IPG
Politécnico da Guarda
Escola Superior de Tecnologia e Gestão

Engenharia de Software II

Problemática Desenvolvimento Software

Maria Clara Silveira

2021-01-19

Problemática Desenvolvimento Software

Factores “crise de software”

- Problemas de comunicação
- Baixa qualidade
- Custos elevados na manutenção
- Ausência de metodologias rigorosas (aplicação rigorosa e sistemática de métodos, procedimentos e normas)
- Resistência à mudança

“Crise Software”

Engenharia de Software apresenta características diferentes quando comparada com outros ramos de engenharia:

- Invisibilidade
- Complexidade
- Entendimento pouco claro do **processo** de software
- Experiência anteriormente adquirida acaba por ter valor limitado na gestão dos novos projetos

Fatores Complexidade - Booch

Complexidade do domínio do problema

Dificuldade em gerir o processo de desenvolvimento

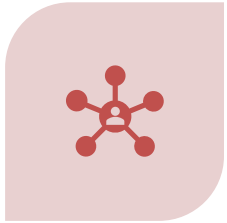
Flexibilidade exigida

Problemas que caracterizam o comportamento dos sistemas discretos.

Complexidade (factores)

CONSTANTE EVOLUÇÃO/MUDANÇA QUER DO NEGÓCIO QUER DAS TECNOLOGIAS;

GRANDE DIVERSIDADE DO SOFTWARE APLICACIONAL;



NECESSIDADE DE COORDENAÇÃO SIMULTÂNEA DE MAIS EQUIPAS E MAIS HETEROGÉNEAS;

MAIOR DISPERSÃO GEOGRÁFICA DAS EQUIPAS;

NECESSIDADE DE DESENVOLVER EM MÚLTIPLAS PLATAFORMAS TÉCNICAS;

MAIOR VISIBILIDADE DOS ERROS E DISFUNÇÕES.

Complexidade

O grau de complexidade relaciona-se com o domínio do problema e entre outros factores com a capacidade intelectual do ser humano.

Segundo as experiências de Miller um indivíduo pode abranger apenas **cerca de sete** (mais ou menos duas) fracções de informação de uma só vez na memória de curta duração.

Complexidade

- A resolução de problemas passa pela divisão em fracções, processo que atualmente se designa por abstração
- Admitimos que o que estamos a considerar é complexo e em vez de tentarmos compreender o todo selecionamos parte dele. Sabemos que existem detalhes adicionais, simplesmente optamos por não os considerar naquele momento.

Esta técnica é uma forma de **gerir a complexidade**

Complexidade

A complexidade dos sistemas implica o aparecimento de estratégias que permitam a gestão da mesma

Um dos avanços do desenvolvimento de *software* nas últimas décadas foram os conceitos:

- objetos
- componentes

UML – Unified Modeling Language

Sommerville

- A intangibilidade dos sistemas de software traz problemas para a gestão de projetos de software
  - Boa gestão é essencial para que se possa cumprir prazos e orçamento
  - Gestores de projeto de software não podem quantificar o progresso - dependem de outras pessoas para produzir a documentação necessária. Se essa documentação é insuficiente ou inexistente o gestor não tem elementos para decidir.

Relatório CHAOS - Standish Group's

Segundo o relatório CHAOS elaborado pelo Standish Group's (2012) 61% dos projetos de Tecnologias Informação não tiveram sucesso (não foram executados dentro do prazo e/ou orçamento), devido:

- má qualidade das entradas (inputs);
- a definição pouco clara dos objetivos;
- a má especificação e definição dos requisitos;
- devido a práticas deficientes de planeamento e gestão de projetos

