

# Gestão de projetos Informáticos

## Apontamentos

Prof. Cédric Grueau

2022 / 2023

### Resumo

Este documento fornece uma introdução aos conceitos de gestão de projetos na área da informática e de produtos digitais. Contém vários exemplos de artefatos produzidos no âmbito da gestão de projeto para ilustrar algumas das técnicas usadas neste domínio.

## Conteúdo

<b>1 Introdução e Definições</b>	<b>3</b>
1.1 Etapas do planeamento . . . . .	4
1.2 Definições importantes . . . . .	5
1.3 O papel do gestor de projetos . . . . .	6
<b>2 Definição do âmbito do projeto</b>	<b>7</b>
2.1 Projetos internos . . . . .	7
2.2 projetos externos . . . . .	7
2.3 Contextualizar o projeto . . . . .	8
2.4 Exemplo de definição do âmbito . . . . .	8
<b>3 Planear a qualidade do produto de software</b>	<b>9</b>
3.1 Exemplo de plano de qualidade . . . . .	10
<b>4 Planear as comunicações do projeto informático</b>	<b>11</b>
4.1 Exemplo de plano de comunicação . . . . .	12
<b>5 Planear as aquisições do projeto</b>	<b>12</b>
5.1 Exemplo de um plano de aquisições . . . . .	13
<b>6 Planear a gestão dos stakeholders</b>	<b>14</b>
6.1 Exemplo de gestão de stakeholders . . . . .	14

<b>7 Planear e definir o cronograma do projeto</b>	<b>15</b>
7.1 A estrutura analítica do trabalho (WBS) . . . . .	15
7.2 Estimar as tarefas . . . . .	15
7.3 Dependências entre tarefas . . . . .	21
7.4 Método do Caminho Crítico (CPM) . . . . .	22
7.5 Gráfico PERT . . . . .	26
7.6 Diagrama de Gantt . . . . .	26
<b>8 Gerir um projeto com uma metodologia em cascata</b>	<b>26</b>
8.1 Recolher e Analisar as necessidades . . . . .	29
8.2 Definir o âmbito do projeto . . . . .	30
8.3 As metodologias clássicas . . . . .	31
8.4 As metodologias ágeis . . . . .	32
8.5 Metodologia clássica ou ágil, eis a questão. . . . .	32
8.6 Planear e definir o cronograma do projeto . . . . .	33
8.7 Atribuir tarefas aos membros da equipa . . . . .	36
<b>9 Planeamento e estimativa dos custos do projeto</b>	<b>37</b>
9.1 Diferenciar itens de despesas . . . . .	37
9.2 Calcular o custo total do projeto . . . . .	38
9.3 Redigir a proposta comercial . . . . .	41
<b>10 Planear a Gestão do risco</b>	<b>44</b>
10.1 Exemplo . . . . .	46
10.2 Anti-padrões da gestão de projetos informáticos . . . . .	48
10.3 Anti-padrões organizacionais . . . . .	49
10.4 Anti-padrões individuais . . . . .	53
<b>11 Planear a execução</b>	<b>55</b>
11.1 Planeamento de responsabilidades . . . . .	55
<b>12 Controlar um projeto</b>	<b>58</b>
<b>13 Encerramento do projeto</b>	<b>59</b>
<b>14 Anexos</b>	<b>61</b>
14.1 Anexo A - Resposta ao concurso Projeto Digital . . . . .	62
14.2 Anexo B - Diferenças entre o planeamento ágil de projeto e o planeamento com metodologias clássicas . . . . .	64

# 1 Introdução e Definições

Se estiver de desenvolver uma aplicação Web, desenhar um carro, mudar uma empresa para uma nova sede ou atualizar um sistema de informação (grande ou pequeno), terá de passar pelas mesmas quatro fases : **planeamento, construção, implementação e encerramento**. Embora as fases tenham características distintas, elas se sobrepõem. Por exemplo, geralmente começa-se com o planeamento, com um orçamento aproximado e uma data de conclusão estimada. Define-se e começa-se a executar os detalhes do plano do projeto quando se chegue às fases de construção e implementação. Com base na informações recolhidas nestas fases, procede-se a uma revisão do orçamento e da data final a luz dos objetivos geral do projeto.



## Definição: PROJETO

Um projeto é um **conjunto de atividades coordenadas e controladas** para **atingir um objetivo** respeitando às **exigências de qualidade, de custos e de prazos**. Um projeto tem uma data de início e uma data de fim

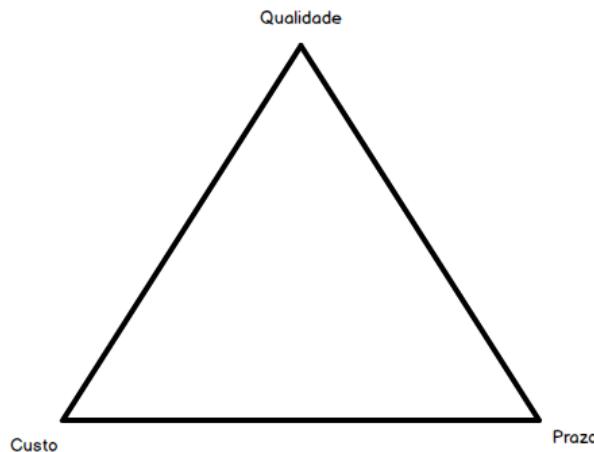


Figura 1: Triângulo qualidade - custo - prazo

A tabela 1 descreve as atividades de cada fase da gestão de projeto, bem como as competências e ferramentas necessárias para concluir o trabalho:

PLANEAMENTO	CONSTRUÇÃO	IMPLEMENTAÇÃO	ENCERRAMENTO
<b>Atividades</b>			
Determinar o problema real a resolver	Montar a equipa	Monitorizar e controlar o processo e o orçamento	Avaliar o desempenho do projeto
Identificar as partes interessadas	Definir as atribuições do plano	Relatar progresso	Fechar o projeto
Definir os objetivos do projeto	Criar o cronograma	Realizar reuniões semanais de equipa	Discussão com a equipa
Determinar o âmbito, os recursos e as tarefas principais	Realizar uma reunião inicial	Gerir os problemas	Desenvolver um relatório de pós-avaliação

PLANEAMENTO	CONSTRUÇÃO	IMPLEMENTAÇÃO	ENCERRAMENTO
Preparar-se para as compensações	Desenvolver um orçamento		
<b>Competências-Chave</b>			
Análise de tarefas	Análise de processos	Supervisão	Acompanhamento
Planejar	Formação de equipes	Liderar e motivar	Planejar
Análise custo-benefício das opções	Delegar	Comunicação	Comunicação
	Negociação	Gestão de conflitos	
	Recrutamento e contratação	Resolução de problemas	
	Comunicação		
<b>Ferramentas</b>			
Estrutura Analítica do Trabalho	Ferramentas de agendamento (CPM, PERT, Gantt)		Relatório de pós-avaliação: análise e lições aprendidas

Tabela 1 - Atividades, competências e ferramentas de cada fase da gestão de projeto

O planeamento de projetos em engenharia de software é o processo de definição das tarefas que precisam ser concluídas para desenvolver um produto de software, bem como os recursos (incluindo pessoal, equipamentos e recursos financeiros) necessários para concluir essas tarefas. Envolve o estabelecimento de um cronograma do projeto, identificando os recursos necessários para concluir o projeto e determinando o orçamento e a linha temporal do projeto.

O objetivo do planeamento do projeto é garantir que o processo de desenvolvimento de software seja bem organizado e que o projeto seja concluído dentro do prazo e dentro do orçamento. É uma parte importante do ciclo de vida de desenvolvimento de software (SDLC) e normalmente é feito no início de um projeto para definir as fundações para o processo de desenvolvimento.

Existem várias ferramentas e técnicas que podem ser usadas para facilitar o planeamento de projetos informáticos, como, por exemplo, softwares de gestão de projetos, gráficos de Gantt e metodologias ágeis.

## 1.1 Etapas do planeamento

Existem várias etapas que normalmente são seguidas para planejar um projeto informático. Essas etapas podem variar dependendo dos requisitos e restrições específicos do projeto, mas geralmente incluem:

- Definição do âmbito do projeto:** A primeira etapa no planeamento de um projeto de software é definir claramente o âmbito do projeto. Isso inclui identificar as metas e objetivos específicos do projeto, bem como os recursos e funcionalidades que serão incluídos no software.
- Realização de um estudo de viabilidade:** Após a definição do âmbito do projeto, é importante realizar um estudo de viabilidade para determinar se o projeto é viável técnica e financeiramente. Isso inclui avaliar os recursos e capacidades da organização, bem como os riscos e desafios potenciais associados ao projeto.
- Desenvolvimento de um cronograma do projeto:** Uma vez que o projeto tenha sido determinado como viável, o próximo passo é desenvolver um cronograma do projeto. Isso envolve identificar as tarefas que precisam ser concluídas para desenvolver o software e organizar essas tarefas numa linha do tempo. O

cronograma do projeto deve incluir marcos e prazos para cada tarefa, bem como os recursos necessários para concluir as tarefas.

4. **Alocação dos recursos:** com base no cronograma do projeto, a próxima etapa é alocar os recursos necessários (incluindo pessoal, equipamentos e recursos financeiros) para concluir o projeto. Isso inclui identificar quem será responsável por cada tarefa e garantir que os recursos necessários estejam disponíveis quando forem necessários.
5. **Criação de um orçamento:** Em conjunto com o cronograma do projeto e a alocação de recursos, um orçamento deve ser desenvolvido para o projeto. O orçamento deve incluir todos os custos associados ao projeto, incluindo pessoal, equipamentos e outras despesas.
6. **Obtenção da aprovação:** Uma vez desenvolvido o plano do projeto, ele deve ser apresentado às partes interessadas apropriadas para aprovação. Isso pode incluir a gestão, os clientes e outros membros da equipa de desenvolvimento.
7. **Monitorização e controlo do projeto:** Após a aprovação do projeto, é importante monitorizar e controlar atentamente o projeto para garantir que ele permaneça no caminho certo e dentro do orçamento. Isso pode envolver a revisão regular do cronograma e do orçamento do projeto e fazer os ajustes necessários.

## 1.2 Definições importantes

De acordo com a metodologia de desenvolvimento do projeto (preditiva vs ágil) as etapas divergem. A tabela 2 descreve algumas etapas específicas da gestão de projeto consoante a abordagem de desenvolvimento do projeto.

Abordagem preditiva	Abordagem ágil
1. calcular o prazo	1. Estimativa relativas das funcionalidades
2. calcular o custo	2. Atribuição das funcionalidades às várias releases (roadmap)
3. identificar as atividades	3. (release) clarificação das funcionalidades (user stories) e afetação às várias iterações
4. calcular a duração das atividades	4. (iteração) levantamento das tarefas para user story tratada
5. ordenar as atividades	5. (daily stand-up meeting ) ajustamento do planeamento, atribuição das tarefas
6. estabelecer o planeamento	
7. ajustar o planeamento	

Tabela 2 - Atividades, competências e ferramentas de cada fase da gestão de projeto

O desenvolvimento de um novo projeto / projeto é uma empreiteira assustadora. Envolve tarefas de desenho, escrita de código, testes, marketing, apoio técnico e gestão de sistemas, etc.

A solução passa por **dividir, planejar, gerir**.



Figura 2: Estratégia da gestão de projeto reduzir a complexidade

A engenharia de software fornece abordagens diferentes para a decomposição :

- as tarefas compõem atividades
- as atividades compõem fases
- Cada fase possui tarefas que são executadas

As práticas ágeis fornecem formas de organizar as tarefas de maneira a que elas estejam efetivamente executadas - Scrum - plano de iterações - Kanban - forma visual de rastrear o progresso das tarefas através de um conjunto de etapas

Todas as atividades estão envolvidas: análise, design, desenvolvimento, teste, documentação... Estas são essenciais para a implementação completa de uma funcionalidade, potencialmente entregável no final da iteração.

Palavra	Definição
<b>Tarefa</b> (Task)	Uma tarefa é uma etapa pequena e manejável de um projeto que deve ser concluída. As tarefas são essenciais para um projeto - tudo num projeto pode ser dividido em tarefas.
<b>Papel</b> (Role)	Um papel é um dever que uma pessoa assume ou desempenha em relação com o produto, Exemplos de funções incluem programador, operador de testes, designer, cliente e gestor de produto.
<b>Produto de trabalho</b> (Work Product)	Um produto de trabalho é uma saída/output produzido por uma tarefa ou processo,
<b>Agendamento</b> (Schedule)	Um agendamento é criado quando as tarefas de um projeto são mapeadas para uma linha de tempo.
<b>Marco</b> (Milestone)	Marcos referem-se a pontos de verificação internos utilizados para medir o progresso. Eles não são baseados no tempo, mas baseados em evento ou ação. Os marcos são mais usados em processos lineares e iterativos precoces de um projeto, Em ágil, os marcos não são geralmente usados. O progresso é, em vez disso, medido pelo software de trabalho em oposição a eventos ou ações. Releases e iterações tendem a serem baseados no tempo também, o que não se enquadra com marcos.

Tabela 3 - Algumas definições importantes da gestão de projeto.

### 1.3 O papel do gestor de projetos

Para muitos, o papel do gestor de projetos parece tão transversal que é difícil de entender. De fato, é muito mais fácil entender o papel de um designer gráfico, fotógrafo ou programador porque eles possuem competências claramente identificáveis:

- o designer gráfico cria visuais com seu software
- o fotógrafo tira fotos com a sua câmera
- o programador escreve código num editor de texto ou numa consola

O gestor de projeto é a interface entre as várias profissões e o patrocinador do projeto (a entidade que encerra o projeto).

Para fazer uma analogia, um gestor de projeto é um pouco como um maestro de orquestra. Ao contrário dos membros da orquestra (designer gráfico, programador, etc.) não necessariamente toca um instrumento (software, editor de texto, etc.) mas garante a coesão geral, harmonia entre todos os membros para que o concerto (entregável) atende às expectativas do público (o cliente).

Se a analogia musical não lhe agradar, tente criar sua própria analogia cinematográfica substituindo "maestro" por "diretor".

Em outras palavras:

Um gestor de projeto coordena todas as atividades do projeto para garantir o alcance do objetivo e o respeito dos requisitos de qualidade, custos e prazos.

Para simplificar ainda mais, o gestor de projeto é responsável pelo bom andamento do projeto.

Observe que quanto mais atores no palco houver para coordenar, mais complexa e crucial se torna a tarefa do gestor de projeto.

Para o gestor de projeto, "organização" e "comunicação" são as duas palavras-chave porque, concretamente, ele passa muito tempo a criar e atualizar documentos e a comunicar.

## 2 Definição do âmbito do projeto

Definir o âmbito do projeto informático é o processo de identificação das metas, objetivos e requisitos específicos de um projeto de software, bem como os limites do projeto. Trata-se de definir claramente o que será incluído no software, bem como o que não será incluído.

Se se verifique que é difícil identificar claramente a(s) necessidade(s) que um projeto aborda, pode não haver necessidade de fazer um projeto. Pode ser útil colocar esta pergunta.

Dependendo de quem expressa as necessidades e de que forma são expressas, o gestor do projeto terá que gerir cenários potencialmente muito diferentes. A primeira distinção a fazer é a diferença entre os projetos internos e externos.

### 2.1 Projetos internos

Em alguns casos, evoluí-se dentro da própria organização que precisa do projeto, como funcionário. Pode-se até estar na origem do projeto para o qual a necessidade foi identificada.

Nesse caso, é necessário saber que fazer parte da organização não é suficiente para garantir o sucesso do projeto. Especialmente se sua função principal nesta organização não é gerir o referido projeto e será necessário fazer isso além de outras responsabilidades, existe o risco de o projeto fracassar rapidamente ou, pior ainda, de ficar à deriva indefinidamente.

Nas grandes organizações, não é raro encontrar projetos iniciados há anos, sem avanços significativos. Esses projetos podem ter parado por falta de recursos ou nunca terminar por falta de alinhamento, de pessoal qualificado, de metodologia de gestão adequada ou dos três.

### 2.2 projetos externos

Em outros casos, uma organização pode identificar uma necessidade, mas não lançar um projeto interno. Isso pode acontecer devido à falta de recursos internos ou à distância do core business. Nesse caso, essa organização pode ser o patrocinador do projeto e trabalhar com uma agência ou outro tipo de prestador de serviço.

Novamente, vários cenários podem surgir. O patrocinador pode contactar directamente os prestadores de serviços, caso em que a manifestação de necessidades é feita à porta fechada. Caso contrário, o promotor pode

decidir abrir um concurso ou um anúncio de concurso. O concurso consiste na publicação de uma manifestação de necessidades num sítio especializado e no convite aos prestadores de serviços para estudar e selecionar o melhor.

Nem todas as propostas são iguais. Alguns são mais ou menos detalhados dependendo da capacidade do patrocinador em identificar e formular as suas necessidades ou mesmo as possíveis soluções para elas. Enquanto alguns patrocinadores ficarão satisfeitos com algumas frases, outros irão tão longe a ponto de escrever um documento de especificação detalhando o problema e a solução que esperam ver executada.

## 2.3 Contextualizar o projeto

No contexto deste documento, iremos usar, mais a frente, um exemplo de empresa/agência, chamada DIGITALIZER, que produz aplicações Web. Esta agência será usada para responder a um concurso e realizar o planeamento de um projeto de um site para um hotel, o hotel Sado (nome fictício).

O processo de definição do âmbito do projeto geralmente inclui as seguintes etapas:

1. Identificar as partes interessadas do projeto (stakeholders): A primeira etapa na definição do âmbito do projeto é identificar todas as partes interessadas que serão impactadas pelo projeto. Isso pode incluir a equipa de desenvolvimento, a gestão, os clientes e os seus parceiros, assim como os utilizadores do software.
2. Determinar as metas e objetivos do projeto: O próximo passo é definir as metas e objetivos específicos do projeto. Isso inclui identificar o que o software pretende fazer e qual valor este irá fornecer às partes interessadas.
3. Definir os requisitos do projeto: Com base nas metas e objetivos do projeto, o próximo passo é definir os requisitos específicos para o software. Isso inclui identificar as características e as funcionalidades que o software deve ter para atender às necessidades das partes interessadas.
4. Definir os limites do projeto: Após a identificação dos requisitos, o próximo passo é definir os limites do projeto. Isso inclui identificar o que será incluído no software, bem como o que não será incluído. É importante ser o mais específico possível para evitar aumento do âmbito, que pode ocorrer quando recursos ou requisitos adicionais são adicionados ao projeto após o seu início.
5. Documentar o âmbito do projeto: Uma vez definido o âmbito do projeto, é importante documentá-lo de forma clara e concisa. Isso pode envolver a criação de uma declaração de âmbito do projeto ou um termo de abertura do projeto, que pode ser partilhado com todas as partes interessadas.

Definir o âmbito do projeto é uma etapa importante no processo de desenvolvimento de software, pois ajuda a garantir que o projeto permaneça no caminho certo e que todas as partes interessadas estejam cientes do que está incluído no projeto. Também ajuda a prevenir o crescimento do âmbito durante o desenvolvimento do âmbito, o que poderia causar atrasos e aumentar o custo do projeto.

## 2.4 Exemplo de definição do âmbito

Segue um exemplo de um plano de âmbito para uma aplicação web:

**Nome do Projeto:** Loja de Vendas Online

**Âmbito:**

A aplicação web destina-se a fornecer uma plataforma para os utilizadores comprarem e venderem itens artesanais e vintage. Ela permitirá que os utilizadores criem perfis, listem itens para venda e comuniquem entre si por

meio de um sistema de mensagens interno. A aplicação web também incluirá um sistema de processamento de pagamentos para que os utilizadores concluam as transações.

#### **Partes interessadas:**

- Clientes: os clientes usarão a aplicação web para navegar e comprar itens. Eles estarão interessados na qualidade e variedade de itens disponíveis, bem como na facilidade de uso da plataforma.
- Vendedores: os vendedores usarão a aplicação web para listar e vender seus itens artesanais e vintage. Eles estarão interessados nas taxas associadas ao uso da plataforma, bem como no número de clientes potenciais.
- Equipa de desenvolvimento: A equipa de desenvolvimento será responsável por desenhar e construir a aplicação web. Eles estarão interessados nos requisitos técnicos do projeto, bem como no cronograma e no orçamento.
- Gestão: A gestão será responsável por supervisionar o desenvolvimento da aplicação web. Eles estarão interessados no desempenho financeiro da plataforma, bem como na satisfação dos clientes e vendedores.

#### **Exclusões:**

As seguintes funcionalidades não serão incluídas na aplicação web:

- Uma plataforma de media social: a aplicação web não incluirá recursos como feed de notícias ou capacidade de seguir outros utilizadores.
- Um sistema de revisão: a aplicação web não incluirá um sistema para os utilizadores avaliarem ou avaliam item ou vendedores.
- Um programa de fidelidade: a aplicação web não incluirá um programa de fidelização ou sistema de recompensas para utilizadores frequentes.

## **3 Planear a qualidade do produto de software**

Existem várias etapas que podem ser tomadas para planear a qualidade num projeto de software. Essas etapas podem ajudar a garantir que o produto final atenda às necessidades das partes interessadas e que esteja livre de defeitos. Aqui estão algumas etapas que podem ser tomadas para planear a qualidade num projeto de software:

1. **Definir os padrões de qualidade:** O primeiro passo no planeamento da qualidade é definir os padrões de qualidade que o software deve atender. Isso pode incluir padrões definidos pela organização, bem como quaisquer padrões relevantes do setor.
2. **Identificar riscos potenciais:** A próxima etapa é identificar quaisquer riscos potenciais que possam afetar a qualidade do software. Isso pode incluir riscos técnicos, como a complexidade do software, ou riscos organizacionais, como falta de recursos.
3. **Desenvolver um plano de qualidade:** Com base nos padrões de qualidade e nos riscos identificados, o próximo passo é desenvolver um plano de qualidade. Este plano deve delinear as etapas específicas que serão tomadas para garantir que o software atenda aos padrões de qualidade exigidos.
4. **Implementar controles de qualidade:** O plano de qualidade deve incluir controles de qualidade específicos que serão implementados ao longo do processo de desenvolvimento de software. Esses controles podem incluir testes e depuração, bem como revisões de código e revisões por pares.

5. **Monitorar e controlar a qualidade:** À medida que o software está sendo desenvolvido, é importante monitorar e controlar regularmente a qualidade do produto. Isso pode envolver a realização de inspeções e testes regulares para garantir que o software atenda aos padrões de qualidade exigidos.
6. **Realize uma revisão final de qualidade:** antes do lançamento do software, é importante realizar uma revisão final de qualidade para garantir que todos os padrões de qualidade exigidos foram atendidos. Isso pode envolver testar o software novamente, bem como revisar toda a documentação e outros materiais associados ao projeto.

Seguindo essas etapas, as organizações podem planejar a qualidade em um projeto de software e aumentar a probabilidade de entregar um produto de software de alta qualidade.

