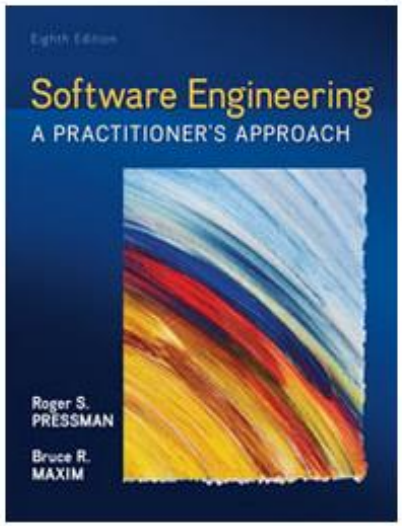
Unidade 15
Conceitos de Gerenciamento de Projeto de Software

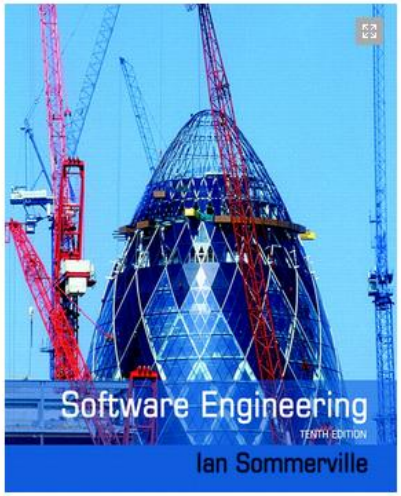
Prof. Aparecido V. de Freitas
Doutor em Engenharia
da Computação pela EPUSP

Bibliografia

- **Software Engineering – A Practitioner's Approach – Roger S. Pressman – Eight Edition – 2014**
- **Software Engineering – Ian Sommerville – 10th edition - 2015**
- Engenharia de Software – Uma abordagem profissional – Roger Pressman - McGraw Hill, Sétima Edição - 2011
- Engenharia de Software – Ian Sommerville – Nona Edição – Addison Wesley, 2007



Software Engineering: A Practitioner's Approach, 8/e



Introdução

- Gerenciamento efetivo de desenvolvimento de software tem um foco nos **4Ps**.

  - ✓ **Pessoas**
  - ✓ **Produto**
  - ✓ **Processo**
  - ✓ **Projeto**

Pessoas

- "Toda organização precisa aprimorar continuamente sua habilidade para atrair, desenvolver, motivar, organizar e reter a **força** de **trabalho** necessária para atingir os objetivos estratégicos de seus negócios". [Curtis, 2001]

- **Práticas-chave** para o pessoal de Software:

- ✓ **Formação de equipe**
- ✓ **Comunicação**
- ✓ **Ambiente de Trabalho**
- ✓ **Gerenciamento do Desempenho**
- ✓ **Treinamento**
- ✓ **Análise de Competência**
- ✓ **Desenvolvimento de Carreira**

Produto

- Antes de se traçar um plano de projeto, deve-se estabelecer os **objetivos** do **produto** e seu **escopo**, deve-se considerar as soluções alternativas e identificar as restrições técnicas e de gerenciamento;

- Sem tais informações, é impossível definir-se de forma razoável (e precisa) a **estimativa** de **custo**, a avaliação efetiva dos **riscos**, a análise realística das **tarefas de projeto** ou um **cronograma** gerenciável do projeto;

- Em geral, os objetivos e o escopo do produto se iniciam com a **Engenharia de Requisitos** do software.

Processo

- Um **processo de software** fornece a **metodologia** por meio da qual um plano de projeto abrangente para o desenvolvimento de software pode ser estabelecido.

- Uma quantidade de diferentes conjuntos de **atividades-tarefas**, pontos de controle, artefatos de software e pontos de garantia de qualidade permitem que as atividades metodológicas sejam adaptadas às características do projeto de software.

Projeto

- Projetos são estabelecidos para se administrar a complexidade;
- Em um estudo(*) de **250** grandes projetos de software entre 1998 e 2004, constatou-se que:

  - ✓ **25** lograram sucesso em cumprir cronograma, custos e objetivos quanto à qualidade;
  - ✓ **50** apresentaram atrasos (no mínimo 35% retardamentos sérios);
  - ✓ **175** tiveram atrasos com **retardamentos sérios** ou **não** conseguiram terminá-lo.

(*) C. Jones – Software Project Management Practices: Failure versus Success – 2004

As pessoas

- Em um estudo publicado pelo **IEEE** [Cur 88], **CIO's** de empresas de engenharia de software foram questionados sobre o **fator mais importante para o sucesso de um projeto de software**:

  - ✓ Ferramentas **não** são importantes, importantes são as **PESSOAS**;
  - ✓ O mais importante em um projeto é selecionar a **equipe**;
  - ✓ Assegurar que se possa **reunir bom pessoal** – bem como **cultivá-los** – e propiciar **ambiente** no qual os bons profissionais possam produzir.

Interessados (comprometidos)

- O processo de software (e todo o projeto de software) é formado por interessados que podem ser categorizados em um dos cinco grupos:
  - ✓ **Gerentes Seniores**: Definem os itens de negócio e exercem influência significativa no projeto;
  - ✓ **Gerentes de Projeto**: Planejam, motivam, organizam e controlam os programadores que executam o trabalho de software;
  - ✓ **Programadores**: Devem ter habilidades técnicas para o desenvolvimento do software;
  - ✓ **Clientes**: Especificam os requisitos para o software a ser desenvolvido;
  - ✓ **Usuários Finais**: Interagem com o software uma vez liberado para uso operacional.

