Engenharia de Software

Processos de Desenvolvimento

Luís Morgado

Instituto Superior de Engenharia de Lisboa
Área Departamental de Engenharia de Electrónica e Telecomunicações e de Computadores

Processo de desenvolvimento de software

- Processo de organização do trabalho de produção de software
- Define e organiza as actividades a realizar
- Utiliza métodos de desenvolvimento e ferramentas de suporte

- Processo

- Métodos

- Ferramentas

Processo de Desenvolvimento

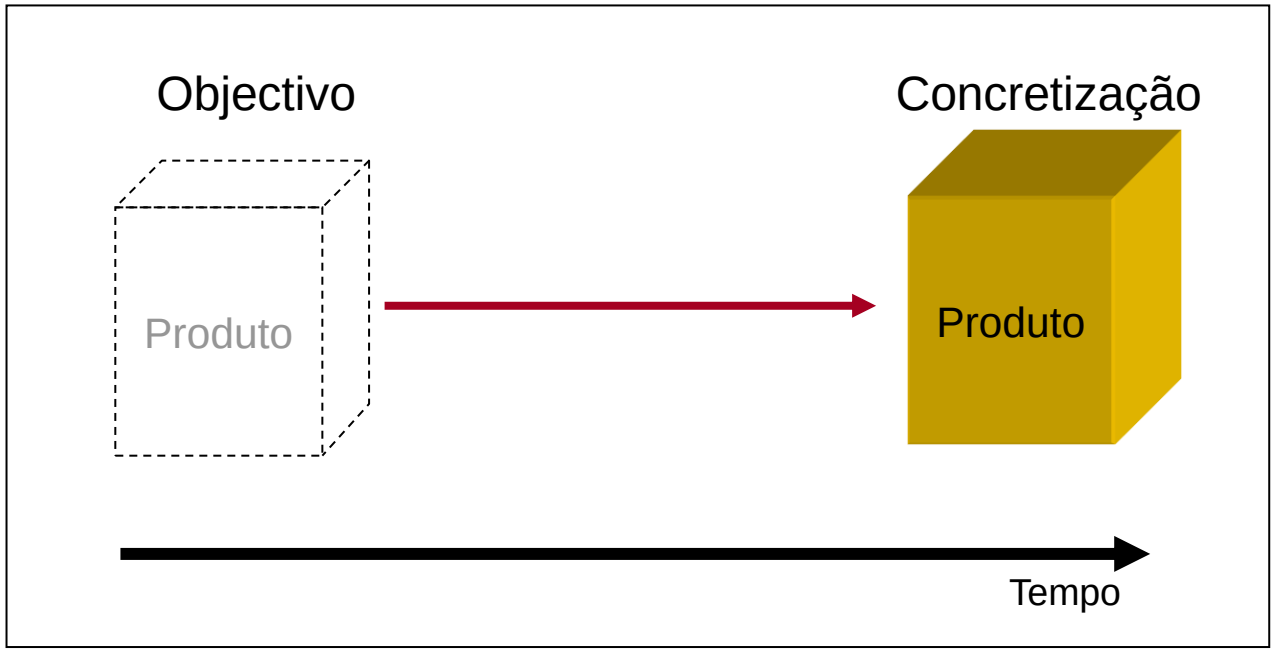
- Um processo de desenvolvimento de software é um **conjunto de actividades estruturadas que orientam e organizam o trabalho desenvolvimento de software**
- Define as *actividades* a realizar, os *artefactos* a criar e as formas de organização das equipas e do trabalho, de modo a produzir produtos de software de qualidade, de forma sistemática e organizada, de acordo com prazos e orçamentos definidos
- Um processo de desenvolvimento de software é caraterizado por várias *etapas* organizadas no tempo, que podem variar de acordo com a metodologia utilizada, mas geralmente incluem as atividades de *análise*, *concepção*, *implementação*, *verificação*, *implantação*, *manutenção*
- O processo de desenvolvimento é **determinante para o sucesso de um projecto de desenvolvimento de software**, nomeadamente, no que se refere à qualidade e ao esforço de desenvolvimento

Processo de Desenvolvimento

Atividades principais de um processo de desenvolvimento de software:

- **Análise**: identificação e definição das necessidades dos utilizadores e especificação dos requisitos do software a produzir
- **Concepção**: elaboração e especificação das diferentes partes que compõem o software a produzir e da forma como interagem, tendo como resultado a arquitectura do software
- **Implementação**: concretização da arquitectura definida sob a forma de código fonte
- **Verificação**: verificação e teste do software de modo a garantir os requisitos definidos
- **Implantação**: instalação, configuração e disponibilização para operação do software
- **Manutenção**: manutenção do software em operação e sua eventual evolução

Processo de Desenvolvimento



Processo

Um processo de desenvolvimento de software define as actividades a realizar, os artefactos a criar e as formas de organização das equipas e do trabalho, de modo a produzir produtos de software de qualidade, de forma sistemática e organizada, de acordo com prazos e orçamentos definidos

Tem início com a definição de uma *visão* do objectivo a atingir (o produto a criar)

Tem como resultado a concretização desse produto, sob a forma de uma solução que realiza a *visão* inicial

Processo de Desenvolvimento

Questões a responder:

- **O que fazer?**
  - Artefactos
- **Como fazer?**
  - Actividades
- **Quem faz?**
  - Participantes

Monitorização

Foco no objectivo

Processo

- Define os ***artefactos*** (documentação, modelos, programas, ...) que devem ser realizados
- Define as ***actividades*** envolvidas e a respectiva sequência de realização
- Guia e enquadra as tarefas dos ***participantes*** como um todo coerente
- Providencia critérios para **monitorização** e **avaliação** dos artefactos e actividades realizadas

Processo de Desenvolvimento

Um processo de desenvolvimento é realmente necessário?

Sem um **processo sistemático**, independentemente de qual seja, o desenvolvimento de sistemas, em particular de software, **ocorre sem rumo**, sendo **impossível garantir qualidade**, ou o atingir do que se pretende

- Complexidade predomina
- Confusão, anarquia, desorganização
- Esforço elevado
- Risco elevado
- Qualidade baixa

Processo de Desenvolvimento

Processo *Tentativa – Erro*

- Parte de uma ideia vaga do que é pretendido
- Ausência de método sistemático de trabalho
- Reagir, corrigir, remediar são a regra
- Caos, ineficiência
- Deficiente gestão de recursos
- Incumprimento de prazos
- Custos não controlados
- Baixa qualidade
- …

Processo sistemático

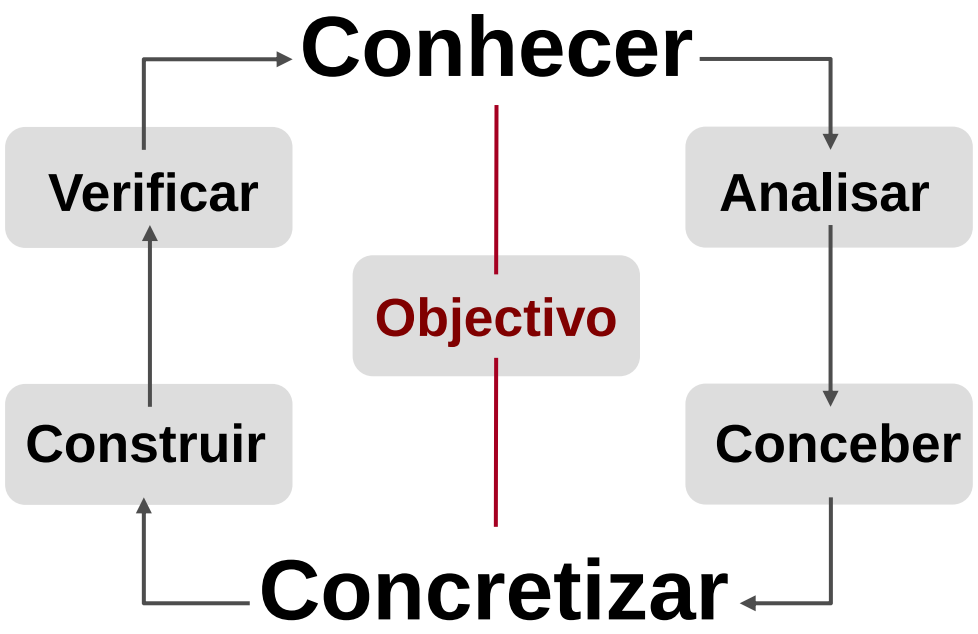
- O desenvolvimento de software deve ser realizado de forma sistemática, que envolve duas vertentes principais de trabalho:
  - **Conhecer**: obter conhecimento em dois âmbitos
    - **Domínio do problema**: âmbito do problema a resolver, envolvendo os utilizadores e outras partes interessadas
    - **Domínio da solução**: âmbito do conhecimento técnico necessário à concepção e produção da solução
  - **Concretizar**: conceber e construir a solução, tendo por base o conhecimento obtido, no âmbito do domínio do problema e do domínio da solução
  - Pode ser organizado de diferentes formas (linear, iterativa, incremental) dando origem a diferentes modelos de processos de desenvolvimento