### 3.1 Exemplo de plano de qualidade

A seguir descreve-se um exemplo de um plano de qualidade para uma aplicação web:

1. **Padrões de qualidade:** a aplicação web atenderá aos seguintes padrões de qualidade:
  - a aplicação web será visualmente atraente e fácil de usar.
  - a aplicação web será acessível a utilizadores com deficiência.
  - a aplicação web será testada e compatível com as versões mais recentes dos principais navegadores web.
  - a aplicação web terá um tempo de resposta inferior a 3 segundos para todas as páginas.
2. **Riscos potenciais:** Foram identificados os seguintes riscos que podem afetar a qualidade da aplicação web:
  - Complexidade da aplicação web: a aplicação web inclui um grande número de características e funcionalidades, o que pode dificultar o teste e a depuração.
  - Mudanças na aplicação web: Espera-se que a aplicação web sofra alterações e atualizações ao longo do tempo, o que pode introduzir novos defeitos.
  - Compatibilidade com diferentes navegadores web: a aplicação web pode não ser mostrada ou funcionar corretamente em alguns navegadores web.
3. **Plano de qualidade:** As seguintes etapas serão realizadas para garantir que a aplicação web atenda aos padrões de qualidade exigidos:
  - Design visual: a aplicação web será projetado por um designer profissional para garantir que seja visualmente atraente e fácil de usar.
  - Acessibilidade: a aplicação web será testado usando ferramentas automatizadas e manualmente para garantir que seja acessível a utilizadores com deficiência.
  - Teste de compatibilidade: a aplicação web será testado nas versões mais recentes dos principais navegadores web para garantir que seja compatível com todos eles.
  - Teste de desempenho: a aplicação web será testado para garantir que tenha um tempo de resposta inferior a 3 segundos para todas as páginas.
  - Rastreamento de defeitos: quaisquer defeitos identificados durante o teste serão rastreados e resolvidos usando uma ferramenta de rastreamento de defeitos.
4. **Controles de qualidade:** Os seguintes controles de qualidade serão implementados ao longo do processo de desenvolvimento:
  - Revisões de código: todo o código será revisado por pelo menos um outro desenvolvedor antes de ser verificado no repositório de código.

- Revisão por pares: Todos os documentos técnicos e de design serão revisados por pelo menos um outro membro da equipe antes de serem finalizados.
- Teste: a aplicação web será testado em vários estágios do processo de desenvolvimento, incluindo teste de unidade, teste de integração e teste de aceitação.

5. **Monitorização e controlo:** As seguintes etapas serão realizadas para monitorar e controlar a qualidade da aplicação web:

- Inspeções regulares: a aplicação web será inspecionado regularmente pela equipe da desenvolvimento para identificar possíveis defeitos.
- Teste: a aplicação web será testado regularmente para garantir que atenda aos padrões de qualidade exigidos.
- Rastreamento de defeitos: Quaisquer defeitos identificados serão rastreados e resolvidos usando uma ferramenta de rastreamento de defeitos.

6. **Revisão de qualidade final:** antes do lançamento da aplicação web, uma revisão de qualidade final será realizada para garantir que todos os padrões de qualidade exigidos foram atendidos. Isso incluirá testar a aplicação web novamente e revisar toda a documentação e outros materiais associados ao projeto.

## 4 Planear as comunicações do projeto informático

A comunicação eficaz é essencial para o sucesso de qualquer projeto informático. Na engenharia de software, o planeamento das comunicações ajuda a garantir que todos os membros da equipa de desenvolvimento e as partes interessadas estejam cientes das metas do projeto, do progresso e de quaisquer problemas que possam surgir. Também ajuda a facilitar a colaboração e a coordenação entre os membros da equipa e pode melhorar a eficiência geral do processo de desenvolvimento.

Algumas razões específicas pelas quais o planeamento de comunicações é importante na engenharia de software incluem:

- Esclarecendo as metas e objetivos do projeto: ao planejar as comunicações, os gerentes de projeto podem garantir que todos os membros da equipe e as partes interessadas estejam cientes das metas e objetivos específicos do projeto. Isso pode ajudar a garantir que todos estejam trabalhando para os mesmos objetivos e que não haja confusão sobre o escopo do projeto.
- Gestão dos riscos do projeto: A comunicação eficaz pode ajudar os gestores de projeto a identificar e gerir os riscos potenciais associados ao projeto. Ao manter os membros da equipa e as partes interessadas informados sobre o status do projeto, os gerentes de projeto podem identificar possíveis questões antes que elas se tornem problemas e desenvolver um plano para resolvê-los.
- Melhorar a colaboração e a coordenação: o planeamento das comunicações pode ajudar a facilitar a colaboração e a coordenação entre os membros da equipa, pois permite que eles partilhem ideias, discutam o progresso e resolvam problemas em tempo útil. Isso pode ajudar a melhorar a eficiência geral do processo de desenvolvimento.
- Reduzir o risco de falha do projeto: ao planejar as comunicações, os gestores de projeto podem garantir que os membros da equipa e as partes interessadas estejam cientes do status do projeto e de quaisquer problemas que possam surgir. Isso pode ajudar a reduzir o risco de falha do projeto, pois permite que os gestores de projeto identifiquem e resolvam os problemas antes que se tornem sérios.

No geral, o planeamento das comunicações é uma parte essencial do processo de desenvolvimento de software, pois ajuda a garantir que todas as partes interessadas sejam informadas sobre o projeto e que o projeto seja concluído com eficiência e sucesso.

## 4.1 Exemplo de plano de comunicação

Descreve-se a seguir um exemplo de um plano de comunicação para um projeto de aplicação web:

1. **Canais de comunicação:** A equipa do projeto usará uma variedade de canais de comunicação, incluindo e-mail, telefonemas e reuniões presenciais. A equipe também usará uma ferramenta de gestão de projetos, como Asana ou Trello, para rastrear as tarefas e comunicar atualizações.
2. **Reuniões:** A equipa do projeto realizará reuniões semanais para discutir o progresso e abordar quaisquer questões ou preocupações. Essas reuniões serão presenciais ou por videoconferência.
3. **Atualizações de status:** o gestor de projeto enviará um e-mail de atualização de status semanal para a equipa e as partes interessadas, descrevendo o progresso feito no projeto e quaisquer desafios ou problemas que tenham surgido.
4. **Tomada de decisão:** As decisões relacionadas com o projeto serão tomadas por meio de uma combinação de reuniões presenciais e discussões por e-mail. O gestor de projeto reunirá informações da equipa e das partes interessadas e apresentará opções para consideração.
5. **Resolução de problemas:** Se surgirem problemas ou desafios durante o projeto, o gestor do projeto trabalhará com a equipa para identificá-los e resolvê-los o mais rápido possível. Se necessário, o gestor de projeto escalará os problemas para a parte interessada apropriada para resolução.
6. **Comunicação com as partes interessadas:** o gestor do projeto manterá as partes interessadas informadas sobre o andamento do projeto por meio de atualizações e reuniões regulares. A equipa também estará disponível para responder a quaisquer perguntas ou preocupações que as partes interessadas possam ter.

Ao seguir este plano de comunicação, a equipa do projeto poderá comunicar efetivamente entre si e com as partes interessadas, garantindo que todos estejam na mesma página e que o projeto seja concluído com sucesso.

## 5 Planear as aquisições do projeto

O planeamento de aquisições de um projeto informático refere-se ao processo de identificação e aquisição de ferramentas ou recursos de software que serão utilizados no desenvolvimento de um projeto de software. Isso pode incluir a aquisição de bibliotecas, estruturas ou outras ferramentas de terceiros necessárias para criar o software.

Existem várias razões pelas quais o planeamento de aquisições é importante na engenharia de software:

1. **Eficiência de custo:** a aquisição de ferramentas de software ou recursos necessários para um projeto geralmente pode ser mais econômica do que desenvolvê-los internamente. Isso é especialmente verdadeiro para ferramentas especializadas ou complexas que exigiriam um investimento significativo de tempo e recursos para serem desenvolvidas.
2. **Economia de tempo:** a aquisição de ferramentas ou recursos de software também pode economizar tempo, pois permite que as organizações aproveitem o trabalho de outras pessoas em vez de começar do zero. Isso pode ajudar a reduzir o tempo geral de desenvolvimento de um projeto.
3. **Qualidade aprimorada:** em alguns casos, a aquisição de ferramentas ou recursos de software pode levar a uma qualidade aprimorada, pois permite que as organizações usem ferramentas bem testadas e confiáveis que foram desenvolvidas por especialistas na área.

4. **Acesso a conhecimentos especializados:** A aquisição de ferramentas ou recursos de software também pode fornecer às organizações acesso a conhecimentos especializados que talvez não tenham internamente. Isso pode ser particularmente útil para organizações que trabalham em projetos complexos ou especializados.

Ao planejar as aquisições, as organizações podem desenvolver produtos de software de maneira mais eficiente e eficaz e gerir melhor os custos, cronogramas e riscos associados ao processo de desenvolvimento.

## 5.1 Exemplo de um plano de aquisições

O plano de aquisições geralmente inclui uma lista dos recursos e ferramentas necessários, bem como um cronograma para adquiri-los.

Segue-se um exemplo de um plano de aquisições para um projeto de engenharia de software:

Projeto: Software XYZ

Plano de Aquisições:

1. **Hardware:** Dois novos servidores são necessários para dar suporte ao desenvolvimento e teste do software. Esses servidores serão usados para hospedar os ambientes de desenvolvimento e teste e precisarão ser configurados com o software e as ferramentas necessárias. A equipa de desenvolvimento também precisará de seis novos laptops, com no mínimo 8 GB de RAM e 250 GB de armazenamento, para dar suporte ao trabalho no projeto.

2. **Programas:** A equipa de desenvolvimento precisará ter acesso ao seguinte software:

- Um ambiente de desenvolvimento integrado (IDE), como Visual Studio ou Eclipse
- Um sistema de controle de versão como o Git
- Um sistema de rastreamento de bugs como o Jira

### 3. Serviços:

- A equipa de desenvolvimento precisará de acesso a um ambiente de teste baseado na nuvem, como AWS ou Azure, para dar suporte a seus esforços de teste.
- O projeto também exigirá os serviços de um designer de UX, que será responsável por projetar a interface do utilizador para o software.

### 4. Linha do tempo:

- O hardware e software necessários para o projeto serão adquiridos até o final do primeiro trimestre.
- O designer de UX será contratado até o final do segundo trimestre.
- O ambiente de teste baseado na nuvem será instalado e configurado até o final do terceiro trimestre.

Este é apenas um exemplo de um plano de aquisições para um projeto de engenharia de software. Os recursos e ferramentas específicos incluídos no plano dependerão dos requisitos e restrições específicos do projeto.

## 6 Planear a gestão dos stakeholders

A gestão dos stakeholders é um aspecto importante da gestão de projeto porque ajuda a garantir que as necessidades e expectativas de todas as partes interessadas sejam levadas em consideração durante o processo de desenvolvimento. As partes interessadas incluem quaisquer indivíduos ou grupos afetados pelo projeto ou que tenham interesse no seu resultado.

Algumas partes interessadas comuns em projetos de engenharia de software incluem:

- **Clientes:** Os clientes geralmente são os principais interessados num projeto de software, pois são eles que usarão o software. É importante gerir as suas expectativas e garantir que as suas necessidades sejam atendidas.
- **Utilizadores:** Os utilizadores são os indivíduos que realmente usarão o software diariamente. É importante considerar as suas necessidades e feedback ao desenvolver o software.
- **Equipa de desenvolvimento:** A equipa de desenvolvimento é responsável por realmente construir o software. É importante gerir as suas expectativas e garantir que eles tenham os recursos e suporte necessários para concluir o projeto no prazo e dentro do orçamento.
- **A Gestão:** A gestão é responsável por supervisionar o projeto e garantir que ele seja bem-sucedido. É importante gerir as suas expectativas e mantê-los informados sobre o andamento do projeto.
- **Outros departamentos:** Outros departamentos dentro da organização também podem ser afetados pelo projeto, e é importante considerar as suas necessidades e preocupações ao desenvolver o software.

Ao gerir adequadamente as partes interessadas, as organizações podem aumentar a probabilidade de entregar um produto de software que atenda às necessidades de todas as partes interessadas e seja bem-sucedido. Isso pode envolver comunicação regular com as partes interessadas, recolhendo e incorporando os seus comentários no processo de desenvolvimento e gestão das suas expectativas.

### 6.1 Exemplo de gestão de stakeholders

Etapas para a gestão de stakeholders em engenharia de software:

- Identificar as partes interessadas: A primeira etapa no gerenciamento das partes interessadas é identificar todos os indivíduos ou grupos que serão afetados pelo projeto de software. Isso pode incluir a equipe de desenvolvimento, gerenciamento, clientes e utilizadores do software.
- Analisar as necessidades e expectativas das partes interessadas: Uma vez que as partes interessadas tenham sido identificadas, o próximo passo é analisar suas necessidades e expectativas. Isso pode envolver a realização de entrevistas ou pesquisas para coletar informações sobre o que as partes interessadas esperam do software.
- Desenvolva um plano de comunicação das partes interessadas: com base nas necessidades e expectativas das partes interessadas, a próxima etapa é desenvolver um plano de comunicação que descreva como as informações serão compartilhadas com as partes interessadas ao longo do processo de desenvolvimento de software. Isso pode incluir reuniões regulares, atualizações e relatórios de progresso.
- Envolver-se com as partes interessadas: é importante envolver-se ativamente com as partes interessadas em todo o processo de desenvolvimento de software. Isso pode envolver a busca de informações e feedback das partes interessadas e a abordagem de quaisquer preocupações ou problemas que possam ter.

- Monitorar a satisfação das partes interessadas: é importante monitorar regularmente a satisfação das partes interessadas para garantir que suas necessidades e expectativas sejam atendidas. Isso pode envolver a realização de pesquisas ou grupos focais para coletar feedback sobre o projeto de software.

Ao seguir essas etapas, as organizações podem gerenciar com eficácia as partes interessadas em um projeto de engenharia de software e garantir que suas necessidades e expectativas sejam atendidas. Isso pode ajudar a aumentar as chances de sucesso do projeto e melhorar a qualidade geral do produto de software.

## 7 Planear e definir o cronograma do projeto

### 7.1 A estrutura analítica do trabalho (WBS)

A A estrutura analítica do trabalho, ou Work Breakdown Structure (WBS) em Inglês, é uma representação hierárquica do trabalho que precisa ser concluído num projeto. Ele é usado para dividir o projeto em partes de trabalho menores e mais fáceis de gerir, chamadas de "entregáveis". Cada entrega é definida em termos das tarefas específicas que precisam de ser concluídas para alcançá-la.

A WBS normalmente começa com o objetivo geral do projeto no nível superior. Este objetivo geral é devidamente dividido em sub-entregas sucessivamente mais detalhadas. Isso permite que a equipa do projeto tenha uma visão geral, ao mesmo tempo em que se concentra nas ações específicas que precisam de ser executadas para levar o projeto adiante.

A WBS geralmente é representada visualmente, usando um diagrama em forma de árvore que mostra as relações entre as diferentes entregas. Ele pode ser usado para ajudar a planejar e agendar o projeto, bem como para acompanhar o progresso e identificar possíveis congestionamento de tarefas ou problemas que possam surgir.

De forma geral, a WBS é uma ferramenta útil para organizar e gerir o trabalho de um projeto, pois ajuda a garantir que todas as tarefas necessárias sejam identificadas e concluídas de maneira lógica e eficiente.

### 7.2 Estimar as tarefas

Na engenharia de software, as estimativas são previsões ou previsões dos recursos (como tempo, custo ou esforço) que serão necessários para concluir uma tarefa ou projeto.

As estimativas são normalmente baseadas nas informações disponíveis sobre a tarefa ou projeto, incluindo os requisitos, os recursos que serão necessários e quaisquer riscos ou incertezas potenciais. Eles são usados para planejar e programar o trabalho e alocar recursos de forma adequada.

As estimativas geralmente estão sujeitas a um certo grau de incerteza, pois pode ser difícil prever exatamente quanto tempo ou esforço será necessário para concluir uma tarefa ou projeto. Como resultado, as equipes de engenharia de software podem usar técnicas como estimativa de três pontos ou análise de ponto de função para gerar uma variedade de estimativas e contabilizar possíveis riscos e incertezas.

Em geral, as estimativas são uma ferramenta importante para gerir projetos de engenharia de software, pois ajudam a garantir que o projeto seja concluído dentro dos recursos alocados e dentro do cronograma.

A estimativa de tarefas em horas pode ser uma técnica útil para planejar e programar um projeto de software, pois permite que a equipa do projeto tenha uma compreensão clara do esforço necessário para cada tarefa. Isso pode ajudar a identificar possíveis problemas que possam surgir e permitir que ajustes sejam feitos conforme necessário.

No entanto, é importante observar que as estimativas de tarefas geralmente estão sujeitas a um certo grau de incerteza, principalmente em projetos de software em que os requisitos podem mudar ou novos problemas podem surgir. Como resultado, é importante ser flexível e estar preparado para ajustar as estimativas à medida que o projeto avança.

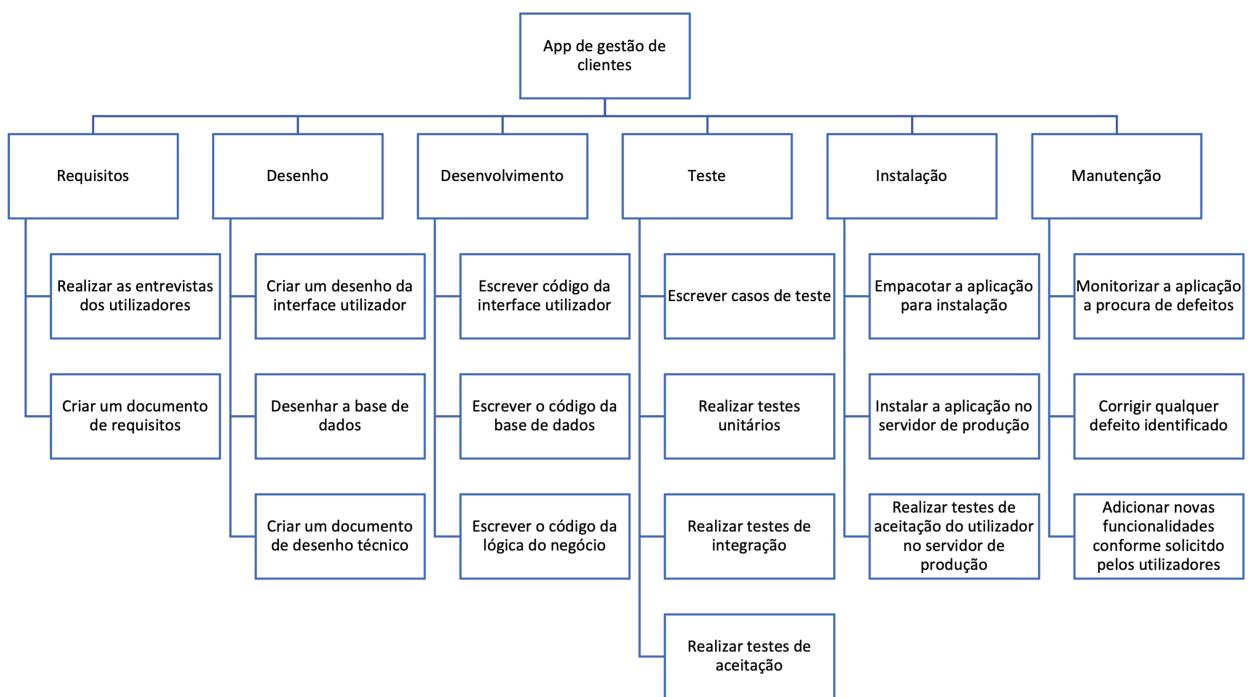


Figura 3: Exemplo de WBS de um projeto, usando uma abordagem em cascata

A variabilidade refere-se aos fatores ou à extensão pela qual uma estimativa pode variar. Esta variabilidade muda em função da evolução do projeto.

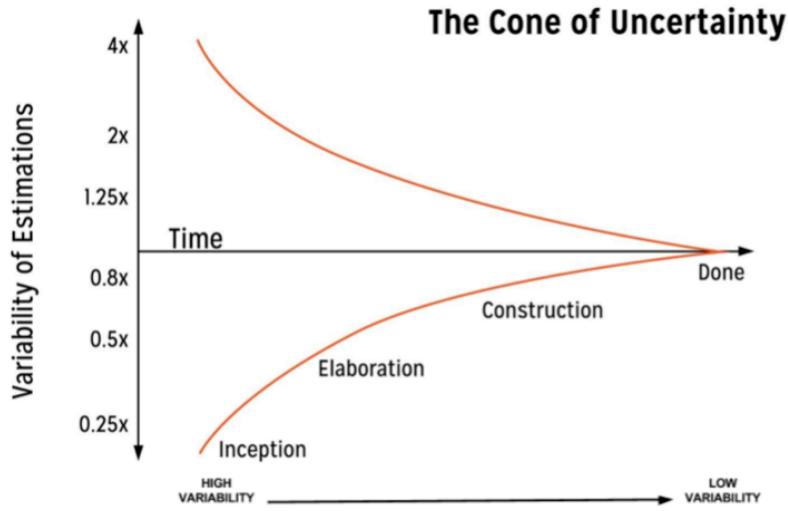
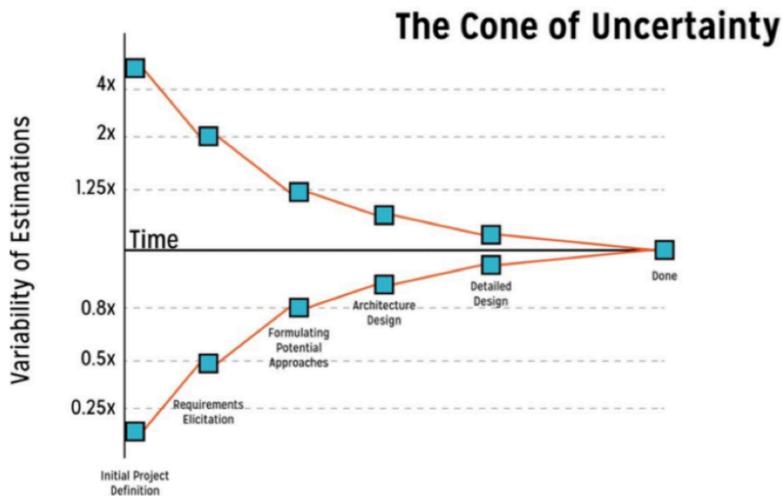


Figura 4: O cone da incerteza - Adaptado de Software Estimation: Demystifying the Black Art (2006) by Steven C. McConnell.