Líderes de Equipe

- Gerenciamento de projeto é uma atividade intensiva de pessoal;
- Por essa razão, pode haver **programadores competentes** que resultam em **maus líderes de projeto**;
- Jerry Weiberg (*) sugere um modelo de liderança: (MOI)
  - ✓ **Motivação**: Habilidade para encorajar o pessoal técnico a produzir com o melhor de suas habilidades;
  - ✓ **Organização**: Habilidade para moldar os processos já existentes (ou criar novos) que irão capacitar o conceito inicial para ser traduzido em um produto final;
  - ✓ **Inovação**: Habilidade de encorajar pessoas para criar e ser criativas, mesmo quando estiverem trabalhando sob padrões estabelecidos;

(*) Edgemon J. – How to recognized it when selecting a Project Manager

Bons Gerentes de Projeto (*)

- ✓ **Solução de Problema**: **Motiva** a equipe a buscar soluções. Põe em prática lições aprendidas de outros projetos. Deve ser flexível para mudar de direção, se necessário;
- ✓ **Identidade Gerencial**: Deve assumir a responsabilidade do projeto. Confiança para **assumir o controle** quando necessário.
- ✓ **Realizações**: **Deve recompensar iniciativas** e realizações para otimizar a produtividade da equipe. Deve demonstrar por seus atos que decisões por riscos controlados não serão punidas;
- ✓ **Formação de equipe e influência**: Deve ser capaz de "**ler**" pessoas ("O corpo fala"). Deve permanecer sob controle em situações de alto estresse.

(*) Edgemon J. – How to recognized it when selecting a Project Manager

Fatores para planejar a Equipe(*)

- ✓ Grau da dificuldade do problema a ser solucionado;
- ✓ Quantidade de Pontos de Função;
- ✓ Tempo de vida da equipe;
- ✓ Grau de modularização do problema;
- ✓ Qualidade e confiabilidade do software a ser construído;
- ✓ Rigidez das datas de entrega;

(*) L. Constantine – Work Organization: Paradigms for Project Management and Organization, 1993

Para equipes de alta performance...

- ✓ Os membros da equipe devem **confiar** uns nos outros;
- ✓ A **distribuição** de habilidades deve ser adequada ao problema;
- ✓ **Estrelismos** devem ser excluídos da equipe para se manter a coesão do grupo.

(*) L. Constantine – Work Organization: Paradigms for Project Management and Organization, 1993

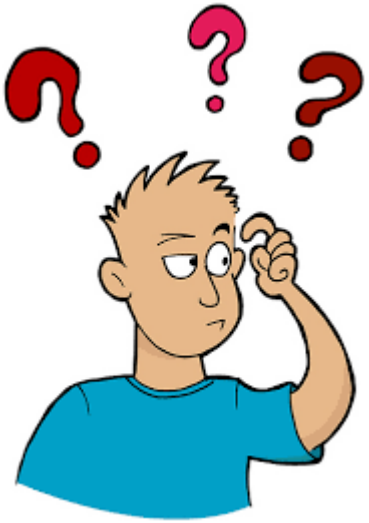
Toxidade da equipe (*)

- Há cinco fatores que fomentam um ambiente potencialmente **tóxico** da equipe:

  - ✓ **Atmosfera** de trabalho **frenética** (agitada) ;
  - ✓ Grau de **frustração** causando atrito entre os **membros** da equipe;
  - ✓ **Processo** de software fragmentado ou **pobremente** coordenado;
  - ✓ Definição **nebulosa** dos **papéis** dentro da equipe de software;
  - ✓ Contínua e repetida **exposição** a **falhas**.

(*) M. Jackman – Homeopathic Remedies for Team Toxicity, 1998

O que o Gerente de Projeto deve fazer para evitar a Toxidade da equipe ?



Atmosfera de Trabalho frenética

- **Frenético**: indivíduo que está em estado extremo de exaltação, que está inquieto ou muito agitado.

  - ✓ O gerente de projeto deve estar certo de que a **equipe** tem **acesso** a todas as **informações** necessárias para realizar o trabalho;

  - ✓ As **metas** e **objetivos** prioritários (papéis), uma vez estabelecidos, não devem ser alterados a menos que absolutamente necessários.

Grau de frustração

- Uma equipe pode evitar **frustrações** se lhes for oferecida, tanto quanto possível, **responsabilidade** para tomada de decisão;
- **Revisões Técnicas** são excelentes meios para se obter responsabilidades;
- Implementar técnicas voltadas à realimentação (**feedback**) e resolução de problemas;

Aspectos pessoais da equipe

- Pessoas são **diferentes**;
- Uns são **extrovertidos**, outros **introvertidos**;
- Uns são desenvolvem as atividades de forma **natural** e espontânea, outros requerem **intervenção** do gerente de projetos;
- Uns se sentem confortáveis em tomada de decisão baseada em algum argumento **lógico** e **ordenado**, enquanto outros baseiam-se em **intuições**;
- Uns trabalham com base em **cronogramas** detalhados e organizados, enquanto que outros preferem um ambiente mais **espontâneo**;
- Uns trabalham arduamente para conseguir que as etapas sejam concluídas bem antes do prazo (para evitar **estresse** da proximidade da data-limite), outros são energizados pela **correria** de última hora.
- O gerente de projetos deve observar que o reconhecimento das **forças humanas** é o primeiro passo em direção à criação de **equipes consistentes**.