Conhecer e concretizar

- Envolve quatro tipos de actividades principais:
  - **Analisar**: estudar o domínio do problema, de modo a identificar e definir as necessidades dos utilizadores e sistematizar o conhecimento obtido numa especificação de requisitos do software a produzir
  - **Conceber**: estudar o domínio da solução, de modo a elaborar um modelo das diferentes partes que compõem o software a produzir e da forma como interagem, sob a forma de uma arquitectura de software, nomeadamente, utilizando linguagens de modelação de software
  - **Construir**: com o conhecimento do domínio da solução, criar os artefactos que constituem a solução a produzir, nomeadamente, código fonte, de modo a realizar a arquitectura definida
  - **Verificar**: verificar e testar o software produzido tendo por base o conhecimento do domínio do problema e do domínio da solução e obtendo novo conhecimento, no sentido de corrigir deficiências ou de melhorar características do software produzido
- Os vários modelos de processos de desenvolvimento concretizam estes tipos de actividade de diferentes modos de acordo com os respectivos princípios e formas de organização

Processo de Desenvolvimento

As actividades de um processo de desenvolvimento, encadeiam-se de modo a aumentar o conhecimento da solução a produzir para atingir o objectivo definido, com esse conhecimento torna-se possível concretizar a solução



Foco no objectivo a atingir

Actividades de suporte:

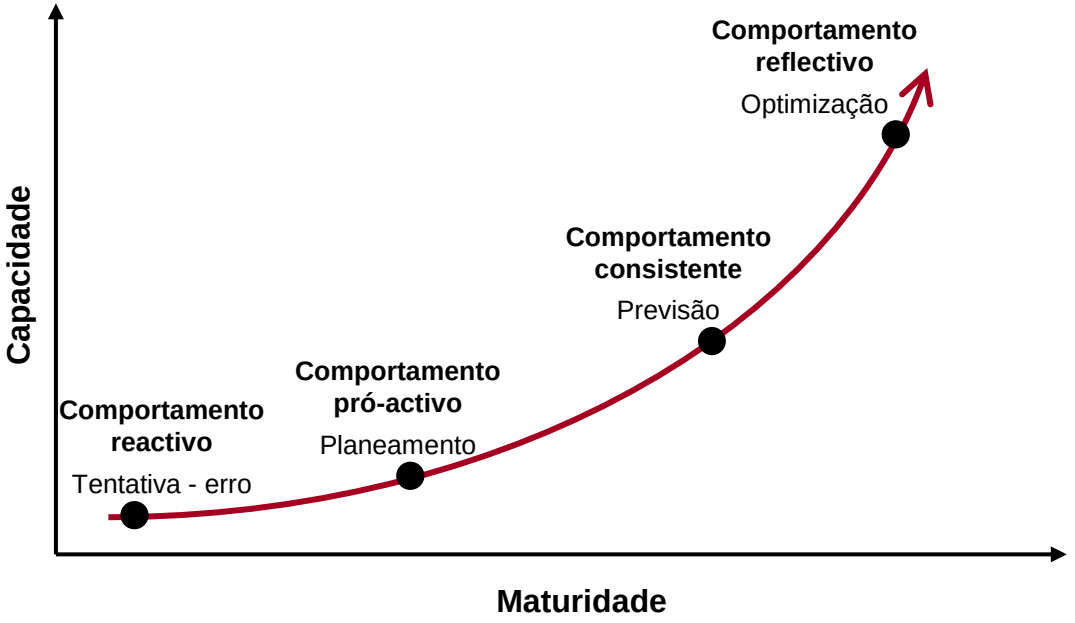
Comunicação **Planeamento** **Disponibilização**