Nas etapas iniciais, a variabilidade da estimativa é maior, enquanto na fase de construção, a variabilidade é menor:

- **Definição Inicial do Produto** -(formulação inicial da ideia do produto). Nesta fase, a variabilidade é elevada (entre o fator aproximado de 3 e 0,375), e as estimativas necessitarão de um grande intervalo.
- **Elicitação de Requisitos** - (quando as necessidades do produto e características foram identificadas). Nesta fase, a variação é menor (entre aproximadamente 2 e 0,5), e os intervalos de estimativa serão menores também.
- **Formulação das Abordagens Potenciais** - quando os métodos usados para fazer o projeto são escolhidos. Nesta fase, a variação(entre o fator aproximado de 1,25 e 0,8)e os intervalos de estimativa tornam-se ainda menores.
- **Desenho de Arquitetura e Desenho Detalhado**, quando a informação é conhecida sobre como e o que será criado no produto final. Nestas fases, a estimativa torna-se ainda mais fiável à medida que a variação diminui(a variação torna-se cada vez mais próxima de um factor de 1).



Os intervalos de estimativa para o projeto podem ser avaliados usando a variabilidade ilustrada no gráfico: A estimativa deve ser multiplicada pela sua variabilidade fatorial correspondente à fase em que se encontra o projeto. Isso permitirá determinar os valores superiores e inferiores do intervalo de estimativa. Por exemplo, se uma tarefa é estimada a 6 horas na fase de elicitação de requisitos, isso significa que tem uma variabilidade de estimação de 2 a 0,5 no eixo y. -  $2 \times 6 = 12$ ,  $0,5 \times 6 = 3$  - Isto dá um intervalo estimado de 3 a 12 horas. - Nesta fase, ainda há muita incerteza.

Usar a variabilidade para criar intervalos de estimativa ajuda a garantir um intervalo mais preciso. As estimativas também devem ser continuamente revisadas à medida que o projeto se move através do Cone.

Uma maneira de lidar com essa incerteza é usar uma técnica como a estimativa de três pontos, que permite que uma variedade de estimativas potenciais seja considerada. Isso pode ajudar a fornecer uma estimativa mais realista do esforço necessário para cada tarefa e melhorar a precisão geral do cronograma do projeto.

### 7.2.1 Criação de estimativas temporais

Existem 4 abordagens diferentes para criar estimativas temporais:

- Bottom-up
- Analogia
- Peritos
- Usar a fórmula de estimativa dos 3 pontos

**7.2.1.1 Abordagem Bottom up** Esta abordagem é baseada na estrutura dos produtos de trabalho. Procede-se a uma estimativa do tempo de cada tarefa individual. Trata-se de uma técnica **mais rápida**, se a informação está disponível. Pressupõe que nenhuma das tarefas na estrutura de divisão de trabalho (WBS) pode ser executada simultaneamente, mas sim que cada tarefa deve ser concluída antes que outra possa ser iniciada.

**7.2.1.2 Abordagem da analogia** A estimativa de tempo é criada **comparando a tarefa ou projeto atual com uma experiência similar**. Se uma tarefa ou projeto semelhante foi realizado no passado com âmbito e custos semelhantes, então pode ser uma boa base para estimar o projeto atual.

Esta técnica deve ser usada com alguma cautela:

- Só pode funcionar se a **equipa for comum aos dois projetos**
- O trabalho, o processo, as tecnologias e os membros da equipa devem ser iguais ou similares ao projeto atual para serem uma base significativa de dados.

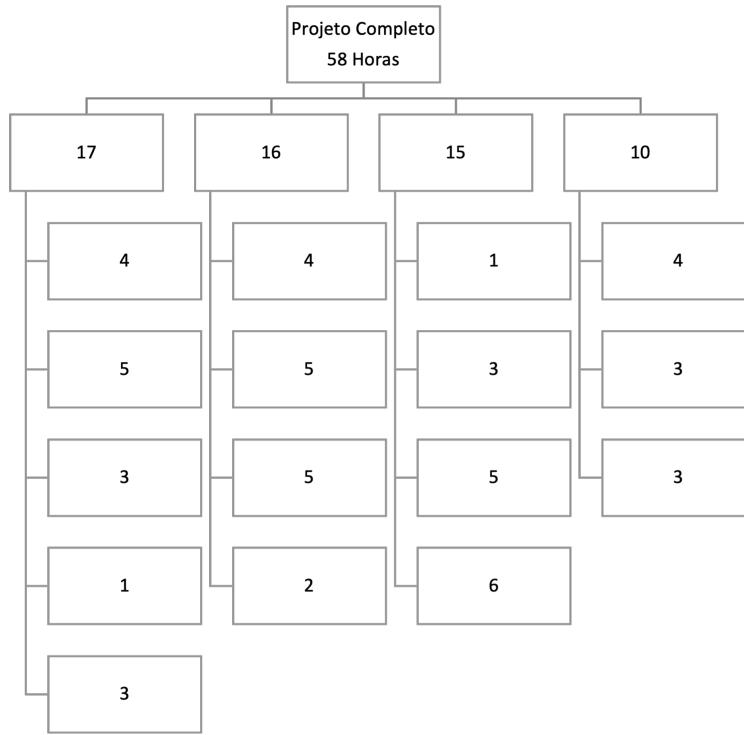


Figura 5: Exemplo de uma abordagem bottom up

**7.2.1.3 Peritos** Esta abordagem consiste em fusionar várias estimativas de peritos (antigos programadores que trabalharam em projetos semelhantes). É usada quando a equipa tem pouca experiência. Assim tem um custo financeiro associado a ter em conta (pagamento dos peritos). Pode ser uma abordagem muito necessária em alguns casos.

**7.2.1.4 Estimativa de três pontos** A estimativa de três pontos é uma técnica para estimar o esforço necessário para concluir uma tarefa. Envolve a geração de três estimativas para cada tarefa:

- um cenário de melhor caso,
- um cenário mais provável e
- um cenário de pior caso.

O melhor cenário representa a quantidade mínima de esforço que seria necessária se tudo corresse perfeitamente de acordo com o planeado. O pior cenário representa a quantidade máxima de esforço que seria necessária se tudo desse errado. O cenário mais provável representa a estimativa mais realista do esforço necessário, com base em todas as informações disponíveis.

- **Tempo mais provável (T<sub>m</sub>)**: a estimativa amigável do tempo que seria necessário para terminar a tarefa. Esta estimativa parte do pressuposto que o projeto ou a tarefa irão encontrar problemas normais. Qualquer um dos métodos de avaliação (baixo para cima, a analogia, ou especialistas) podem ser usados para gerar esta estimativa.
- **Tempo otimista (T<sub>o</sub>)**: É a melhor estimativa de cenário. Ele assume que tudo vai dar certo durante o projeto ou a tarefa e que a equipa de desenvolvimento irá trabalhar de forma eficiente. É, portanto, a menor quantidade de tempo na qual a tarefa ou o projeto poderia ser concluído.

- **Tempo pessimista (Tp):** É a pior das hipóteses. Assume-se que tudo ou quase tudo o que poderia correr mal ao longo do projeto ou da tarefa irá acontecer. É, portanto, a quantidade máxima de tempo na qual a tarefa ou o projeto poderia ser concluído.

Para calcular uma estimativa final, normalmente é usada uma média ponderada dessas três estimativas. Isso envolve atribuir pesos diferentes a cada uma das estimativas com base em sua probabilidade de ocorrência. Por exemplo, o cenário mais provável pode receber um peso de 0,6, o melhor cenário um peso de 0,2 e o pior cenário um peso de 0,2. A estimativa final é então calculada multiplicando cada estimativa por seu peso e somando os resultados.

$$\text{Tempo expectável (Te): } Te = \frac{(To + 4 Tm + Tp)}{6}$$

Esta fórmula coloca ênfase no tempo mais provável que é multiplicado por 4, dando-lhe mais peso na fórmula. Depois, é calculada a média com o Tempo Optimista e o Tempo Pessimista. O tempo esperado é geralmente representado como **um intervalo**. Para criar um intervalo, precisamos de algumas informações sobre a probabilidade de que a estimativa esteja correta. Isto é feito calculando os desvios do Tempo Esperado.

$$\text{Desvio padrão: } \sigma = \frac{(Tp - To)}{6}$$

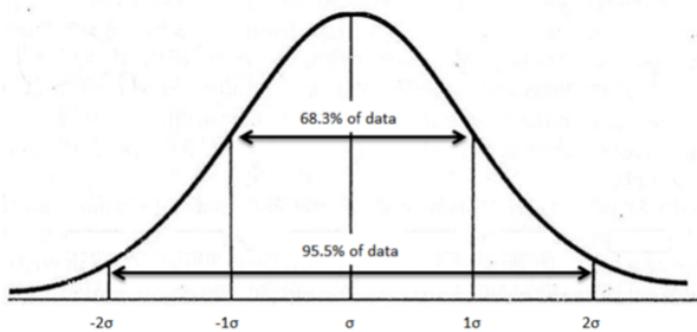


Figura 6: Triangulo qualidade - custo - prazo

Os desvios são representados pela letra grega sigma ( $\sigma$ ). O desvio padrão representa a forma como as medições podem ser distribuídas a partir do valor esperado. Estes são baseados em parcelas de distribuição normal com base em percentagens. Uma distribuição normal pressupõe que os valores estão repartidos de acordo com o desvio padrão (neste caso, o Tempo Esperado) e que esses valores dentro de uma unidade de desvio ( $\sigma$ ) do desvio padrão têm um grau de precisão provável de 68,3%. Se considerados duas unidades de desvio ( $(2\sigma)$ ), há uma probabilidade de 95,5% do valor estar certo.

**7.2.1.4.1 Exemplo** Considere um exemplo em que, para uma determinada tarefa, o Tempo mais provável, o Tempo otimista e o Tempo pessimista foram estimados da seguinte forma:

- $To = 5$  dias
- $Tm = 8$  dias
- $Tp = 17$  dias

$$Te = \frac{(5 + 4*8 + 17)}{6} = \frac{54}{6} = 9$$

O tempo expectável Te para esta tarefa é de 9 dias. Vamos olhar agora para o desvio padrão:

$$\sigma = \frac{(17 - 5)}{6} = 2$$

Desta forma, a estimativa do bloco de trabalho será de 7 a 11 dias, com um grau de certeza de 68,3%.

Se o resultado da estimativa for um valor real, convém arredondar o valor calculado (ex: 6,1 para 6 dias)

Para o cálculo da estimativa, é por vezes usada uma outra fórmula:  $Te = \frac{(To + Tm + Tp)}{3}$

### Estimativa de três pontos



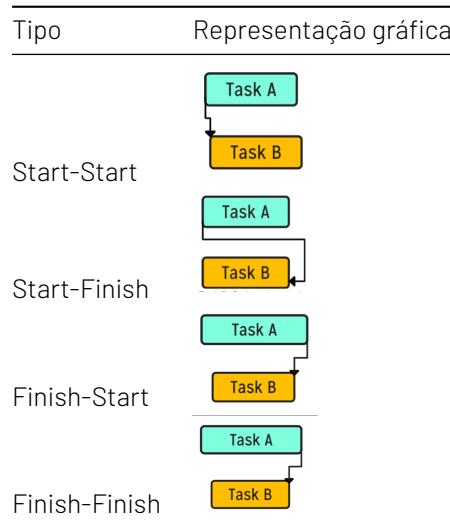
- Conhecida como a formula PERT de estimativa de 3 pontos
- Estimativa do tempo de cada tarefa individual
- Técnica **mais rápida** se a informação está disponível
- Pressupõe que nenhuma das tarefas na estrutura de divisão de trabalho (WBS) pode ser executada simultaneamente, mas sim que cada tarefa deve ser concluída antes que outra possa ser iniciada.

A estimativa de três pontos é uma técnica útil para lidar com a incerteza e o risco na estimativa do projeto, pois permite que uma variedade de resultados potenciais seja considerada. É freqüentemente usado em conjunto com outras técnicas de estimativa para melhorar a precisão das estimativas.

De forma geral, pode valer a pena estimar tarefas em horas num projeto de software, desde que as estimativas sejam baseadas numa sólida compreensão do trabalho que precisa de ser feito e dos riscos e incertezas envolvidos. No entanto, também é importante estar preparado para ajustar as estimativas conforme necessário e ficar de olho no progresso real do projeto para garantir que ele permaneça no caminho certo.

## 7.3 Dependências entre tarefas

Em qualquer tipo de projeto, existe uma dependência entre tarefas que rege o relacionamento entre as mesmas. Por exemplo, a conclusão de uma tarefa depende da conclusão de uma ou mais outras tarefas. Em outras palavras, certas tarefas não podem ser iniciadas ou concluídas até que outras tarefas tenham sido concluídas primeiro. Isso pode incluir tarefas como escrever código, testar e instalar uma aplicação. Compreender as dependências entre as tarefas é importante no desenvolvimento de software, pois ajuda a planear e gerir o cronograma geral do projeto e garantir que os recursos sejam alocados da maneira mais eficiente.



### 7.3.1 Dependência entre tarefas do tipo start-start

Este tipo de dependência aplica-se as dependências onde a primeira tarefa deve ser iniciada antes que a segunda tarefa possa ser iniciada. A altura em que estas tarefas terminam neste tipo de dependência não é im-

portante. As dependências **start-start** são representadas visualmente da forma acima descrita, com uma seta que vai desde o início da Tarefa A até o início da Tarefa B. As duas tarefas são mostradas mais ou menos em paralelo. A tarefa B depende do início da Tarefa A. Este método de representação pode ser usado em qualquer tipo de representações visuais, incluindo gráficos de Gantt e PERT. Um exemplo de uma dependência start-start pode ser: estar a trabalhar num projeto e criar um glossário. O projeto deve ser iniciado antes do início do glossário. É possível trabalhar nestas duas em simultâneo, no entanto, não importa qual é delas termina primeiro.

### 7.3.2 Dependência entre tarefas do tipo start-finish

Aplica-se às tarefas onde a primeira tarefa deve começar antes da segunda tarefa termine. Essas dependências geralmente envolvem algum tipo de data de entrega. Nas dependências de start-finish, o início da segunda tarefa não é importante. Em outras palavras, a primeira tarefa pode ser iniciada antes ou depois da segunda tarefa ser iniciada. Uma seta que vai desde o início da Tarefa **A** até o final da Tarefa **B**. Ambas as tarefas são mostradas mais ou menos em paralelo. A tarefa **A** deve ser iniciada antes da tarefa **B** terminar. Este método de representação pode ser usado em qualquer número de representações visuais, incluindo gráficos de Gantt e PERT. Um exemplo de uma dependência de início-fim poderia ser: - estar a planejar o próximo sprint durante o sprint atual enquanto este está em curso. Isso permite que a equipa de desenvolvimento comece a trabalhar imediatamente após iniciar o próximo sprint. O planeamento(Tarefa B)deve ser iniciado antes que o sprint atual (Tarefa A) termine, portanto, é dependente do sprint atual do tipo start-finish.

### 7.3.3 Dependência entre tarefas do tipo finish-start

Aplica-se às tarefas onde a primeira tarefa deve terminar antes que a segunda comece. É o tipo mais comum de dependência. A tarefa **B** não se está sobreposto com a tarefa **A**. A deve terminar antes de B começar. Pode ser aplicada a qualquer tipo de representação visual de tarefas, incluindo os gráficos de Gantt e PERT. Um exemplo de uma dependência finish-start pode ser a necessidade de concluir uma funcionalidade e saber o que esta irá fazer antes de elaborar um manual de utilizador ou um tutorial para um produto.

### 7.3.4 Dependência entre tarefas do tipo finish-finish

Acontece quando a primeira tarefa deve terminar antes da segunda ser concluída. As duas tarefa avançam em paralelo. As setas mostram que **A** deve terminar antes de **B** ser concluída. Pode ser usada em qualquer tipo de representação visual de tarefas, incluindo os gráficos de Gantt e PERT. Um exemplo de dependência finish-finish pode ser finalizar a faturação do cliente antes de entregar a totalidade do projeto para evitar falhas de pagamento.

## 7.4 Método do Caminho Crítico (CPM)

Um diagrama do Método do Caminho Crítico (CPM) é uma **representação visual das tarefas e dependências** envolvidas num projeto. É usado para analisar e planejar o cronograma do projeto. É um tipo de diagrama de rede que mostra a sequência de tarefas, a duração de cada tarefa e as dependências entre as tarefas.

O caminho crítico é a sequência de tarefas que determina a duração mínima do projeto, e qualquer atraso numa tarefa do caminho crítico atrasará a conclusão de todo o projeto. O diagrama CPM permite que os gestores de projeto identifiquem o caminho crítico e analisem o impacto de possíveis atrasos ou alterações no cronograma.

É uma ferramenta poderosa para ajudar os gestores de projeto e as equipas a entender o escopo do projeto, identificar possíveis atrasos e engarrafamentos e tomar decisões informadas sobre o cronograma e a alocação de recursos.

A construção desta rede corresponde ao método das tarefas potenciais ou método do caminho critico. Pode também ser chamado PDM (Precedence Diagramming Method). Cada atividade é representada por uma caixa. As atividades são conectadas por ligações de dependência representadas por setas. É construído da esquerda para a direita para representar a cronologia do projeto.



### Vantagem

forma uma rede de antecedentes que permite visualizar de forma clara a lógica das dependências.  
Dá a possibilidade de relações com prazos ( intervalo / sobreposição).

É uma representação existente em várias ferramentas de gestão de projetos: PMW (Project Management Work-bench) e Microsoft Project, etcs.



### Método

1. Fazer uma lista das tarefas necessárias para concluir o projeto / produto de trabalho e criar a seguir uma Estrutura de Composição do Trabalho (WBS) para essas tarefas.
2. Adicionar estimativas de tempo para cada tarefa.
3. Organizar as tarefas agrupando as dependências horizontalmente. Por exemplo, uma tarefa que é concluir-iniciar dependente de outra deve ser agrupada ao lado da tarefa que é dependente. As setas podem então ser usadas dependendo da tarefa para descrever a dependência.
4. A partir destes grupos de tarefas, é possível criar caminhos. Um caminho é uma sequência coletadas por setas que pode levar de uma tarefa para outra (com as dependências ilustradas).

**7.4.0.1 Exemplo** Considere um exemplo simples em que se pretende realizar o jantar. Esta refeição é constituída por um prato de esparguete e almôndegas e uma sala César. Pretendemos saber qual será a duração total de preparação. Para isso, é necessário identificar todas as tarefas, estimar a sua duração e definir as dependências entre elas. Vamos olhar primeiro para a estrutura analítica do trabalho e realizar uma decomposição dos três pratos.

O objetivo a seguir será identificar os caminhos que vão desde o início até o final do gráfico. Estes são chamados caminhos beginning-to-end (início até o fim). As tarefas em diferentes caminhos de início até o fim podem ser concluídas de forma independente e, portanto, ao mesmo tempo. Se dois ou mais caminhos de início até o fim encontram-se, este é um ponto de coordenação para dependências de tarefas - significa que as duas ou mais tarefas devem ser concluídas antes que a tarefa para a qual apontam possa começar. Em outras palavras, as tarefas devem ser coordenadas.

Um gráfico de CPM pode ter tantos caminhos de início-até-fim e pontos de coordenação quanto necessário para concluir o projeto.

Existem 4 caminhos, com 3 pontos de coordenação:

- "Boil the Water" -> "Add Meatballs to Dish"
- "Heat the Pan" -> "Add Meatballs to Dish"
- "Prepare Ingredients" -> "Add Meatballs to Dish"
- "Pre-heat Oven" -> "Add Meatballs to Dish"

No nosso exemplo, existem 3 caminhos beginning-to-end:

- "Add Croutons"

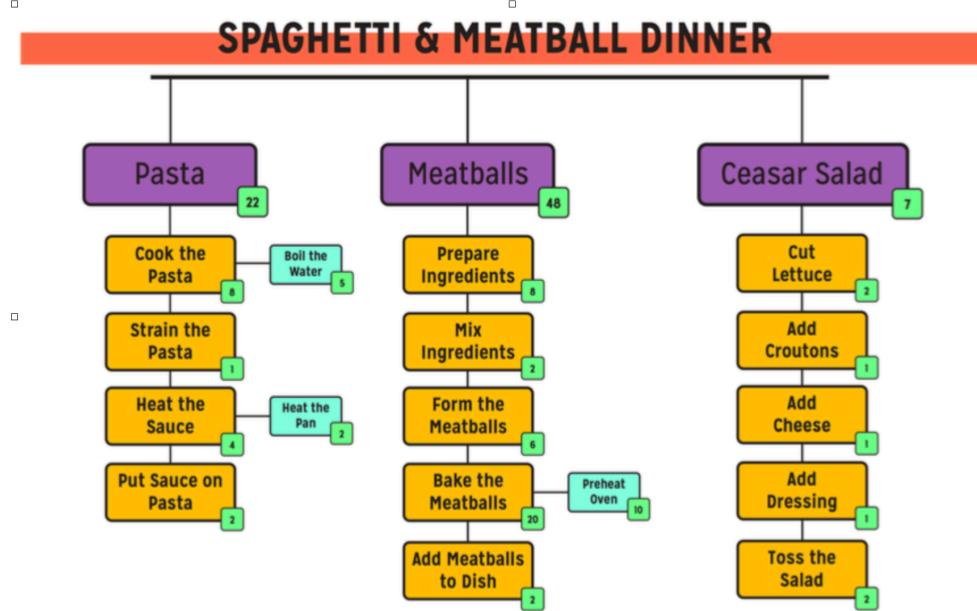


Figura 7: Estrutura analítica do trabalho para o exemplo do jantar

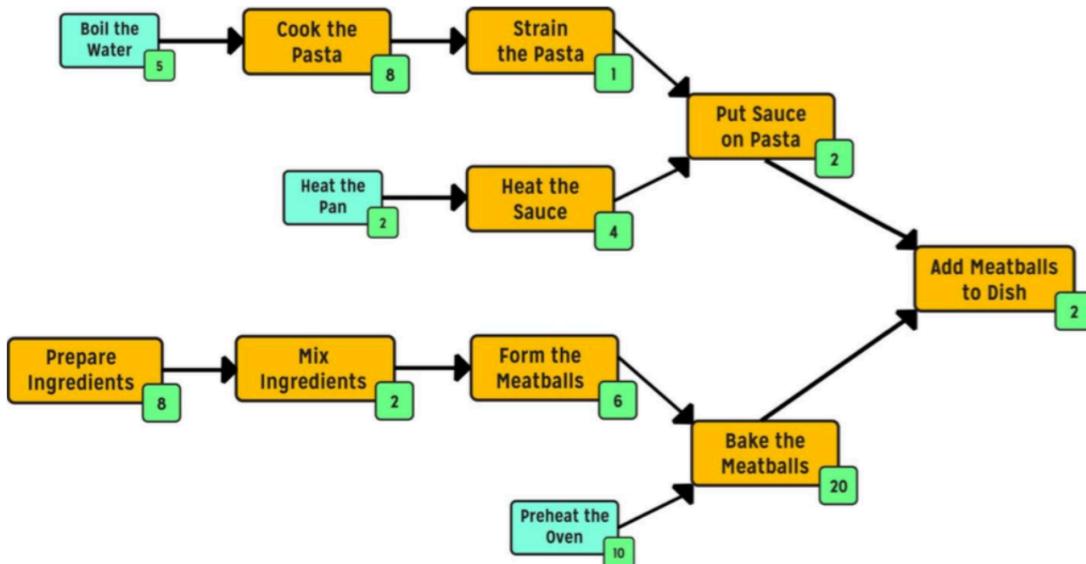


Figura 8: Triangulo qualidate - custo - prazo

- “Add Cheese”
- “Add Dressing”

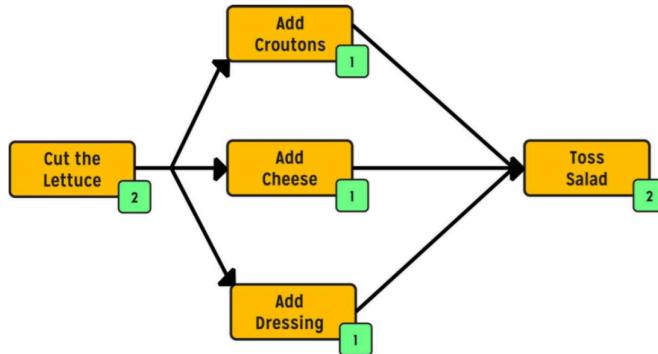


Figura 9: Tarefas paralelas num CPM

Combinação de todos os caminhos beginning-to-end para criar um CPM completo.