O dilema do Projeto

Dilema do Projeto

- Um gerente de projeto de software confronta-se com um **dilema** sempre que se inicia um projeto;
- É preciso **estimativas quantitativas** e um **plano organizado**, entretanto **informações sólidas não** estão **disponíveis** ainda;
- Análise dos requisitos frequentemente, levam **semanas** ou mesmo **meses** para estarem completas;
- **Pior ainda**, os requisitos podem **mudar** durante o correr do projeto;
- Gostando-se ou não, deve-se examinar o produto e problema que se pretende solucionar logo no início do projeto;
- No mínimo**, o escopo do produto deve ser estabelecido e delimitado**.

Fonte: Pressman

Escopo de Software

- **Contexto**: Como o software a ser desenvolvido se ajusta a um sistema maior, a um produto ou contexto de negócio e quais são as restrições impostas resultantes do contexto?

- **Objetivo da informação**: Quais objetos de dados visíveis ao cliente são produzidos como saída do software? Quais objetos de dados são necessários como entrada?

- **Função e performance**: Qual a função que o software desempenha para transformar os dados de entrada em dados de saída? Há quaisquer características especiais de desempenho a ser acessada?

Escopo de Software

- **Não** deve apresentar **ambiguidades**;
- Deve ser **compreensível** tanto no nível gerencial quanto no técnico;
- Devem-se levantar **dados quantitativos** tais como: número de usuários simultâneos, ambiente-alvo, tempo máximo de resposta, etc.
- Devem-se levantar **restrições/limitações** tais como: **prazo de entrega do projeto**, **linguagem de programação a ser desenvolvida**, etc.

Decomposição do Problema

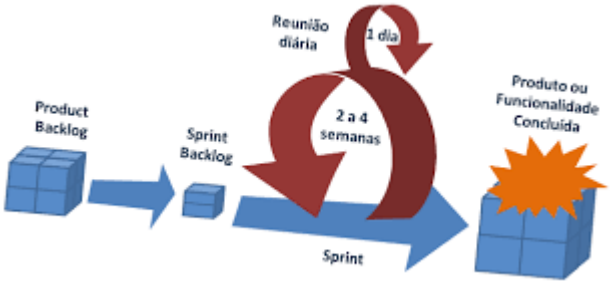
- Durante a atividade de escopo, aplica-se a decomposição em duas áreas vitais:
  - ✓ **Funcionalidade** e no conteúdo (**informação**) que deve ser entregue;
  - ✓ **Processo** que será utilizado para entregar o software.
- Essa é a **estratégia** que se deve aplicar no início do **planejamento** do projeto;
- **Estimativas**, **custos** e **cronogramas** podem ser levantados com base na decomposição das funções e dos objetos de dados manuseados pelo software.

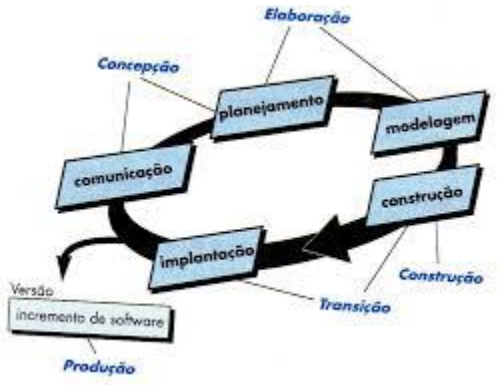
O Processo

- As atividades de modelagem que caracterizam o processo de software são **aplicáveis** a todos os projetos de software;
- A **dificuldade** está em **selecionar** o modelo de processo apropriado ao software a ser desenvolvido pela equipe;
- Quando um modelo de processo é selecionado, a equipe define o planejamento preliminar do projeto com base no conjunto de atividades estabelecidas no modelo do processo;
- Com base nas atividades genéricas do processo adotado, uma matriz com as **funções principais** do produto serão relacionadas às **atividades** do modelo do processo.

A escolha do Processo

- Um projeto relativamente pequeno poderia ser mais bem realizado por meio da abordagem sequencial **linear** (modelo cascata);
- Similarmente, projetos com outras características, por exemplo: requisitos indefinidos, tecnologias avançadas recentes, clientes difíceis, etc. podem ser conduzir à seleção de modelos **iterativos** ou **incrementais** (RUP ou modelos ágeis);





O projeto

- John Reel [Ree 99], apresenta **10** sinais de que o projeto de software está em **perigo**:

- ✓ O pessoal de software não compreende as **necessidades** de seus clientes;
- ✓ O **escopo** do produto está parcialmente definido;
- ✓ As alterações são **mal** gerenciadas;
- ✓ A **tecnologia** escolhida muda;
- ✓ As necessidades de **negócio** são mal definidas;
- ✓ Os **prazos** não são realistas;
- ✓ Os **usuários** mostram-se resistentes;
- ✓ O **patrocínio** é perdido (ou nunca existiu);
- ✓ Faltam **profissionais** habilitados ou capacitados para o projeto;
- ✓ **Gerentes** de projeto evitam práticas e lições aprendidas.