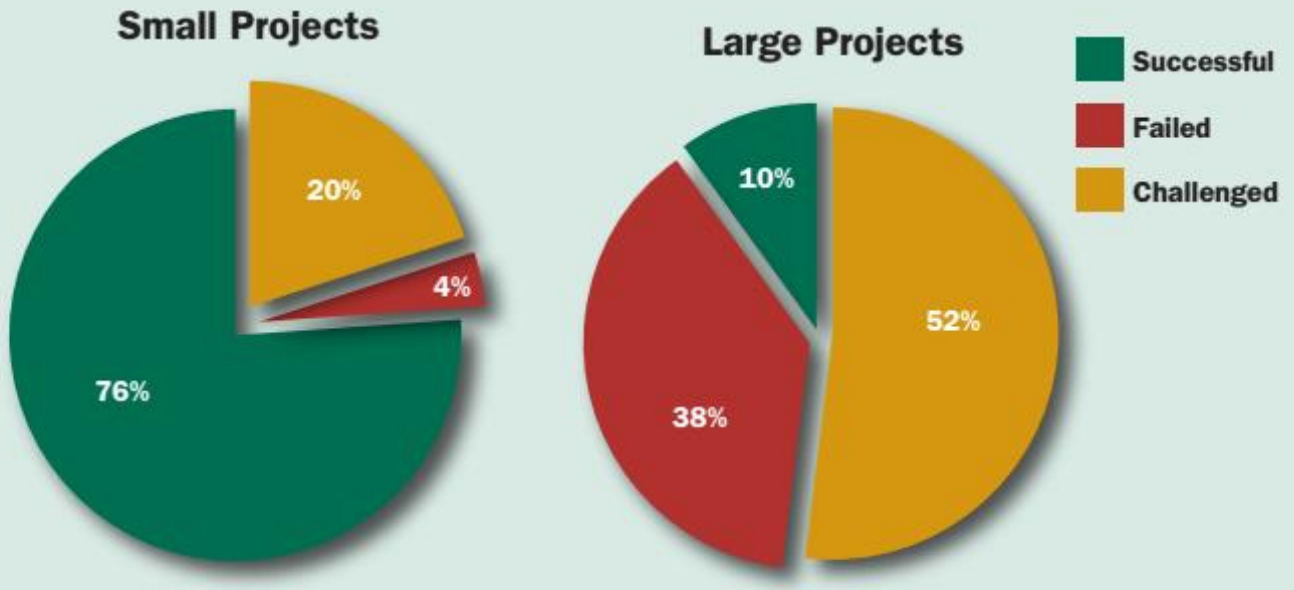
Standish Group -resultados

- projetos são cancelados
- não cumprem com as expectativas
- em média os projetos custam mais 189%
- em média os projetos ultrapassam os objetivos de prazo e custo em 220%
- O custo de fracasso ascende aos 145 mil milhões de dólares

Relatório CHAOS - Standish Group's

CHAOS RESOLUTION BY LARGE AND SMALL PROJECTS

Project resolution for the calendar year 2012 in the new CHAOS database. Small projects are defined as projects with less than $1 million in labor content and large projects are considered projects with more than $10 million in labor content.



FATORES DE SUCESSO (pequenos projetos)

Factors of Success	Points
Executive management support	20
User involvement	15
Optimization	15
Skilled resources	13
Project management expertise	12
Agile process	10
Clear business objectives	6
Emotional maturity	5
Execution	3
Tools and infrastructure	1

Tom De Marco

"56% dos erros de software podem ser detectados na fase análise de requisitos"

- Quanto mais tarde um erro é detectado, maior o custo de correcção;
- Erros típicos: factos incorretos, omissões, inconsistências e ambiguidades;
- Erros nos requisitos constituem uma das maiores preocupações da indústria de software.

Boas práticas

Sucesso dos projetos de desenvolvimento de Software implica a utilização de soluções integradas e a implementação de boas práticas de engenharia de software tais como:

- Desenvolvimento iterativo;
- Aplicação de técnicas de modelação visual;
- Garantia de qualidade através de uma verificação (testes) contínua;
- Gestão da mudança (incluindo o planeamento de tarefas/acções) e das configurações dos inúmeros objectos manipulados...

COMO EVITAR QUE PROJETOS FALHEM

Porque é que alguns projetos orientados a objetos não funcionam?

Na maior parte das vezes deve-se:

A uma incapacidade em lidar de forma adequada com os riscos;

À construção errada de alguma coisa;

À desorientação provocada pela tecnologia.

COMO EVITAR QUE PROJETOS FALHEM

- *A maior parte dos projetos não são bem sucedidos pelo facto de não existir uma vigilância adulta. São traçados planos e listas altamente irrealistas, sem ninguém que tenha a coragem de enfrentar e reconhecer a realidade.*

- A gestão do projeto tem de atacar ativamente os riscos de um projeto, caso contrário, eles atacá-lo-ão a si.

COMO EVITAR QUE PROJETOS FALHEM

Envolver os utilizadores finais no decorrer do processo de desenvolvimento de software; a presença deles relembra constantemente o porquê e para quem o software está a ser desenvolvido.

Esta é a razão pela qual os *use case* funcionam tão bem durante a análise OO: criam uma situação na qual os utilizadores finais e os analistas são obrigados a falar.

COMO EVITAR QUE PROJETOS FALHEM

A equipa de desenvolvimento deverá possuir uma visão partilhada do problema a resolver. À medida que o projeto avança, a equipa deve perguntar-se continuamente:

"Será que ainda estamos a produzir um sistema que satisfaz as características escolhidas?"

Se a resposta é positiva, então, a equipa tornou o seu percurso realmente válido.

Será que ainda estamos a produzir um sistema que satisfaz as características escolhidas?

Resposta **negativa**:

- Dividir, explicitamente, o plano de desenvolvimento, eliminar algumas funcionalidades ou reduzir certas necessidades de performance;
- Abandonar o projeto

Se um projeto reagir de outra forma significa que está, apenas, a ignorar a realidade

Projeto de Software bem sucedido

- Satisfaz e, possivelmente, excede as expectativas do cliente, que foi desenvolvido de uma forma económica e oportuna e é resistente à mudança e à adaptação
- Através desta medida, observam-se dois traços comuns a projetos bem sucedidos:

1. A existência de uma nítida visão arquitetónica
2. A aplicação de um ciclo de vida de desenvolvimento iterativo e incremental bem gerido