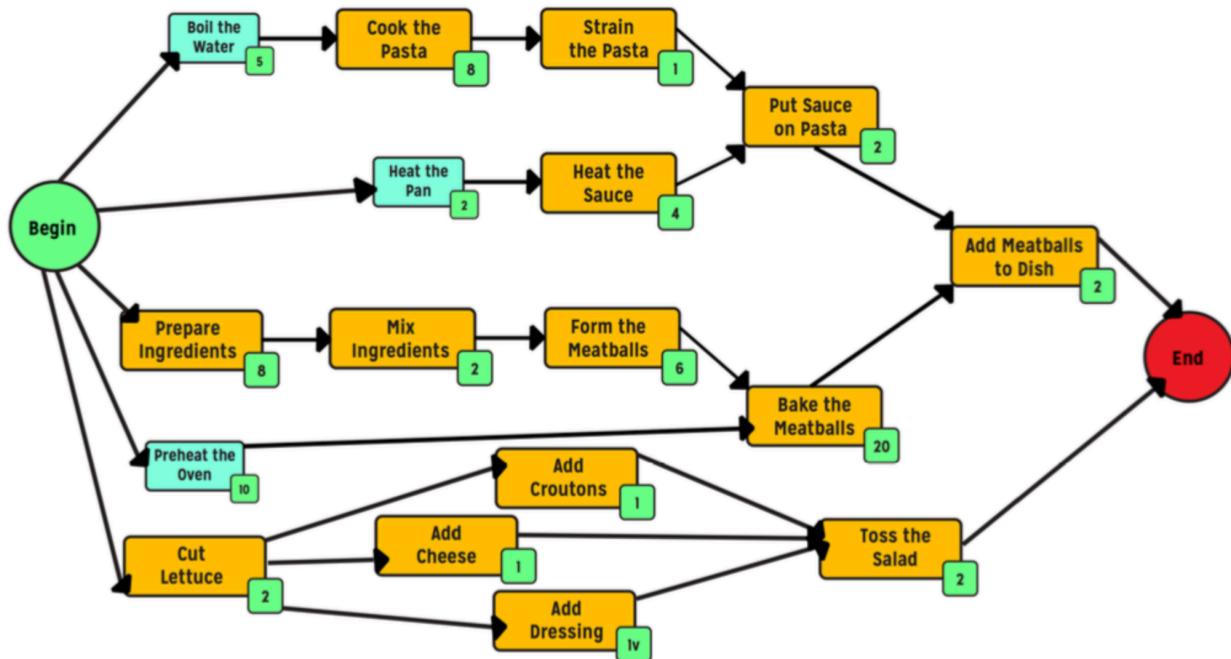


Figura 10: CPM completo.

Os gráficos de CPM são uma ferramenta útil para gerar um cronograma do projeto e determinar estimativas de tempo mínimo. Isto é feito com base em caminhos críticos. Caminho crítico: é o caminho de maior duração de tarefas entre pontos lógicos. Em outras palavras, é o **caminho beginning-to-end com a maior estimativa de tempo**.

Representa a quantidade de tempo necessária para completar nível de projeto, assumindo que há recursos suficientes para que outros caminhos beginning-to-end possam ser executados simultaneamente.

Os caminhos críticos fornecem informações sobre a **maior** e a **menor estimativa de tempo** para as tarefas do projeto. Evidenciam as **tarefas críticas** para o projeto. Evidenciam também as tarefas que têm **folga**. São aquelas que não estão no caminho crítico.

As estimativas de tempo geradas a partir de caminhos críticos em gráficos CPM podem diferir das estimativas da Estrutura de Repartição do Trabalho (WBS). Num gráfico CPM, **o caminho crítico determina a estimativa mínima e assume-se (onde os recursos estão disponíveis) que outros caminhos serão executados em simultâneo**. Numa estimativa WBS, no entanto, cada tarefa é adicionada como se for feita uma após a outra.

**7.4.0.2 Tarefas críticas** As tarefas críticas no caminho crítico devem ser as **primeiras a serem agendadas** a partir de um gráfico CPM. Da facto, as tarefas de outros caminhos podem ter tempo extra para serem concluídas (dentro do limite de tempo do caminho crítico), sem adicionar ao projeto geral.

A **folga** é útil para os projetos, pois fornece um certo grau de flexibilidade para certas tarefas, sem afetar necessariamente a data de conclusão desejada.

## 7.5 Gráfico PERT

O gráfico PERT - Program Evaluation and Review Technique - é outra ferramenta de visualização de projeto.

Não será analizada no âmbito deste documento, mas ficam desde já a saber que permite desenhar o caminho crítico, tal como CPM.

## 7.6 Diagrama de Gantt

Um gráfico de Gantt, também conhecido como diagrama de Gantt, é um tipo de gráfico de barras usado de forma comum na gestão de projetos para representar o cronograma de tarefas e atividades num projeto. É nomeado após Henry Gantt, um engenheiro mecânico americano e consultor de gestão que desenvolveu o conceito no início do século 20.

Um gráfico de Gantt geralmente consiste num eixo horizontal que representa o tempo e um eixo vertical que representa as diferentes tarefas do projeto. Cada tarefa é representada por uma barra horizontal, com o comprimento da barra indicando a duração da tarefa. As datas de início e término da tarefa são representadas pelas bordas esquerda e direita da barra, respectivamente.

O gráfico de Gantt também permite mostrar dependências entre tarefas, isso pode ser feito conectando uma barra de uma tarefa a outra barra de uma tarefa da qual ela depende, isso é útil para identificar qual tarefa depende de outra e quando uma tarefa pode iniciar com base após a conclusão da tarefa anterior.

Os gráficos de Gantt são usados para planejar, organizar e acompanhar o progresso de um projeto e são usados em projetos de construção, fabrico e desenvolvimento de software. Eles também podem ser usados para identificar possíveis conflitos ou atrasos no cronograma do projeto e para fazer ajustes no plano do projeto conforme necessário.

## 8 Gerir um projeto com uma metodologia em cascata

Vamos, portanto, colocar-nos na pele de um gestor de projeto contratado por esta agência para responder a um concurso muito simples para um hotel de luxo, o Hotel Sado.

É, portanto, um projeto externo que vamos gerir. Dito isso, a metodologia que vai ser apresentada aplica-se em muitos projetos internos.

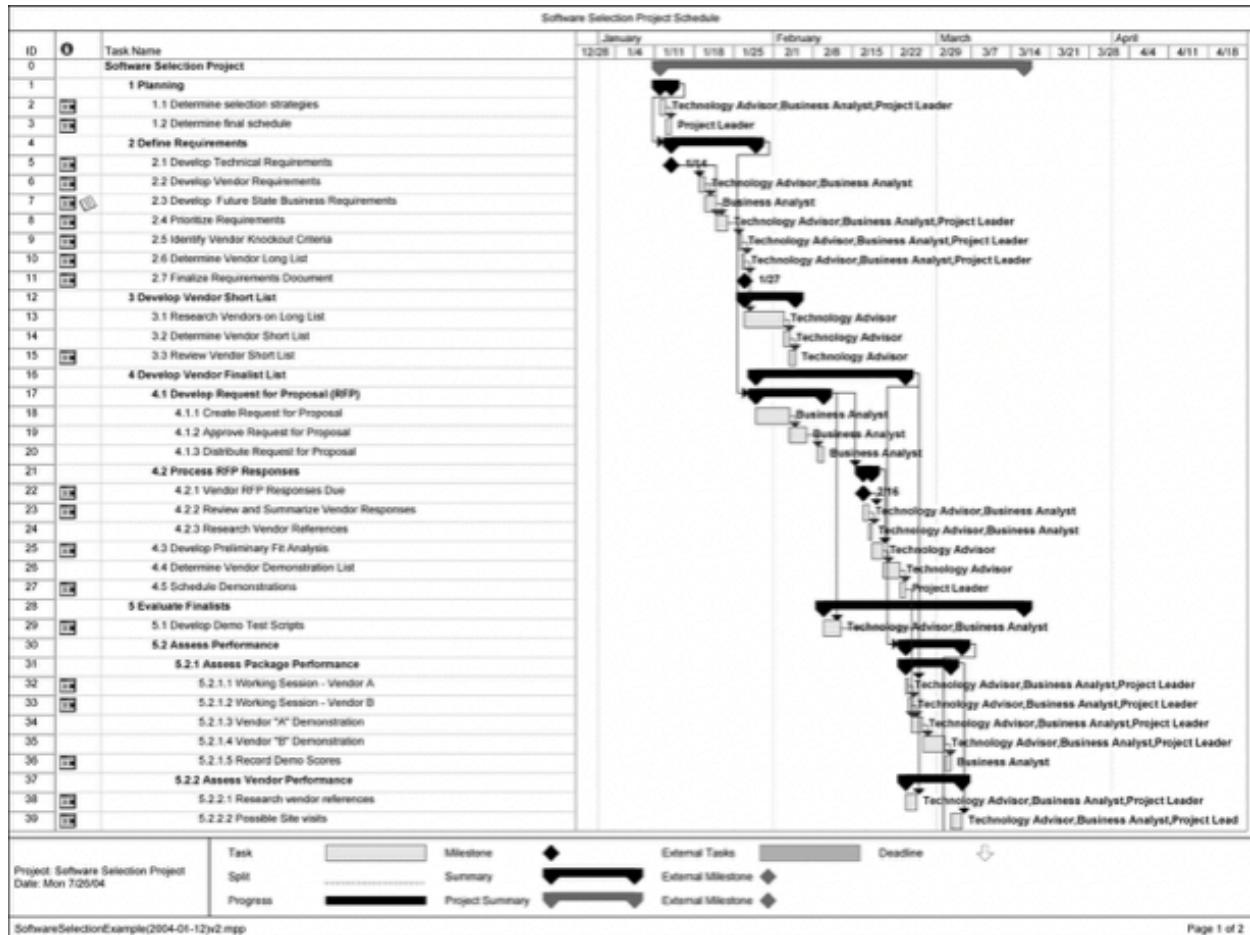


Figura 11: Exemplo de um diagrama de Gantt completo.

Software Selection Project Schedule							
ID	Task Name	Work	Duration	Start	Finish	Predecessors	Resource Names
0	Software Selection Project	1,112 hrs	46.5 days	Mon 1/12/04	Tue 3/16/04		
1	1 Planning	32 hrs	2 days	Mon 1/12/04	Tue 1/13/04		
2	1.1 Determine selection strategies	24 hrs	8 hrs	Mon 1/12/04	Mon 1/12/04		Technology Advisor,Business Analyst,Project Leader
3	1.2 Determine final schedule	8 hrs	8 hrs	Tue 1/13/04	Tue 1/13/04	2	Project Leader
4	2 Define Requirements	144 hrs	11 days	Wed 1/14/04	Wed 1/28/04	1	
5	2.1 Develop Technical Requirements	24 hrs	24 hrs	Wed 1/14/04	Fri 1/16/04		Technology Advisor
6	2.2 Develop Vendor Requirements	10 hrs	8 hrs	Mon 1/19/04	Mon 1/19/04	5	Technology Advisor,Business Analyst
7	2.3 Develop Future State Business Requirements	10 hrs	10 hrs	Tue 1/20/04	Wed 1/21/04	6	Business Analyst
8	2.4 Prioritize Requirements	40 hrs	10 hrs	Thu 1/22/04	Fri 1/23/04	5,6,7	Technology Advisor,Business Analyst,Project Leader
9	2.5 Identify Vendor Knockout Criteria	12 hrs	4 hrs	Mon 1/26/04	Mon 1/26/04	8	Technology Advisor,Business Analyst,Project Leader
10	2.6 Determine Vendor Long List	12 hrs	4 hrs	Mon 1/26/04	Mon 1/26/04	9	Technology Advisor,Business Analyst,Project Leader
11	2.7 Finalize Requirements Document	16 hrs	16 hrs	Tue 1/27/04	Wed 1/28/04	10	Business Analyst
12	3 Develop Vendor Short List	66 hrs	7 days	Tue 1/27/04	Wed 2/3/04		
13	3.1 Research Vendors on Long List	40 hrs	5 days	Tue 1/27/04	Mon 2/2/04	10	Technology Advisor
14	3.2 Determine Vendor Short List	8 hrs	1 day	Tue 2/3/04	Tue 2/3/04	13	Technology Advisor
15	3.3 Review Vendor Short List	8 hrs	1 day	Wed 2/4/04	Wed 2/4/04	14	Technology Advisor
16	4 Develop Vendor Finalist List	120 hrs	16.5 days	Thu 1/29/04	Tue 2/24/04		
17	4.1 Develop Request for Proposal (RFP)	64 hrs	8 days	Thu 1/29/04	Mon 2/3/04	4	
18	4.1.1 Create Request for Proposal	32 hrs	32 hrs	Thu 1/29/04	Tue 2/3/04		Business Analyst
19	4.1.2 Approve Request for Proposal	24 hrs	24 hrs	Wed 2/4/04	Fri 2/6/04	16	Business Analyst
20	4.1.3 Distribute Request for Proposal	8 hrs	8 hrs	Mon 2/6/04	Mon 2/6/04	16	Business Analyst
21	4.2 Process RFP Responses	24 hrs	1.5 days	Mon 2/6/04	Wed 2/8/04	17F+5 days	
22	4.2.1 Vendor RFP Response Due	0 hrs	0 days	Mon 2/6/04	Mon 2/6/04		
23	4.2.2 Review and Summarize Vendor Responses	16 hrs	8 hrs	Tue 2/7/04	Tue 2/7/04	22	Technology Advisor,Business Analyst
24	4.2.3 Research Vendor References	8 hrs	4 hrs	Wed 2/8/04	Wed 2/8/04	22	Technology Advisor,Business Analyst
25	4.3 Develop Preliminary Fit Analysis	16 hrs	16 hrs	Wed 2/8/04	Fri 2/20/04	21	Technology Advisor
26	4.4 Determine Vendor Demonstration List	8 hrs	8 hrs	Fri 2/20/04	Mon 2/23/04	25	Technology Advisor
27	4.5 Schedule Demonstrations	8 hrs	8 hrs	Mon 2/23/04	Tue 2/24/04	26	Project Leader
28	5 Evaluate Finalists	460 hrs	25.5 days	Tue 2/16/04	Tue 3/16/04		
29	5.1 Develop Demo Test Scripts	48 hrs	24 hrs	Tue 2/16/04	Thu 2/12/04	17	Technology Advisor,Business Analyst
30	5.2 Assess Performance	224 hrs	8 days	Tue 2/16/04	Fri 2/19/04	16	
31	5.2.1 Assess Package Performance	128 hrs	8 days	Tue 2/16/04	Wed 3/3/04		
32	5.2.1.1 Working Session - Vendor A	12 hrs	4 hrs	Tue 2/16/04	Tue 2/20/04	16	Technology Advisor,Business Analyst,Project Leader
33	5.2.1.2 Working Session - Vendor B	12 hrs	4 hrs	Wed 2/25/04	Wed 2/25/04	32	Technology Advisor,Business Analyst,Project Leader
34	5.2.1.3 Vendor 'A' Demonstrations	40 hrs	10 hrs	Wed 2/25/04	Fri 2/27/04	32,29,27,33	Technology Advisor,Business Analyst,Project Leader
35	5.2.1.4 Vendor 'B' Demonstrations	40 hrs	10 hrs	Fri 2/27/04	Tue 3/2/04	34	Technology Advisor,Business Analyst,Project Leader
36	5.2.1.5 Record Demo Scores	8 hrs	8 hrs	Tue 3/2/04	Wed 3/3/04	35	Business Analyst
37	5.2.2 Assess Vendor Performance	96 hrs	8 days	Tue 2/24/04	Fri 3/6/04		
38	5.2.2.1 Research vendor references	48 hrs	16 hrs	Tue 2/24/04	Thu 2/26/04	16	Technology Advisor,Business Analyst,Project Leader
39	5.2.2.2 Possible Site visits	48 hrs	16 hrs	Wed 3/3/04	Fri 3/5/04	21	Technology Advisor,Business Analyst,Project Leader
40	5.3 Develop Final Fit Analysis	80 hrs	8 days	Fri 2/6/04	Fri 3/13/04	30	
41	5.3.1 Perform Fit-Gap analysis for each option	16 hrs	8 hrs	Fri 2/6/04	Mon 2/8/04	41	Technology Advisor,Business Analyst
42	5.3.2 Assess risk for each option	16 hrs	8 hrs	Mon 2/8/04	Tue 3/9/04	41	Technology Advisor,Business Analyst
43	5.3.3 Develop high-level schedule and budget forecast	16 hrs	8 hrs	Tue 3/9/04	Wed 3/10/04	42	Technology Advisor,Business Analyst

SoftwareSelectionExample(2004-01-12).xspf

Page 1 of 2

Figura 12: Exemplo de um diagrama de Gantt sob a forma de uma tabela

## 8.1 Recolher e Analisar as necessidades

Qualquer projeto começa com uma expressão ou identificação de uma necessidade e esta pode ser mais ou menos completa. O primeiro trabalho do gestor de projeto é, portanto, analisar esta ou essas necessidades, completá-las se necessário e reformulá-las na forma de objetivos e entregas.



### Definição: Entregável

O termo "entrega" ou "entregável" é usado para designar o resultado de um serviço ou uma concretização. Uma entrega é, portanto, literalmente o que entregaremos ao cliente, mesmo que não seja um produto físico.

No caso do projeto do Hotel Sado, à primeira vista, as necessidades parecem relativamente bem identificadas embora pouco detalhadas, pelo que podemos facilmente traduzi-las em entregáveis:

Precisa	Potencial(is) produto(s)
Desenvolvendo uma presença online	Site de demonstração, plano de marketing na web
Modernizar uma imagem de marca	Plataforma de marca com norma gráfica e logotipo
Permitir que os clientes reservem online	sistema de reserva

A priori, temos aqui os entregáveis mais óbvios. Resta saber se todas correspondem às reais expectativas do patrocinador e se este não esqueceu ou expressou mal algumas necessidades. Para ter a certeza e apresentar uma proposta comercial a medida do patrocinador, é necessário comunicar com ele para extrair mais informações. Para isso, cabe ao gestor encontrar a forma mais adequada de contatá-lo, por telefone, videoconferência, LinkedIn ou e-mail.

Durante estes contatos exploratórios, deverá prestará uma atenção especial a 4 coisas:

- as necessidades explícitas do cliente
- as necessidades implícitas do cliente
- as entregas potenciais e
- a sua suposta grelha de leitura

### 8.1.1 Necessidades explícitas

Por "necessidades explícitas", entenda-se as necessidades expressas de forma clara, sem ambiguidades e sobre as quais existe consenso. Para saber se é esse o caso, tente reformulá-las e analise a reação do seu interlocutor.

No nosso exemplo, a necessidade de desenvolver a presença online do Hotel Sado através da criação de um website é uma necessidade bastante explícita.

### 8.1.2 Necessidades implícitas

Se verificamos, por exemplo, que a clientela do hotel é maioritariamente estrangeira, existirá implicitamente a necessidade de ter um site multilíngua. Essa necessidade, por outro lado, não foi claramente expressa pelo patrocinador. Diremos portanto que é uma necessidade implícita e um constrangimento que deve ser tido em conta desde o início do projeto. Nesse caso, essa restrição foi descoberta enquanto o cliente comentava os problemas com seu sistema de reservas. Daí o interesse de fazer perguntas bem abertas durante uma recolha de necessidades.

Necessidades explícitas	Necessidades implícitas	Potencial(is) produto(s)
Desenvolver presença online	Torne o site acessível em 4 idiomas: francês, inglês, chinês e russo.	Site (multilingue)
Modernizar a imagem da marca	Manter o nome e logotipo do estabelecimento.	Adaptação da norma gráfica
Sistema de reservas	Permita que a equipa do hotel gere os seus preços e reservas no back office.	Sistema de reservas (e gestão)

Por vezes, algumas necessidades do patrocinador podem parecer contraditórias, como aqui, a vontade de modernizar a imagem da marca mantendo o logótipo do estabelecimento. Cabe ao gestor julgar se é melhor confrontar o patrocinador com as suas contradições ou chegar a um compromisso de forma diplomática.

### 8.1.3 Descodificar uma grelha de leitura

Enquanto um gestor comunica com um potencial cliente, ele aproveite para fazer o seguinte exercício: Descodificar a **grelha de leitura** do cliente!

A grelha de leitura do patrocinador é o conjunto de critérios a partir do qual ele selecionará a agência candidata. Alguns dão mais importância ao entendimento do problema encontrado ou ao orçamento proposto, enquanto outros privilegiam a estética dos entregáveis ou o relacionamento com o prestador de serviços.

Esteja ou não esta grelha formalizada para o seu patrocinador, é fundamental que o compreenda para poder antecipar as suas prioridades e assim elaborar a sua proposta comercial em conformidade.

Para fazer isso, basta listar os critérios de seleção que aparecem com mais frequência e atribuir-lhes uma prioridade com base na análise que se faz do patrocinador.

Aqui está um exemplo de uma suposta grade de leitura:

Critérios de seleção	Importância
Compreensão do problema	2
Estética	3
Solução Técnica	4
Metodologia	5
orçamento	1

Se se perguntar explicitamente ao cliente quais são os seus critérios, ele frequentemente dirá que todos são importantes. Deve-se procurar, portanto, detectar "sinais fracos", ou seja, informações parciais, fragmentárias e implícitas fornecidas pelo cliente.

## 8.2 Definir o âmbito do projeto

Na secção anterior, vimos como recolher e analisar as necessidades ou como descodificar uma grelha de leitura.

Nesta fase ainda não temos todas as informações para formular uma proposta comercial. Por outro lado, temos o suficiente para apresentar o projeto à equipa do projeto e pedir que eles forneçam informações adicionais.

Ou seja, no início de um projeto, na fase de início, vamos começar por recolher e analisar as informações do cliente, depois partilhar estas informações com a equipa para planear, orçamentar e sintetizar tudo num documento contratual. ■

Vamos chamar de "estudo de viabilidade" todas as etapas que antecedem a proposta comercial. Para o gestor de projeto, esta fase é uma espécie de busca de informações. Isso termina quando se sabe o suficiente sobre os prós e contras do projeto para informar a tomada de decisão e avaliar o interesse.

