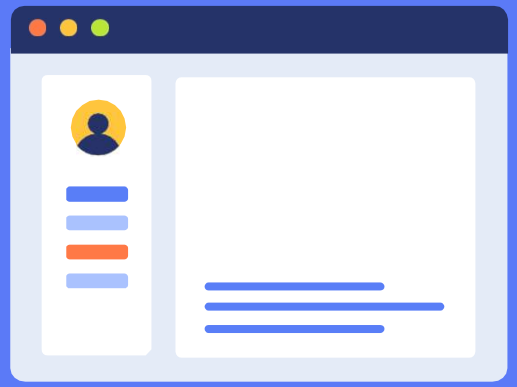
P.PORTO

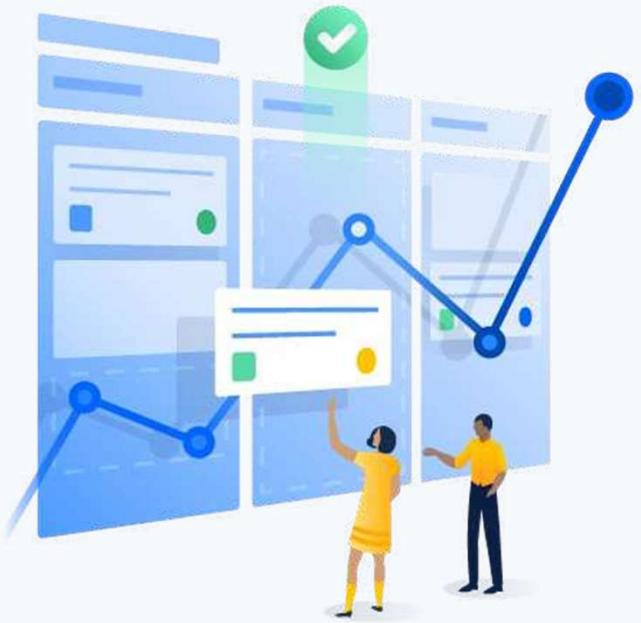
ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

Laboratório de desenvolvimento de software



Índice

- Engenharia de Software
- Desenvolvimento de Software
- Application Lifecycle Management (ALM)
- Modelos de Desenvolvimento de Software
- Ferramentas Case



Engenharia de software

Engenharia de Software é uma disciplina que diz respeito a todos os aspetos da produção de software, desde o período inicial de especificação de requisitos até à manutenção do sistema depois de entrar em utilização. (Somervile, 2016)

É a aplicação de técnicas de engenharia ao desenvolvimento de software.

A necessidade de melhorar a qualidade do software e de implementar métricas de controlo de qualidade resultou no aparecimento da disciplina de engenharia de software:

- Aplicação sistemática de metodologias e modelos de desenvolvimento de software
- Avaliação de software
- Auditoria a processos

Desenvolvimento de software

O software pode ser desenvolvido para uma variedade de propósitos. Os mais comuns são:

- Necessidades de um cliente ou negócio
- Para necessidades de utilizadores específicos
- Uso pessoal

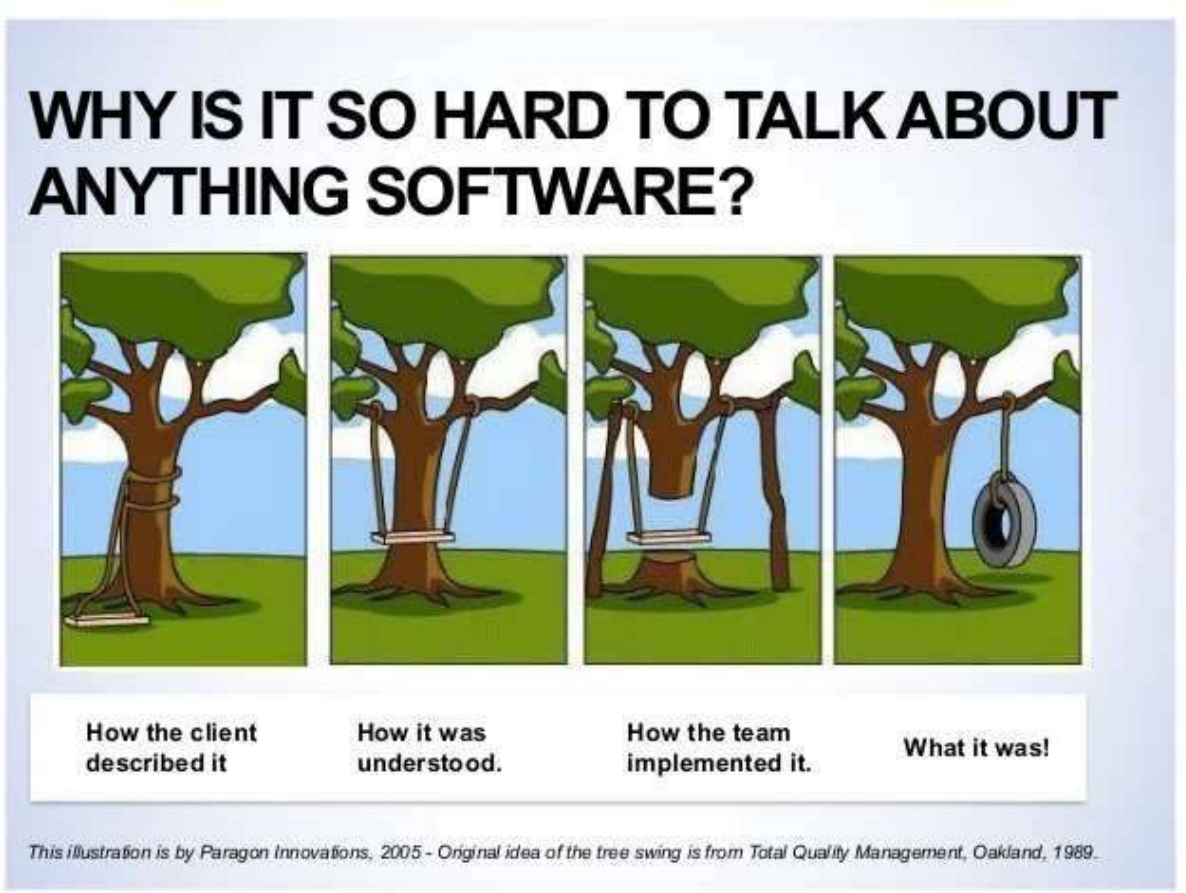
O software desenvolvido é normalmente categorizado como:

- Genérico
- Específico ou Customizado

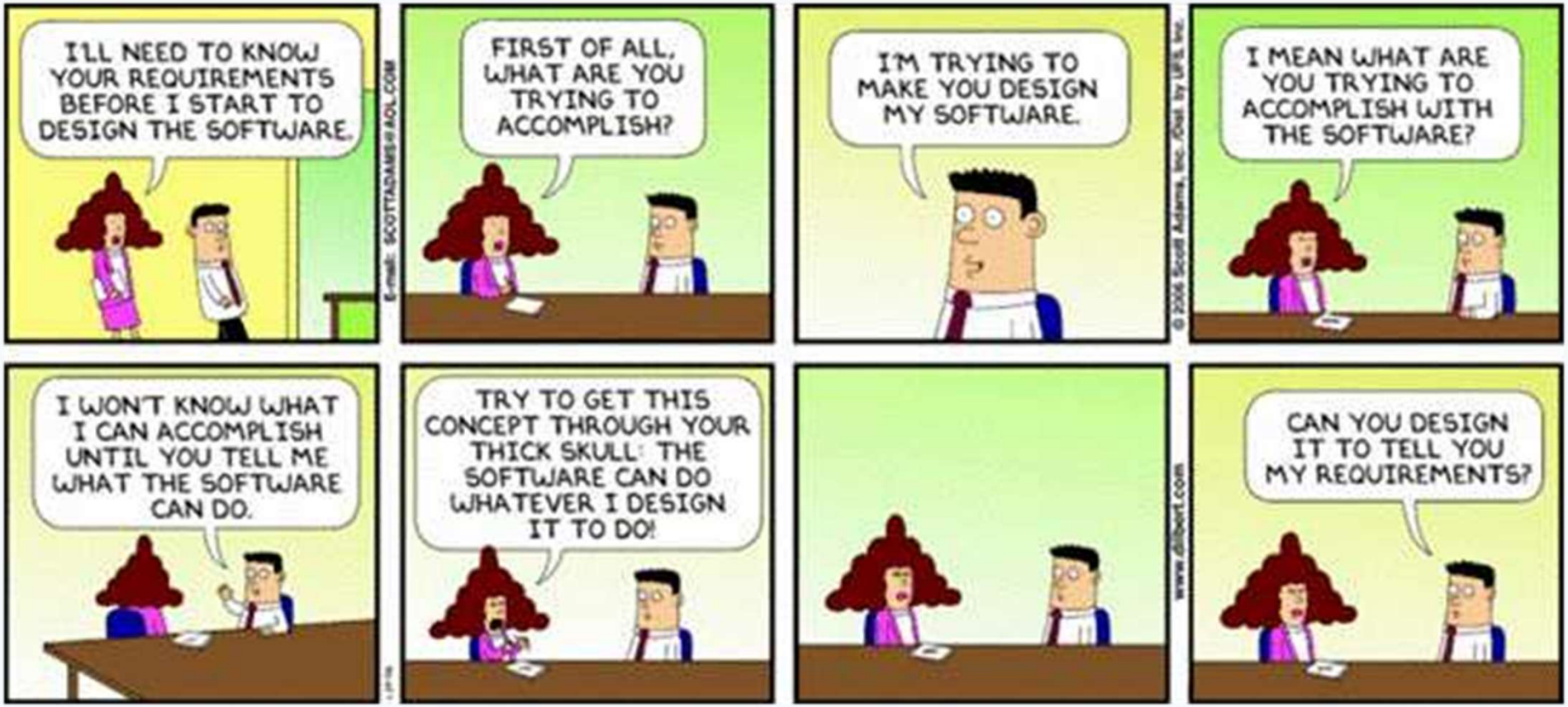
Desenvolvimento de software

Desenvolver software é uma atividade complexa!

Desenvolvimento de software



Desenvolvimento de software



Desenvolvimento de software

O processo de desenvolvimento de software tem essencialmente 4 tarefas (Sommerville, 2016):

- Especificação de Software – onde clientes e engenheiros definem o software que irá ser produzido e todas as condicionantes ao seu desenvolvimento.
- Desenvolvimento de Software – onde é programado o produto de software.
- Validação de Software – onde o produto de software é testado avaliado e validado de acordo com os requisitos do cliente.
- Evolução de Software – onde o produto de software é modificado para refletir alterações pedidas por clientes ou para atividades de manutenção.