Suporte **Operação** **Manutenção**

Processo de Desenvolvimento

Maturidade e Capacidade

- Os conceitos de *capacidade* e *maturidade*, definem dimensões de desempenho individual ou de uma organização no desenvolvimento de software, reflectindo conhecimento e competência
  - **Maturidade** refere-se à capacidade de desenvolver software com qualidade, eficiência e consistência, seguindo um conjunto de processos definidos e padronizados, produzindo resultados de forma previsível
  - **Capacidade** refere-se ao grau de eficiência e qualidade com o qual se atinge um resultado esperado

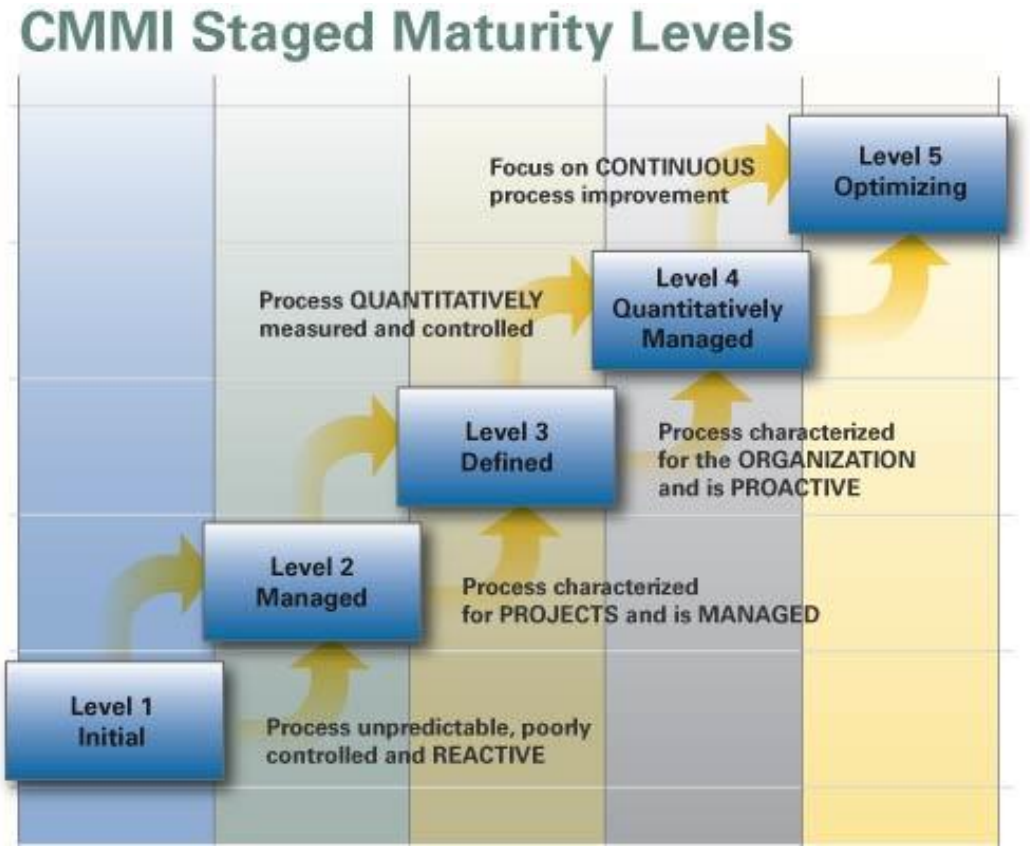


Maturidade e Capacidade

- Os conceitos de *capacidade* e *maturidade c*onstituem um modelo com base no qual o desempenho de uma organização pode ser medido e melhorado
- Existem vários modelos de capacidade e maturidade para desenvolvimento de software, nomeadamente, o modelo CMMI (*Capability Maturity Model Integration*) do *Software Engineering Institute*
- A adopção de modelos de capacidade e maturidade pode ajudar as organizações a aumentarem a qualidade e eficiência do produto final, nomeadamente, no que se refere à satisfação de requisitos de clientes ou de entidades de supervisão e regulação, bem como nos métodos de trabalho das equipas de desenvolvimento de software

