gerar relatórios mensais com o custo dos medicamentos receitados por cada clínica, durante o mês em questão”.

- Requisitos do Sistema
  - Requisitos de sistema são descrições mais detalhadas das funções, serviços e restrições operacionais do sistema de software. O documento de requisitos do sistema (as vezes, chamado especificação funcional) deve definir exatamente o que deve ser implementado. Pode ser parte do contrato entre o comprador do sistema e os desenvolvedores do software.

Ex: “No último dia de cada mês, deve ser feito um gerado um relatório com todos os medicamentos receitados, o seu custo e a clínica que os receitou.”

System stakeholders

- Qualquer pessoa
  - que esteja relacionada com o sistema;
  - que tenha algum interesse no sistema.
- Tipos de stakeholders
  - Utilizadores finais;
  - Gestores do sistema;
  - Donos do sistema;
  - Stakeholders externos;

Requisitos funcionais

- São declarações de serviços que o sistema deve fornecer, de como o sistema deve reagir a entradas especificas e de como o sistema deve se comportar em determinadas situações. Em alguns casos, os requisitos funcionais também podem explicitar o que o sistema não deve fazer.

- Os requisitos funcionais de um sistema descrevem o que ele deve fazer. Eles dependem do tipo de software a ser desenvolvido, de quem são os seus possíveis utilizadores e da abordagem geral adotada pela organização ao escrever os requisitos.
  - Quando expressos como requisitos de utilizador, os requisitos funcionais são normalmente descritos de forma abstrata.
  - No entanto, requisitos de sistema funcionais mais específicos descrevem em detalhes as funções do sistema, suas entradas e saídas, exceções etc.

Ex: "Um utilizador deve ser capaz de procurar consultas na lista de agendamentos de todas as clínicas."

Requisitos nao funcionais

- Os requisitos não funcionais, como o nome sugere, são requisitos que não estão diretamente relacionados com os serviços específicos oferecidos pelo sistema aos seus utilizadores. Eles podem estar relacionados às propriedades emergentes do sistema, como confiabilidade, tempo de resposta e ocupação de espaço.
- Requisitos não funcionais são frequentemente mais críticos que requisitos funcionais individuais.

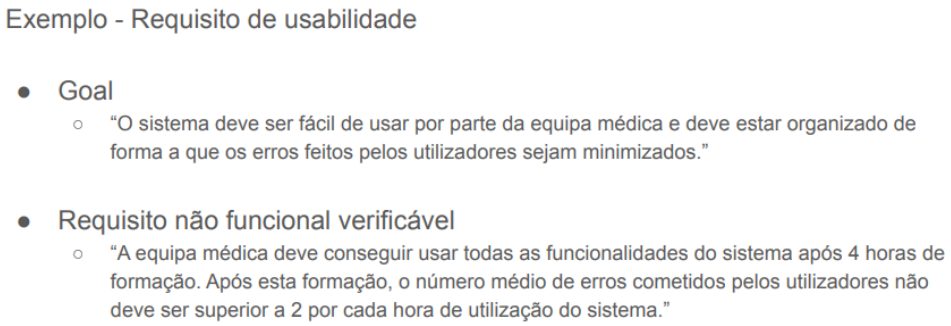
Os requisitos não funcionais podem ser provenientes das características requeridas para o software (requisitos de produto), da organização que desenvolve o software (requisitos organizacionais) ou de fontes externas:

- **Requisitos de produto**. Esses requisitos especificam ou restringem o comportamento do software.
- **Requisitos organizacionais**. Esses são os requisitos gerais de sistemas derivados das políticas e procedimentos da organização do cliente e do desenvolvedor.
- **Requisitos externos**. Esse tipo abrange todos os requisitos que derivam de fatores externos ao sistema e seu processo de desenvolvimento.