Postura do Gerente de Projeto

- John Reel [Ree 99], sugere uma abordagem de **5** partes para projetos de software:

- ✓ Compreender bem o **problema do usuário** e estabelecer expectativas e objetivos realísticos para todos os envolvidos no projeto. Isso é reforçado pela equipe correta;
- ✓ Manter a **velocidade do projeto**. Gerente do projeto deve fornecer incentivos para que a rotatividade de pessoal fixe-se num nível absolutamente mínimo;
- ✓ O projeto deve ser rastreado e mapeado com os artefatos produzidos e aprovados por meio de revisões técnicas como parte da **garantia de qualidade**;
- ✓ Manter a **simplicidade**. Sempre que possível utilize componentes de software existentes;
- ✓ Fazer sempre análise post-mortem. Estabelecer um mecanismo consistente para extrair **aprendizados** de cada projeto.

Cronograma de Projetos de Software

Prof. Aparecido V. de Freitas
Doutor em Engenharia
da Computação pela EPUSP
aparecidovfreitas@gmail.com

Por que projetos de software têm atrasos ?

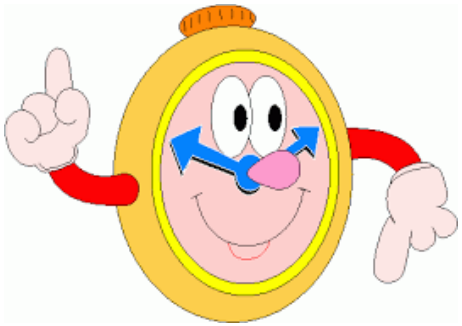
Atrasos na entrega do Software

- Prazos de entrega **não** realísticos;
- Alterações nos **requisitos** do cliente não refletidas no cronograma do projeto;
- **Subestimativa** do esforço e/ou recursos necessários ao projeto;
- **Riscos** previsíveis e/ou não previsíveis não considerados no início do projeto;
- Dificuldades **técnicas** não previstas;
- Dificuldades **humanas** não previstas;
- Falhas de **comunicação** entre pessoal de projeto;
- Falta de **ações corretivas** para na detecção de atrasos de cronograma.

Prazos

- Se as melhores estimativas indicam que o prazo de entrega não é realístico, um **gerente de projeto competente** deverá "proteger sua equipe contra pressões indevidas sobre o cronograma..."

- " E enviar a pressão de volta para aqueles que a originaram." [Page Jones,85]



O que fazer?

- Uma empresa de software recebeu a incumbência de produzir um software em **9** meses;
- Após estimativa e análise de riscos, chegou-se a conclusão de que o software, da maneira como foi solicitado, levará **14** meses para ser desenvolvido com a equipe disponível.

O que fazer?

- **Alternativa** 1: Pedir ao cliente que a data de entrega seja alterada...
- **Alternativa** 2: Declinar o projeto...

O que fazer?

- **Alternativa** 1: Pedir ao cliente que a data de entrega seja alterada... **Cliente pode não concordar com a alteração da data, por pressões de Marketing Externo...**
- **Alternativa** 2: Declinar o projeto... **Sendo uma empresa de software, essa alternativa profissionalmente não é viável e aconselhável...**

O que fazer?

Negociação do Projeto

- Reunir-se com o cliente e, usando a estimativa detalhada, justificar o **porquê** do prazo de entrega imposto ser **impraticável**;

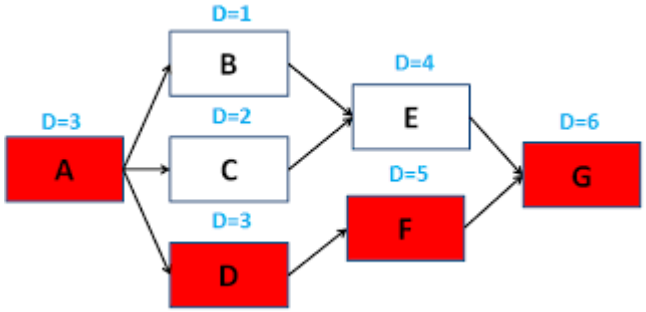
- Mas, propor uma estratégia de desenvolvimento **incremental**; Priorizar funcionalidades mais críticas para serem entregues dentro do prazo (9 meses) e demais funções em prazos mais adequados (14 meses);

Cronograma de Projeto

- Independentemente do tamanho do projeto, algumas tarefas do projeto podem estar no "**caminho crítico**". Se essas tarefas atrasarem, o prazo da entrega do projeto inteiro é ameaçado;

- Cabe ao **gerente de projetos**, definir todas as tarefas, criar uma rede que mostre suas interdependências, identificar as tarefas críticas dentro da rede e acompanhar o progresso para controle do projeto;

- **Cronograma de projeto de software** é uma atividade que distribui o esforço por toda a duração planejada do projeto.



Cronograma de Projeto – Princípios Básicos

- **Divisão do Trabalho**. Projeto deve ser dividido em uma série de atividades e tarefas gerenciáveis;
- **Interdependência**. Algumas atividades devem ocorrer em sequência, outras em paralelo;
- **Alocação de tempo**: À cada atividade, deve ser alocado um certo número de unidades de trabalho (pessoas-dias), além de uma data de início e uma data de término;
- **Validação de esforço**: Foi alocado mais esforço do que pessoas disponíveis para fazer o trabalho?
- **Definição de responsabilidades**;
- **Definição de resultados**. Que artefato deve ser gerado pela atividade?
- **Definição de pontos de controle** (milestones). Um ponto de controle é atingido quando um ou mais artefatos teve sua qualidade examinada e foi aprovado.

