Módulo 4

Engenharia de Software

Lição 7

Introdução ao Gerenciamento do Projeto de Software

Versão 1.0 - Jul/2007

Autor
Ma. Rowena C. Solamo

Equipe
Jaqueline Antonio
Naveen Asrani
Doris Chen
Oliver de Guzman
Rommel Feria
John Paul Petines
Sang Shin
Raghavan Srinivas
Matthew Thompson
Daniel Villafuerte

Necessidades para os Exercícios

Sistemas Operacionais Suportados

NetBeans IDE 5.5 para os seguintes sistemas operacionais:

- Microsoft Windows XP Profissional SP2 ou superior
- Mac OS X 10.4.5 ou superior
- Red Hat Fedora Core 3
- Solaris™ 10 Operating System (SPARC® e x86/x64 Platform Edition)

NetBeans Enterprise Pack, poderá ser executado nas seguintes plataformas:

- Microsoft Windows 2000 Profissional SP4
- Solaris™ 8 OS (SPARC e x86/x64 Platform Edition) e Solaris 9 OS (SPARC e x86/x64 Platform Edition)
- Várias outras distribuições Linux

Configuração Mínima de Hardware

Nota: IDE NetBeans com resolução de tela em 1024x768 pixel

Sistema Operacional	Processador	Memória	HD Livre
Microsoft Windows	500 MHz Intel Pentium III workstation ou equivalente	512 MB	850 MB
Linux	500 MHz Intel Pentium III workstation ou equivalente	512 MB	450 MB
Solaris OS (SPARC)	UltraSPARC II 450 MHz	512 MB	450 MB
Solaris OS (x86/x64 Platform Edition)	AMD Opteron 100 Série 1.8 GHz	512 MB	450 MB
Mac OS X	PowerPC G4	512 MB	450 MB

Configuração Recomendada de Hardware

Sistema Operacional	Processador	Memória	HD Livre
Microsoft Windows	1.4 GHz Intel Pentium III workstation ou equivalente	1 GB	1 GB
Linux	1.4 GHz Intel Pentium III workstation ou equivalente	1 GB	850 MB
Solaris OS (SPARC)	UltraSPARC IIIi 1 GHz	1 GB	850 MB
Solaris OS (x86/x64 Platform Edition)	AMD Opteron 100 Series 1.8 GHz	1 GB	850 MB
Mac OS X	PowerPC G5	1 GB	850 MB

Requerimentos de Software

NetBeans Enterprise Pack 5.5 executando sobre Java 2 Platform Standard Edition Development Kit 5.0 ou superior (JDK 5.0, versão 1.5.0_01 ou superior), contemplando a Java Runtime Environment, ferramentas de desenvolvimento para compilar, depurar, e executar aplicações escritas em linguagem Java. Sun Java System Application Server Platform Edition 9.

- Para **Solaris**, **Windows**, e **Linux**, os arquivos da JDK podem ser obtidos para sua plataforma em http://java.sun.com/j2se/1.5.0/download.html
- Para **Mac OS X**, Java 2 Plataform Standard Edition (J2SE) 5.0 Release 4, pode ser obtida diretamente da Apple's Developer Connection, no endereço: http://developer.apple.com/java (é necessário registrar o download da JDK).

Para mais informações: http://www.netbeans.org/community/releases/55/relnotes.html

Colaboradores que auxiliaram no processo de tradução e revisão

Aécio Júnior
Alexandre Mori
Alexis da Rocha Silva
Allan Souza Nunes
Allan Wojcik da Silva
Anderson Moreira Paiva
Anna Carolina Ferreira da Rocha
Antonio Jose R. Alves Ramos
Aurélio Soares Neto
Bruno da Silva Bonfim
Carlos Fernando Gonçalves
Daniel Noto Paiva
Denis Mitsuo Nakasaki
Fábio Bombonato
Fabrício Ribeiro Brigagão
Francisco das Chagas
Frederico Dubiel
Jacqueline Susann Barbosa
João Vianney Barrozo Costa
Kleberth Bezerra G. dos Santos
Kefreen Ryenz Batista Lacerda
Leonardo Ribas Segala
Lucas Vinícius Bibiano Thomé
Luciana Rocha de Oliveira
Luiz Fernandes de Oliveira Junior
Marco Aurélio Martins Bessa
Maria Carolina Ferreira da Silva
Massimiliano Giroldi
Mauro Cardoso Mortoni
Mauro Regis de Sousa Lima
Paulo Afonso Corrêa
Paulo Oliveira Sampaio Reis
Ronie Dotzlaw
Seire Pareja
Sergio Terzella
Thiago Magela Rodrigues Dias
Vanessa dos Santos Almeida
Wagner Eliezer Rancoletta

Auxiliadores especiais

Revisão Geral do texto para os seguintes Países:

- **Brasil** – Tiago Flach
- **Guiné Bissau** – Alfredo Cá, Bunene Sisse e Buon Olossato Quebi – ONG Asas de Socorro

Coordenação do DFJUG

- **Daniel deOliveira** – JUGLeader responsável pelos acordos de parcerias
- **Luci Campos -** Idealizadora do DFJUG responsável pelo apoio social
- **Fernando Anselmo -** Coordenador responsável pelo processo de tradução e revisão, disponibilização dos materiais e inserção de novos módulos
- **Rodrigo Nunes -** Coordenador responsável pela parte multimídia
- **Sérgio Gomes Veloso -** Coordenador responsável pelo ambiente JEDI™ (Moodle)

Agradecimento Especial

John Paul Petines – Criador da Iniciativa JEDI™
Rommel Feria – Criador da Iniciativa JEDI™

1. Objetivos

O desenvolvimento de software para computador envolve uma variedade de fases, atividades e tarefas. Algumas dessas atividades são realizadas iterativamente até que o software fique pronto e entregue aos usuários finais para sua utilização.

Entretanto, antes que o desenvolvimento possa começar, usuários e desenvolvedores precisam estimar qual será a duração do projeto e quanto ele irá custar. Não é intenção deste capítulo discutir detalhadamente o gerenciamento do projeto de software. Isto está reservado para o Curso de Gerenciamento do Projeto de Software. A discussão nesta lição será realizada em torno dos conceitos de gerenciamento de projeto que auxiliarão a gestão do gerenciamento de projetos de software baseado em classes.

Ao final desta lição, o estudante será capaz de:

- Conhecer conceitos de gerenciamento básico de projeto de software
- Entender definição e identificação do problema
- Saber como organizar o projeto
- Saber como agendar atividades do projeto
- Saber como alocar recursos para o projeto
- Aprender métricas de software
- Saber calcular estimativas de custo e esforço para um projeto
- Entender gerenciamento de risco
- Entender gerenciamento de configuração de software

2. Gerenciamento do Projeto de Software

Gerenciamento do projeto de software é definido como o processo de gerenciar, alocar e disponibilizar recursos para o desenvolvimento do software que atenda aos requisitos. É a integração sistemática da técnica, recursos humanos e financeiros para atingir metas e objetivos do desenvolvimento de software de uma maneira eficiente e vantajosa. A Figura 1 mostra o processo de gerenciamento de projeto.

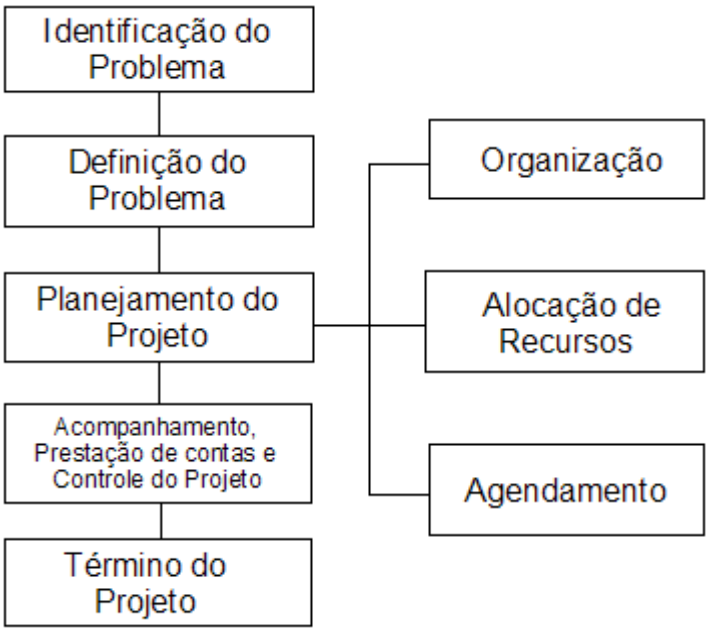

Figura 1: Processo de Gerenciamento de Projeto

Consiste em oito(8) tarefas.