Ex: "O sistema deve estar disponível para todas as clínicas, durante as horas normais de trabalho (Seg-Sex, 08:30h – 17:30h). O sistema não pode estar em baixo mais do que 5 segundos por dia, em qualquer dia." (Requisito de produto)

Objetivos(goals) e requisitos

Requisitos não funcionais são muito abstratos e normalmente imprecisos. São difíceis de verificar se estão satisfeitos. A solução passa por criar “goals”, que são uma descrição genérica sobre a intenção do utilizador, e associar aos goals requisitos não funcionais verificáveis. Os goals são uteis para a equipa de desenvolvimento. Definem as intenções de uso dos utilizadores.



Documento de requisitos de software

O documento de requisitos de software, às vezes chamado Especificação de Requisitos de Software, é uma declaração oficial de o que os desenvolvedores do sistema devem implementar. Deve incluir tanto os requisitos de utilizador para um sistema quanto uma especificação detalhada dos requisitos de sistema.

Processos Engenharia de Requisitos

- Variam dependendo de:
  - Domínio de aplicação;
  - Pessoas envolvidas;
  - Entidade que desenvolve os requisitos;

- No entanto, existem atividades comuns a todos os processos:
  - Identificação dos requisitos;
  - Análise dos requisitos;
  - Validação dos requisitos;
  - Gestão dos requisitos;

Na prática são atividades feitas de forma iterativa e intercalada.

Identificacao dos Requisitos

```
^       para cima       ^
```

Os engenheiros de software trabalham com clientes e utilizadores finais do sistema para obter informações sobre o domínio da aplicação, os serviços que o sistema deve oferecer, o desempenho do sistema, restrições de hardware e assim por diante.
A identificação de requisitos pode envolver todos os stakeholders.

- Engenheiros trabalham com os stakeholders por forma a descobrir:
  - Domínios de aplicação;
  - Que serviços devem ser fornecidos;
  - Desempenho do sistema;
  - Restrições de hardware.

- Deve incluir:
  - Descoberta dos requisitos;
  - Classificação e organização dos requisitos;
  - Priorização e negociação dos requisitos;
  - Especificação dos requisitos.

- Problemas na identificação dos requisitos

- Stakeholders não sabem o que realmente querem;
- Stakeholders expressam os requisitos nos seus termos próprios;
- Diferentes stakeholders podem ter requisitos com conflitos;
- Fatores organizacionais e políticos podem influenciar os requisitos de sistema;
- Os requisitos mudam durante o processo de análise:
- Podem surgir novos stakeholders;
- As regras de negócio podem alterar-se;

Processo de identificacao e análise de requisitos

1. Descoberta dos Requisitos
   - Interagir com todos os stakeholders por forma a descobrir os requisitos
   - Requisitos de domínio são também identificados
   - Fazer a distinção entre requisitos de utilizador e de sistema
2. Clarificação e organização dos requisitos
   - Agrupar requisitos relacionados
   - Organizá-los em conjuntos coerentes
3. Priorização e negociação
   - Atribuir prioridades
   - Resolver conflitos entre requisitos
4. Especificação de requisitos
   - Escrita dos requisitos

Cenarios

```
^        para cima        ^
```

- User story estruturada;
- Devem incluir:
  - Descrição da situação inicial (ou ponto de partida);
  - Descrição do fluxo normal de eventos;
  - Descrição do que pode correr mal;
  - Informação sobre outras atividades concorrentes;
  - Descrição do estado do sistema quando o cenário termina.

Exemplo: Fazer upload de fotografias/imagens

- Estado inicial:
  - "A user or a group of users have one or more digital photographs to be uploaded to the picture sharing site. These are saved on either a tablet or laptop computer. They have successfully logged on to KidsTakePics."
- Comportamento normal:
  - "The user chooses upload photos and they are prompted to select the photos to be uploaded on their computer and to select the project name under which the photos will be stored. They should also be given the option of inputting keywords that should be associated with each uploaded photo. Uploaded photos are named by creating a conjunction of the user name with the filename of the photo on the local computer."
  - "On completion of the upload, the system automatically sends an email to the project moderator asking them to check new content and generates an on-screen message to the user that this has been done."