Modelo de Maturidade e Capacidade

O modelo **CMMI** define diferentes níveis de maturidade, possibilitando a medição do grau de progresso atingido por uma organização na implementação de projetos de software



[Software Engineering Institute]

Modelo de Maturidade e Capacidade

Nível de maturidade 1: *Inicial*

- Processo informal, mal definido, ou por tentativa-erro
- Desempenho imprevisível
- Resultados resultam de grande esforço individual, em vez de um processo organizado, sistemático
- Dificuldade em repetir sucessos anteriores, devido à imprevisibilidade do método de trabalho
- Fracas especificações de requisitos, arquitectura e planeamento
- Elevado esforço, baixa eficiência

Modelo de Maturidade e Capacidade

Nível de maturidade 2: *Gerido*

- Planeamento e gestão de projectos baseados na experiência de projectos anteriores
- Processo planeado, executado e controlado
- O estado dos produtos em desenvolvimento é conhecido em marcos de desenvolvimento definidos e as respectivas mudanças são controladas
- Os requisitos de um projecto são geridos e as mudanças a esses requisitos são controladas tendo em conta o impacto em prazos e orçamento
- Existem auditorias independentes ao processo

Modelo de Maturidade e Capacidade

Nível de maturidade 3: *Definido*

- Processos padronizados que suportam toda a organização
- Esses processos garantem consistência da forma como os projectos são desenvolvidos em toda a organização
- Existem directrizes acerca de como os processos da organização devem ser adaptados e aplicados a cada projecto
- Existem práticas bem definidas de tomada de decisão e de gestão de risco
- Processos definidos de forma mais rigorosa

Modelo de Maturidade e Capacidade

Nível de maturidade 4:
Gerido quantitativamente

- A organização define objectivos quantificados de desempenho dos processos principais
- Processos controlados com base em métricas quantitativas, nomeadamente utilizando técnicas estatísticas
- Processos estáveis e executados dentro de limites bem definidos
- São definidos e geridos objectivos de qualidade de produtos e processos
- Desempenho previsível, com variações identificadas e as causas de variação corrigidas

Modelo de Maturidade e Capacidade

Nível de maturidade 5: *Optimização*

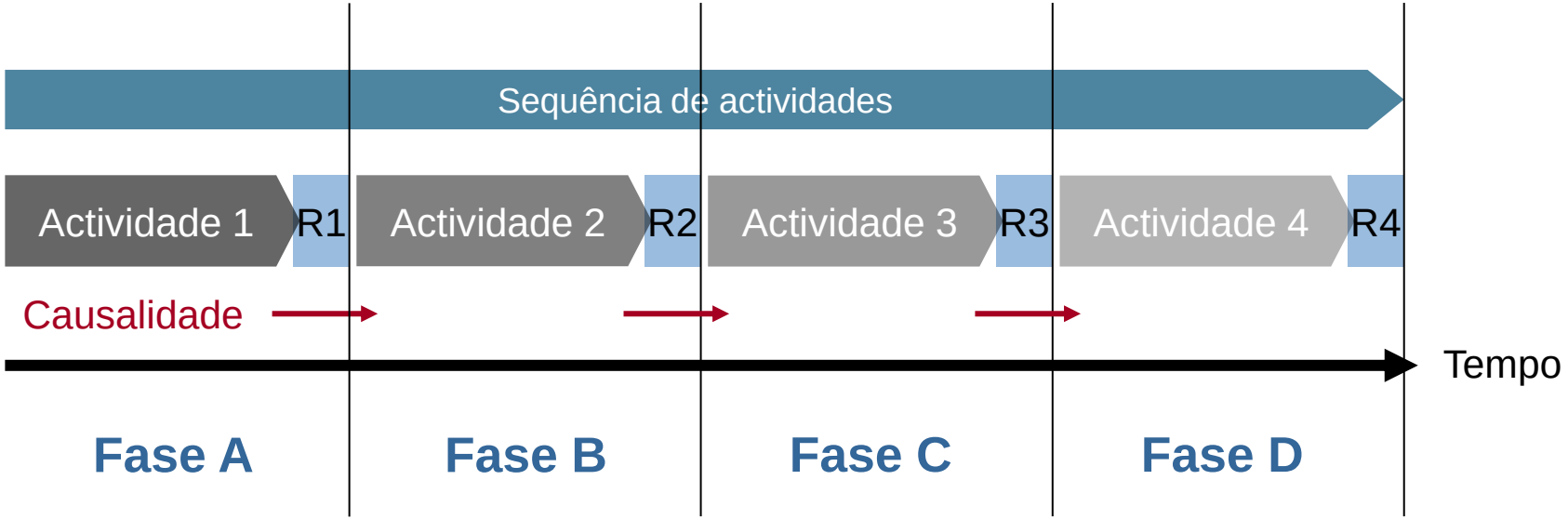
- Melhoria contínua dos processos tendo por base a compreensão de forma quantitativa das causas de variação
- A prevenção de defeitos é parte integral do ciclo de vida dos produtos
- Novas tecnologias são avaliadas e introduzidas na organização onde é adequado
- Processos melhorados incrementalmente, ou através de processos inovadores e melhorias tecnológicas