Relação entre pessoas e esforço

- Em um pequeno projeto de software, uma **única** pessoa pode levantar os requisitos, fazer o projeto, gerar o código e realizar os testes;
- Mas, para grandes projetos, um time de desenvolvimento é necessário;
- É praticamente impossível, um projeto de **dez pessoas-ano** ser executado por **um só indivíduo** trabalhando por **10 anos** !

Em atrasos de projeto, basta acrescentar pessoas nas últimas fases do projeto?

Atrasos em projetos

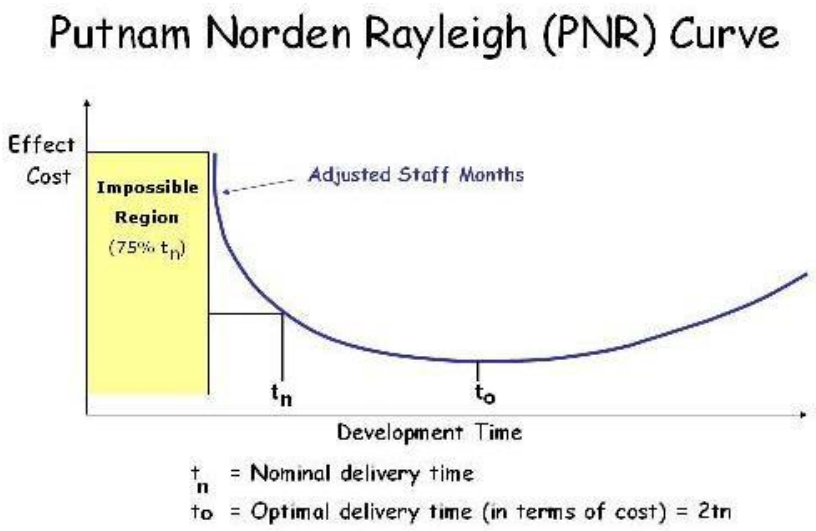
- Infelizmente, acrescentar pessoas nas últimas fases do projeto, muitas vezes tem um **efeito prejudicial**, fazendo o cronograma se **arrastar** ainda mais;

- Os profissionais inclusos no projeto, precisam **aprender** sobre o software e os encarregados de ensiná-los são os mesmos que estavam trabalhando no projeto;

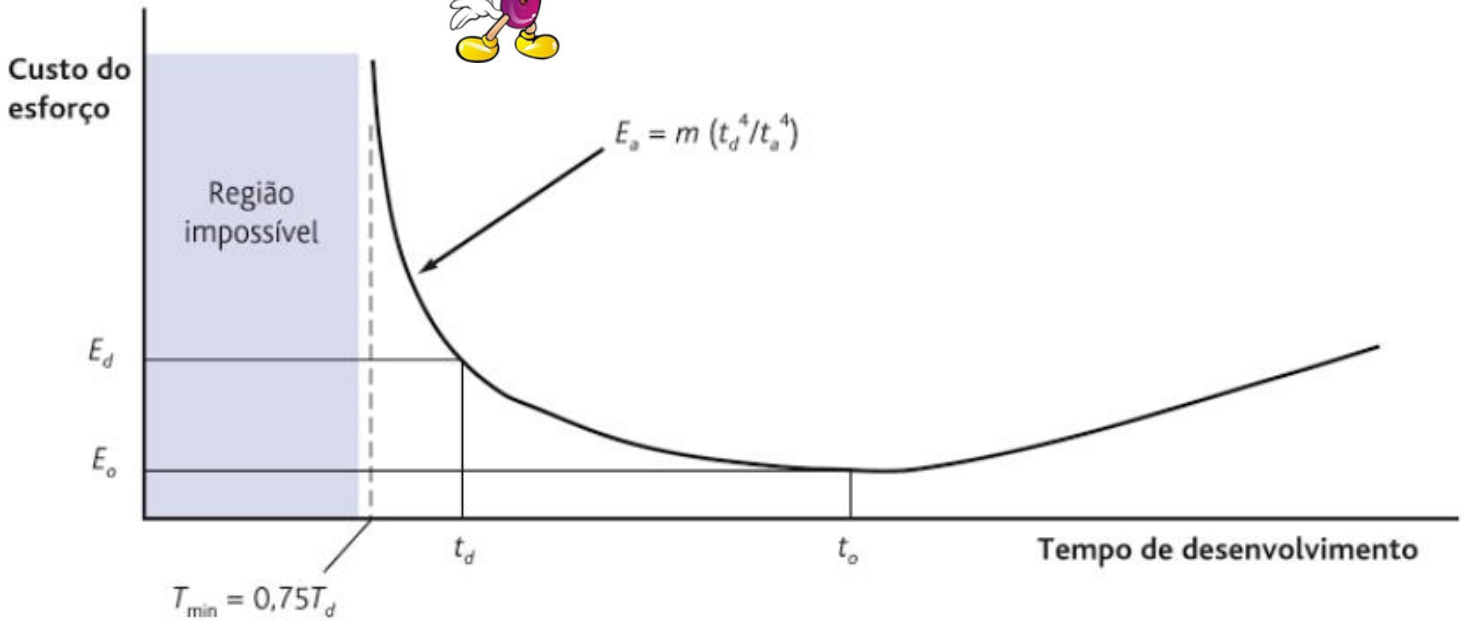
- Enquanto explicam, nada é feito, e o projeto torna-se ainda **mais atrasado**!!!