Identificação do Problema

É a tarefa onde um projeto proposto é identificado, definido e justificado. Um projeto proposto pode ser um novo produto, implementação de um novo processo ou um aprimoramento de um recurso existente.

Definição do Problema

É a tarefa onde o propósito do projeto é esclarecido. A declaração da missão do projeto de software é a principal saída desta tarefa. Ela deverá especificar como o gerenciamento de projeto deverá ser utilizado para evitar perda de prazos, agendamentos mal elaborados, alocação inadequada de recursos, falhas de coordenação, baixa qualidade e conflito de prioridades.

Planejamento do Projeto

Esta tarefa define o **Plano de Projeto**. Descreve uma série de ações ou passos necessários para o desenvolvimento do produto de trabalho ou meta de acompanhamento. Isto é especificado como uma inicialização do projeto e executa estes objetivos. Componentes comuns do plano de projeto são incluídos nos objetivos do projeto, definição do projeto, time e critérios de performance ou critérios de validação.

Organização do Projeto

É a tarefa que especifica como integrar as funções dos recursos humanos no projeto. Ele é usualmente realizado concorrentemente com o planejamento do projeto. Requer habilidade de direção, orientação e supervisão da equipe do projeto. É necessário tornar claras as expectativas de cada pessoa da equipe e identificar como suas funções contribuirão no conjunto de metas globais do projeto.

Alocação de Recursos

É a tarefa de alocar recursos para o projeto tais como, dinheiro, pessoas, equipamentos, ferramentas, instalações, informações, técnicas, com o propósito de atingir as metas e objetivos

do software.

Cronograma do Projeto

É a tarefa de alocar recursos para que todos os objetivos do projeto sejam alcançados dentro de um espaço de tempo razoável. Envolve a definição de períodos de tempo específicos para tarefas dentro da agenda de trabalho. Utiliza técnicas como: análise de disponibilidade de recursos (humano, material, dinheiro) e técnicas elaboração de cronogramas de *PERT*, *CPM* e Gráficos de *Gantt*.

Acompanhamento, Prestação de contas e Controle

Tarefa que envolve verificação de conformidade dos resultados do projeto com o planejamento e as especificações de desempenho elaborados. Controle envolve identificação e tomada de ações apropriadas para correção de desvios inaceitáveis em relação ao desempenho esperado. Inclui o processo de avaliação da relação entre o desempenho planejado e o real, no que diz respeito aos objetivos do projeto. As variáveis a serem medidas, as escalas de medição e a abordagem para realizar as medições deverão ser claramente especificadas durante a atividade de planejamento. Ações corretivas podem requerer reagendamento, realocação de recursos ou aceleração do desempenho para determinada tarefa.

Término do Projeto

É a tarefa que envolve a submissão do relatório final, a colocação em funcionamento do novo equipamento ou o sinal para liberação de uma versão. Ela pode dar início a um projeto subsequente ou complementar.

3. Identificação e Definição do Problema

O desenvolvimento de um projeto começa sua identificação e definição. O propósito desta tarefa é informar as decisões tomadas quanto a aprovação, rejeição e priorização de projetos. Ela requer que o estudo da requisição de produto e uma proposta sejam dadas para oferecer uma imagem clara do projeto proposto e as razões por trás dele.

O principal produto deste trabalho é a **Proposta de Projeto**. Ela tipicamente vem de uma variedade de fontes tais como clientes ou requisitos de usuários, conselhos de engenharia, requisitos legais ou regulatórios, pesquisas de mercado e novas tecnologias. Ela deverá indicar informações históricas, escopo e limites iniciais do projeto, capacidade técnica, custo/benefício e risco. Ela não inclui detalhes dos requisitos, planejamento e design. Criar uma proposta de projeto requer uma série de passos.

1. Definir a necessidade.
2. Identificar abordagens alternativas parar atender à necessidade.
3. Recomendar no mínimo duas alternativas.
4. Obter a aprovação.

Uma equipe pode ser estabelecida para fazer a proposta do projeto. Ela deverá incluir especialistas das seguintes áreas funcionais:

- Software/Hardware
- Suporte de Rede
- Centro de Processamento de Dados
- Segurança de Dados
- Administração de Banco de Dados
- Clientes e Usuários
- Auditores Internos e Externos
- Outros interessados ou grupos de suporte

Todas as deliberações da proposta de projeto devem ser documentadas no plano de desenvolvimento ou arquivo do projeto. A estrutura da proposta de projeto pode ser vista abaixo.

I. Sumário Executivo
  A. Escopo do projeto
  B. Objetivos do Negócio e do Sistema de Informação
    i. Objetivos do Negócio
      a) Objetivo primário
      b) Objetivo secundário
    ii. Objetivos do Sistema de Informação
      a) Objetivo primário
      b) Objetivo secundário
II. Critérios de sucesso
III. Alternativas
IV. Cronograma
V. Custos
VI. Benefícios
VII. Risco
VIII. Recomendações

Sumário Executivo

Ele oferece uma introdução para o projeto que inclui informações de suporte indicando o início do projeto e o sumário do trabalho a ser realizado. Ele responde às questões de porque o projeto é importante, qual é a oportunidade de negócio ou necessidade que irá atender e qual é o suporte de informações pertinente.

Escopo

Define os limites do projeto. Descreve as funções e processos que estarão envolvidos no projeto.

Responde à questão de qual é a realização final esperada desta proposta, o que estará incluído no projeto, o que não estará incluído no projeto, quem é o cliente e que solicitou o desenvolvimento do projeto.

Objetivos do Negócio e do Sistema de Informação

Oferece alvos específicos e define resultados mensuráveis. Ao definir objetivos é utilizado o princípio SMART – ***S****pecific* (específico), ***M****easurable* (mensurável), ***A****ttainable* (atingível), ***R****esult-oriented* (orientado a resultado) *and* ***T****ime-oriented* (orientado ao prazo) e M&M's – ***M****easurable* (mensurável) *and* ***M****anageable* (gerenciável). Resultados mensuráveis em termos de tempo, custo e desempenho. Responde à seguinte questão: qual é o resultado desejado ou benefícios do projeto e qual é a estratégia, tática ou objetivos operacionais suportados por este projeto.

Critério de Sucesso

Identifica os principais resultados que deverão ser atingidos para que o projeto esteja completo. Oferece medidas específicas para utilização no julgamento de sucesso. Deverá dar suporte aos objetivos do projeto. Responde às questões de como saberemos quando o projeto está completo, quem irá julgar o sucesso do projeto e como irá julgar esse sucesso.

Alternativas

Define soluções alternativas para o problema do negócio ou necessidade. A alternativa pode ser "fazer ou comprar" ou abordagem técnica tais como codificação tradicional, desenvolvimento orientado a objetos ou ferramenta CASE, integração das edições, reengenharia de processo de negócio. Cada alternativa deverá ser avaliada para determinar um cronograma. Responde às questões de quais os principais produtos, qual relação lógica entre os principais produtos, que pressupostos serão utilizados para definir o cronograma, qual é a dimensão razoável do cronograma baseado no tamanho, esforço, dependências e compromissos.

Custos

Representa um custo estimado para o desenvolvimento do software. Ele é representado como uma gama de valores. Responde às questões de quais são os custos associados com o esforço requerido para produzir cada um dos principais subprodutos, quais são os custos associados com hardware e software, quais são ou outros custos não laborais que devem ser considerados e qual é o custo operacional que precisa ser identificado e considerado.

Benefícios

Podem ser benefícios de curto ou longo prazo. Inclui potenciais economias, recuperações, posicionamento de produtos ou da empresa, aumento de quantidade ou qualidade, crescimento de produtividade e outros benefícios tangíveis ou intangíveis.

Riscos

Identifica riscos de cada alternativa e como os riscos podem ser reduzidos. Deverá ser analisado com base na probabilidade de ocorrência e impacto.