Desenvolvimento de software

O processo de desenvolvimento de software envolve tradicionalmente:

- Especificação de requisitos
- Desenho da arquitetura
- Programação e Desenvolvimento
- Documentação
- Testes
- Correção de bugs
- Manutenção

Desenvolvimento de software

Pode também haver lugar a:

- Investigação
- Prototipagem
- Reengenharia
- Outros

Desenvolvimento de software

Especificação de Requisitos

- As atividades de especificação de requisitos envolvem reuniões com os clientes do software e a produção da respetiva documentação.
- Envolve o levantamento dos pedidos/necessidades dos stakeholders e a sua transformação num conjunto de requisitos de forma a que o produto de software funcione de acordo com o desejado.
- É necessário fornecer informação detalhada sobre aquilo que o sistema deve fazer.

Desenvolvimento de software

Desenho da Arquitetura

- A arquitetura de um sistema de software pode conter uma representação abstrata do sistema.
- A arquitetura deve dar a garantia de que o sistema de software irá ao encontro dos requisitos do produto, como também assegurar que futuros requisitos possam ser atendidos.
- A arquitetura também direciona as interfaces entre os sistemas de software e outros produtos de software.

Desenvolvimento de software

Implementação, Testes e Documentação

- Implementação implica a própria programação do produto de software.
- Testes de Software devem ser usados para detetar erros no software e garantir a sua qualidade. Devem:
  - Verificar a interação entre objetos.
  - Verificar a integração adequada de todos os componentes do software.
  - Verificar se todos os requisitos foram corretamente implementados.
  - Identificar e garantir que os defeitos são abordados antes da implantação do software.
  - Garantir que todos os defeitos são corrigidos, reanalisados e fechados.
- Documentação é a atividade responsável pela produção de toda a informação relevante sobre o software permitindo auxiliar futuras tarefas de manutenção e de melhorias do software.

Desenvolvimento de software

Here is what they always forget to tell new testers in the job interview

Desenvolvimento de software

Deployment e Manutenção

- O Deployment é a atividade que corresponde à instalação do produto de software em ambiente real. Tipicamente diz-se que o produto de software passou a produção. Esta tarefa começa após os testes de software serem bem sucedidos.
- As atividades de manutenção e suporte à aplicação também estão previstas. Nestas deve ser dada resposta a eventuais falhas do software, problemas ocasionais e correção de falhas detetadas em modo de produção.

Desenvolvimento de software

http://dilbert.com/

Desenvolvimento de software

Não esquecer:

- Engenharia de software é uma disciplina que cobre todos os aspetos da produção de software.
- Atributos essenciais dos produtos de software são: manutenção, confiança, segurança, eficiência e aceitação.
- Processos de software incluem todas as atividades envolvidas no desenvolvimento de software.
- Atividades de alto nível como especificação, desenvolvimento, validação e evolução fazem parte do processo de desenvolvimento de software.
- As noções fundamentais de engenharia de software são universalmente aplicáveis a todos os tipos de desenvolvimento de sistemas de software.

Application Lifecycle management

Definir ALM não é uma tarefa fácil, nem se pode dizer que exista uma definição consensual. A sua definição varia de acordo com a perspetiva pela qual é analisada.

"Like a human life, an application's lifecycle is demarcated by significant events." (David Chappel, 2008)

A ALM cobre toda a história da aplicação, desde a sua ideia inicial até ao fim de vida da aplicação.

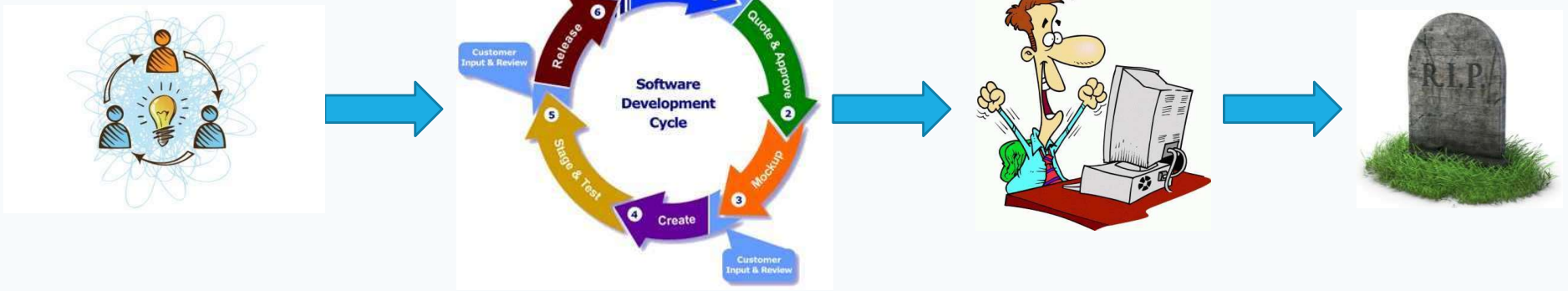
Application Lifecycle management

É comum considerar-se que ALM é o mesmo que SDLC (Software Development Lifecycle). No entanto, ALM é muito mais do que SDLC. SDLC é uma parte da ALM.

Software Development Lifecycle

Application Lifecycle management

Application Lifecycle Management



ALM estende o SDLC nos dois extremos.

A área de tecnologias de informação (TI) está hoje perante uma nova realidade. Existe obrigatoriamente a necessidade de otimizar projetos de desenvolvimento de aplicações.

Application Lifecycle management

Existe ainda a forte necessidade de gerir inovações aplicacionais, assim como, novas necessidades de negócios que vão surgindo.