Cronogramas são elásticos

- Dados empíricos e análises teóricas, têm demonstrado que os cronogramas de projeto são **elásticos**;
- É possível **abreviar** uma data de conclusão desejada para um projeto (acrescentando recursos) até certo ponto.
- É possível também estender a data de conclusão de um projeto, reduzindo o número de recursos;
- A curva **PNR – Putnam Norden Rayleigh**, fornece uma indicação da relação entre esforço e prazo de entrega para um projeto de software.



Curva PNR – Putnam-Norden-Rayleigh



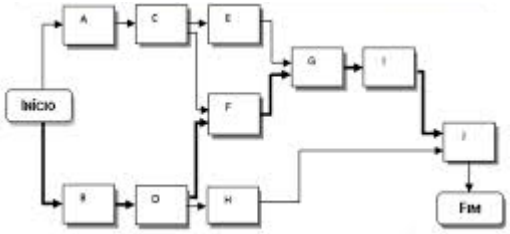
- A curva PNR indica um valor mínimo t_0, que representa o custo mínimo para a entrega (o prazo de entrega que resultará no trabalho mínimo dispendido;

- Quando nos movemos para a esquerda de t_0 (quanto tentamos acelerar a entrega), a curva não sobre linearmente.

Distribuição do Esforço

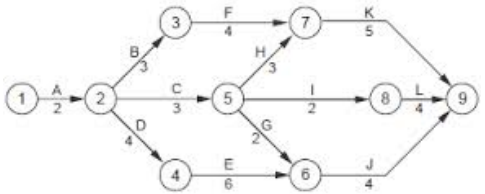
- ✓ Uma distribuição recomendada do trabalho durante o processo de software é conhecida como regra **40**-**20**-**40**;
- ✓ **40**% de todo o esforço é alocado na **análise preliminar** e **projeto**;
- ✓ **20**% de todo o esforço é alocado à **construção**;
- ✓ **40**% de todo o esforço é aplicado ao **teste**;
- ✓ Essa distribuição do esforço é apenas utilizada como um guia;
- ✓ A criticidade do software muitas vezes determina o volume de teste necessário. Se o software estiver relacionado com vidas humanas, as porcentagens podem ser tipicamente **mais altas**.

Definindo uma rede de tarefas



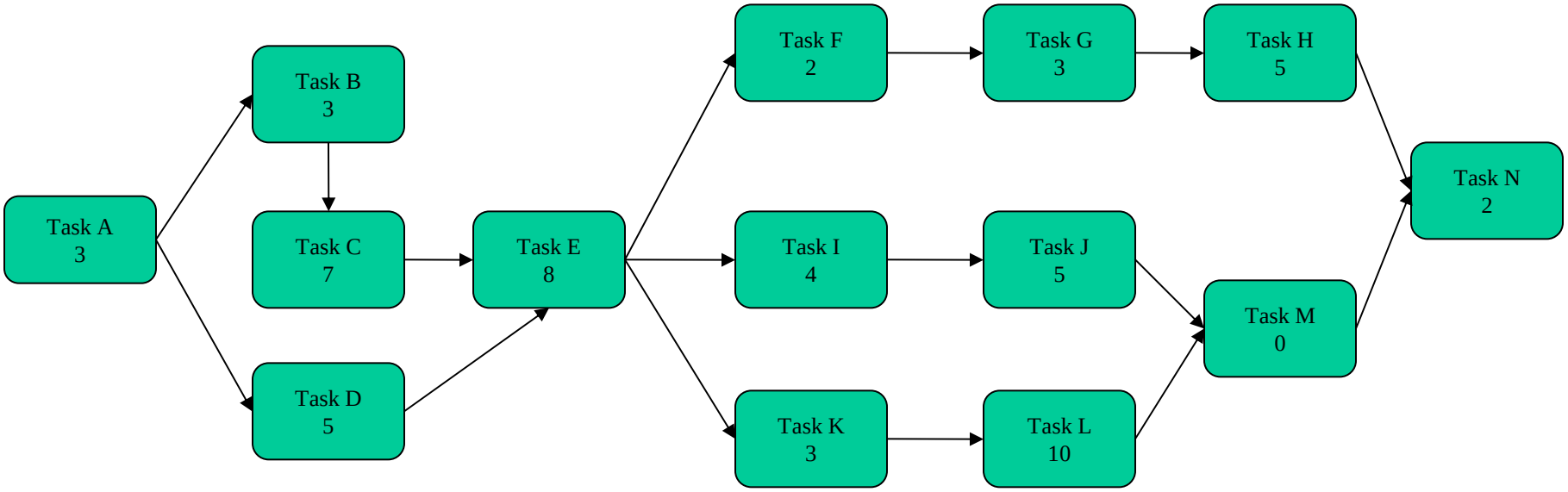
- ✓ As tarefas individuais têm interdependências baseadas em sua sequência;
- ✓ Quando há mais de uma pessoa envolvida em um projeto de engenharia de software, é provável que as tarefas sejam executadas em paralelo;
- ✓ Uma **rede de tarefas**, também chamada de **rede de atividades**, é uma representação gráfica do fluxo de tarefas de um projeto.