Recomendações

A equipe da proposta de projeto deverá trabalhar junto para selecionar a melhor alternativa que ofereça maior equilíbrio em termos de cronograma, custo/benefício e riscos do projeto. Deverá atender aos objetivos do projeto da melhor forma possível.

4. Organização do Projeto

A ***organização do projeto*** integra as funções das pessoas em um projeto. É necessário identificar claramente as expectativas de cada pessoa e que seja comunicado como sua função contribuirá no conjunto de metas e objetivos do projeto para que haja colaboração. Esta tarefa exige a compreensão da estrutura de desenvolvimento de sistemas da empresa. Figura 2 ilustra esta estrutura.

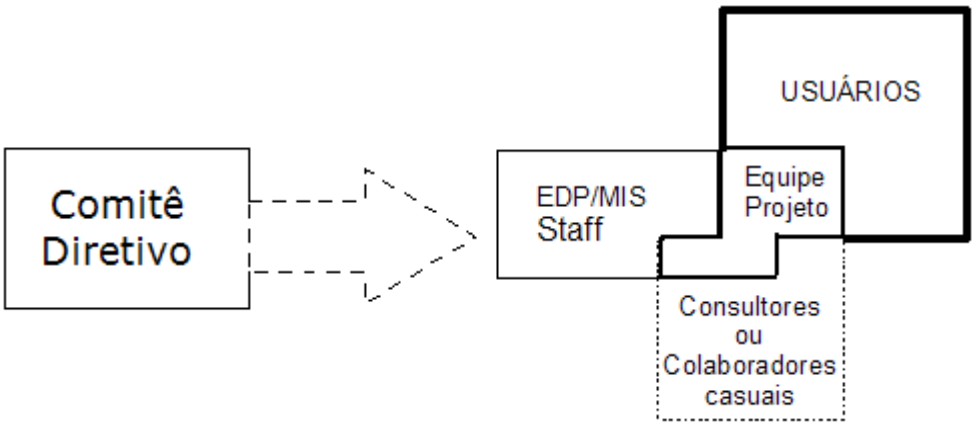

Figura 2: Organização da Estrutura de Desenvolvimento de Sistema

Comitê Diretivo

Define alguns problemas da empresa que têm grande efeito no desenvolvimento do projeto. Ele é composto pela cúpula gerencial (Presidente, CEO ou Diretor Administrativo ou Parceiro), linha de Gerentes (Vice-presidentes ou Chefes de Divisão), Gerentes (Planejamento da Empresa, Controladores), MIS ou gerentes EDP e opcionalmente Consultores. Suas funções consistem em:

- formalizar a composição da equipe do projeto;
- revisar e aprovar os planos de projeto e recomendações;
- aprovar e alocar recursos para o projeto;
- sincronizar o esforço de desenvolvimento do sistema com as demais atividades da organização;
- averiguar regularmente o progresso do projeto;
- resolver situações de conflito;
- aprovar requisições de alterações críticas; e
- liberar versões operacionais do sistema para o usuário.

Equipe de Projeto

Composta de dois grupos, denominados, desenvolvedores e usuários finais. Desenvolvedores consistem de gerente de projeto, analistas, projetistas, programadores e membros do controle de qualidade. O grupo de usuários finais consiste de EDP e Equipe MIS, equipe de usuários, colaboradores casuais e consultores. É um pequeno grupo com um líder ou gerente. Eles regularmente interagem para que as metas sejam atingidas. Requer espírito de equipe, contribuição e coordenação.

4.1. Estrutura da Equipe de Projeto

A estrutura da equipe pode ser organizada como democrática descentralizada, controlada descentralizada e controlada centralizada.

Democrática Decentralizada (DD)

A estrutura de equipe não tem um líder permanente. Preferencialmente, haverá coordenadores para tarefas de curta duração que serão substituídos depois por outros que podem ser necessários para coordenar outras tarefas. Decisões são tomadas pelo consenso do grupo. A comunicação é principalmente horizontal.

Controlada Descentralizada (CD)

A estrutura de equipe tem um líder permanente que primariamente coordena as tarefas. Ela tem um líder secundário responsável por subtarefas. Decisões são tomadas pelo consenso do grupo mas a implementação é feita pelos subgrupos. A comunicação durante a implementação é horizontal mas o controle é vertical.

Controlada Centralizada (CC)

A estrutura de equipe tem um líder responsável pela coordenação da equipe e tomada de decisões de alto nível. A comunicação entre o líder e os membros da equipe é vertical.

A decisão de qual estrutura empregar dependerá das características do projeto. Utilize a Tabela 1 para determinar a estrutura da equipe a ser utilizada para um projeto.

Características do Projeto	DD	CD	CC
Nível de Dificuldade do Problema	Alto	Baixo	Baixo
Tamanho do Software (Linhas de Código ou Pontos por Função)	Pequeno	Grande	Grande
Duração da Equipe	Longa	Curta	Curta
Modularidade do Programa	Baixa	Alta	Alta
Requisito de Confiabildade do Sistema (*Reliability*)	Alta	Alta	Baixa
Rigidez na Data de Entrega	Flexível	Flexível	Rígido
Grau de Sociabilidade	Alta	Baixa	Baixa

Tabela 1: Característica do Projeto versus Estrutura da Equipe

4.2. Gráfico de Responsabilidade do Projeto

É uma matriz que consiste de colunas de áreas funcionais ou individuais, e linhas com ações requeridas. Ele representa a necessidade de comunicar expectativas e responsabilidades entre o pessoal envolvido no desenvolvimento do software. Evita problemas encontrados na comunicação e na possibilidade de indefinição de obrigações. Ele responde às seguintes questões:

- Quem faz o que?
- Quanto tempo levará?
- Quem informará quem sobre algo?
- Quem deverá aprovar o que?
- Quem é responsável por cada resultado?
- Quais interfaces pessoais são necessárias?
- Qual suporte é necessário para quem e quando?

Um modelo de matriz de responsabilidade do projeto é mostrada na Tabela 2.

Atividade	Presid.	Líder Proj.	Analista	Progr.	05/Nov	06/Nov	07/Nov	08/Nov
1. Tarefa Reunião Conjunta								
1.1 Redigir pedido do produto	R				T			
1.2 Marcar reunião conjunto	R	R			T			
1.2.1 Marcar data e hora	I	R	R		T			
1.2.2 Marcar lugar	I	R	R		T			
1.2.3 Identificar particip. e facilitadores	C	I	R			E		
1.3 Convidar participantes			R			E		
1.4 Distribuir pedido do produto	C		R					

Tabela 2: Matriz de Responsabilidade do Projeto

Legenda:

- **Código de Responsibilidade**: **R**-Responsável **I**-Informa **C**-Consulta **S**-Suporte
- **Código Tarefa**: **T**-Terminada **E**-Execução **A**-Atrasada

5. Cronograma do Projeto

Cronograma do Projeto é a tarefa que descreve o processo de desenvolvimento do software para um projeto particular. Enumera fases ou estágios do projeto, e divide cada um dentro de tarefas ou atividades discretas que serão executadas, identifica as interações entre as partes do trabalho e estima o tempo para cada tarefa ou atividade ser executada. É uma sequência de atividades planejadas no tempo sujeitas a relacionamentos de precedências, restrições de tempo e limitações de recursos com a finalidade de atingir objetivos específicos.

É um processo da equipe que dá início à formulação do programa de trabalho. É um processo iterativo que deve ser flexível para acomodar mudanças. Há certos princípios básicos utilizados na criação do cronograma do projeto. Alguns deles estão enumerados abaixo:

1. **Compartimentalização**. O produto e o processo são decompostos em atividades e tarefas tangíveis.
2. **Interdependência**. Cada atividade ou tarefa compartimentalizada deve ser determinada. Tarefas podem ocorrer em sequência ou em paralelo. Tarefas podem ocorrer independentemente.
3. **Alocação de Tempo**. Cada tarefa deverá ser alocada em algum número de unidades de trabalho (pessoa/dia ou homem/dia). Cada tarefa deverá ter uma data de início e fim sujeitas às interdependências e pessoas responsáveis pela tarefa (*part-time* ou *full-time*).
4. **Validação de Esforço**. Em nenhum momento o número de pessoas alocadas é maior que o número de pessoas disponíveis.
5. **Definição de Responsabilidade**. Cada tarefa deverá ter um dono. Ele deverá ser um membro da equipe.
6. **Definição de Resultado**. Cada tarefa deverá ter um resultado definido. Os produtos do trabalho são combinados em entregas.
7. **Definição de *Milestones***. Cada tarefa ou grupo de tarefas deverão ser associadas com um marco do projeto. *Milestones* do projeto são pontos de revisão para qualidade e para aprovação do patrocinador do projeto.