É imprescindível a utilização de uma ferramenta/solução que permita a redução de custos com projetos de desenvolvimento.

ATENÇÃO: Muitos projetos de TI falham devido a ultrapassarem timings e custos. Ou então por não permitirem obter o retorno adequado.

Estudos mostram que mais de metade do orçamento é gasto em questões operacionais e de manutenção.

Talvez uma das maiores dificuldades das organizações são as estratégias de comunicação entre o lado do negócio e o lado das TI.

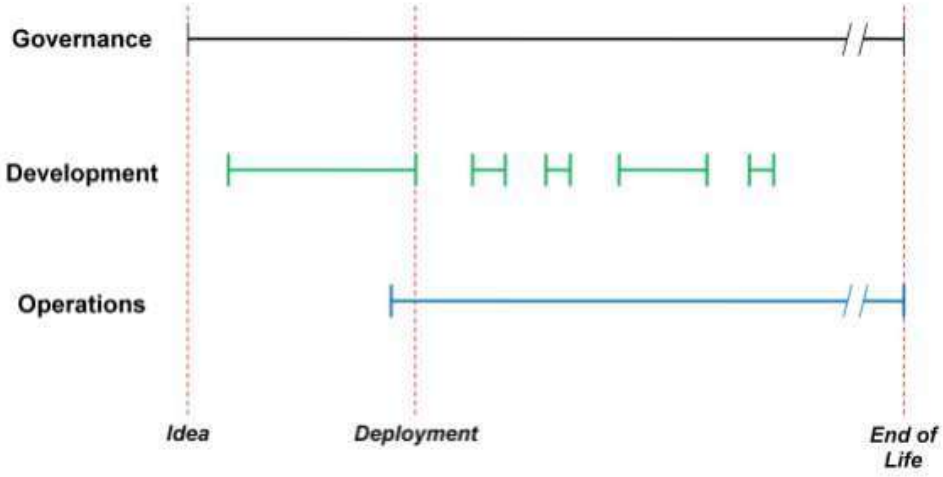
ALM – 3 PRINCIPAIS aspectos

ALM pode ser dividido em 3 áreas distintas: governance, development e operations.

Governance: estende-se ao longo de todo o período, incluindo todo o processo de decisão e de gestão do projeto para uma determinada aplicação.

Development: o processo que visa desenvolver a aplicação. Acontece depois da ideia e antes do deployment.

Operations: O trabalho necessário para colocar a aplicação a correr, assim como tratar da sua gestão. Geralmente começa um pouco antes do deployment e continua até que a aplicação seja removida.



ALM – gOVERNANCE

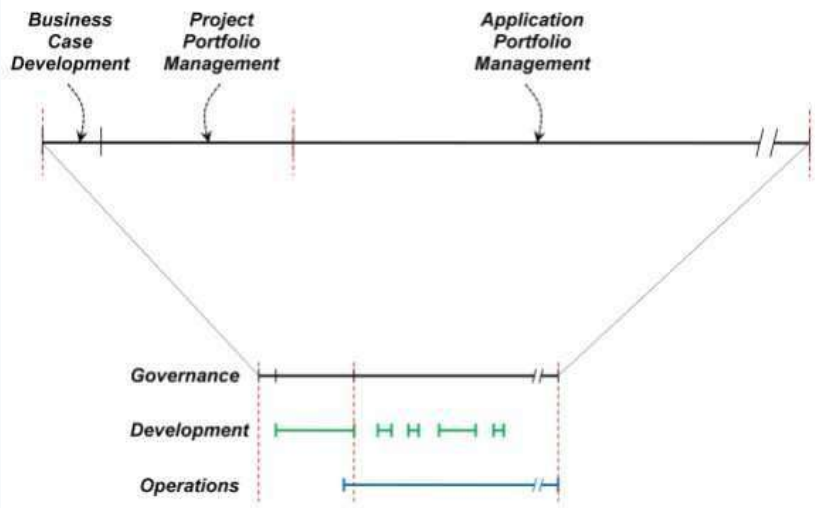
O propósito da "Governance" é garantir que a aplicação está sempre de acordo com as necessidades da empresa.

Como vimos a "Governance" acompanha todo o ciclo de vida a aplicação.

O primeiro passo da "Governance" num ALM é o "Business Case Development" (antes do início do desenvolvimento da aplicação).

Depois do BCD ser aprovado a "Governance" é implementada através do "Project Portfolio Management".

Finalmente, depois de ser feito o deployment da aplicação, é iniciada a parte de "Application Portfolio Management".



ALM – GOVERNANCE

O que é afinal o "Business Case Development"? Para que serve?

No BCD é percebida a razão para iniciar um projeto (ou tarefa).

- A lógica do "Business case" deve ajudar avaliar os custos e os benefícios relacionados com o consumo de recursos.
- "Business cases" são normalmente utilizados para avaliar os projetos, permitindo tomar decisões, por exemplo: "Devemos iniciar este projeto? Sim-Não"; "Se começarmos o projeto... Que prioridade devemos dar a este projeto?"

ATENÇÃO: um "Business Case" não é a mesma coisa que um modelo financeiro.

(Resumindo) Mas afinal porque deve ser criado um "Business Case"?

- Ajuda a organização a estimar os custos e os benefícios do esforço necessário para a criação de um novo projeto.
- Comparando os resultados esperados dos diferentes esforços, a gestão pode determinar onde é que deve alocar os recursos.