### **8.2.1 Consultar os especialistas para definir a solução**

A reunião de definição do escopo é o momento perfeito para envolver e alinhar com os colaboradores. O objetivo do gestor de projeto será traduzir da forma mais clara possível as necessidades do cliente para que os especialistas de cada área possam identificar riscos e soluções.

Uma agenda típica (esboço de reunião) para uma reunião de definição de escopo seria:

- Restituição de necessidades
- Desenvolvendo uma solução
- Estimativa de carga de trabalho

Assim, teoricamente, no final desta reunião deverá ser recolhida todas as informações necessárias para planejar o projeto (assunto da próxima secção).

### **8.2.2 Escolher uma metodologia de gestão de projeto**

A reunião de definição do escopo também é o momento ideal para escolher e anunciar a metodologia de gestão de projeto que será utilizada. Assim como um programador escolhe a tecnologia mais adequada para a realização de um produto, o gestor de projeto terá que escolher a metodologia mais adequada para a gestão do projeto.

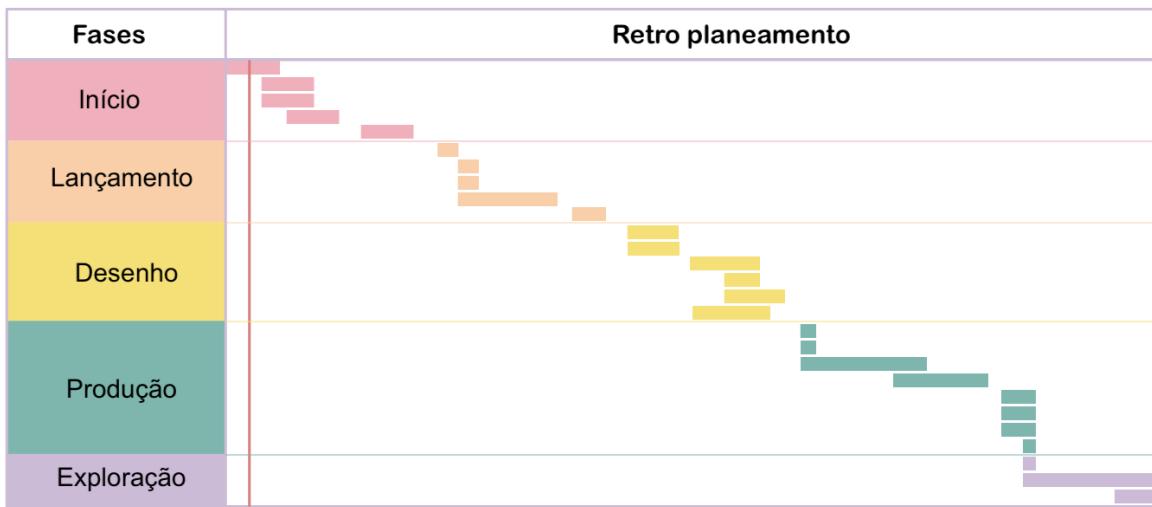
Existem muitas metodologias de gestão de projetos. A Unidade curricular de Engenharia de Software do curso da Licenciatura em Engenharia Informática apresenta em detalhe algumas destas metodologias. Nesta unidade curricular, vamos apenas distingui-las e escolher uma delas entre as duas famílias principais: metodologias clássicas e metodologias ágeis.

## **8.3 As metodologias clássicas**

As metodologias **clássicas** de gestão de projetos, também chamadas de métodos **preditivos** e **sequenciais**, como o método **cascata** ou o **ciclo V**, envolvem um processo em etapas, em sequências, em fases. Essas fases correspondem a conjuntos de tarefas predefinidas num cronograma de implementação preciso.

Um projeto em cascata, por exemplo, geralmente ocorre em 5 fases:

- A fase de **início**: que antecede a proposta comercial e que termina com a aceitação do orçamento.
- A fase de **lançamento**: na qual se reespecifica e se formaliza todos os elementos do projeto numa “especificação”. Esta fase termina com a assinatura de um documento.
- A fase de **design**: na qual se faz o acompanhamento da equipa na criação dos elementos gráficos e textuais a partir dos quais o site será desenvolvido, produzido.
- A fase de **produção**: em que sua equipe criará o produto final a partir dos elementos feitos na fase de design. Esta fase termina com a entrega do produto.
- A fase **operacional**: na qual o produto é entregue de acordo com as expectativas e é suficiente mantê-lo ou desenvolvê-lo.



Como todas as metodologias, a cascata tem as suas vantagens e desvantagens:

#### Vantagens | Desvantagens |

precisão de prazos | clareza e visualização do estado de andamento | orçamentação mais fácil | detecção tardia de riscos | pouca margem de erro |

### 8.4 As metodologias ágeis

Ao contrário dos métodos tradicionais, as metodologias ágeis pretendem ser **iterativas, incrementais** e **adaptativas**. Ou seja, ao invés de seguir rigidamente as etapas uma após a outra, o desenvolvimento do produto é feito em camadas, cada uma trazendo novos desenvolvimentos. Assim, o produto pode ser testado e melhorado continuamente (pouco a pouco) ao longo do projeto e não no final.

#### Vantagens | Desvantagens |

melhor capacidade de resposta ao inesperado | detecção rápida de riscos | planeamento impreciso | orçamento difícil |

### 8.5 Metodologia clássica ou ágil, eis a questão.

Para saber se você deve optar pelo clássico ou pelo ágil, atente-se para a previsibilidade do seu projeto. Em outras palavras, pergunte-se o quanto previsível ou imprevisível é o seu projeto. O gestor pode consultar a equipa para ajudá-lo.

A lógica é a seguinte: quanto mais incógnitas o projeto tiver, mais agilidade será necessária para lidar com elas. Por outro lado, se for muito previsível, será do seu interesse apostar na precisão para se organizar e comunicar o progresso de forma mais eficaz.

No caso do Hotel Sado, como o programador identifica poucas ou nenhuma incógnita, o risco tecnológico é quase nulo e, portanto, uma metodologia clássica parece ser apropriada.

Se o risco tecnológico é insignificante, o próprio projeto também não é isento de riscos. Este risco tecnológico pode então ser substituído por um risco humano. Assim, um projeto considerado "muito fácil" ou "pouco estimulante" pode, por vezes, ser encarado de forma leviana e levar à falta de rigor.

Assim como uma equipa pode ser sob qualificada para um projeto, ela também pode ser sobre qualificada.

Assim, dentro da família de metodologias clássicas, o gestor ainda pode hesitar entre diferentes métodos primos.

Anteriormente, na definição dos métodos clássicos, abordamos o método em cascata. O ciclo V, por outro lado, leva mais ou menos as mesmas etapas às quais adiciona-se baterias de teste mais avançadas para limitar o regresso às etapas anteriores.

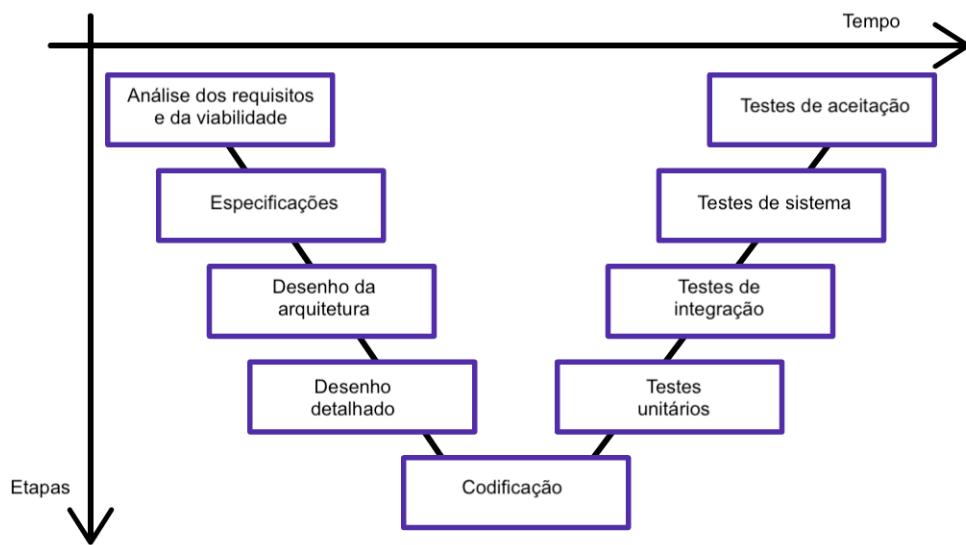


Figura 13: O ciclo em V

Também poderíamos ter optado por esta metodologia, mas dada a simplicidade do projeto, vamos usar uma metodologia em cascata. A especificidade da metodologia em cascata é a sua ferramenta de planeamento, o gráfico de Gantt.

Na próxima secção, vamos ver como criar um gráfico de Gantt e como descobrir quanto tempo será necessário à equipa para concluir o projeto.

## 8.6 Planear e definir o cronograma do projeto

Depois de receber a estimativa de carga de trabalho dos membros da equipa, será possível passar para o planeamento. É sobretudo nesta fase que se irá definir os requisitos que irão garantir o cumprimento do prazo no decorrer do projeto.

O planeamento das tarefas irá resultar num **agendamento** (Schedule em Inglês) que consiste no mapeamento de tarefas numa linha temporal.

Nesta secção vamos ver como planear um projeto com um gráfico de Gantt.

### 8.6.1 Criar um gráfico de Gantt

O gráfico de Gantt é um método clássico de planeamento e gestão que representa visualmente o andamento de um projeto.

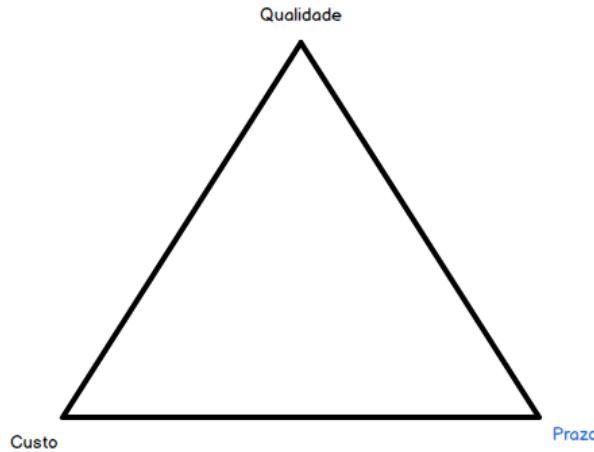


Figura 14: As exigências de qualidade, de custos e de prazos de um projeto



Figura 15: Um exemplo de diagrama de Gantt

**8.6.1.1 Escolher uma ferramenta** Existem muitas aplicações e softwares especializados. Cada um deles tem suas vantagens e desvantagens. No âmbito desta disciplina vamos olhar para as folhas de cálculo como o MS Excel.

De fato, em alguns casos, usar uma aplicação de gestão de projeto pode ser mais complexo do que usar um software especializado. Pelo menos se se começar do zero. Para economizar tempo, preparei [este modelo](#). Funciona como um verdadeiro software pequeno, só que este é simplificado ao máximo para obrigar a se concentrar nos elementos constitutivos de um gráfico de Gantt e não nas funcionalidades que estão ao seu redor. Aprenda a criar um gráfico de Gantt numa folha de cálculo não representa nenhum problema.

Vamos!

- Abra o modelo clicando [aqui](#).
- Crie sua cópia (conforme explicado abaixo).
- Complete para chegar a uma renderização final como esta: Gráfico de Gantt do hotel Sado.



Figura 16: Diagrama de gantt do projeto hotel Sado

**8.6.1.2 Listar as tarefas e sua duração** Como referido anteriormente, quando se quer gerir um projeto em cascata, assume-se que todas as tarefas que o compõem são previsíveis (e que, portanto, pode-se planeá-las).

A primeira tarefa será projetar-se através do progresso das 5 fases do projeto, detalhando as tarefas correspondentes a cada uma delas. Da primeira vez, este exercício pode ser difícil, mas rapidamente irá notar que, de um projeto para outro, certas tarefas são repetidas quase sistematicamente.

### Como identificar todas as tarefas do lado da programação ou do grafismo?

Basta perguntar às pessoas envolvidas.

Na Web, uma carga de trabalho geralmente é contada em dias-homem ou meio-dia. Em outras palavras, quantos dias são necessários para uma pessoa concluir a tarefa. No entanto, algumas tarefas levarão menos de meio dia para serem concluídas. Tente agrupá-los o máximo possível.

**Granularidade da tarefa:** se sua lista de tarefas for muito detalhada, perderá a legibilidade. Por outro lado, se não for detalhado o suficiente, o diagrama perde seu interesse. Cabe ao gestor encontrar um equilíbrio.

## 8.7 Atribuir tarefas aos membros da equipa

Agora que se tem a lista de tarefas e a sua duração estimada, pode-se atribuí-las aos colegas de equipa. Para cada tarefa, identificará-se, portanto, a pessoa mais capaz de realizá-la.

Pode parecer contra intuitivo, mas nem sempre essa pessoa será a mais qualificada, competente ou experiente. De fato, nem todos os funcionários produzem com o mesmo nível de qualidade, com o mesmo custo e nos mesmos prazos. As chances são, portanto, de que a pessoa mais competente e mais rápida seja também aquela que mais custa à sua organização.

Os membros de uma equipa podem estar envolvidos em vários projetos simultaneamente. Nesse caso, não estará disponível 100% do tempo. Antes de atribuir uma tarefa a alguém, certifique-se de que essa pessoa está disponível!

Tudo o que é preciso fazer é dar uma data de início e de fim para cada tarefa e cada fase. Para chegar a um cronograma realista, é preciso prestar muita atenção a duas coisas: dependências e sobreposições.

### 8.7.1 Dependências

Certas tarefas não podem ser executadas em boas condições até que outra tarefa tenha sido finalizada anteriormente. Isso é conhecido como a dependência de duas tarefas.

No exemplo abaixo, chamo a atenção para a dependência direta entre a recolha de requisitos e as três tarefas que seguem. Com efeito, não se pode fazer um estudo de viabilidade em boas condições se não tiver sido feita uma recolha prévia. Da mesma forma, uma proposta comercial não pode ser finalizada até que o estudo de viabilidade e o escopo sejam concluídos.

Para que um projeto seja executado de forma otimizada, procura-se minimizar o tempo de inatividade, ou seja, os períodos de inatividade durante os quais um ou mais membros da equipa ficam bloqueados devido a uma dependência.

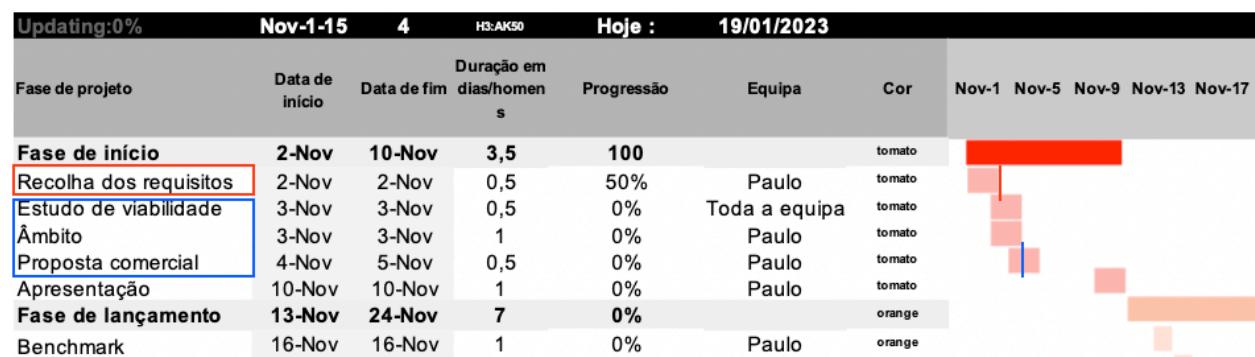


Figura 17: Diagrama de gantt do projeto hotel Sado

### 8.7.2 Sobreposições

Enquanto algumas tarefas são dependentes, outras, pelo contrário, são independentes. Essas tarefas podem, portanto, ser realizadas em paralelo por diferentes membros da equipa. No exemplo abaixo, chamo a atenção para três tarefas que podem ser executadas ao mesmo tempo.

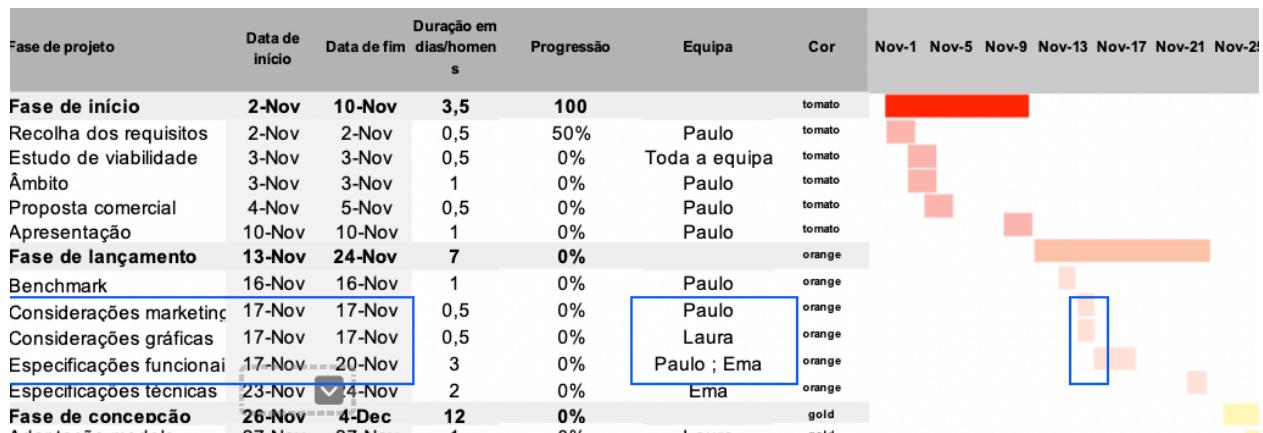


Figura 18: Diagrama de gantt do projeto hotel Sado

### 8.7.3 Atualização do progresso

Quando todas as tarefas do projeto têm uma estimativa de tempo, um responsável, uma data provisória de início e de fim, o gráfico de Gantt não será usado apenas para planeamento, mas também para gestão diária. Assim, poderá-se acompanhar o andamento do projeto, preenchendo o nível de andamento de cada tarefa.

Neste ponto, três cenários podem surgir:

- As estimativas foram muito otimistas e está-se atrasado.
- As estimativas foram perfeitamente calibradas e está-se dentro do timing.
- As estimativas foram muito pessimistas e está-se adiantado.

Quanto mais se fica para trás, mais a margem diminui. É até possivelmente perder-se dinheiro num projeto. Por outro lado, quanto mais adiantado, mais dinheiro economiza-se e, portanto, aumenta-se seu lucro.

### 8.7.4 Vantagens e limitações do gráfico de Gantt

#### Vantagens | Desvantagens |

simples de criar | preciso fácil de entender para todas as partes interessadas | requer atualizações regulares possíveis problemas de visualização em projetos complexos |

## 9 Planeamento e estimativa dos custos do projeto

A elaboração de um cronograma do projeto num gráfico de Gantt permite saber quais tarefas precisam ser realizadas, quando, por quem e em quanto tempo. Com esta informação poderá estabelecer o orçamento provisório do seu projeto.

### 9.1 Diferenciar itens de despesas

Para começar, saiba que o orçamento que vai propor ao seu cliente não será igual ao que vai calcular internamente. Neste capítulo, calcularemos o orçamento internamente, ou seja, o preço de custo do projeto do **lado da**

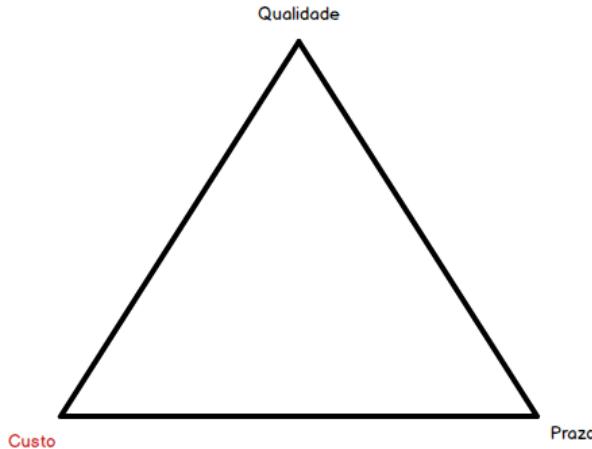


Figura 19: Atender aos requisitos de orçamento

**empresa que fornece o serviço.** É a partir dela que iremos elaborar a nossa proposta comercial e o orçamento do **lado do cliente**.

Num projeto simples como, pode-se simplesmente começar por distinguir entre diferentes itens de despesa, por exemplo:

- Recursos **materiais**
- Recursos de **informática** (licenças, software)
- Recursos **humanos** (salários)



#### Nota

Dependendo da natureza do seu projeto, alguns itens de despesa serão mais importantes do que outros.

Num projeto informático, antes mesmo de calcular com precisão o orçamento, pode-se esperar que os recursos humanos sejam o primeiro item de despesa.

## 9.2 Calcular o custo total do projeto

Calcular o orçamento provisório de um projeto equivale a calcular o custo de cada uma das 5 fases que o compõem.

$$\begin{aligned} \text{Custo total do projeto} = & \text{ custo da fase de inicialização} + \text{custo da fase de lançamento} \\ & + \text{custo da fase de projeto} + \text{custo da fase de produção} \\ & + \text{custo da fase operacional} \end{aligned}$$

Essas fases são compostas por tarefas; aqueles que foram listadas no gráfico de Gantt:

$$\text{Custo total de uma fase} = \text{custo da tarefa 1} + \text{custo da tarefa 2} + \text{custo da tarefa n...}$$

Se calcular o custo total de um projeto equivale a somar o custo das fases e, portanto, das tarefas, como calcular o custo de uma tarefa?

**Custo estimado de uma tarefa = duração estimada da tarefa \* custo das pessoas envolvidas**

Para dar um exemplo, vamos calcular o custo da tarefa "Elaboração da proposta comercial":

- Duração estimada: 0,5 dias
- Pessoa envolvida: Gestor de projeto (Paulo)
- Custo diário da pessoa envolvida: 130,80€



**Nota**

Em função do escopo da tarefa (meses, semanas, dias, horas), usa-se a escala salarial correspondente (mensal, semanal, diário, horário).

Custo anual	Custo mensal	Custo semanal	Custo diário	Custo horário	Outros
Paulo - chefe de projeto	€34 000,00	€2 833,00	€653,80	€130,80	€18,70
Ema - Programador Web	€36 000,00	€3 000,00	€692,30	€138,50	€19,80
Laura - Designer	€30 000,00	€2 500,00	€576,80	€115,40	€16,50
João - Designer estagiário	€7 200,00	€600,00	€140,00	€28,00	€4,00
Tradutor					
Fotógrafo				€450,00	

Exemplo de grelha de salário



**Nota**

Para a empresa, um funcionário custa mais do que o seu salário bruto. Para prever e gerir melhor os custos do seu projeto, peça ao seu contabilista ou gestor para lhe enviar o custo global/total de um funcionário.