O Cronograma do Projeto identifica conjuntos de atividades ou tarefas, marcos e entregas.

1. **Conjuntos de Atividades ou Tarefas**. Uma coleção de atividades da engenharia de software, marcos e entregas que deverão ser realizadas para completar um projeto em particular. É a parte do projeto que tem lugar sobre um período de tempo. É descrito como um verbo no gerúndio.
2. ***Milestones***. Uma indicação de que alguma coisa foi completada. Faz referência a um momento particular no tempo. Significa pontos de realização dentro do cronograma do projeto. Ele não é a duração do trabalho. Exemplos de *milestones* de projeto são aprovação do usuário, aprovação do projeto do sistema e data de recuperação do investimento no sistema.
3. **Entregáveis**. Uma lista de itens que um cliente espera ver durante o desenvolvimento do projeto. Pode incluir documentos, demonstrações de funcionalidades, demonstrações de subsistemas, demonstrações de precisão, confiabilidade, segurança ou velocidade.

5.1. Estrutura de Divisão do Trabalho do Projeto (WBS)

A ***WBS - Work Breakdown Structure*** é uma ferramenta que permite aos gerentes de projeto definir o conjunto de tarefas, *milestones* e entregáveis. É uma abordagem de análise sistemática de detalhamento do projeto como um conjunto discreto de unidades de trabalho. As tarefas e os *milestones* podem ser usados no projeto para acompanhar o desenvolvimento e a manutenção. Dois métodos são utilizados na definição da estrutura de divisão do trabalho. São denominados Análise da Divisão do Trabalho, Decomposição *top-down* e Integração *bottom-up*.

Análise da Divisão do Trabalho - Work Breakdown Analysis

A Análise da Divisão do Trabalho consiste dos seguintes passos:

1. Dividir o projeto em blocos de atividades relacionadas.
2. Organizar os blocos dentro de uma lógica hierárquica.
   - A análise inicia pela identificação das maiores fases e dos principais subprodutos que cada uma produz.
   - Dividir cada atividade e definir subatividades e produtos do trabalho destas.
   - Continuar a subdividir uma atividade até obter uma atividade que não possa ser subdividida. Esta atividade atômica é chamada de unidade de trabalho ou pacote.
3. Definir a Unidade de Trabalho ou Pacote. A unidade de trabalho ou pacote é de responsabilidade de uma pessoa. Deverá ser executada sem interrupção alguma até estar finalizada. Sua duração e custo podem ser medidos e requerem uso contínuo de um conjunto de recursos.

Processo WBS de Decomposição *top-down* e Integração *bottom-up*

Este método define duas principais atividades como especifica seu nome, decomposição *top-down* e integração *bottom-up*.

1. Decomposição *top-down*
   - Identifica de 4-7 componentes principais do trabalho, não importando a sequência.
   - Identifica entregáveis intermediários e finais para cada grupo.
   - Executa decomposição funcional até que uma tarefa tenha um dono, define claramente os entregáveis, estimativas confiáveis e que possam ser verificadas.
   - Utiliza um verbo para designar uma tarefa no nível mais elementar. O número recomendado de níveis é quatro (4).
   - Múltiplas iterações são requeridas.
2. Integração *bottom-up*
   - Todas as tarefas possíveis são consideradas (Brainstorm).
   - Organiza tarefas em agrupamentos de 4-7 refletindo como o projeto será gerenciado.

5.2. Formato de Cronograma da Divisão do Trabalho

Há dois formatos comuns que podem ser utilizados para representar a WBS, denominados diagrama gráfico ou hierárquico e esquema de tópicos.

Diagrama Gráfico ou Hierárquico

Um exemplo de um diagrama hierárquico da Tarefa Reunião Conjunta Prévia da Engenharia de Requisitos é mostrada na Figura 3.

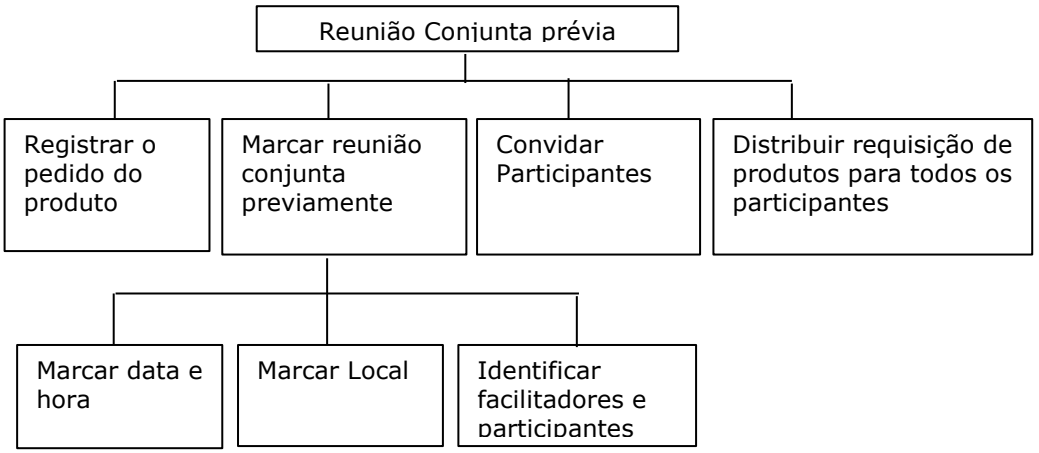

Figura 3: Diagrama Hierárquico

Esquema de Tópicos (*Outline Format*)

Um exemplo de esquema de tópicos é mostrado abaixo.

1. Tarefa - Reunião Conjunta Prévia
   - 1.1 Redigir pedido do produto
   - 1.2 Marcar reunião conjunta prévia
     - 1.2.1 Marcar data e hora
     - 1.2.2 Marcar lugar
     - 1.2.3 Identificar facilitadores e participantes
   - 1.3 Convidar participantes
   - 1.4 Distribuir pedido do produto para todos os participantes

Gráfico de GANTT

Este é um modo fácil de escalonar tarefas. Ele é um diagrama que consiste de barras que indicam a duração da tarefa. A dimensão horizontal indica o tempo, enquanto que a dimensão vertical indica a tarefa. Tabela 3 mostra um exemplo de gráfico de GANTT.

	1	**2**	**3**	**4**	**5**	**6**
1. Tarefa Reunião conjunta prévia						
1.1 Redigir pedido do produto						
1.2 Marcar reunião conjunta prévia						
1.2.1 Marcar data e hora						
1.2.2 Marcar lugar						
1.2.3 Identificar facilitadores e participantes						
1.3 Convidar participantes						
1.4 Distribuir pedido do produto						

Tabela 3: Exemplo de Gráfico de GANTT

6. Alocação de Recursos do Projeto

É o processo de alocar ou destinar dinheiro, pessoas, equipamentos, ferramentas, facilidades, informações, habilidades e outros para tarefas identificadas no projeto. Eles serão necessários para atingir os objetivos e metas do software. Há três restrições principais que devem sempre ser consideradas em um projeto de desenvolvimento de software. São elas:

- Restrições de Tempo
- Restrições de Recursos
- Restrições de Desempenho

6.1. Base de Dados de Disponibilidade de Recursos

O gerente de projeto deverá utilizar a base de dados de disponibilidade de recursos para gerenciar os recursos alocados para cada grupo de tarefas. A base de dados de disponibilidade de recursos especifica quais recursos são necessários diante dos recursos que estão disponíveis. A Tabela 4 mostra um exemplo.