Criar um "Business Case" requer pensamento crítico. Existem 4 componentes importantes que devem ser incluídos nesse pensamento:

- Perceber custos/recursos que um projeto irá necessitar.
- Definir os benefícios que esse projeto irá trazer.
- Identificar os riscos que podem ocorrer devido ao esforço realizado.
- Capturar os pressupostos das estimativas.

ALM – GOVERNANCE

O que é afinal o "Project Portfolio Management"? Para que serve?

PPM é a coordenação e a gestão do portfolio de projetos com o objetivo de alcançar os objetivos da organização.

Permite a uma organização definir, perceber, dar prioridade e medir o impacto dos projetos.

Permite definir a magnitude dos recursos envolvidos de forma a gerir melhor as várias equipas. "do the right projects right"

O PPM analisa o risco-benefício de cada projeto, os fundos disponíveis, a expectativa da duração de um projeto e os resultados esperados.

Muitas vezes esta função é realizada por um Project Manager ou por um técnico da equipa de desenvolvimento. No entanto, em organizações com uma abordagem mais formal esta função está centralizada num chamado Project Management Office (PMO).

PPM não está relacionado com executar os projetos mas sim com escolher que projetos executar e como financiá-los. PPM ajuda os decisores a definir sobre que projetos devem ser acelerados, priorizados, selecionados, abrandados e terminados.

ALM – GOVERNANCE

O que é afinal o "Application Portfolio Management"? Para que serve?

Serve para gerir (um género de inventário) as aplicações de software/serviços baseados em software da organização.

APM disponibiliza aos gestores um inventário das aplicações da organização bem como métricas que permitem entender os benefícios de cada aplicação.

Existem APM que usam algoritmos que avaliam as várias aplicações e que geram reports sobre o valor de cada aplicação.

Através das métricas que um APM disponibiliza (idade da aplicação, quantas vezes é usada, quanto custa manter a aplicação, de que forma se relaciona com outras aplicações, etc.) os gestores podem mais facilmente decidir sobre se uma aplicação deve ou não continuar, se deve ser atualizada ou substituída.

ALM – development

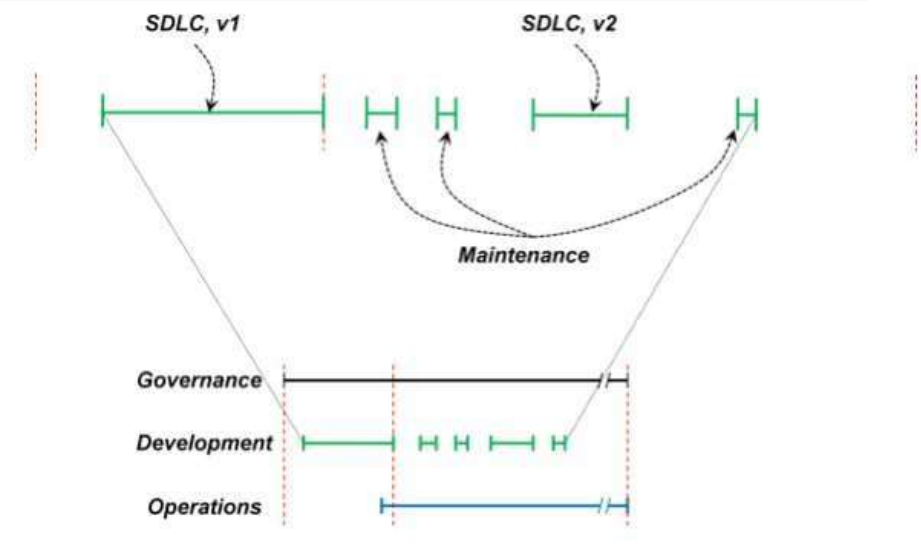
"Development" diz respeito exatamente ao desenvolvimento da aplicação.

O desenvolvimento começa na primeira parte do ciclo de vida da aplicação e depois acontece periodicamente de cada vez que a aplicação é atualizada.

O "Development" começa logo após o "Business Case" ser aprovado.

Quando a primeira versão do "SDLC" está completa, é feito o deployment da aplicação.

Podem ocorrer situações de "Maintenance" que resultam em updates da aplicação, ou então mesmo novos "SDLC" que resulta em novas versões da aplicação.



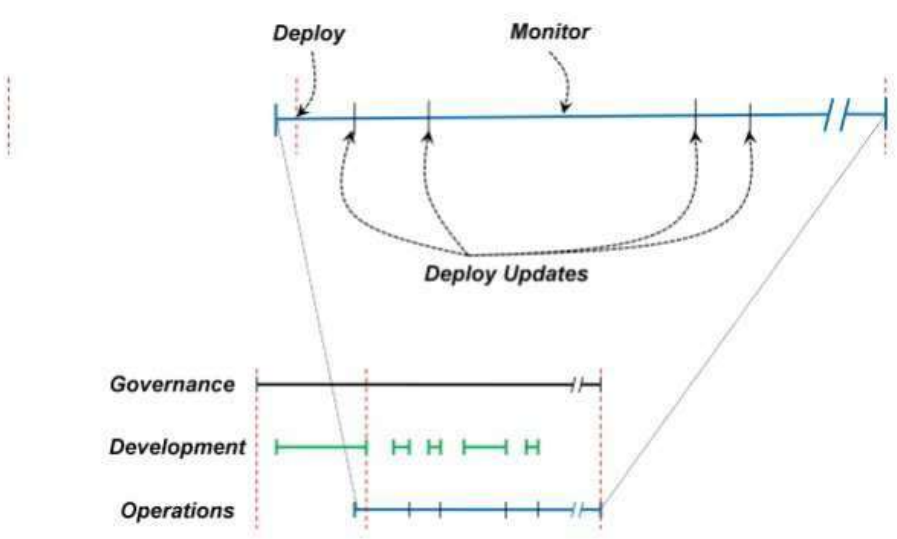
ALM – operations

Todas as aplicações que são implementadas têm de ser monitorizadas e geridas. Sendo essas as responsabilidade das "Operations".