Modelo linear

- O modelo de processo de desenvolvimento linear, também designado *modelo em cascata*, devido ao encadeamento sequencial das várias fases que o constituem, é um modelo de desenvolvimento de software que segue uma abordagem linear, sequencial, onde cada fase do desenvolvimento é concluída antes de se iniciar a próxima fase
- Principais características:
  - As fases do processo são executadas numa ordem linear, sequencial, sem sobreposição de atividades
  - Cada fase do processo deve ser concluída antes do início da fase seguinte
  - Modelo adequado para projetos em que os requisitos são bem definidos e estáveis
  - Modelo menos adequado para projetos em que os requisitos mudam com frequência ou não são bem definidos, por exemplo, devido a incerteza no domínio do problema a resolver

Modelo linear

- Define um encadeamento sequencial na realização das actividades
- Organizado em fases
  - Delimitadas por marcos de desenvolvimento ("*milestones*")



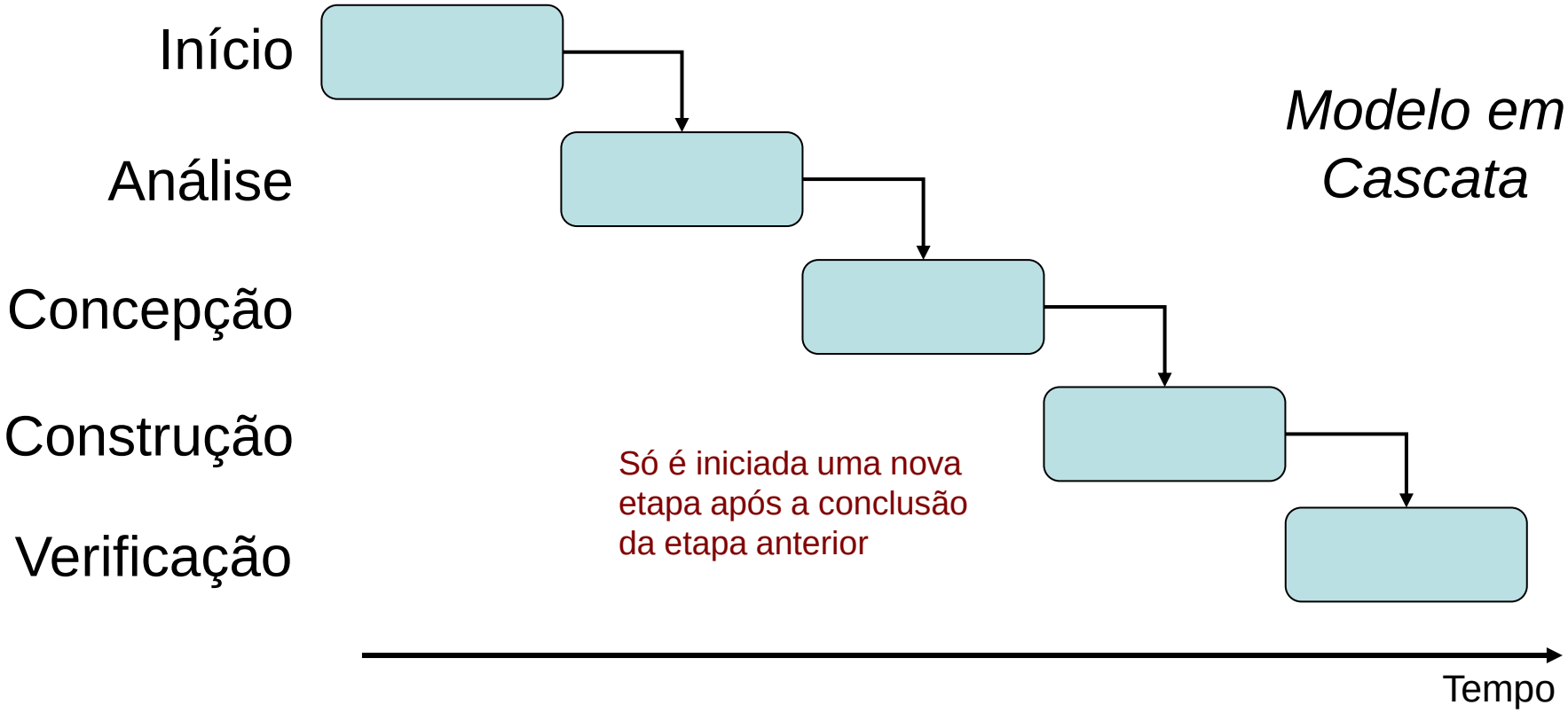
R_i – Resultado de actividade / marco de desenvolvimento (*milestone*)

Modelo Linear de Desenvolvimento

- Fases de actividade
  1. Início
     - Produz uma **descrição do problema** que define o âmbito, objectivos, prazos e orçamento para realização de uma solução para o problema
  2. Análise
     - Produz uma **definição dos requisitos**, expectativas e prioridades dos utilizadores para uma solução do problema
  3. Concepção
     - Produz a **arquitectura** e as especificações para uma solução que satisfaça os requisitos
  4. Construção