Tipo Rec.	Recurso ID	Habilidades	Data Neces.	Duração	Nº Recursos
Tipo 1	Gerente Projeto	Planejamento Gerenciamento	1/1/XX	10 meses	1
Tipo 2	Analista	Desenvolvimento Análise e Modelagem	25/12/XX	Indefinido	2
Tipo 3	Projetista	Desenvolvimento Software Arquitetura e Componentes	Agora	36 meses	2
…	…	…	…	…	…
…	…	…	…	…	…
…	…	…	…	…	…
Tipo n-1	Operador	Conhecimento de Equipamentos	Imediato	Indefinido	4
Tipo n	Programador	Ferramentas Software	9/2/XX	12 meses	5

Tabela 4: Base de Dados da Disponibilidade de Recursos

Há diversos fatores que devem ser considerados quando da alocação de recursos. Alguns deles são listados abaixo.

- Limitações na disponibilidade do recurso
- Restrições de precedências
- Restrições de subdivisões de atividades
- Atividades imprevistas
- Data de término do projeto
- Substituição de recursos
- Assinalamento parcial de recursos
- Atividades mutuamente exclusivas
- Disponibilidade variável de recursos
- Variação na duração das atividades

Alocação de recursos para um grupo de tarefas do Projeto.

Grupo de Tarefas do Projeto	Recursos
1. Tarefa reunião conjunta prévia	
1.1 Redigir pedido do produto	Usuários finais que irão redigir o pedido do produto

Grupo de Tarefas do Projeto	Recursos
	Computador Material de escritório
1.2 Marcar reunião conjunta prévia	
1.2.1 Marcar data e hora	Líder do projeto
1.2.2 Marcar lugar	Líder do projeto
1.2.3 Identificar facilitadores e participantes	Líder do projeto
1.3 Convidar participantes	Equipe Administrativa Email Material de escritório
1.4 Distribuir pedido do produto	Líder do projeto Computador Material de escritório

7. Métricas de Software

Referem-se a uma variedade de indicadores utilizados em software de computadores. Esses indicadores podem ser aplicados ao processo de software com o objetivo de melhoria contínua do processo. Podem ser utilizadas através do projeto para auxiliar em estimativas, controle de qualidade, avaliação de produtividade e controle do projeto. Podem ainda ser utilizadas pelos engenheiros de software para assegurar a qualidade dos produtos e auxiliar na tomada de decisões táticas conforme o projeto evolui.

Métricas de Software são utilizadas como indicadores para propiciar melhor compreensão do processo de software, do projeto ou do software. Permitem aos engenheiros de software tomar decisões de como prosseguir com o esforço de desenvolvimento.

Categorias de Métricas

Estas são as linhas gerais de duas categorias métricas:

1. **Métricas Diretas**. No processo de software elas podem ser o custo e o esforço dispendido. No software, podem ser linhas de código (*LOC – Lines of Codes*) produzidas, número de classes, velocidade de execução, espaço em memória e defeitos reportados em um período de tempo.
2. **Métricas Indiretas**. Podem ser medidas de produtos tais como funcionalidade, qualidade, complexidade, eficiência, confiabilidade, manutenibilidade, etc.

Exemplos de métricas que são aplicadas a projetos de software são brevemente enumeradas abaixo.

1. **Métricas de Produtividade**. *Productivity Metrics*. Referem-se a medição de saídas ou resultados do processo de software.
2. **Métricas de Qualidade**. Referem-se às medidas que indicam com que precisão o software se aproxima dos requisitos. Também conhecidas como capacidade do software para uso.
3. **Métricas Técnicas**. Refere-se à medição das características do produto tais como complexidade, grau de colaboração, modularidade, etc.
4. **Métrica Orientada a Tamanho**. É utilizado para coletar medidas diretas do software produzido e qualidade baseada nas linhas de código (LOC) produzidas.
5. **Métrica Orientada a Função**. É utilizado para coletar medidas diretas do software produzido e de qualidade baseada nas funcionalidades ou utilidade dos programas.
6. **Métrica Orientada a Recursos Humanos**. Oferece medidas coletadas sobre a maneira como as pessoas desenvolvem software de computador e a percepção humana sobre a eficiência de métodos e ferramentas.

Diferentes fases ou atividades do desenvolvimento de projetos requerem diferentes métricas. Cada capítulo oferece as métricas que normalmente são utilizadas para auxiliar os engenheiros de software a garantir a qualidade do software e do processo.

7.1. Métrica Orientada a Tamanho – Linhas de Código (LOC)

Um medida direta utilizada para métrica de software é Linhas de Código (LOC). Ela é uma métrica orientada ao tamanho utilizada para estimar esforço e custo. É importante que as métricas de outros projetos sejam colocadas em uma tabela para servir de referência durante a realização da estimativa. Tabela 5 mostra um exemplo disto.

Projeto	**Esforço** (em pessoas-mês)	**Custo** (em P)	***KLOC*** (mil linhas de cód.)	**Págs** (de Doc.)	**Erros**	**Pessoas**
Projeto Um	24	120	12.1	365	29	3
Projeto Dois	62	450	27.2	1224	86	5
Projeto Três	43	213	20.2	1050	64	6

Projeto	Esforço (em pessoas-mês)	Custo (em P)	*KLOC* (mil linhas de cód.)	Págs (de Doc.)	Erros	Pessoas
Projeto Quatro	36	30	15	17	10	5
…	…	…	…	…	…	…

Tabela 5: Histórico de Métricas de Software

Nesta tabela, **esforço** em pessoas-mês especifica o número de meses em que o projeto foi desenvolvido, **custo** é o dinheiro gasto na produção do software, ***KLOC*** são as linhas de código produzidas, **páginas** é o número de páginas de documentação, **erros** é o número de erros reportados depois que o software foi entregue para o cliente e **pessoas** é o número de pessoas que desenvolveram o software.

Alguém poderia derivar métricas desta tabela. Exemplos de cálculos são mostrados abaixo.

Produtividade = KLOC / pessoas-mês

Qualidade = erros / KLOC

Custo = Custo / KLOC

Documentação = Páginas / KLOC

7.2. Métrica orientada por Função: Pontos por Função (PF)

Outra métrica comumente utilizada é a métrica de Pontos por Função (PF). É focada na funcionalidade ou utilidade do programa. Utiliza um relacionamento empírico baseado em mensurações contáveis de domínio de informação e estimativas de complexidade do software.

Calculando Ponto por Função

PASSO 1. Determinar o valor de domínio da informação usando a tabela mostrada na Tabela 6.

Preencher a coluna de quantidade e multiplicar o valor pelo fator de peso escolhido. Como exemplo, se o número de entrada for 3 e a entrada de dado for simples, escolha o fator de peso 3, multiplicando, 3 x 3 = 9.

Domínio de Informação		Fator de Peso			
Parâmetro de Medição	Qtde.	Simples	Média	Complexa	
Número de Entradas Externas		x 3	4	6	=
Número de Saídas Externas		x 4	5	7	=
Número de Consultas Externas		x 3	4	6	=
Número de Arquivos		x 7	10	15	=
Número de Interfaces Externas		x 5	7	10	=
Total Geral					

Tabela 6: Cálculo do Domínio de Informação

Encontre a soma (Total Geral).

PASSO 2. Determinar o valor de ajuste de complexidade.

Para determinar o valor de ajuste de complexidade, utilize a escala na Tabela 7 para responder às questões especificadas a seguir.

0	1	2	3	4	5
Sem Influência	Casual	Moderada	Média	Significativa	Essencial

Tabela 7: Escala de ajuste de complexidade

1. Sistema exige cópia de segurança e recuperação confiáveis?
2. Exige a comunicação entre os dados?
3. Existem funções de processamento distribuído?

4. Performance é crítica?
5. Sistema irá rodar em ambiente operacional conhecido e bastante utilizado?
6. Sistema requer entrada de dados on-line?
7. Entrada de dados on-line requer múltiplas telas e operações ?
8. Arquivos principais são atualizados on-line?
9. Entradas, saídas, arquivos ou consultas são complexos?
10. Processamento interno é complexo?
11. Código modelado pode ser reusável?
12. Procedimentos de instalação e migração foram incluídos no projeto?
13. Sistema foi projetado para múltiplas instalações em diferentes organizações?
14. Sistema foi projetado para facilitar alterações e utilização pelo usuário?

PASSO 3. Calcular o ponto de função.

Utilize a fórmula indicada abaixo.

$$\text{Ponto de Função} = \text{Quantidade Total X } [0.65 + 0.01 \text{ X soma}(F_i)]$$

Pode-se utilizar outros tipos de métricas orientadas a tamanho. Exemplos disso são listados abaixo.