Cronograma



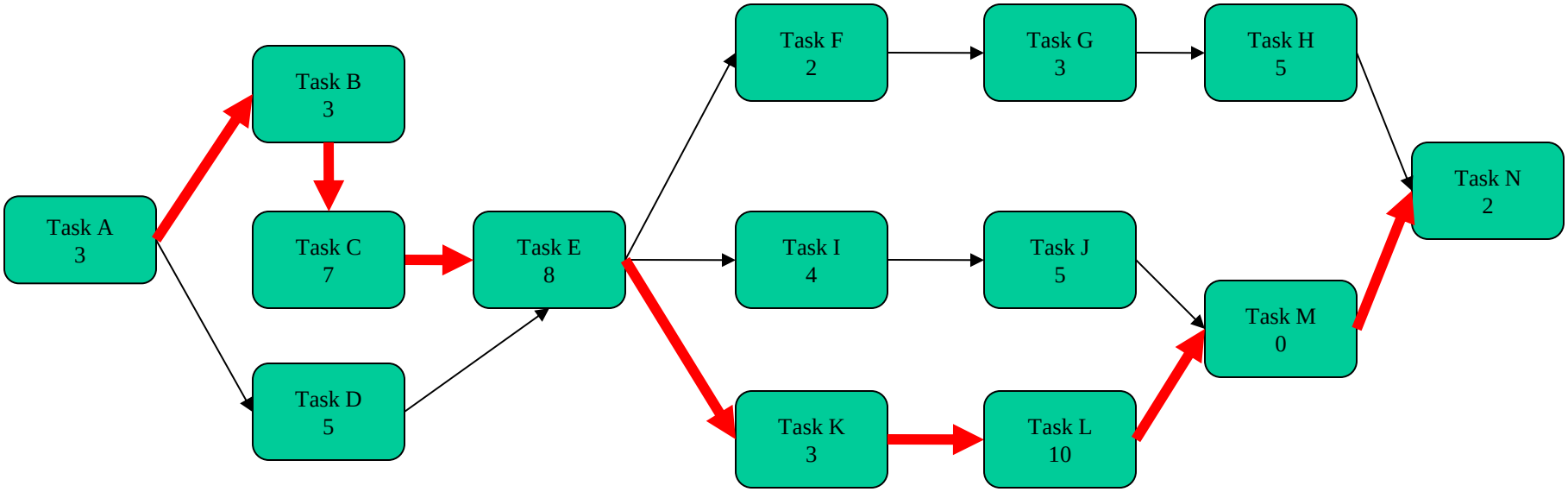
- O cronograma de um projeto de software não difere muito do cronograma de qualquer esforço de engenharia multitarefa;
- As técnicas **PERT** (Program Evaluation and Review Techique) e **CPM** (Critical Path Method) são usualmente aplicados ao desenvolvimento de software;
- Tanto **PERT** quanto **CPM** fornecem ferramentas quantitativas que permitem:
  - Determinar o **caminho crítico** – a cadeia de tarefas que determinam a duração do projeto;
  - Estabelecer estimativas de tempo "**mais prováveis**" para tarefas;
  - Calcular "**tempos-limite**" que definem uma "janela de tempo" para um tarefa particular.

Exemplo – Rede de Tarefas



Qual o caminho crítico e quais tarefas estão nele?

Exemplo – Rede de Tarefas



Caminho Crítico: A-B-C-E-K-L-M-N

Gráfico de Gantt

- ✓ Tarefas do projeto são listadas na coluna da esquerda;
- ✓ Próximas colunas indicam: duração, data de início, data de término, dependências, etc;
- ✓ Barras horizontais indicam a duração de cada tarefa;
- ✓ Ocorrência de múltiplas barras horizontais indicam concorrência de tarefas;
- ✓ Um losango na área de calendário indica um marco do projeto, com duração zero. (milestone)

						Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct
Task #	**Task Name**	**Duration**	**Start**	**Finish**	**Pred.**										
1	**Task A**	**2 months**	**1/1**	**2/28**	**None**										
2	**Marco N**	**0**	**3/1**	**3/1**	**1**										

Exercício

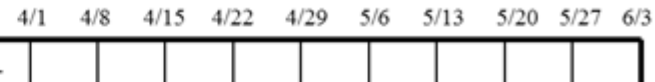
Timeline chart:

Task #	Task Name	Duration	Start	Finish	Pred.	4/1	4/8	4/15	4/22	4/29	5/6	5/13	5/20	5/27	6/3
A	Establish increments	3	4/1		None										
B	Analyze Inc One	3			A										
C	Design Inc One	8			B										
D	Code Inc One	7			C										
E	Test Inc One	10			D										
F	Install Inc One	5			E										
G	Analyze Inc Two	7			A, B										
H	Design Inc Two	5			G										
I	Code Inc Two	4			H										
J	Test Inc Two	6			E, I										
K	Install Inc Two	2			J										
L	Close out project	2			F, K										