**Custo de um funcionário = salário bruto + contribuições dos empregadores**

Agora se tem todas as informações para calcular o custo da sua tarefa:

**Custo da tarefa 'escrita de proposta comercial' = 0,5 \* 130,8 = € 65,4**

Agora basta fazer esse cálculo para todas as tarefas programadas.



**Custo dos recursos**

Quando tem-se recursos de hardware ou software, não se esqueça de incluí-los nos seus cálculos. Aqui a compra da licença completa do tema WordPress inflaciona o orçamento estimado de 2700€; não é desprezível.

Assim pode-se ver que algumas fases do projeto custam mais do que outras. Neste caso, no projeto do Hotel Sado, pode-se estimar que a fase de produção custará quase 9 vezes mais que a fase de inicialização!

Observe que na etapa do orçamento provisório, esses custos são apenas estimativas e não custos reais. À medida que o projeto avança, acompanhará-se a evolução desses custos e atualizará-se o orçamento em função dessa evolução. No final do projeto, fará-se uma comparação entre a previsão e o real.

Tarefas	Equipa	Custo diário	Duração estimada	Custo estimado
<b>Fase de iniciação</b>				<b>€584,75</b>
Recolha dos requisitos	Chefe de projeto	€130,80	0,50	€65,40
Estudo de viabilidade	Chefe de projeto	€130,80	0,50	€65,40
	Programador	€138,50	0,50	€69,25
	Designer	€115,40	0,50	€57,70
Âmbito	Chefe de projeto	€130,80	1,00	€130,80
Proposta comercial	Chefe de projeto	€130,80	0,50	€65,40
Defesa	Chefe de projeto	€130,80	1,00	€130,80
<b>Fase de lançamento</b>				<b>€931,00</b>
Benchmark	Chefe de projeto	€130,80	1,00	€130,80
Considerações marketing	Chefe de projeto	€130,80	0,50	€65,40
Considerações gráficas	Designer	€115,40	0,50	€57,70
Especificações funcionais	Chefe de projeto	€130,80	2,00	€261,60
	Programador	€138,50	1,00	€138,50
Especificações técnicas	Programador	€138,50	2,00	€277,00
<b>Fase de concepção</b>				<b>€2 674,80</b>
Adaptação carta gráfica	Designer	€115,40	1,00	€115,40
	Designer estagiário	€28,00	1,00	€28,00
Carta editorial	Chefe de projeto	€130,80	1,00	€130,80
Redação dos conteúdos	Chefe de projeto	€130,80	2,00	€261,60
Traduções	Chefe de projeto	€130,80	3,00	€392,40
	Tradutor	€1 069,20	1,00	€1 069,20
Fotos	Fotógrafo	€450,00	1	€450,00
Mockups	Designer	€115,40	1,00	€115,40
	Designer estagiário	€28,00	4,00	€112,00
<b>Fase de Produção</b>				<b>€4 146,55</b>
Hospedagem e nome de domínio	Programador	€138,50	0,50	€69,25
Instalação	Programador	€138,50	0,50	€69,25
Compra do tema	Programador	€138,50	0,00	2700
Customização do tema	Programador	€138,50	5,00	€692,50
Internacionalização	Programador	€138,50	1,00	€138,50
Instalação de serviços externos	Programador	€138,50	1,00	€138,50
Criação de emails	Programador	€138,50	0,50	€69,25
Testes	Chefe de projeto	€130,80	1,00	€130,80
Deployment	Programador	€138,50	1,00	€138,50
<b>Fase de exploração</b>				<b>€392,40</b>
Formação CMS	Chefe de projeto	€130,80	1,00	€130,80
SEO	Chefe de projeto	€130,80	2,00	€261,60
<b>Total</b>				<b>€8 729,50</b>

Figura 20: Orçamento provisório

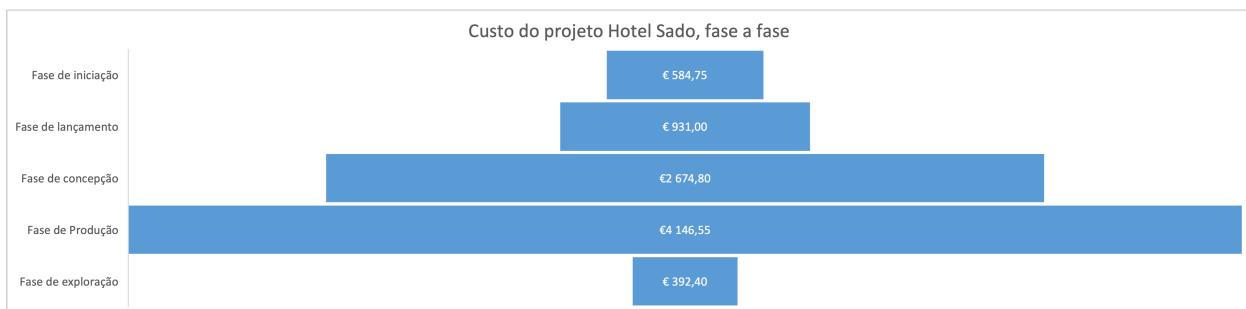


Figura 21: Custo do projeto Hotel Sado, fase a fase

### Resumindo



- Comece por identificar os diferentes itens de despesa.
- Calcular o custo total do projeto equivale a calcular o custo de cada fase.
- Calcular o custo de cada fase é equivalente a calcular o custo de cada tarefa.
- Lembre-se que o orçamento provisório é apenas uma estimativa. Portanto, cabe a você acompanhar a evolução dos custos reais ao longo do projeto.

Agora pode-se estimar que o projeto custará € 8.729 à agência Digitalizer.

## 9.3 Redigir a proposta comercial

Após a recolha e **análise das necessidades do cliente**, a **reunião de definição do âmbito** com a sua equipa e o **planeamento** do projeto, elaborou-se um **orçamento provisório**. Podemos, portanto, dizer que temos todos os elementos para concluir o **estudo de viabilidade**.

A partir disso, já se poderá escrever a proposta comercial. E para dar um exemplo concreto, vamos mais uma vez nos basear no cenário comum, reconstruindo a **proposta comercial** endereçada ao hotel Sado.

Objetivo: destacar-se da concorrência inspirando confiança e entusiasmo suficientes para ser selecionado pelo cliente.

Se a sua proposta não conseguir convencer o cliente a seleccioná-lo, todo o trabalho feito até agora terá sido uma perda de tempo e, portanto, de dinheiro...

### 9.3.1 Estrutura do documento

O que se deve incluir na proposta comercial?

Pelo menos,

- Uma capa
- Um sumário executivo
- Um orçamento

mas também,

- condições de venda em anexo
- uma versão mais redigida do estudo de viabilidade

Quão detalhada deve ser a proposta comercial?

Novamente, dependendo do escopo do projeto em questão, o nível de complexidade, o risco e o perfil do cliente, a proposta comercial será mais ou menos detalhada.

Assim, não se enviará a mesma proposta ao gestor de uma organização de 10 pessoas para um projeto de 10.000 € e ao diretor de uma organização de 5.000 pessoas para um projeto de 500.000 €. Isso pode soar como senso comum, mas não custa repetir.

Para o pequeno projeto do Hotel Sado, um documento de 3 páginas serve.

**9.3.1.1 Cuide da apresentação da sua capa** Gastar tempo a elaborar uma capa pode parecer superficial e fútil. Dito isso, se o patrocinador do projeto não o conhece, essa capa será uma das primeiras coisas que eles verão vindo da sua parte

Escrever uma proposta comercial é, por natureza, um exercício de vendas. Nas vendas, dependendo da pessoa que se tem à sua frente, a forma pode valer tanto que a substância.

A primeira impressão que uma pessoa projeta determina a imagem que temos dela. Da mesma forma, o primeiro documento que se envia a um potencial cliente determina a imagem que ele tem de si ou da sua empresa.

No exemplo do anexo A, notará que a capa já inclui uma aproximação do site para criar um efeito agradável de surpresa e de projeção. Na abordagem clássica de um projeto em cascata, a equipa só produzirá modelos de alta fidelidade durante a fase de projeto.

No entanto, ao competir com outras agências por um projeto, é preciso criar uma proposta de negócios que seja impactante o suficiente para se diferenciar. Para isso, pode-se antecipar a produção de alguns entregáveis para dar suporte à proposta, como o layout da página inicial. Nas agências de comunicação, chamamos esse tipo de investimento em trabalho pseudovoluntário de trabalho “pro bono”.

Fortalecer uma proposta com trabalho pro bono aumenta as hipóteses de ganhar uma licitação, mas também as perdas em caso de fracasso.

### 9.3.2 Redigir um sumário executivo

Como o próprio nome sugere, um sumário executivo deve ser... conciso. O formato não é fixo, dito isso, sugerem-se que três subseções são suficientes para demonstrar a compreensão do projeto e a capacidade de realizá-lo:

- Uma **descrição do projeto e das necessidades** específicas.
- A **solução recomendada** para atender a essas necessidades.
- O **objetivo datado e, se possível, quantificado** do projeto.

Ao escrever um sumário executivo, tenha em mente a suposta grelha de leitura do cliente.

### 9.3.3 Formalizar um orçamento

Após o sumário executivo, incluí-se uma estimativa, ou seja, uma proposta de preço que não se deve modificar até que o cliente tenha assumido um compromisso contratual (e, portanto, definitivo).

### 9.3.4 Coloque a trama

Uma vez que o orçamento é um documento legal, certifique-se de que cumpre as normas e regulamentos em vigor no seu país. Normalmente possuí:

- O período de validade da oferta
- A indicação manuscrita, datada e assinada pelo cliente: “Lido e aceito”
- A menção “Lido e aceito”, datado e assinado pelo contratante

Se existir requisitos legais específicos para o negócio, poderá-se incluí-los como um apêndice à proposta comercial. No caso de um projeto informático como o do Hotel Sado, pode-se, por exemplo, anexar um contrato de apoio detalhando os termos e condições do seu serviço pós-venda, durante a fase de exploração do projeto.

### 9.3.5 Calcula a margem de vendas

Uma vez que a estrutura esteja definida, é necessário preenchê-la com as informações básicas, a lista de entregáveis do projeto e os preços destes entregáveis.

**Portanto, será confrontado com 2 riscos principais:**

- Fazer uma proposta demasiada cara, correndo o risco de não ganhar a licitação
- Fazer uma oferta que não seja cara o suficiente, que não apresenta lucros suficientes e que origina um prejuízo

Para se proteger ao máximo contra esses riscos, é importante calcular com cuidado o **preço de venda** e, portanto, a **margem comercial**.

Simplificando, a margem de vendas é a diferença entre o preço de venda e o preço de custo.

**Margem comercial = preço de venda - preço de custo**

E, inversamente, o preço de venda é a soma do preço de custo com a sua margem comercial.

**Preço de venda = margem de lucro + preço de custo**

No exemplo do hotel Sado, vimos que o preço de custo total é de **8729€**

**Margem comercial = preço de venda - 8729€**

... ou dito de outra forma:

**Preço de venda = margem de venda + 8729€**

Nesta fase, sabe-se quanto custará produzir esses entregáveis, mas ainda não se sabe quanto se cobrará ao cliente por eles. Para definir a margem de manobra, comece-se limitando-a, ou seja, procure-se identificar uma margem máxima e uma margem mínima.

Vamos começar com a margem máxima porque é a mais simples. Aqui, o orçamento do cliente atua como o preço máximo de venda e é de € 15.000:

**Margem máxima = 15.000 - 8.729 = € 6.271 €**

Agora vamos admitir que a agência Digitalizer tem uma política que incentiva os gestores de projeto a não embarcar em projetos com taxa de margem inferior a 50%, por segurança. Ou seja, a agência não quer cobrar por projetos que pagam menos da metade do que custam. Aqui, a taxa de margem é simplesmente uma atualização da margem de negociação como uma percentagem.

**Taxa de margem = (margem comercial / preço de custo) \* 100**

**Margem mínima de negociação = 8.729 \* 0,5**

**Margem comercial mínima = € 4.364**

Adicionando a margem comercial mínima ao seu preço de custo, obtém-se o preço mínimo de venda:

$$\text{Preço mínimo de venda} = 4.213 + 8.427 = € 13.093$$

Assim sabe-se que se pode oferecer um preço de venda entre € 13.093 e € 15.000, ou seja, uma taxa de margem entre 50% e 72%.

$$\text{Taxa de margem máxima} = (6.271 / 8.729) * 100 = 72\%$$

Cabe agora ao gestor do projeto manobrar entre:

- Uma **margem mínima de 50%** para um preço de venda de **€ 13.093**, o que seria muito competitivo.
- Uma **margem máxima de 72%** para um preço de venda de **€ 15.000**, o que seria menos competitivo mas mais lucrativo.

Para decidir sobre um número, considere a saúde financeira da empresa, bem como a do cliente potencial. Ainda no nosso exemplo, digamos que a agência Digitalizer está a ter melhor resultados do que o hotel Sado. Neste caso seria adequado fazer uma proposta que tenda para baixo. Quanto sugeriria?

Uma proposta razoável seria um preço bruto de venda de **€ 13.500** sem IVA. Isso representaria uma taxa de margem de **55%** que parece equilibrada.

Portanto, poderá-se impactar essa margem de forma mais ou menos proporcional em todas os entregáveis listados na estimativa.

Último ponto: quanto pior a saúde financeira do cliente, mais será necessário solicitar um grande depósito para compensar o risco de insolvencia. No nosso cenário, pediremos 40% do valor (5.400€) antes de mobilizar recursos no projeto hoteleiro Sado



### Resumindo

- Não existe uma maneira certa de escrever uma proposta de negócios. Adapte-o ao potencial cliente.
- No mínimo, a proposta deve incluir uma capa de rosto, um sumário executivo para persuadir e um orçamento para convencer.
- A proposta deve ajudá-lo a ganhar o projeto, é claro, mas não a qualquer preço. Para definir o preço de venda, primeiro calcule sua margem.

## 10 Planear a Gestão do risco

O planeamento de riscos em projetos informáticos é o processo de identificação e avaliação de riscos potenciais que podem afetar a boa conclusão de um projeto de software e o desenvolvimento de estratégias para mitigar ou gerir esses riscos.

Durante o processo de planeamento de riscos, as equipas de projeto geralmente identificam uma ampla gama de riscos potenciais, incluindo riscos técnicos (por exemplo, problemas com arquitetura de software, tecnologias não comprovadas), riscos de agendamento (por exemplo, atrasos na conclusão de tarefas), riscos de recursos (por exemplo, escassez de pessoal qualificado) e riscos relacionados a fatores externos (por exemplo, mudanças nas condições de mercado).

Tipos de risco

- Os riscos do **escopo**, que se referem a riscos que envolvem a expansão de requisitos.
- Riscos **tecnológicos**, que se referem a riscos que a tecnologia usada num projeto falhará.
- Riscos do **cliente** e dos **stakeholders**, que se refere aos riscos associados os clientes e as partes interessadas .
- Riscos de **pessoal**, que se referem aos riscos colocados pelo pessoal da equipa de desenvolvimento.
- Muitos outros riscos, tais como **questões jurídicas, problemas de segurança**, destruição de locais físicos, perda de interesse de equipas de gestão ou de desenvolvimento, espionagem industrial, etc.



### Estratégias para lidar com o risco

- Contingência
- Mitigação.

Depois de identificar os riscos potenciais, as equipas avaliarão a probabilidade e o impacto potencial de cada risco e desenvolverão um plano de gerenciamento de riscos que inclua estratégias para mitigar ou gerenciar os riscos. Isso pode incluir atividades como a implementação de processos de gerenciamento de riscos, atribuição de responsabilidades de gerenciamento de riscos, preparação de planos de resposta a riscos e monitoramento e revisão de riscos continuamente.

O planeamento de riscos é um aspecto importante do desenvolvimento de software, pois ajuda a garantir que as equipas de projeto estejam cientes dos riscos potenciais e tenham um plano para respondê a estes riscos de forma eficaz, reduzindo a probabilidade de falha do projeto.

Os métodos ágeis são bem equipados para abordar os riscos de escopo, porque a abordagem ágil tem boas medidas de controle de mudança. Em outras palavras, Agile lida bem com mudanças e exigências em mudança.

- Isso inclui falha de hardware, protocolos de software tornando-se obsoletos ou sem suporte, falta de escalabilidade com a tecnologia ou falta de entendimento tecnológico por parte da equipe de desenvolvimento. Estes geralmente ocorrem quando os clientes prometem fornecer material ou informações para a equipe e eles esquecem ou entregam isso tarde. Exemplos de riscos de clientes ou partes interessadas incluem clientes tornando-se apático ou uma mudança no principal ponto de contato entre a equipe e o cliente.
- Por exemplo, os membros da equipe podem sair parcialmente de um projeto. Isso pode ser especialmente problemático se a pessoa que sai tem competências especializadas que não são compartilhadas por outros membros da equipe. Os riscos de pessoal também podem referir-se à falta de competências necessárias, conflito entre os membros da equipe, ou problemas de comunicação, como pensamento de grupo.

#### 10.0.1 Avaliação de Risco, Probabilidade e Impacto

É importante identificar os riscos potenciais, assim como a probabilidade de um risco acontecer no decorrer de um projeto. Os riscos podem incluir os anti-padrões a discutir na próxima secção, mas também problemas específicos do projeto que podem não ser comum em outros projetos.

1. criar uma **lista de riscos**. É importante reconhecer que se pode tomar medidas para evitar alguns riscos, mas nem sempre é possível.
2. **Atribuir prioridades** a estes riscos, usando, por exemplo, uma matriz de impacto vs probabilidade.
  - Plano de gestão de riscos
    - Matriz de riscos
    - Plano de mitigação

**10.0.1.1 Matriz de riscos** Uma vez listados os riscos, estes são colocados numa matriz de risco em função da sua probabilidade de acontecer e do seu impacto sobre o projeto. Ambas as dimensões são classificadas numa escala de 3 valores: baixo, médio, alto. Considere os 3 riscos seguintes identificados no contexto de um projeto:

1. Esgotamento do orçamento: risco = ?
2. Asteroide destrói a terra: risco = ?
3. O computador do João morrer: risco = ?

2- Potencialmente um grande risco. É um grande impacto. Mas, se tem uma probabilidade média de ocorrência, então deve constar da coluna do meio da matriz, na parte superior da grelha. 3- impacto grande, por isso é na linha superior. No entanto, tem uma probabilidade baixa, de modo que estará na terceira coluna. Será portanto situado no canto inferior direito, e: tem um ranking de risco de 3.

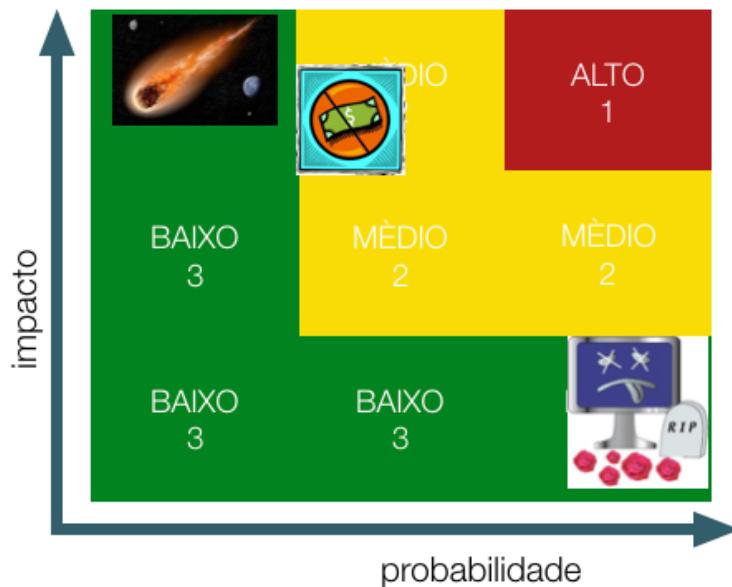


Figura 22: Custo do projeto Hotel Sado, fase a fase

**10.0.1.2 Ações a prever** Com base na classificação do risco, são recomendadas as seguintes medidas:

Classificação | Medidas :-:  
**1** | A equipa precisa de mitigar os danos. O risco precisa de ser tratado o mais rapidamente possível, na medida que é o mais perigoso. **2** | A equipa dever estar preocupada com o risco mas não precisa de tomar uma ação imediata, uma vez que é pouco provável que aconteça. Todavia, uma estratégia deve ser delineada, no caso do risco acontecer. **3** | A equipa não deve perder muito tempo com o risco. É o menos ameaçador.

## 10.1 Exemplo

Considere o desenvolvimento de uma aplicação móvel genérica. A tabela a seguir apresenta uma matriz de risco para desenvolvimento para este tipo de aplicação.

### 10.1.1 Matriz de riscos no desenvolvimento de aplicações móveis

Nível de risco	Descrição do risco	Impacto	Probabilidade	Estratégia
alta	Violão de dados ou acesso não autorizado	Perda de dados confidenciais do utilizador	alta	Implemente medidas de segurança robustas, como criptografia e autenticação de dois fatores
alta	O aplicação é lenta ou apresenta baixo desempenho	Perda de cativação e retenção do utilizador	alta	Teste e otimize regularmente o desempenho da aplicação
Médio	Incompatibilidade com certos tipos de dispositivos	Perda de potenciais utilizador e receitas	Médio	Aplicação de teste numa variedade de tipos de dispositivos e plataformas
Médio	Falta de adoção ou cativação do utilizador	Perda de receita e retenção de utilizadores	Médio	Realizar pesquisas e testes de utilizadores para garantir que a aplicação atenda às necessidades e preferências do utilizador
Baixo	Atraso na entrega ou atraso no lançamento da aplicação	Danos à reputação e potencial perda de receita	Baixo	Estabelecer cronogramas e marcos claros para desenvolvimento e teste
Baixo	Questões legais ou de conformidade	Potenciais multas e danos à reputação	Baixo	Consulte especialistas jurídicos e de conformidade para garantir que a aplicação atenda a todos os regulamentos e padrões necessários

### 10.1.2 Plano de mitigação para riscos de desenvolvimento de aplicações móveis

Em resposta aos riscos, o seguinte plano de mitigação foi elaborado:

1. Violão de dados ou acesso não autorizado:

- Implementar medidas de segurança robustas, como criptografia e autenticação de dois fatores, para proteger os dados confidenciais do utilizador.
- Realizar regularmente auditorias de segurança e testes de penetração para identificar e abordar possíveis vulnerabilidades.
- Educar os utilizadores sobre como proteger as informações das suas contas e incentivá-los a usar senhas fortes.
- Trabalhar com especialistas jurídicos e de conformidade para garantir que a aplicação esteja em conformidade com todos os regulamentos e padrões necessários.