Produtividade = PP / pessoa-mês

Qualidade = Defeitos / PF

Custo = Custo / PF

Documentação = Páginas de Documentação / PF

7.3. Harmonizando Métricas de PF e LDC

Para harmonizar as métricas LDC e PF, utilizamos a tabela 8[1].

Linguagem de Programação	*LDC/PF (Média)*
Linguagem Assembly	337
COBOL	77
Java	63
Perl	60
Ada	154
Visual Basic	47
SQL	40
C	162
C++	66

Tabela 8: Harmonização de LDC e PF

1 Pressman, Software Engineering: A Practitioner's Approach, p. 657)

8. Métricas de Software

Avaliações de Projeto

São necessárias para auxiliar engenheiros de software a determinar o esforço e o custo requeridos na construção do software. Para fazer a avaliação, é necessário experiência, acesso a uma boa série histórica de dados, e comprometimento para quantificar medidas quando dados qualitativos é tudo o que existe.

Pode-se usar LDC e PF para o provimento de estimativas de custo e esforço. Ambos têm variáveis para quantificar o "tamanho" de cada elemento de software. Ambos têm métricas de *baseline* coletadas de projetos anteriores e utilizadas em conjunto com variáveis de estimativa para o cálculo de projeções de esforço e custo.

Como exemplo, considere as estimativas de projeto de um sistema de manutenção de associados de um clube. Utilize LDC e PF para determinar as estimativas de custo e esforço. Calculamos primeiro o PF.

Domínio de Informação		**Fator de Peso**			
Parâmetro de Medição	**Qtd**	**Simples**	**Média**	**Complexa**	
Número de Entradas Externas	2	x 3	4	6	6
Número de Saídas Externas	5	x 4	5	7	20
Número de Consultas Externas	3	x 3	4	6	9
Número de Arquivos	2	x 7	10	15	14
Número de Interfaces Externas	2	x 5	7	10	10
				Qtd-Total	**59**

1. O sistema exige cópia de segurança e recuperação confiáveis? 3
2. É exigida comunicação de dados? 3
3. Existem funções de processamento distribuído? 1
4. A performance é crítica? 2
5. O sistema irá rodar em ambiente operacional conhecido e muito utilizado? 3
6. O sistema requer entrada de dados on-line? 4
7. A entrada de dados on-line requer múltiplas telas e operações ? 2
8. Os arquivos principais são atualizados on-line? 2
9. As entradas, saídas, arquivos ou consultas são complexos? 1
10. O processamento interno é complexo? 1
11. O código modelado pode ser reusável? 4
12. Procedimentos de instalação e migração foram incluídos no projeto? 3
13. O sistema foi projetado para múltiplas instalações em diferentes organizações? 1
14. O sistema foi projetado para facilitar alterações e facilidade de uso pelo usuário? 1

Total = 31

PF = 59 X [0.65 + 0,01 X 31] = 56,64 PF

O projeto utiliza JAVA extensivamente. A LDC estimada é calculada a seguir:

LDC = 56,64 * 63 = 3568,32 LDC

Estimativas Derivadas: Assuma que a Manutenção do Título do Clube é similar ao projeto 4.

Estimativas de Custo são calculadas a seguir:

Custo de Produção de uma única LDC (do banco de dados): Custo / KLDC

Custo/KLDC = P30000/15000LDC = P2,00/LDC

Custo do projeto = 3568,32 * 2,00 = P7136,64

Estimativas de Esforço são calculadas a seguir:

Número de LDC produzidas em um mês (do banco de dados): KLOC/Esforço

KLDC/Esforço = 15000LDC / 8 meses = 1875 LDC / mês

Esforço do projeto = 3568,32 / 1875 = 1,9 ou 2 meses

9. Escrevendo o Plano de Projeto

O plano de projeto é um documento que contém o plano de desenvolvimento do software. Não é necessário ser extenso e complexo. Seu propósito é ajudar a estabelecer a viabilidade do desenvolvimento do software. O plano se concentra em declarações genéricas de "O Quê" vai ser feito e declarações específicas de "Quanto vai custar" e "Quanto tempo" vai levar para ser feito. Os detalhes serão apresentados no decorrer do desenvolvimento. O plano de projeto teria a seguinte estrutura.

I. Introdução
   A. Histórico
   B. Objetivos e Escopo do Projeto
   C. Dados e Funções Principais
   D. Questões de Performance
   E. Restrições e Limitações
II. Estimativas do Projeto
   A. Estimativas de Esforço
   B. Estimativas de Custo
III. Cronograma do Projeto
IV. Recursos do Projeto
V. Organização do Projeto

9.1. Histórico

Consiste em dois ou mais parágrafos discutindo brevemente o histórico do sistema. Identifica os dados principais, informações utilizadas e os processos principais dentro do sistema em condições gerais. Identifica estudos prévios feitos com o histórico do sistema.

9.2. Declaração de Problema

Define um conjunto de problemas ou áreas de melhoria que podem ser vistas dentro do histórico do sistema.

9.3. Objetivos do Projeto

Declaração do que será alcançado ou será feito. Define as metas globais do projeto sem considerar como estas metas serão alcançadas. Deveria ser **SMART** (Esperto):

Specific – Específico
Measurable – Mensurável
Attainable – Atingível
Result-oriented – Orientado a resultado
Time-bound – De tempo limitado

9.4. Estimativas do Projeto

Identifica os dados primários, funções e comportamento que caracterizam o problema e, mais importante, tenta limitar estas características de uma maneira quantitativa. Declaração do que será e não será feito.

9.5. Cronograma do Projeto

Envolve a tarefa de períodos de tempo a tarefas específicas dentro de um horário de trabalho. Para cada atividade identificada no WBS, terá:

Entregável. É o produto de trabalho que precisa ser desenvolvido.
Dono. Responsável que assegura que a tarefa seja finalizada e foram conhecidos os entregáveis.
Recursos. São as pessoas responsáveis por realizar a tarefa.
Recursos técnicos. São os equipamentos, ou seja, as ferramentas que são necessárias para fazer o trabalho e criar o entregável.
Custo. É quanto gasta para realizar a atividade.
Linha do Tempo. Em quanto tempo o trabalho será terminado.

9.6. Recursos do Projeto

Lista os recursos necessários. Banco de dados de Disponibilidade de Recurso.

9.7. Organização do Projeto

Identifica as pessoas necessárias para realizar o projeto e os papéis que eles precisam desempenhar.

10. Gerenciamento de Risco

Gerenciamento de Risco consiste em etapas que ajudam engenheiros de software a entender e gerenciar incertezas no processo de desenvolvimento de software. Um ***risco*** é considerado um problema potencial, ou seja, ele pode ou não acontecer. É melhor identificá-lo, avaliar sua probabilidade de ocorrer, estimar seu impacto e estabelecer um plano de contingência caso ele ocorra. Várias coisas podem acontecer durante o progresso do desenvolvimento de software. É por essa razão que entender os riscos envolvidos e ter medidas que os evitem, influenciará o quão bem o projeto é gerenciado.

Duas importantes características de risco são sempre analisadas e identificadas em um gerenciamento de risco. Riscos sempre envolvem incertezas. Isto significa que riscos podem ou não acontecer. Se um risco torna-se realidade, ocorrerão perdas e consequências negativas. O ***nível de incerteza*** e ***grau de perda*** associados a cada risco são identificados e documentados.

Existem diferentes categorias de riscos que engenheiros de software precisam entender. Elas devem ser consideradas como problemas potenciais que podem surgir. Entendê-los permitirá a criação de um plano de contingência com o impacto mínimo no desenvolvimento do projeto.

1. **Riscos de Projeto**. Esses riscos estão associados ao plano do projeto. Se tal risco vier a se tornar realidade, poderiam causar problemas referentes a atrasos, orçamento e recursos humanos.
2. **Riscos Técnicos**. Esses riscos estão associados a aspectos relativos a implementação do software. Caso vierem a se tornar realidade, a implementação do projeto torna-se dificultosa. Podem ser problemas de modelagem e interface, problemas relativos a componentes, banco de dados e problemas de infraestrutura.
3. **Riscos de Negócio**. Esses riscos estão associados com a viabilidade do software. Se tal risco vier a acontecer, no contexto do negócio o software, será um problema. Pode ser a construção de um software que não atenda às necessidades, que não seja utilizado, ou que perca orçamento e patrocínio de área demandante.
4. **Riscos Conhecidos**. Esses riscos podem ser detectados com avaliação cuidadosa do plano de projeto, análise do ambiente de negócios, ou outras boas fontes de informação.
5. **Riscos Previsíveis**. Esses riscos ocorreram com projetos similares e a expectativa é que possam ocorrer novamente.
6. **Riscos Imprevisíveis**. Esses riscos não são detectados.