- O que pode correr mal:
  - "No moderator is associated with the selected project. An email is automatically generated to the school administrator asking them to nominate a project moderator. Users should be informed that there could be a delay in making their photos visible."
  - "Photos with the same name have already been uploaded by the same user. The user should be asked if they wish to re-upload the photos with the same name, rename the photos or cancel the upload. If they chose to re-upload the photos, the originals are overwritten. If they chose to rename the photos, a new name is automatically generated by adding a number to the existing file name."
- Outras atividades:
  - "The moderator may be logged on to the system and may approve photos as they are uploaded."
- Estado final depois de terminar:
  - "User is logged on. The selected photos have been uploaded and assigned a status 'awaiting moderation'. Photos are visible to the moderator and to the user who uploaded them."

Especificacao de requisitos

- Processo de escrever os requisitos
  - requisitos de sistema e de utilizador
  - documento de requisitos
- Os requisitos devem ser de fácil compreensão
- Os requisitos devem ser detalhados
- Os requisitos devem ser o mais completo possiveis

Especificação em língua natural

- Requisitos são escritos em língua natural, complementados com diagramas e tabelas;
- A linguagem natural é:
  - Expressiva
  - Intuitiva

Escrita dos requisitos

- "Criar" um formato standard para todos os requisitos;
- Usar linguagem de forma consistente:
  - Usar "deve" para requisitos obrigatórios
  - Usar "pode" para requisitos desejáveis
- Usar text highlight para identificar aspetos importantes do requisito
- Evitar termos técnicos;
- Explicar o porque do requisito

Problemas da Linguagem Natural

- Falta de clareza
- Confusão de requisitos, os requisitos funcionais e nao funcionais misturam-se entre si;
- Junção dos requisitos

Especificações estruturadas

- Baseadas em lingua natural
  - Seguem um standar

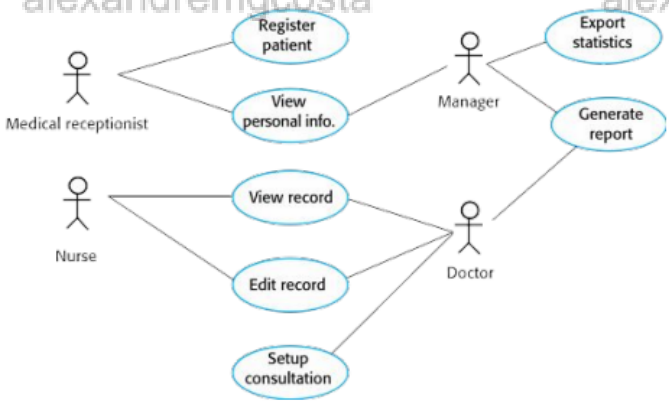
    - Em vez de usar texto livre

      Mantém alguma uniformidade
- Abordagem para escrever requisitos:
    - Liberdade limitada para descrever os requisitos;
    - Escritos seguindo um padrao espeicfico.
- Funciona bem para alguns tipos de sistema

Use Cases

- Tipo de cenários:
    - Fazem parte do UML
- Identificam:
    - Atores;
    - Numa interação com o sistema
- Modelo gráfico de alto nível:
    - Complementado com uma descrição tabular
- Diagrama de sequências (UML):
    - Podem ser usados para detalhar os use cases;

Use Cases - Exemplo



Validação dos requisitos

- A validação de requisitos é o processo pelo qual se verifica se os requisitos definem o sistema que o cliente realmente quer. A validação de requisitos é importante porque erros em um documento de requisitos podem gerar altos custos de retrabalho quando descobertos durante o desenvolvimento ou após o sistema já estar em serviço.
- Como validar:
    - Validade dos requisitos;
    - Consistência;
    - Completude;
    - Realismo;
    - Verificabilidade;

Gestão dos requisitos

A gestão de requisitos é o processo de compreensão e controle das mudanças nos requisitos do sistema. É preciso se manter a par das necessidades individuais e manter as ligações entre as necessidades dependentes para conseguir avaliar o impacto das mudanças nos requisitos. É necessário estabelecer um processo formal para fazer propostas de mudanças e a ligação destas às exigências do sistema.