2. A aplicação crasha ou apresenta baixo desempenho:

- Testar e otimizar regularmente o desempenho da aplicação para garantir que ela funcione sem problemas em todos os dispositivos e plataformas.
- Usar a análise de desempenho e ferramentas de monitorização para identificar e resolver possíveis problemas.
- Testar continuamente a aplicação em cenários de alta carga para garantir que ela possa lidar com alto tráfego.

3. Incompatibilidade com certos tipos de dispositivos:

- Testar a aplicação em vários tipos de dispositivos e plataformas para garantir a compatibilidade.
- Permitir que os utilizadores relatem problemas de compatibilidade para que possam ser resolvidos imediatamente.
- Monitorizar continuamente o desempenho da aplicação em diferentes tipos de dispositivos e faça os ajustes necessários.

4. Falta de adoção ou cativação do utilizador:

- Realizar pesquisas e testes de utilizadores para garantir que a aplicação atenda às necessidades e preferências do utilizador.
- Reúnir feedback de testadores beta e utilizadores da primeira hora para identificar quaisquer problemas ou áreas de melhoria.
- Usar análises e métricas de adoção do utilizador para rastrear o comportamento do utilizador e fazer os ajustes necessários.

5. Atraso na entrega ou atraso no lançamento do aplicação:

- Estabelecer cronogramas e marcos claros para o desenvolvimento e os testes.
- Verificar regularmente o progresso e identificar possíveis atrasos ou obstáculos.
- Comunicar quaisquer atrasos às partes interessadas e fornecer atualizações regulares sobre o progresso.

6. Questões legais ou de conformidade:

- Consultar especialistas jurídicos e de conformidade para garantir que a aplicação atenda a todos os regulamentos e padrões necessários.
- Monitorizar regularmente as mudanças legais e regulamentares que possam afetar a aplicação.
- Desenvolver um plano de resposta a incidentes para resolver rapidamente quaisquer problemas legais ou de conformidade que surjam.

Deve-se notar que este plano é baseado em riscos comuns que as aplicações móveis enfrentam. Dependendo das especificidades da aplicação e do domínio ao qual a aplicação pertence, alguns riscos podem não estar presentes, enquanto outros podem ter diferentes níveis de impacto e probabilidade. Portanto, é necessário continuar a monitorizar os riscos e a atualizar em função disso.

### 10.1.3 Notas sobre riscos

**10.1.3.1 O papel do gestor de produto** Mesmo com planos de gestão de risco, é muito importante que o gestor de equipa de desenvolvimento reconheça que haverá sempre riscos que não podem ser previstos ou planeados, já que cada projeto irá enfrentar riscos próprios. - No entanto, isso não significa que o planeamento de risco deve ser ignorado. Os gestores de produtos devem desenvolver uma competência para lidar com o desconhecido e o inesperado. - Em casos de riscos imprevistos, o sucesso do projeto dependerá em grande parte do modo como esses riscos são tratados na altura.

## 10.2 Anti-padrões da gestão de projetos informáticos



### Anti-padrões - Definição

Padrão de projeto de software que pode ser usado com alguma frequência mas é **ineficiente** e/ou **contra-productivo** na prática.

### 10.2.1 anti-padrões de grupo

PT	GB
Parálisia de análise	Analysis paralysis
A carrinha antes dos bois	Cart before the horse
Pensamento de Grupo	Groupthink
Silos	Silos
Bloqueio do fornecedor	Vendor lock-in
Sobre-engenharia	Over-engineering
Folhear a Ouro	Gold plating
Engenharia de visão técnica	Viewgraph engineering
Boca de incêndio e heroísmo	Fire drill and heroics
Marcha da morte	Death march

### 10.2.2 Anti-padrões de gestão de projeto

- **Marcha da morte:** Todos sabem que o software será um desastre - exceto o CEO - então a verdade é escondida
- **Pensamento de Grupo:** Durante o projeto, os membros da equipa evitam demonstrar pontos de vista fora da zona de conforto
- **Fumaça e espelhos:** Demonstrar funções não implementadas como se já tivessem sido implementadas
- **Modelo em cascata:** Um velho método de desenvolvimento que lida de forma inadequada com mudanças inesperadas

## 10.3 Anti-padrões organizacionais

- **Parálisia de análise:** Dedicar esforço desproporcional à fase de análise do projeto
- **Barraca de bicicleta:** Dedicar grande esforço a questões triviais.
- **Bleeding Edge:** Usar tecnologias de ponta instáveis e/ou não testadas, sendo causa de transbordamento do orçamento, baixo desempenho e/ou atraso na entrega.
- **Vaca do dinheiro:** Um produto herdado que às vezes leva à complacência sobre novos produtos
- **Projeto por submissão:** O resultado de ter vários pessoas a contribuir para o projeto, mas nenhuma visão unificada
- **Agravamento de compromisso:** Falhar em revogar uma decisão quando provada errada
- **Gestão por perkele:** Estilo autoritário de gestão sem tolerância à divergências
- **Departamentalização matricial:** Estrutura organizacional desfocada que resulta em lealdades divididas e falta de direção
- **Risco moral:** Isolar alguém das consequências da sua decisão
- **Gestão cogumelo:** Manter funcionários desinformados e mal-informados
- **Aprisionamento tecnológico:** Fazer um sistema excessivamente dependente de um componente externo

### **10.3.1 Paralisia de análise**

#### **10.3.1.1 Anti-padrão de grupo**

- A equipa de desenvolvimento fica **presa** ou **parada** numa fase do projeto, geralmente a fase de análise dos requisitos, levando à paralisia do projeto.

Isso geralmente acontece na **fase de especificação** do projeto. É durante essa fase que os projetos atrasam-se porque os clientes e / ou os gestores de produto passam muito tempo a analisar os requisitos e não podem decidir arrancar com o projeto numa direção até que a análise seja aperfeiçoada.

#### **10.3.1.2 Solução**

- Uma boa estratégia é executar um projeto com releases incrementais, como preconiza a abordagem Agile. Com releases incrementais, nem tudo necessita de ser conhecido antecipadamente. entende-se que o produto deve ser flexível e que irá mudar e refinar-se ao longo do tempo.

---

---

"Deliver as fast as possible"

---

---

### **10.3.2 Carrinha antes dos bois**

#### **Anti-padrão**

- Esta expressão significa que algo foi colocado numa ordem contraproducente. Na gestão de produtos de software significa que foi colocada muita ênfase numa parte do projeto que deve ser feito mais tarde.

**10.3.2.1 Exemplo** Menos tempo e recursos são gastos a desenvolver funcionalidades críticas de um produto do que numa funcionalidade menos crítica ou que não é necessária no momento.

Acontece quando um programador se auto-atribui trabalhos que não afetam diretamente as funcionalidades atuais e que se quer dar algum avanço sobre o trabalho futuro. O perigo deste processo é que uma equipa pode encontrar-se com um conjunto de funcionalidades mal desenvolvidas com graus de prioridade heterogêneos.

**10.3.2.2 Solução** É importante que **a equipa de desenvolvimento compreenda as prioridades** de desenvolvimento e se concentre nessas prioridades. Em outras palavras, a equipa deve entender claramente o trabalho que deve ser feito agora e que trabalho deve ser feito mais tarde.

### **10.3.3 Pensamento de grupo**

- Refere-se a forma como as pessoas tendem a seguir as opiniões gerais de um grupo, independentemente das suas próprias opiniões individuais serem diferentes. O termo vem das ciências sociais. O pensamento de grupo pode levar a decisões e julgamentos pobres no desenvolvimento de projetos.

**10.3.3.1 Exemplo** Numa reunião de projeto só 1 ou 2 elementos do grupo intervém e os outros não. Podem levar a um mau produto porque opções melhores não foram exploradas.

- Aceitar as divergências de opinião / reconhecer direito a discordar
- **Lean Software Design:** Permitir que uma equipa se separe em grupos menores ou indivíduos e incentivar esses grupos menores a propor alternativas. Essas sugestões são então combinadas. Esta abordagem tem o benefício de não colocar as pessoas em foco.

#### 10.3.4 Silos

Falta de comunicação entre os grupos de uma equipa de desenvolvimento, o que leva a uma perda de foco unificado e a criação de estratégias e funcionalidades contraproducentes do produto.

Os recursos desenvolvidos em silos são desenvolvidos em um vácuo, levando a que o trabalho de uma parte da equipa seja diferente ou incompatível com o de outro grupo, e que a junção do trabalho das duas equipas seja difícil.

##### 10.3.4.1 Exemplo

- Estruturas de gestão fortemente hierarquizadas onde os programadores falam poucos entre eles e reportam diretamente aos gestores.

#### Solução

- Incentivar um ambiente aberto e positivo, onde a comunicação e a colaboração da equipe é favorecida.
- Reavaliar a estrutura de gestão também pode ser benéfico, de forma a torná-la mais plana e com mais interações entre programadores.
- Encorajar a comunicação cara-a-cara.

#### 10.3.5 Bloqueio de fornecedor

- Ocorre quando uma equipa de desenvolvimento cria um produto que depende muito de uma única solução ou fornecedor de tecnologia. É diferente de uma escolha racional por uma tecnologia que é de facto a melhor opção. Resulta em geral, de uma falta de flexibilidade na mudança para outra tecnologia sem custos substanciais.

A forte dependência de uma única tecnologia à medida que o projeto avança é problemática se a tecnologia não cobrir o que é necessário, vier a ficar desatualizada ou não se adaptar à mudança. Em casos extremos, a equipa de programadores pode mesmo deixar uma organização em resultados de um bloqueio de fornecedor.

##### 10.3.5.1 Solução

- Para evitar esse risco, é importante pesquisar antes de se comprometer com uma tecnologia ou uma solução, independentemente das recomendações feitas ao projeto.

#### 10.3.6 Sobre-engenharia

- Refere-se a forma como a equipa de desenvolvimento cria o próprio produto. Ocorre quando um produto construído é mais complexo do que é necessário. Pode acontecer tanto na interface do utilizador (UI) de um produto como nos seus processos internos.

### **10.3.6.1 Exemplo**

- funcionalidades extras ou desnecessárias são adicionadas a um produto, tornando-o confuso para a maioria dos utilizadores: câmeras digitais com muitas características que devem ser definidas antes da funcionalidade é ativada, players de música com muitas opções, carros que podem atingir velocidades que nunca será atingidas, ou editores de texto com muitas opções exportação.

### **10.3.6.2 Solução**

- É importante que a equipa de desenvolvimento comprehenda claramente as necessidades para um produto (o que o produto precisa ser capaz de fazer) versus os desejos para um produto (o que seria bom que o produto fizesse, mas desnecessárias para o sucesso).
- Atribuir prioridades aos requisitos e assegurar que um projeto é viável.

### **10.3.7 Folhear a ouro**

Ocorre quando tantos esforços são colocados numa parte de um projeto que ele **atinge um ponto em que o lucro decresce**. Em outras palavras, tanto esforço extra é focado numa parte do projeto que já não contribui para o valor do produto. A equipa de desenvolvimento adiciona recursos extras a um produto para impressionar o cliente, especialmente quando a equipe termina cedo.

No entanto, os recursos adicionados podem adicionar trabalho imprevisto e extra a um projeto e funcionalidades que o cliente realmente não deseja no produto (lembre-se de que os requisitos do utilizador devem ser especificados pelo cliente e não pela equipe de desenvolvimento).

### **10.3.7.1 Solução**

- A equipa de desenvolvimento deve parar de trabalhar quando o produto funciona bem. Se a equipe sente que novos recursos devem ser adicionados, os recursos devem primeiro ser examinados com o cliente.

As boas perguntas a fazer ao considerar adicionar características extra incluem:

- Essa funcionalidade melhora o produto?
- Essa funcionalidade torna o produto confuso de alguma forma?
- Esta funcionalidade tira da funcionalidade principal?

Realização de um estudo do utilizador.

### **10.3.8 Engenharia de visão Técnica**

Ocorre quando muito pouco esforço é colocado num projeto. É um pouco o oposto do folhear a ouro. No entanto, se folhear a ouro é trabalho desfocado, Engenharia de visão Técnica corresponde a trabalhar no que não é importante.

### **10.3.8.1 Exemplo**

- dificulta o trabalho do projeto para os programadores, exigindo que eles trabalhem em outras coisas além do trabalho de desenvolvimento, como documentação, relatórios ou criação de apresentações. O tempo gasto nessas atividades não deve ser maior do que o tempo gasto escrevendo código ou desenvolvendo um projeto.

### **10.3.8.2 Solução**

- Usar métodos ágeis: o foco está no produto e no seu desenvolvimento
- Remover quaisquer barreiras para que os programadores possam se dedicar ao desenvolvimento de um projeto.
- Criar um protótipo básico é muitas vezes mais informativo do que qualquer relatório ou apresentação.

### **10.3.9 Boca de incêndio e heroísmo**

A equipa produz muito pouco trabalho durante a maior parte do projeto e faz um forcing no final do projeto, resultando em horas extras de trabalho, a fim de cumprir prazos.

A equipa depende de uma pessoa com grandes competências técnicas para realizar o projeto.

#### **10.3.9.1 Solução\***

- Estabelecer expectativas com o cliente no início do projeto e seguir as práticas de desenvolvimento de software ágil.

Por exemplo, usando time-boxing e produzindo sempre um produto de trabalho no final dessas caixas de tempo. Essas estratégias manter a equipe trabalhando num ritmo regular.

### **10.3.10 Marcha da morte**

#### **Anti-padrão**

- Acontece quando a equipa de desenvolvimento e a equipa de gestão estão em desacordo, porque a equipa de gestão está a impor um caminho para o projeto no qual a equipa de desenvolvimento não acredita.
- Isso leva a equipe de desenvolvimento a perder convicção e paixão pelo projeto, o que aumenta a probabilidade Falha do projeto. Apesar disso, todos continuam trabalhando por um senso de obrigação.

#### **10.3.10.1 Exemplo**

- Podem ocorrer por razões financeiras, ou quando a administração é muito teimosa em reconhecer outras ideias e o potencial fracasso do projeto. Em ambos os casos, se a moral do programador é baixa, então a qualidade do produto irá sofrer.

#### **10.3.10.2 Solução**

- é importante que a equipa de gestão mantenha uma comunicação aberta com a equipa de desenvolvimento. Os gestores devem ouvir o que a equipa pensa sobre a direção do projeto e deve estar dispostos a explorar outras alternativas no desenvolvimento do projeto.

## **10.4 Anti-padrões individuais**

### **10.4.0.1 Microgestão**

**10.4.0.1.1 anti-padrão dos Gestores** Um gestor é muito controlador e quer ser envolvido em cada detalhe do projeto, não importa quanto pequeno. O gestor precisa constantemente saber o que seus programadores estão a fazer. Isso se traduz por uma falta de confiança na equipa de desenvolvimento, o que afeta o moral da equipa e a qualidade do produto e pode se agravar se a equipa é culpada pelo comportamento do gestor.

#### **Exemplo**

- É o resultado dos próprios medos internos do gestor, inseguranças, ou stresses básicos. Também pode ser causada por prazos excessivamente ambiciosos, qualidade de produto deficiente, ou medo de que os programadores não estão à altura do trabalho.

#### **Solução**

- pode ser difícil, como não há nenhuma correção rápida e fácil. A solução parte dos micro gestores devem admitir que seu comportamento prejudica a equipa e estar disposto a tomar medidas para melhorar seu comportamento.
  - Curso de gestão pessoal.

#### **10.4.0.1.2 Gestor gaivota Anti-padrão**

- Acontece quando um gestor só está presente quando surgem problemas ou antes dos prazos.

#### **Exemplo**

- um gestor ausente que despeja informações ou pedidos sobre a equipa e, que desaparece novamente. Isso pode causar muito estresse para a equipa e pode facilmente empurrar um projeto para se tornar uma boca de incêndio. A equipa de desenvolvimento trabalha com o medo constante da próxima leva de pedidos.

#### **Solução**

- Aliviar a carga de trabalho do gestor.
- Se suspeitar que pode ser um gestor gaivota, é uma boa ideia pensar em reavaliar sua agenda e a forma como interage com a equipa de desenvolvimento.

#### **10.4.0.1.3 E-mail como meio principal de comunicação** Membros da equipa

Tão comum que se tornou um anti-padrão. Cria gestores gaivotas que só escrevem para a equipa com grandes cargas de trabalho ou grandes lixeiras de informações, e não se comunicam com frequência.

#### **Exemplo**

- o e-mail muitas vezes não é a melhor maneira para comunicar, porque pode ser difícil fornecer contexto em e-mails, ou, alternativamente, a informação pode ser facilmente mal interpretada. Os e-mails também podem ser muito formais para a maioria das comunicações.

#### **Solução**

- comunicar pessoalmente. Curso de gestão pessoal.
- outros meios de comunicação como serviços de chat, videoconferência e chamadas telefónicas.
- permite dar respeito e atenção aos membros da equipa, aumenta a motivação da equipa

#### **10.4.0.1.4 Canhões soltos** Membros da equipa

Comportamento imprevisível de uma pessoa que tem efeitos negativos a não ser que seja gerido de forma particular.

##### **Exemplo**

Pessoas que tomam decisões significativas de projeto sem consultar o resto da equipa. Esse comportamento é imprudente e pode criar mais trabalho para outros membros do grupo.

##### **Solução**

- Como muitos anti-padrões individuais, não há uma solução rápida para um canhão solto.
- Muitas vezes, lidar com canhões soltos envolve a compreensão das motivações do comportamento do indivíduo. Está a tentar causar problemas? Está a tentar evidenciar o seu trabalho para o resto da equipa?
- Um gestor ou a equipa pode tentar e fazer entender ao indivíduo que o seu comportamento é destrutivo e incentivar o indivíduo a tomar medidas para alterá-lo.
- Os canhões soltos devem ser tratados com rapidez ou o risco de um projeto irá aumentar ao longo do tempo.

#### **10.4.0.1.5 Violência intelectual** Membros da equipa

Diz respeito a uma pessoa que afirma sua opinião sobre cada tema e impede o progresso questionando todas as decisões e ações ou usando seus conhecimentos avançados numa área para intimidar ou olhar com desdengo os outros membros da equipa.

##### **Exemplo**

Esses indivíduos tendem a repetir tanto as suas opiniões que o resto do grupo consente apenas para evitar o confronto. Os canhões soltos são às vezes a causa da violência intelectual.

##### **Solução**

O gestor de projeto pode tentar falar com o indivíduo em particular sobre o seu comportamento e sugerir quaisquer alterações adequadas. Alternativamente, o gestor de projeto pode encorajar uma política de perguntas abertas (por exemplo, reforçando a ideia de que não existe uma questão estúpida) e desencorajar opiniões antecipadas ou inexperiência. A violência intelectual deve ser abordada rapidamente para evitar que um projeto de tomar uma direção ruim e manter o moral da equipa.

## **11 Planear a execução**

### **11.1 Planeamento de responsabilidades**

O planeamento de responsabilidade num projeto infprmático refere-se ao processo de identificação e atribuição de papéis e responsabilidades para várias tarefas e atividades relacionadas com o desenvolvimento e manutenção do produto. Isso inclui determinar quem será responsável por aspectos específicos do projeto, como a recolha de requisitos, o design, a codificação, os testes e a instalação.

O planeamento de responsabilidades é um aspecto crucial do desenvolvimento de software porque ajuda a garantir que todas as tarefas sejam concluídas no prazo e com alto padrão, e que todos na equipe estejam cientes de sua função e do que se espera deles. Isso pode ajudar a evitar confusão, atrasos e trabalho extra e garantir que o projeto de software seja concluído com sucesso.

O planeamento de responsabilidade também envolve a criação de uma cadeia clara de comando e comunicação, para que todos saibam a quem pedir ajuda, a quem reportar e quem é responsável pelos diferentes aspectos do projeto. Isso pode ajudar a garantir que o projeto seja executado sem problemas e que todos estejam cientes do que está acontecendo todo o tempo.

### 11.1.1 Atribuições de responsabilidade (Matriz RASIC)

RASIC é um acrônimo para "Matriz de Atribuição de Responsabilidade", também conhecida como "matriz RACI". É uma ferramenta usada para definir e comunicar claramente as funções e responsabilidades de diferentes indivíduos ou equipes dentro de um projeto ou organização.

A matriz de responsabilidade é muitas vezes referida como uma matriz RACI ("Ray-Cee") ou RASIC matriz ("Ray-Sick"). As siglas representam cada nível de responsabilidade potencial.

- R: Responsibility ou. Responsible
- A- Accountable. ou. Approve
- C: Consulted. ou Support
- I: Informed. ou Informed
- C: Consulted

A matriz RASIC é normalmente representada como uma tabela com linhas que representam as várias atividades ou tarefas que precisam ser concluídas e colunas que representam as diferentes funções ou indivíduos envolvidos no projeto. A cada célula da tabela é atribuída uma letra indicando o nível de responsabilidade para aquela função ou indivíduo para aquela tarefa específica. As letras mais usadas são:

R: Responsibility - a pessoa ou equipa encarregada de concluir a tarefa  
A: Accountable. - a pessoa ou equipa responsável pela conclusão da tarefa no prazo e no padrão exigido  
S: Consulted. - a pessoa ou equipa que fornece suporte para a tarefa  
I: Informed. - a pessoa ou equipa que precisa ser informada sobre o andamento da tarefa  
C: Consulted - a pessoa ou equipa que é consultada quando as decisões são tomadas em relação à tarefa

A matriz RASIC é uma ferramenta útil para evitar confusão, atrasos e retrabalho e garante que o projeto seja concluído com sucesso.

Role	Team Member	Approved Budget	Current State Process Flow	Requirements Document	Change Impact Analysis	Configuration Mgmt Plan	Disaster Recovery Strategy	Deployment Plan	Project Plan	QA Plan	Risk Response Plan	Future State Process Flow	Test Cases	Test Plan	Training Plan	System Design Document
Project Sponsor	T Terrific	A	I		A		A	A	A	A	A	I	I	I	A	I
Project Manager	M Yost	R	C		R	R	S	R	A	R	C	C	A	C	A	
Technical Leader	B Gates	S	C	A	C	I	R	S	A	A	A	C	C	C	C	A
Business Process Leader	S Jones	C	A		A	C		A	A	A	A	A	A	A	R	A
Lead Developer	L Gregory		C	C		S		C	C	C	I	C		I	R	
Lead Analyst	E Michael		R	R	C	I		R	A	C	C	R	S	S	S	S
QA Manager	N Reed	I	I	I	C	I		C	A	R	C	I	A	A	A	I
Test Manager	Q Victoria		I	I		I		R	A	C	C	R	R	R	C	C
Developer	R Alexander	I	I		S		I	C	I	C	I	C		C	S	

Legend: R=Responsible; A=Approve; S=Support; C=Consulted; I=Informed

Figura 23: Exemplo de matriz RASIC

Pode ser acompanhado ou substituída por um plano de responsabilidade que indica as responsabilidades de cada membro da equipa.