10.1. A Tabela de Risco

A ***Tabela de Risco*** é uma ferramenta que permite ao engenheiro de software identificar e gerenciar riscos inerentes ao projeto de desenvolvimento de software. A Tabela 9 mostra um exemplo de tabela de risco.

Riscos Identificados	*Categoria*	*Probabilidade*	*Impacto*	*MMGR*
Estimativa do tamanho do projeto pode ser muito baixa	TP	60%	2	
Um número grande de grupos de usuários finais não estão identificados	TP	30%	3	
Componentes não podem ser reutilizados	TP	70%	2	
Membros do comitê patrocinador não estão interessados no projeto	IN	40%	3	
Prazo do projeto não é negociável	IN	50%	2	
Rotação de recursos será alta	TE	60%	1	
…	…	…	…	…

Tabela 9: Exemplo de Tabela de Risco

A primeira coluna lista todos os riscos possíveis, inclusive aqueles que muito provavelmente não

irão acontecer. Cada risco é então categorizado. A categorização identifica os tipos de risco, enumerados abaixo de forma genérica.

1. **Tamanho do Produto**. São os riscos relacionados com o tamanho geral do software que precisa ser desenvolvido.
2. **Impacto no Negócio**. São os riscos relacionados às restrições de mercado e gerenciamento.
3. **Característica do Cliente**. São os riscos relacionados aos requisitos do usuários final e a habilidade da equipe de desenvolvimento de entender-se e comunicar-se com os mesmos.
4. **Definição de Processo**. São os riscos relacionados ao processo de software e como as pessoas envolvidas no esforço de desenvolvimento estão organizadas.
5. **Ambiente de Desenvolvimento**. São os riscos relacionados à disponibilidade e qualidade das ferramentas utilizadas no desenvolvimento do software.
6. **Tecnologia**. São os riscos relacionados à complexidade do sistema e ao grau de inovação que faz parte do software.
7. **Tamanho e Experiência da Equipe**. São os riscos relacionados à técnica em geral e experiência em projetos dos engenheiros de software.

O valor provável do risco é identificado como valor percentual dele ocorrer. A seguir, o seu impacto quando verificada sua ocorrência. É estimado utilizando os seguintes valores.

1. **Catastrófico**. Resultaria em fracasso no atendimento dos requisitos e a não aceitação do software.
2. **Crítico**. Resultaria em fracasso no atendimento de requisitos relacionados com a degradação de performance do sistema a um ponto em que seu sucesso seja questionável.
3. **Marginal**. Resultaria em fracasso no atendimento de requisitos, o que acarretaria degradação da missão secundária.
4. **Desprezível**. Resultaria em inconveniências e problemas de uso.

A última coluna contém plano de MMGR(Mitigação, Monitoração e Gerenciamento de Risco) de cada risco. Tem os seguintes componentes:

1. **Identificador (ID) do Risco**. É um número único de identificação atribuído ao risco.
2. **Descrição**. Breve descrição do risco.
3. **Contexto de Risco**. Define a condição pela qual o risco pode ser concretizado.
4. **Mitigação e Monitoração**. Define os passos necessários a serem executados a fim de mitigar e monitorar o risco.
5. **Plano de Contingência**. Define os passos necessários a serem executados quando o risco é concretizado.

10.2. Lista de Identificação de Risco

Pode-se utilizar a lista abaixo para determinação dos riscos inerentes ao projeto de desenvolvimento de software.

Riscos de Tamanho do Produto

O tamanho do produto é diretamente proporcional ao risco do projeto. Como engenheiros de software, devemos prestar atenção no seguinte:

- O quão grande foi estimado o software em termos de linhas de código e pontos por função?
- Qual a quantidade de programas, arquivo e transações?
- Qual o tamanho do banco de dados?
- Qual a quantidade de usuários estimada?
- Quantos componentes são reutilizados?

Riscos de Negócio

- Qual é o valor de negócio do software para a empresa ou departamento?
- Qual é a visibilidade do software para a alta direção?
- Quão razoável é a data de entrega?
- Qual a quantidade de clientes?
- Quantos produtos e sistemas interagem com o software?
- Qual o tamanho da documentação do produto?
- Que qualidade tem a documentação do produto?
- Quais são as restrições governamentais inerentes ao desenvolvimento do projeto?
- Qual o custo associado a uma entrega tardia?
- Qual o custo associado a produto defeituoso?

Riscos Relacionados ao Cliente

- Como é o relacionamento de trabalho com o cliente?
- Qual é o nível técnico do cliente para definir seus requisitos?
- Os clientes estarão dispostos a perder tempo para coleta de requisitos?
- Os clientes estarão dispostos a participar de revisões técnicas formais?
- Qual é o nível de conhecimento dos clientes em termos de aspectos tecnológicos do software?
- O cliente conhece o processo de desenvolvimento do software e o seu papel nesse desenvolvimento?

Riscos de Processo – Problemas de Processo

- A alta direção apóia ou tem compromisso total com o esforço de desenvolvimento do software?
- A empresa tem a descrição do processo de software a ser utilizada no projeto?
- A equipe conhece o processo de desenvolvimento do software?
- O processo de software foi utilizado antes em outros projetos pelo mesmo grupo de pessoas?
- A empresa desenvolveu ou adquiriu cursos de treinamento para os gerentes e equipe técnica?
- Existem padrões de engenharia de software publicados que serão utilizados pelos desenvolvedores e gerentes?
- Existem esboços e exemplos de entregáveis?
- As revisões técnicas formais de especificação de requisitos, modelagem e código são feitas regularmente? Têm procedimentos definidos?
- As revisões técnicas formais de procedimentos de teste e casos de teste são feitas regularmente? Têm procedimentos definidos?
- Os resultados de cada revisão técnica formal são documentados, incluindo erros encontrados e recursos utilizados? Têm procedimentos definidos?
- Têm procedimentos que assegurem que o trabalho efetuado no projeto esteja em conformidade com os padrões de engenharia de software?
- Têm gerenciamento de configuração para manter consistência entre sistema/requisitos de software, modelagem, casos, e casos de teste?
- Existem documentos para declaração de trabalho, especificação de requisitos e plano de desenvolvimento de software de cada subcontrato?

Riscos de Processo – Problemas Técnicos

- Existe algum mecanismo que facilite uma clara comunicação entre cliente e desenvolvedor?
- Existem métodos específicos utilizados para análise de software?
- Existem métodos específicos para modelagem de componentes?

- Existem métodos específicos para modelagem de arquitetura e dados?
- Os códigos são escritos em linguagem de alto nível? Qual o percentual?
- Padrões de codificação e documentação são observados?
- Existem métodos específicos para modelagem de casos de teste?
- Ferramentas de software são utilizadas para suporte de planejamento e rastreamento de atividades?
- Ferramentas de gerenciamento de configuração de software são utilizadas para controlar e rastrear atividades de mudança através do processo de software?
- Ferramentas CASE são utilizadas?
- As métricas de qualidade foram coletadas para todos os projetos de software?
- Existem métricas de produtividade coletadas para projetos de software anteriores?

Riscos Tecnológicos

- A empresa é recente em tecnologia?
- Será necessário criar novos algoritmos ou novas tecnologias de entrada e saída?
- O software precisa utilizar hardware novo ou sem suporte?
- O software precisa utilizar um sistema de banco de dados não testado na área de aplicação?
- Será necessário definir interface de usuário especializada?
- Serão necessários novos métodos de teste, análise e modelagem?
- Será necessário utilizar métodos não convencionais de desenvolvimento de software, tais como métodos formais, abordagens baseadas em inteligência artificial e redes neurais artificiais?