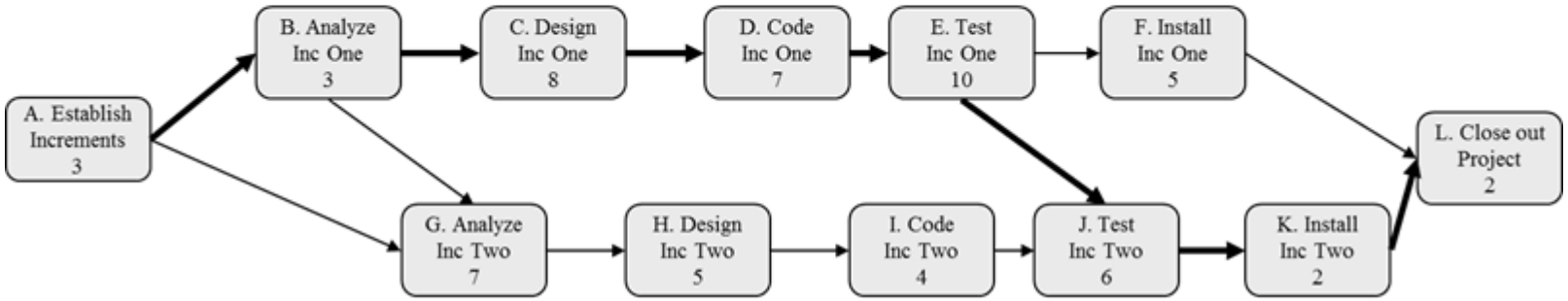
Qual a rede de tarefas e qual o caminho crítico?

Timeline chart:

Task #	Task Name	Duration	Start	Finish	Pred.
A	Establish increments	3	4/1	4/3	None
B	Analyze Inc One	3	4/4	4/6	A
C	Design Inc One	8	4/7	4/14	B
D	Code Inc One	7	4/15	4/21	C
E	Test Inc One	10	4/22	5/1	D
F	Install Inc One	5	5/2	5/6	E
G	Analyze Inc Two	7	4/7	4/13	A, B
H	Design Inc Two	5	4/14	4/18	G
I	Code Inc Two	4	4/19	4/22	H
J	Test Inc Two	6	5/2	5/7	E, I
K	Install Inc Two	2	5/8	5/9	J
L	Close out project	2	5/10	5/11	F, K

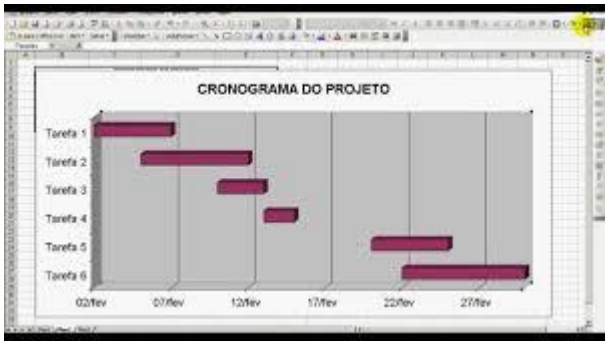


Task network and the critical path: A-B-C-D-E-J-K-L



Controle do Projeto

- ✓ O **cronograma** do projeto torna-se um **roteiro** que define as tarefas e pontos de controle a serem **acompanhados** e **controlados** pelo **gerente** do **projeto**, à medida em que o projeto avança;



Como o gerente de projetos deve fazer o acompanhamento do projeto?

Acompanhamento do Projeto feitas pelo Gerente do Projeto

- ✓ **Reuniões periódicas**, conduzidas pelo gerente do projeto, no qual cada membro da equipe relata o progresso e problemas;
- ✓ **Avaliação** dos resultados de todas as revisões feitas durante o processo de Engenharia de Software;
- ✓ Verificação se os **pontos de controle formais** (milestones) foram atingidos na data programada;
- ✓ Comparação da data de início **real** de cada tarefa com a data de início **programada**;
- ✓ **Reunião informal** com os profissionais para avaliação subjetiva do progresso e problemas previstos.

Jan Feb Mar Apr May Jun Jul Aug Sep Oct

Task #	Task Name	Duration	Start	Finish	Pred.
1	Task A	2 months	1/1	2/28	None
2	Marco N	0	3/1	3/1	1

Como o gerente de projetos deve fazer o controle do projeto?

Controle do Projeto feitas pelo Gerente do Projeto

- ✓ Se tudo está indo **bem**, o controle é **fácil**. Basta **acompanhar** o projeto;
- ✓ Quando ocorrem **problemas**, deve-se inicialmente diagnosticá-lo;
- ✓ **Recursos adicionais** podem ser alocados na área problemática;
- ✓ O **pessoal** envolvido pode ser **realocado**;
- ✓ **Cronograma** do projeto pode ser **redefinido**;

Jan Feb Mar Apr May Jun Jul Aug Sep Oct

Task #	Task Name	Duration	Start	Finish	Pred.
1	Task A	2 months	1/1	2/28	None
2	Marco N	0	3/1	3/1	1

E quando há severas pressões de prazo de entrega?

Técnica Time-boxing

- ✓ Quando uma tarefa chega ao limite de sua caixa de tempo, é provável que 90% da tarefa tenha sido completada. Nesse caso, o trabalho é interrompido e inicia-se a próxima tarefa;
- ✓ Os 10% restantes, embora importantes, podem ser adiados até o próximo incremento ou serem concluídos mais tarde, se necessário;
- ✓ A ideia da técnica time-boxing é: "Ao invés de ficar 'preso' em uma tarefa, o projeto prossegue em direção à data de entrega".

						Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct
Task #	Task Name	Duration	Start	Finish	Pred.										
1	Task A	2 months	1/1	2/28	None										
2	Marco N	0	3/1	3/1	1										

Ditado popular: Os primeiros **90**% do sistema tomam **90**% do tempo. Os **10**% restantes tomam também **90**% do tempo.