     - Produz o **sistema** que concretiza a solução de acordo com a arquitectura e as especificações definidas
  5. Verificação
     - Produz **informação de validação** referente à concretização dos requisitos definidos

Processo de Desenvolvimento

Desenvolvimento linear



Modelo em Cascata

Foco: Tarefas a realizar

Modelo em Cascata

- **Características**
  - O mais antigo e mais utilizado
  - Bem documentado e padronizado
  - Ênfase no planeamento antes da construção
  - Escalonamento linear das tarefas facilita a gestão
  - Conteúdo dos documentos e critérios de revisão bem definidos
  - Separação clara entre actividades
    - Análise / Concepção / Construção / Verificação
  - Favorece uma abordagem do geral para o específico (“*top-down*”)

Modelo em Cascata

- **Problemas**
  - Os projectos reais raramente tem um encadeamento sequencial
  - Não há concorrência na concretização das diferentes etapas
  - Não tem em conta a necessidade de contínua adaptação à mudança
  - Produto tardio

Bibliografia

[Pressman, 2003]
R. Pressman, *Software Engineering: a Practitioner's Approach,* McGraw-Hill, 2003.

[Weaver, 1948]
W. Weaver, *Science and Complexity*, American Scientist, 36: 536, 1948.

[Hitchins, 1992]
D. Hitchins, *Putting Systems to Work*, John Wiley, 1992.

[Boehm, 2000]
B. Boehm, *Software Cost Estimation with COCOMO II*, 2000.

[DAUP, 2001]
Systems Engineering Fundamentals. Defense Acquisition University Press, 2001

[O'Regan, 2017]
G. O'Regan, *Concise Guide to Software Engineering: From Fundamentals to Application Methods,* Springer, 2017