Riscos do Ambiente de Desenvolvimento

- Existem ferramentas de gerenciamento de projeto de software?
- Existem ferramentas para análise e modelagem?
- Existem ferramentas de teste?
- Existem ferramentas de configuração de software?
- As ferramentas de software estão integradas entre si?
- Será necessário treinamento dos membros da equipe de desenvolvimento para usar as ferramentas de desenvolvimento?
- Existem ajudas on-line ou grupos de suporte?

Risco Associado ao Tamanho e Experiência da Equipe

- A rotatividade da equipe é alta?
- Temos recursos com expertise?
- O número de recursos da equipe de desenvolvimento é suficiente?
- Será necessário hora extra?
- Haverá problemas de comunicação?

11. Gerenciamento de Configuração de Software

Gerenciamento de Configuração de Software é uma atividade guarda-chuva que dá suporte ao software durante o seu ciclo de vida. Pode ocorrer a qualquer tempo como produto dos trabalhos que são desenvolvidos ou que sofreram manutenção. É focado no gerenciamento de mudanças no processo de desenvolvimento de software. Consiste na identificação e controle da mudança, assegurando que esteja sendo adequadamente implementada e reportando-a aos que possam ser afetados pela mudança. O gerenciamento de configuração efetua o gerenciamento de itens chamados **Itens de configuração de Software (ICS) ou unidades**, os quais podem ser programas de computador (tanto códigos-fonte como executáveis), documentos (sejam técnicos ou do usuário) e dados.

Os possíveis módulos que servem como unidades ou itens de configuração de software são listados a seguir. São baseados nas atividades ou fases da engenharia de software que são produzidas.

1. Fase de Planejamento de Software
   a) Plano de Projeto de Software
2. Requisitos de Engenharia
   a) Especificações de Sistema
   b) Modelos de Requisitos
   c) Modelos de Análise
   d) Protótipos de Telas
3. Engenharia de Modelagem
   a) Modelagem do Banco de Dados
   b) Modelagem de Telas e Caixas de Diálogo
   c) Modelagem de Relatórios
   d) Modelagem de Formulários
   e) Arquitetura de Software
   f) Modelagem de Componentes
   g) Modelagem Distribuição
4. Implementação
   a) Listagem de Código-Fonte
   b) Executável ou Código Binário
   c) Módulos Ligados
5. Testes
   a) Especificações de Teste
   b) Casos de Teste
6. Manuais
   a) Manuais de Usuário
   b) Manuais Técnicos
   c) Procedimentos e Instalação de Engenharia de Software
7. Padrões e Procedimentos de Engenharia de Software

11.1. Baseline

O gerenciamento de ICSs envolve o conceito de *baseline* (linha de partida). Uma ***baseline*** é um conceito de gerenciamento de configuração de software que auxilia engenheiros de software a efetuar o controle de mudança. É a especificação ou produto que tenha sido formalmente revisado e acordado para ser a base de desenvolvimentos futuros. Somente pode ser alterada através de um procedimento formal de mudança. Figura 4 mostra a dinâmica do gerenciamento de baseline dos ICSs.

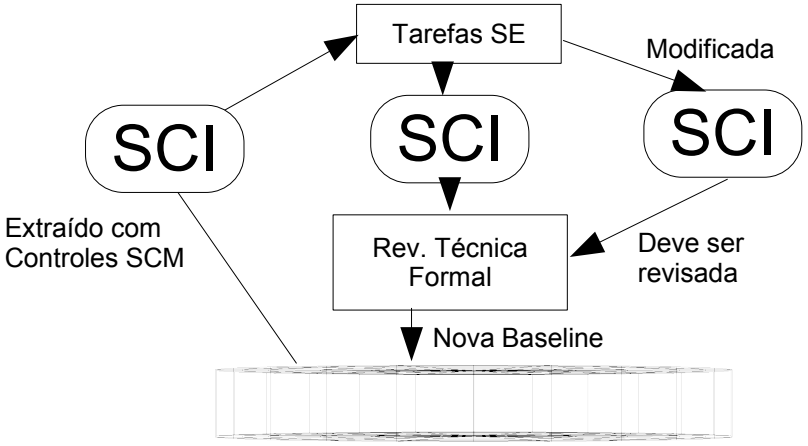

Figura 4: Dinâmica de Baseline de ICSs

Uma tarefa de engenharia de software desenvolve um módulo que torna-se um item ou unidade de configuração de software. Passa pela revisão técnica formal para detecção de erros e falhas. Uma vez que o módulo passe na verificação ou seja aceitável, torna-se uma *baseline* e é salva no banco de dados do projeto. Às vezes, este módulo é extraído do banco de dados do projeto como entrada de alguma outra tarefa de engenharia de software. Observe que "atualizar" uma ICS envolve controles e procedimentos. Durante a execução de uma tarefa de ES (Engenharia de Software), é possível que a ICS tenha sido modificada. Neste caso, a ICS modificada irá passa por uma outra revisão técnica formal. Se aprovada ou aceitável, a versão modificada torna-se a nova baseline.

11.2. Tarefas de Configuração de Software

O gerenciamento de configuração de software tem a responsabilidade do controle de mudança. O gerenciamento identifica os itens de configuração e as várias versões do software. Também audita os itens de configuração do software a fim de assegurar que os mesmos tenham sido propriamente desenvolvidos e que os relatórios de mudança sejam aplicados na configuração. Cinco tarefas são executadas: Identificação de mudança, controle de versão, controle de mudança, auditoria e relatório de configuração.

Identificação de Mudança

É o processo de identificação de itens que alteram o processo de desenvolvimento de software. Tal processo identifica de forma específica os itens de configuração de software (ICS) e atribui um nome e identificador únicos.

Controle de Versão

É o conjunto de procedimentos e ferramentas utilizados para gerenciar as diferentes versões de objetos de configuração que são criados durante o processo de desenvolvimento de software. Uma versão é uma coleção de ICS que pode ser utilizada de forma quase independente.

Controle de Mudança

Consiste em procedimentos manuais e ferramentas automatizadas que proporcionam o mecanismo para o controle de mudança. Tem o seguinte subconjunto de tarefas:

- Uma requisição de mudança é submetida e avaliada para estimar o mérito técnico, potenciais efeitos colaterais e impacto geral nos outros itens de configuração.
- Um relatório de mudança é criado para identificar a decisão final no status e prioridade da mudança.

- Uma Ordem de Engenharia de Mudança (OEM) é criada para descrever a mudança a ser efetuada, as restrições e os critérios para revisão e auditoria.
- Os ICS são conferidos para modificação.
- É, então, verificado numa revisão posterior e passado ao controle de versão.

Auditoria de Configuração

É o processo de avaliar a ICS por características que geralmente não são consideradas durante um revisão técnica formal. Atende às seguintes questões:

- A mudança especificada na OEM foi efetuada? Existem modificações adicionais que foram executadas?
- A revisão técnica formal avaliou a correção técnica do produto?
- Os procedimentos de engenharia de software foram observados?
- Os procedimentos de GQS foram observados?
- Todos os ICSs foram apropriadamente atualizados?

Relatório Informativo

É o processo de informar às pessoas que são afetadas pela mudança. Responde às seguintes questões:

- O que aconteceu?
- Quem fez?
- Quando aconteceu?
- O que mais será afetado?

12. Exercícios

O objetivo da tarefa é reforçar os conhecimentos e habilidades obtidas nesta lição.

1. Contacte outros alunos do ***JEDI*** e crie um grupo de trabalho.
2. Como exercício, o grupo deve criar o plano de projeto para seu projeto.

Parceiros que tornaram JEDI™ possível

Instituto CTS
Patrocinador do DFJUG.

Sun Microsystems
Fornecimento de servidor de dados para o armazenamento dos vídeo-aulas.

Java Research and Development Center da Universidade das Filipinas
Criador da Iniciativa JEDI™.

DFJUG
Detentor dos direitos do JEDI™ nos países de língua portuguesa.

Banco do Brasil
Disponibilização de seus *telecentros* para abrigar e difundir a Iniciativa JEDI™.

Politec
Suporte e apoio financeiro e logístico a todo o processo.

Borland
Apoio internacional para que possamos alcançar os outros países de língua portuguesa.

Instituto Gaudium/CNBB
Fornecimento da sua infra-estrutura de hardware de seus servidores para que os milhares de alunos possam acessar o material do curso simultaneamente.