"Operations" começam um pouco antes da aplicação ter sido implementada, continuando sempre até que esta seja removida.

Como se verifica a linha das "Operations" está relacionada com a linha do "Development".

Organizar o lançamento de uma nova versão ou update é da responsabilidade das "Operations".



ALM – Desafios e ferramentas

Problemas de comunicação.

Equipas com diferentes papeis.

Equipas que estão geograficamente dispersas.

Integração de ferramentas.

Falta de processos.

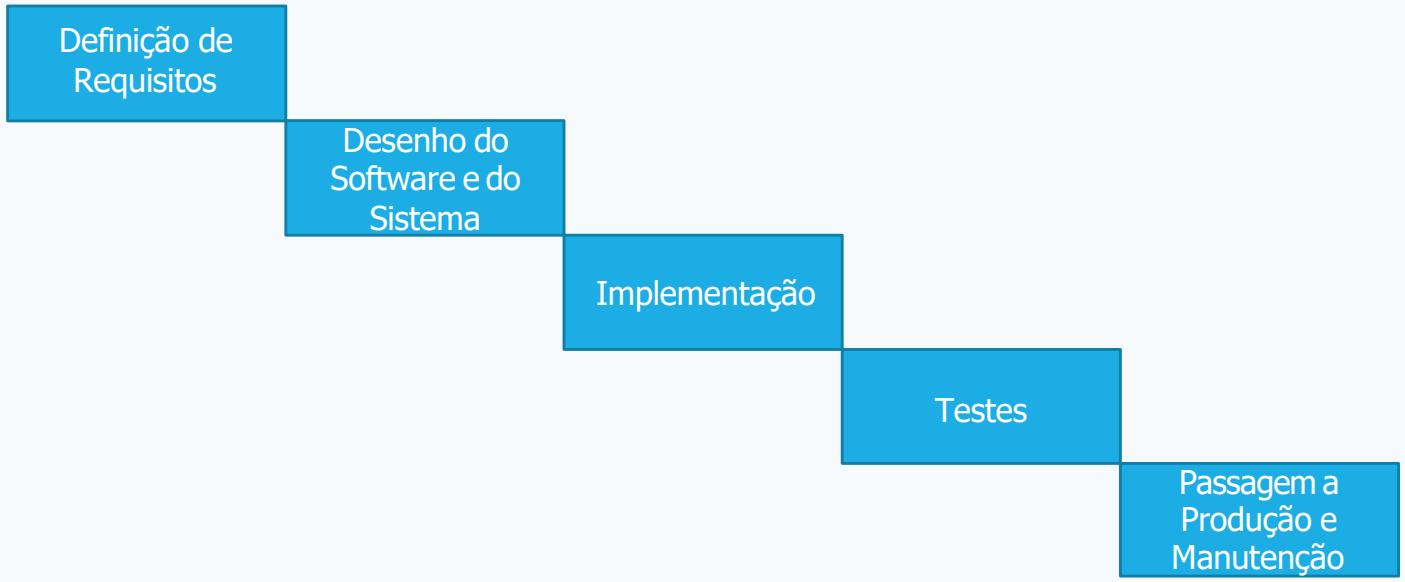
Como é percetível a utilização de determinadas ferramentas tem um papel fundamental no papel da ALM.

As ferramentas devem permitir ligar as várias camadas.

Devem melhorar os processos automáticos.

Modelos de desenvolvimento de software

Modelo em cascata



Modelos de desenvolvimento de software

Modelo em cascata

Vantagens:

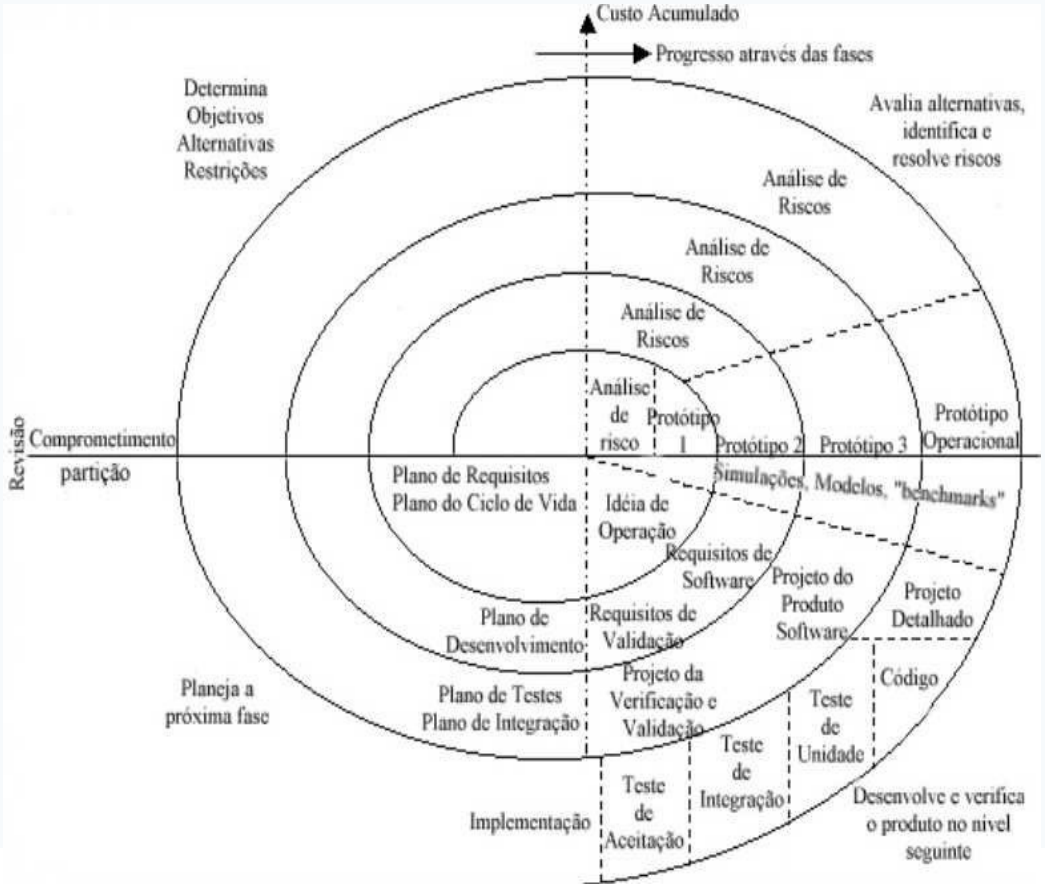
- Documentação produzida em cada fase.
- Similar a outros modelos de engenharia.

Desvantagens

- Os projetos têm dificuldade em seguir fluxos sequenciais.
- Dificuldade em estabelecer todos os requisitos com o cliente.
- O produto de software aparece apenas nas últimas fases do modelo.

Modelos de desenvolvimento de software

Modelo em espiral



Modelos de desenvolvimento de software

Modelo em espiral

Vantagens:

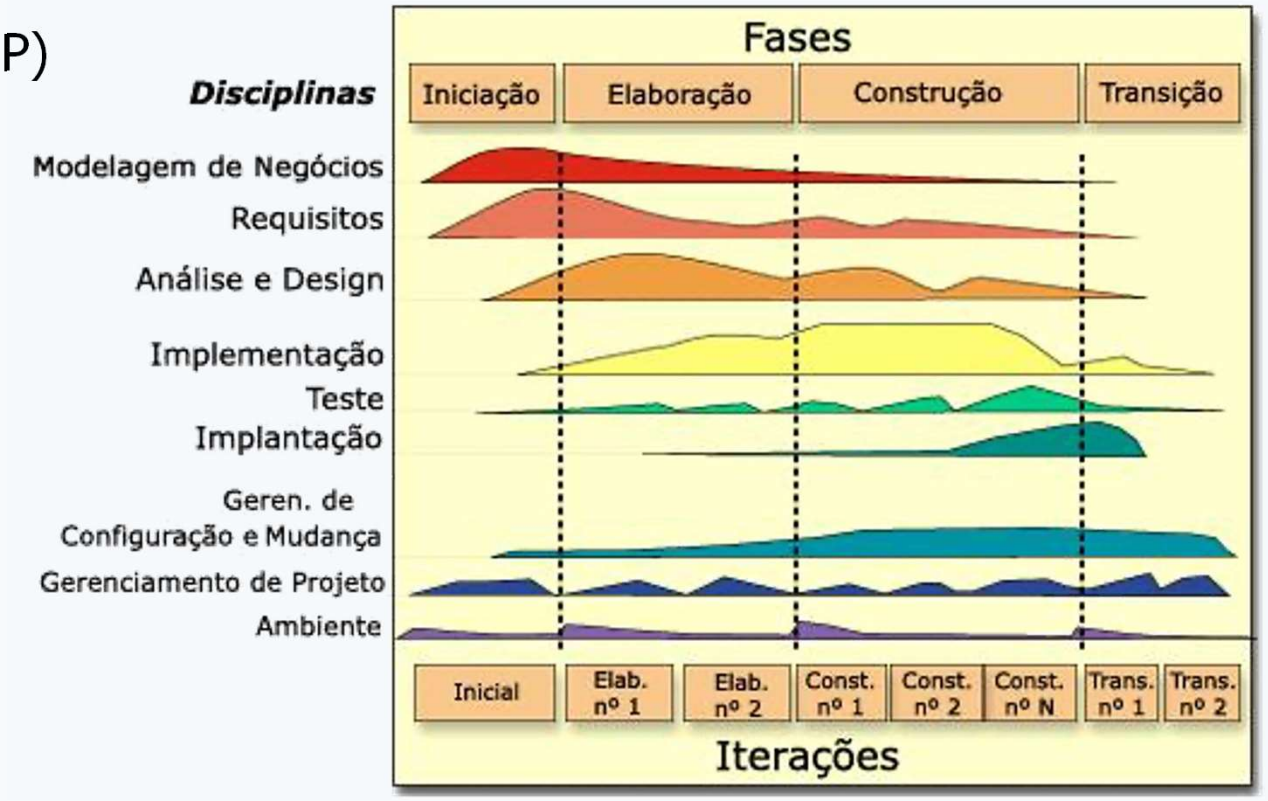
- Abordagem considerada realista.
- Permite desenvolvimento iterativo.
- Permite o desenvolvimento do produto de software com atenção aos riscos e atividades em cada fase.