Project Role	Project Responsibilities	Assigned Team Member
Project Sponsor	<ul style="list-style-type: none"> <li>* Responsible for championing the project and communicating all aspects of the project to other senior management stakeholders.</li> <li>* Has ultimate authority over and is responsible for the project and/or the program.</li> <li>* Approves changes to the scope and provides the applicable funds for those changes.</li> </ul>	T. Terrific
Project Manager	<ul style="list-style-type: none"> <li>* Provides direction and oversight to the initiative</li> <li>* Works with stakeholders to ensure that expectations are met</li> <li>* Develop and manage project plan</li> <li>* Design and execution of a project communications plan</li> <li>* Measure, evaluate, and report progress against the project plan</li> <li>* Provide project status reports</li> <li>* Coordinate and manage activities of project personnel</li> <li>* Resolve project issues</li> <li>* Conduct scheduled project status meetings</li> <li>* Establish documentation and procedural standards for the project</li> <li>* Perform quality review of deliverable documents</li> <li>* Maintain project communication with the Client Project Manager</li> <li>* Review and administer Project Change Control Procedures.</li> </ul>	M. Yost
Technical Leader	<ul style="list-style-type: none"> <li>* Provide technical leadership on the design of application architecture</li> <li>* Lead resolution of any application development issues</li> <li>* Facilitates technical design sessions</li> <li>* Provides quality assurance to technical deliverables</li> </ul>	B. Gates
Quality Assurance Manager	<ul style="list-style-type: none"> <li>* Provides quality assurance to the overall project processes, procedures, and deliverables.</li> <li>* Works with Project Leadership to ensure project expectations are met</li> </ul>	N. Reed
Business Process Leaders	<ul style="list-style-type: none"> <li>* Provide business competence to the project team</li> <li>* Participate in information gathering sessions</li> <li>* Provide pertinent strategic business documentation and information</li> <li>* Assist in the identification of business critical processes</li> <li>* Validate viability of recommendations</li> <li>* Serve as primary user acceptance testers</li> </ul>	S. Jones G. Griffey

Figura 24: Exemplo de plano de responsabilidade

## 12 Controlar um projeto

Durante a execução do projeto, o gestor do projeto controla o projeto, faz a gestão das alterações, a gestão das entregas, a gestão dos problemas e dos riscos, assim como da qualidade do projeto.

Quando o gestor de projeto monitoriza e controla um projeto, pode seguir cinco etapas básicas:

### 1. Acompanhe as atividades do projeto

É importante verificar com os membros da equipe regularmente para garantir que eles estejam concluindo suas tarefas e atendendo aos padrões de qualidade. Você pode fazer isso de forma mais eficaz por meio de reuniões de equipe se todos trabalharem no mesmo local. No entanto, dado o quanto comum são as equipes distribuídas no ambiente de negócios atual, pode ser difícil reunir todo o grupo. Quando for esse o caso, conduza sessões de trabalho separadas com indivíduos ou pequenos grupos necessários para atividades específicas. Também uso "verificações de amigos" para verificar se as tarefas são realizadas corretamente. Quando alguém conclui uma atividade, outro membro da equipe analisa os resultados. Esta não é uma análise técnica aprofundada; é uma verificação rápida para confirmar se a pessoa que fez o trabalho não esqueceu algo acidentalmente ou não entendeu os requisitos. Um membro da equipe verificando um plano de treinamento para um novo sistema, por exemplo, certificar-se-ia de que todos os departamentos que precisam de treinamento foram incluídos.

### 2. Recolher dados / métricas de desempenho

Algumas empresas possuem sistemas de informações de gerenciamento de projetos que geram relatórios automaticamente. Se tiver acesso a um, deve usá-lo sem dúvida - mas também procure dados de desempenho por meio de reuniões curtas, nas quais os membros da equipe compartilham atualizações de status sobre atividades e avaliam riscos, pessoalmente ou virtualmente. Lembre-se de limitar a 10 minutos e discutir apenas as tarefas iniciadas ou concluídas desde a última reunião. O objetivo é ter uma noção rápida de onde as coisas estão, não arreganchar as mangas. Se a equipe identificar algum problema ou risco, o gestor de projeto resolva-os numa sessão de trabalho separada com as pessoas apropriadas. Normalmente, mede-se o pulso dos projetos semanalmente, o que permite acompanhar o progresso de forma adequada e identificar problemas a tempo de respondê-los. No entanto, quando um projeto está em modo de crise, a "taxa de pulso" acelera.

### 3. Analisar o desempenho para determinar se o plano ainda se mantém

As atividades raramente acontecem exatamente como previsto. Eles podem levar mais ou menos tempo; eles podem ultrapassar ou ultrapassar o orçamento. Um afastamento do plano não é um problema, a menos que possa comprometer os objetivos da equipe. Num projeto, um engenheiro relatou em uma reunião de pulso que um novo módulo estaria atrasado duas semanas. Mas como o módulo não estava em nosso caminho crítico e tínhamos quase seis semanas de folga nessa parte do cronograma, a equipe não precisou tomar nenhuma ação especial.

Quando um plano precisa de ser revisado, pode ser necessário estender a data final, aplicar reservas orçamentárias, remover entregas do escopo do projeto ou até cancelar o projeto.

### 4. Relatar o progresso às partes interessadas

Alguns gestores de projeto e membros da equipe percebem as revisões das partes interessadas - que envolvem a preparação de relatórios e a realização de reuniões de progresso - como um esforço desperdiçado, porque levam tempo longe de outras atividades. No entanto, se gerirmos adequadamente, essas revisões impulsionam um projeto para o sucesso. Existem três tipos: revisões de gestão, revisões de portfólio e revisões técnicas. O objetivo da análise pela gestão é gerir o risco. As partes interessadas podem examinar vários projetos ao mesmo tempo para ver se o portfólio como um todo gerará o desempenho de negócios desejado e serve para identificar

fraquezas sistémicas. Eles também analisarão os projetos individuais por seus próprios méritos. Essas revisões são normalmente realizadas em intervalos regulares – mensalmente, por exemplo. Ao conduzi-los, lembre-se de que seus stakeholders se preocupam em atingir os objetivos do negócio, e não em acompanhar o dia a dia da equipa.

Criar um painel de projeto é uma ótima maneira de resumir seus objetivos e mostrar às partes interessadas se o projeto, conforme planeado e gerido atualmente, os alcançará. (Você pode dividi-lo em componentes como cronograma, custo e desempenho.) Isso geralmente é chamado de gráfico de semáforo, pois geralmente indica status de atividade em vermelho, amarelo e verde. A maioria das empresas tem um formato padrão para ajudar os gerentes seniores a avaliar com rapidez e eficiência o progresso e os riscos em muitos projetos.

Ao usar o código de cores, certifique-se de que todos entendam exatamente o que cada cor significa. Por exemplo, pode-se listar todas as tarefas incompletas em vermelho? Ou alguns deles são verdes, porque o plano de conclusão está aprovado e em andamento?

#### 5. Gerir mudanças no plano

Ao revisar um plano, pode-se fazer grandes mudanças ou apenas pequenos ajustes que permitirão que a equipa atinja os seus objetivos. Se se propor mudanças importantes aos stakeholders, especifique os custos e riscos de adotá-las e os de manter o plano original.

Ao fazer essas revisões em grande escala, registe-as (junto com a justificativa) em algum tipo de registo de alterações nos registos do projeto. Pode-se usar os processos e técnicas normais de planeamento de projetos para rever o plano. Envie o novo plano aos membros da equipe e explique as alterações que os afetam.

Pequenas mudanças podem surgir enquanto se implementa um plano de contingência ou se elabora detalhes de uma parte do projeto que foi planeada apenas em alto nível.

Durante a fase de execução de um projeto de software, os membros da equipe geralmente trabalham em tarefas relacionadas à codificação, teste e implementação dos requisitos do projeto. Isso pode incluir escrever e depurar código, criar e executar planos de teste e trabalhar com outros membros da equipe para garantir que o projeto seja concluído no prazo e dentro do orçamento. Os membros da equipe também podem estar envolvidos na solução de problemas e na resolução de quaisquer problemas que surjam durante o projeto, além de fornecer suporte e manutenção contínuos para o software depois de implantado. Além disso, os membros da equipe também podem estar envolvidos nas atividades de gerenciamento do projeto, como monitorar o progresso, comunicar-se com as partes interessadas e fazer ajustes no plano do projeto conforme necessário.

## 13 Encerramento do projeto

### Manipulação de autoridade e controle Captura das lições aprendidas

Durante a fase de encerramento do projeto informático, várias atividades são normalmente executadas para garantir que o projeto seja concluído com sucesso e que todas as etapas necessárias sejam tomadas para fazer a transição do software para sua próxima fase de desenvolvimento ou operação. Essas atividades podem incluir:

1. Conduzir uma revisão do projeto: A equipe do projeto analisa o progresso, o desempenho e os resultados do projeto para identificar o que correu bem e o que pode ser melhorado no futuro.
2. Finalizar a documentação: Todos os documentos relacionados ao projeto são revisados, atualizados e arquivados. Isso pode incluir o plano do projeto, documentos de requisitos, documentos de design, planos de teste e manuais do utilizador.
3. Obter a aceitação do cliente: O cliente é solicitado a aceitar formalmente o software e assinar o projeto, indicando que está satisfeito com o software e que atende aos seus requisitos.

4. Encerrar os contratos: Todos os contratos estabelecidos durante o projeto são encerrados, incluindo contratos de fornecedores e acordos de subcontratados.
5. Conduzir uma revisão pós-implementação: A equipe do projeto realiza uma revisão de todo o projeto, incluindo as fases de planeamento, execução e encerramento, para identificar as lições aprendidas e as melhores práticas para projetos futuros.
6. Entrega do software à equipe de suporte: O software é entregue à equipe de suporte para manutenção e suporte contínuos.
7. Libertação de recursos: Quaisquer recursos que foram dedicados ao projeto, como equipamentos e instalações, são liberados para outros usos.
8. Celebrar a conclusão do projeto: os membros da equipa comemoram a conclusão bem-sucedida do projeto e refletem sobre o trabalho árduo e a dedicação que o tornaram possível.

## **14 Anexos**

## 14.1 Anexo A - Resposta ao concurso Projeto Digital

### 14.1.1 Sumário executivo

**14.1.1.1 Descrição do Projeto** O patrocinador do projeto, Sra Silva, gerente do Hotel Sado expressou as seguintes necessidades:

- desenvolver a presença online do hotel
- modernizar sua imagem
- permitir que seus clientes reservem sua estadia no hotel sem precisar ligar para eles

Nos últimos anos, a concorrência no mercado de hospedagem em Setúbal intensificou-se. O AirBnB e os agregadores de ofertas de acomodação transformaram a maneira como as pessoas reservam acomodações. Ter um sistema de reservas online tornou-se um padrão na indústria hoteleira. Ter um website com funcionalidades adaptadas permitiria ao Hotel Sado assegurar a sustentabilidade do seu estabelecimento (atualmente comprometida).

**14.1.1.2 Solução** Para turistas estrangeiros e viajantes profissionais que precisam de um quarto luxuoso em Setúbal, [www.hotel-sado.com](http://www.hotel-sado.com) permite reservas on-line automatizadas no idioma do visitante (ao contrário do antigo sistema de reservas do hotel) e em todos os tipos de dispositivos .

**14.1.1.3 Objetivo** O objetivo do projeto é montar um sistema de comunicação online que permita ao hotel reencontrar o equilíbrio financeiro.

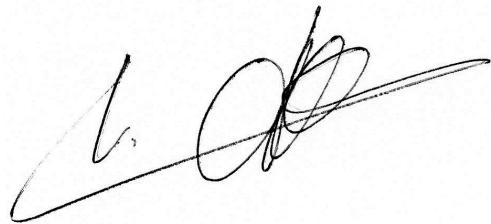
Data de início do projeto: 2 de novembro Data de entrega: 6 de janeiro

**14.1.1.4 Orçamento Data de emissão:** 27/10/2022 **Válido até:** 27/11/2022 **De:** Sr. Pedro Jorge ([pedro.jorge@digitalizer.com](mailto:pedro.jorge@digitalizer.com) - 96.12.19.96.90) Digitalizador | rua da serra, 11, Setúba - Portugal. NIF: PT123456789 Aos cuidados de: Sra. Antónia Silva Hotel Sado | rua do paraíso, 1, setúbal (2940-234) - Portugal.

Quantidade	Designação	Redução	Preço unitário sem IVA	Valor com IVA
1	Criação de um site montra multilingue adaptado a todos os ecrãs e equipado com sistema de reservas (baseado em Wordpress)	-	€7.600	€9.500
1	Adaptação da norma gráfica para a Web	-	€ 240	€300
1	Fotografias	-	€400	€500
1	Editor-edição	-	€320	€400
1	Internacionalização em Inglês, Russo e Chinês	-	€1,520	€1,900
1	Formação em gestão de conteúdos em Wordpress	-	200€	250€
1	Referenciação	-	360€	450€
1	Gestão de alojamento e domínio (assinatura anual)	-	160€	200€
1	Criação de emails dedicados 100%	100%	120€	
Total excluindo IVA			10.800 €	
<b>IVA total 20%</b>				2700 €
Total com IVA				€13.500
Líquido a pagar (€)				<b>€13.500</b>

Nota. Entregáveis feitos de acordo com as especificações projetadas com o direito de inspeção do cliente.

**Assinatura do representante da agência: Assinatura do representante do hotel:**

A handwritten signature in black ink, appearing to read "L. O. S." followed by a stylized surname.

Li e aceito Li e aceito

## **14.2 Anexo B - Diferenças entre o planeamento ágil de projeto e o planeamento com metodologias clássicas**

O planeamento ágil de projetos é uma abordagem iterativa e flexível para planear e gerir projetos, enquanto a abordagem em cascata, também conhecida como método Waterfall, é uma abordagem linear e sequencial. No método Ágil, o trabalho é dividido em partes pequenas e fáceis de gerir, chamadas sprints, e o progresso é revisto e adaptado ao final de cada sprint. Em oposição, a abordagem Waterfall segue uma progressão estrita, na qual cada fase do projeto é concluída antes de passar para a próxima. A abordagem ágil é mais adequado para projetos com alta incerteza ou requisitos que mudam rapidamente, enquanto que a abordagem em cascata é mais melhor para projetos com requisitos bem definidos e um objetivo final claro.

### **14.2.1 O planeamento Ágil**

O planeamento de projeto para um projeto ágil segue, de forma genérica, as seguintes etapas:

1. Estrutura Analítica do Trabalho (WBS):

- Definir o escopo e os objetivos do projeto
- Dividir o projeto em partes menores e gerenciáveis (épicos, histórias e tarefas)
- Atribuir propriedade e estimar esforço para cada item na WBS

2. Avaliação das Tarefas:

- Definir prioridades para as tarefas com base no valor comercial, dependências e viabilidade
- Identificar e mitigar os riscos associados a cada tarefa
- Estabelecer critérios de aceitação para cada tarefa

3. Cronograma:

- Crie um cronograma de sprint, normalmente de 2 a 4 semanas
- Definir as tarefas que serão concluídas durante cada sprint
- Agendar reuniões regulares, como planeamento de sprint, reuniões diárias, revisões de sprint e retrospectivas

4. Métricas:

- Definir indicadores-chave de desempenho (key performance indicators - KPIs) para acompanhar o progresso e medir o sucesso
- Medir e relatar a velocidade, com gráficos de burndown e outras métricas ágeis
- Use métricas para identificar e resolver quaisquer problemas ou obstáculos

### **14.2.2 Planeamento de um projeto em cascata**

O planeamento de projeto para um projeto em cascata segue, de forma genérica, as seguintes etapas:

1. Estrutura Analítica do Trabalho (WBS):

- Definir o escopo e os objetivos do projeto
- Dividir o projeto em partes menores e geríveis (fases, entregas e tarefas)

- Atribuir as propriedades e estimar esforço para cada item na WBS

2. Avaliação das Tarefas:

- Definir a prioridade das tarefas com base no valor comercial, dependências e viabilidade
- Identificar e mitigar os riscos associados a cada tarefa
- Estabelecer critérios de aceitação para cada tarefa

3. Cronograma:

- Crie um cronograma de projeto com datas claras de início e fim para cada fase
- Definir as tarefas que serão concluídas durante cada fase
- Agendar reuniões regulares, como revisões de projeto, relatórios de progresso e reuniões do conselho de controle de mudanças

4. Métricas:

- Definir indicadores-chave de desempenho (KPIs) para acompanhar o progresso e medir o sucesso
- Medir e relatar os marcos do projeto, entregas e outras métricas relevantes
- Usar métricas para identificar e resolver quaisquer problemas ou obstáculos

Além disso, vale a pena notar que no modelo em cascata, uma vez que uma fase é concluída, esta fase é considerada "definida" e qualquer alteração nos requisitos, cronograma ou orçamento só podem ser feitas por meio de um processo formalizado de controle de alterações.

### 14.2.3 Principais diferenças

**14.2.3.1 Estrutura Analítica do Trabalho (WBS)** A Estrutura Analítica do Trabalho das abordagens Agile e em cascata tem algumas diferenças importantes em termos de estrutura e de uso.

WBS ágil:

- Agile normalmente divide o projeto em pedaços menores, como épicos, histórias e tarefas.
- A WBS ágil é usada para definir o escopo e os objetivos do projeto e para atribuir propriedade e estimar o esforço de cada item da WBS.
- A WBS ágil é flexível e pode ser ajustado ao longo do projeto conforme os requisitos mudam.
- As equipes ágeis usarão a WBS para planear e organizar o trabalho de cada sprint.

WBS de cascata:

- A WBS de cascata normalmente divide o projeto em fases, entregas e tarefas.
- A WBS é usada para definir o escopo e os objetivos do projeto e para atribuir propriedade e estimar o esforço de cada item da WBS.
- A Waterfall WBS é mais rígida e menos flexível, e as mudanças só podem ser feitas por meio de um processo formal de controlo de mudanças.
- As equipes Waterfall usarão a WBS para planear e organizar o trabalho de cada fase.

#### **14.2.4 Cronograma**

O cronograma ágil e cronograma em cascata são duas abordagens diferentes para o planeamento e a gestão de projetos, e eles têm algumas diferenças importantes em termos de agendamento e cronograma.

Cronograma ágil:

- O cronograma ágil é normalmente dividido em sprints curtos, que geralmente duram de 2 a 4 semanas.
- Os sprints são usados para entregar pequenos pedaços do projeto, e o progresso é revisado e adaptado no final de cada sprint.
- O cronograma ágil é flexível e permite que mudanças e ajustes sejam feitos ao longo do projeto.
- O cronograma é focado na entrega de valor de forma incremental e é menos focado em atingir marcos específicos.
- O cronograma ágil é baseado no princípio da melhoria contínua, e espera-se que a equipe se adapte e melhore à medida que o projeto avança.

Cronograma de cascata:

- O cronograma de cascata é normalmente dividido em fases distintas, como requisitos, design, desenvolvimento, teste e implantação.
- As fases são concluídas sequencialmente, com uma fase sendo concluída antes de passar para a próxima.
- O cronograma de cascata é mais rígido e menos flexível, e as mudanças só podem ser feitas por meio de um processo formal de controlo de mudanças.
- O cronograma é focado em atingir marcos específicos e entregar o produto final.
- O cronograma de cascata é baseado no princípio de completar cada fase antes de passar para a próxima, com pouco espaço para desvios.

Em resumo, o cronograma ágil é uma abordagem iterativa e flexível, enquanto que o cronograma Waterfall é uma abordagem linear e sequencial.

#### **14.2.5 Planeamento de riscos**

O planeamento de riscos é um aspecto importante da gestão de projetos e envolve a identificação e mitigação de riscos potenciais para o projeto. O planeamento de riscos ágil e de cascata têm algumas diferenças importantes em termos de abordagem e de gestão dos riscos.

Planeamento ágil de riscos:

- As equipas ágeis usam uma abordagem contínua de gestão de riscos, e os riscos são identificados e mitigados ao longo do projeto.
- A abordagem ágil foca em identificar e abordar os riscos desde o início, em vez de esperar até uma fase posterior do projeto.
- As equipas ágeis usam uma abordagem proativa para a gestão de riscos e espera-se que se adaptem e melhorem continuamente à medida que o projeto avança.
- As equipas ágeis usam métricas como gráficos de burndown para identificar e resolver quaisquer problemas ou obstáculos.

Planeamento de Risco de Cachoeira:

- As equipas de projeto em cascata usam uma abordagem de gestão de risco sequencial, e os riscos são identificados e mitigados durante fases específicas do projeto.

- A abordagem de projeto em cascata foca na identificação e tratamento de riscos durante a fase de planeamento do projeto, em vez de esperar até uma fase posterior do projeto.
- As equipas de projeto em cascata usam uma abordagem reativa na gestão dos riscos e espera-se que sigam uma progressão estrita, com cada fase do projeto concluída antes de passar para a próxima.
- As equipas de projeto em cascata usam métricas como marcos do projeto e entregas para identificar e resolver quaisquer problemas ou obstáculos.

#### **14.2.6 Abordagem híbrida**

É possível, e até recomendado em equipas heterogéneas, realizar um planeamento de projeto informático usando uma abordagem híbrida que combina elementos das metodologias ágil e em cascata. É uma abordagem flexível que permite que os melhores aspectos de ambas as metodologias sejam utilizados para atingir os objetivos do projeto.

A seguir, apresenta-se uma visão geral de um plano de projeto de software usando uma abordagem híbrida para o desenvolvimento de uma aplicação Web.

1. Fase de início:

- Definir os objetivos e o escopo do projeto
- Levantar os requisitos das partes interessadas
- Identificar e definir as prioridades das principais funcionalidades
- Criar um plano de projeto e um orçamento de alto nível

2. Fase de planeamento:

- Criar um plano de projeto detalhado e orçamento
- Definir o cronograma do projeto, incluindo datas de início e de fim para cada fase
- Desenvolver uma estrutura analítica detalhada do trabalho (WBS)
- Identificar e definir um plano de mitigação dos riscos
- Criar um cronograma de sprint, normalmente de 2 a 4 semanas

3. Fase de desenho:

- Criar os mockups e maquetes do site
- Desenvolver um design visual e diretrizes de marca
- Definir a arquitetura de informação do site
- Agendar as revisões de projeto e relatórios de progresso

4. Fase de desenvolvimento:

- Construir o site usando metodologias de desenvolvimento ágil
- Criar e implementar planos de teste
- Agendar reuniões regulares, revisões de sprint e retrospectivas
- Integrar e testar continuamente o site

5. Fase de teste:

- Testar o site no que diz respeito à funcionalidade e usabilidade
- Identificar e corrigir quaisquer bugs ou problemas
- Agendar teste de aceitação do utilizador (UAT)

6. Fase de instalação:

- Instalar o site num ambiente live
- Realizar testes finais e garantia de qualidade
- Agendar um evento de lançamento
- Identificar e mitigar os riscos associados à instalação

7. Fase de manutenção:

- Monitorizar e manter continuamente o site
- Implementar as atualizações regulares de software e patches de segurança
- Resolver quaisquer problemas ou bugs que surgirem
- Agendar revisões regulares de desempenho e relatórios de progresso

Nota: Este é uma agenda de projeto geral para um site usando uma abordagem híbrida e pode variar com base no projeto, na equipa e no contexto específicos.