Desvantagens

- Adoção difícil no processo de desenvolvimento de software de grandes projetos pois os custos e cronograma de atividades são alterados ao longo do projeto.

Modelos de desenvolvimento de software

Rational Unified Process (RUP)



Modelos de desenvolvimento de software

Rational Unified Process (RUP)

Vantagens:

- O RUP usa a abordagem de orientação a objetos na sua conceção e é projetado e documentado utilizando a notação UML (Unified Modeling Language) .
- Processo robusto e bem definido com a geração de artefactos importantes.

Desvantagens:

- É um processo considerado pesado e preferencialmente aplicável a grandes equipes de desenvolvimento e a grandes projetos. Porém o fato de ser amplamente adaptável torna possível que seja adaptado para projetos de qualquer escala.
- Exige experiência da equipa de trabalho.

Modelos de desenvolvimento de software

Modelos Ágeis

- As definições modernas de desenvolvimento de software ágil evoluíram a partir da metade de 1990
- Reação contra modelos considerados pesados como o modelo de desenvolvimento em cascata
- Seguem muitas das ideias do manifesto ágil publicado em 2001 (http://agilemanifesto.org/)

Manifesto ágil

- Focar mais nos indivíduos e nas interações do que nos processos e ferramentas
- Software funcional é mais importante do que documentação completa
- Colaboração com o cliente é mais importante do que negociação de contratos
- Adaptação a mudanças é mais importante do que seguir o plano inicial

Modelos de desenvolvimento de software

Princípios dos Modelos Ágeis:

- Garantir a satisfação do consumidor entregando rapidamente e continuamente software funcionais.
- Até mesmo mudanças tardias no âmbito no projeto são bem-vindas para garantir a vantagem competitiva do cliente.
- Software funcionais são entregues frequentemente (semanas, ao invés de meses).
- Cooperação diária entre pessoas que entendem o "negócio" e programadores.
- Projetos surgem através de indivíduos motivados, entre os quais existe relação de confiança.
- A maneira mais eficiente e efetiva de transmitir informações é conversar cara a cara.

Modelos de desenvolvimento de software

Princípios dos Modelos Ágeis:

- Software funcionais são a principal medida de progresso do projeto.
- Processos ágeis promovem desenvolvimento sustentável. Os patrocinadores, programadores e utilizadores devem ser capazes de manter um ritmo constante.
- Design do software deve prezar pela excelência técnica.
- Simplicidade é essencial.
- As melhores arquiteturas, requisitos e projetos emergem de equipes auto-organizadas.
- Em intervalos regulares, a equipe reflete sobre como tornar-se mais eficaz, então sintoniza e ajusta o seu comportamento apropriadamente.

FERRAMENTAS CASE

Ferramentas CASE (Computer-Aided Software Engineering) é um termo que se aplica a ferramentas que auxiliam o processo de desenvolvimento de software.

Podem ser consideradas como ferramentas automatizadas que tem como objetivo auxiliar o desenvolvimento de software.

Nos últimos anos, as ferramentas CASE têm evoluído em direções diferentes, abrangendo desde a especificação de sistemas até à geração automática de código fonte.

A geração atual de ferramentas CASE é composta por softwares complexos que auxiliam as equipas de desenvolvimento a projetar sistemas.

FERRAMENTAS CASE

Controle de Versões
- CVS, Subversion, Git

Gestão de projetos
- Microsoft Project

Edição
- Microsoft Word, Wiki, Eclipse, NetBeans, Visual Paradigm

Ferramentas de prototipagem
- Adobe PageMaker, NetBeans, Jbuilder

Base de Dados
- MySQL, Postgres

Testes
- Unit

Automação de tarefas
- Apache Ant, Apache Maven, Gradle

Teste
- JUnit (unitários)

Geração de código
- Visual Paradigm

FERRAMENTAS CASE

Vantagens:

- Qualidade no produto final
- Produtividade
- Agilizar o tempo para tomada de decisão
- Menor quantidade de códigos de programação
- Melhoria e redução de custos na manutenção
- Agilidade no retrabalho do software
- Maior facilidade para desenvolvimento

Bibliografia

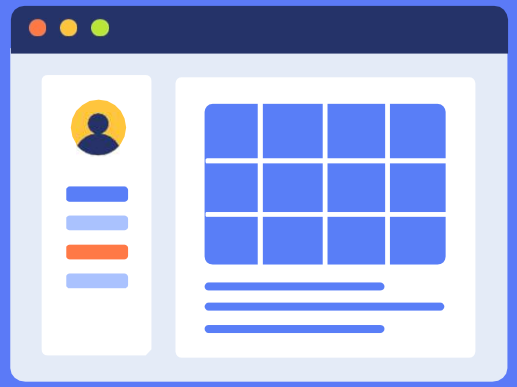
Ian Sommerville, "Software Engineering", 9th. Edition (Pearson - Addison-Wesley, 2011).

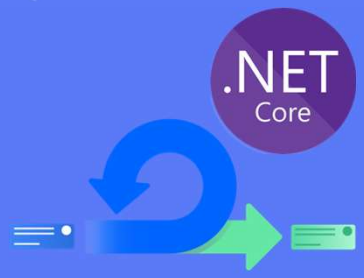
Roger S. Pressman, "Engenharia de Software – Uma Abordagem Profissional", 7a. Edição (McGraw-Hill, 2011)

P.PORTO

ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

Laboratório de desenvolvimento de software





Licenciatura em Engenharia Informática 2022/2023