Introdução À Engenharia De Software Com Foco No RUP: Rational Unified Process

Prof. Dr. Jorge Henrique C Fernandes
(jorge@dimap.ufrn.br)
POTI – Pólo De Tecnologia Da Informação
Departamento De Informática E Mat. Aplicada
Universidade Federal Do Rio Grande Do Norte

Introdução à Engenharia de Software com Foco no RUP: Rational Unified Process

- A estrutura deste curso é baseada, mas não substitui, o uso nos seguintes materiais
  - Livro:
    - [Kruchten, 2000] Introduction to the Rational Unified Process, de Philippe Kruchten, Addison-Wesley, 2000
  - Software:
    - [Rational, 2000] Rational Unified Process – RUP. Rational Software Corporation. 2000
- Rational e Rational Unified Process são marcas comerciais da Rational Software Corporation

Outras Referências

- [Kruchten, 2000] Introduction to the Rational Unified Process, de Philippe Kruchten, Addison-Wesley, 2000.
- [Clements et alli, 1998] Software Architecture in Practice. Len Bass, Paul Clements and Rick Kazman. Addison-Wesley, 1998.
- [J2EE, 2002] Designing Enterprise Applications with the J2EETM Platform, Second Edition. Inderjeet Singh, Beth Stearns, Mark Johnson, and the Enterprise Team. Addison-Wesley. 2002.
- [Sun, 2002] Designing Wireless Enterprise Applications Using Java™ Technology; A Java BluePrints for Wireless White Paper. Sun Microsystems, Inc. 2002.
- [SPEM, 2002] Software Process Engineering Metamodel Specification. November 2002. Version 1.0. formal/02-11-14
- [Kobryn, 2001] Introduction to UML: Structural and Use Case Modeling. Cris Kobryn. OMG UML Tutorial Series.2001

Características do Treinamento

- Introdutório
- Carga de 30 (trinta) horas-aula
- Pré-requisitos
  - Conhecimentos básicos de desenvolvimento de software
- Conhecimentos desejáveis
  - Orientação a objetos
  - UML – Unified Modeling Language
  - Gerencia de projetos de software
- Audiência
  - Gerentes de projetos, desenvolvedores de software, engenheiros de qualidade, de processos e sistemas, analistas de negócios e estudantes de cursos profissionalizantes de informática, computação, sistemas de informação e engenharia de software

Objetivos do treinamento

- Apresentar e discutir abordagens efetivas para desenvolvimento de software
- Entender o que é o RUP, seus vocabulário e conceitos
- Entender como as disciplinas do RUP colaboram para a estruturação efetiva e eficaz de tarefas e fluxos de trabalho de profissionais, atuando em equipes de desenvolvimento de software

Conteúdo: RUP – Módulo Introdutório

Parte I

A1. Abordagens Efetivas Para Desenvolvimento De Software

A2. O Que é o Rational Unified Process - RUP?

A3. Conceitos e Organização

A4. Estágios, Fases, Iterações, Disciplinas e Ciclo de Vida

A5. Foco em Arquitetura

A6. Foco em Casos de Uso

Parte II

A7. Gestão de projetos

A8. Modelagem de Negócios

A9. Requisitos

A10. Gestão de Configuração e Mudanças

A11. Gestão de Ambiente

A12. Análise e Projeto

A13. Implementação

A14. Testes

A15. Instalação

Introdução À Engenharia De Software Com Foco No RUP: Rational Unified Process

Parte II
Disciplinas do RUP

Descrição típica de um workflow

- Propósito
- Definições e Conceitos Chave
- Trabalhadores e Artefatos
- Fluxo Típico
- Suporte de Ferramentas
- Sumário

Aula 7 - Fluxo do RUP: Gestão de Projetos

- Propósito
- Planejamento de Projetos Iterativos
  - Plano de Fase e Plano de Iteração
- Conceitos de Risco
- Conceito de Métricas
- Trabalhadores e Artefatos
- Fluxo Típico
  - Detalhes de Workflow
- Construindo um Plano de Iteração

Gestão de Projetos: Propósito

- Função da Gestão de Projetos
  - Balancear recursos, necessidades e expectativas de todos os envolvidos no projeto, entregando um produto que satisfaça às necessidades de seus usuários e clientes
- A Disciplina de Gestão de Projetos do RUP
  - Base para gerenciamento de sistemas baseado em software
  - Aspectos práticos da gestão: Planejamento, organização, direção e controle
  - Gestão de riscos

Planejamento de Projetos Iterativos

- Questões a responder
  - Quantas iterações?
  - Quão longas as iterações são?
  - Como determinar o conteúdo e objetivos de uma iteração?
  - Como rastrear progresso de uma iteração?

Planejamento de Projetos Iterativos

- O Problema
  - Planejamento de longo prazo não pode ser detalhado
  - Não há previsibilidade no desenvolvimento exceto quando o domínio é extremamente bem conhecido
- Solução
  - Planos em dois níveis de granularidade
  - Plano de Fases (alta granularidade)
    - Um para o ciclo de vida inteiro. Atualizado freqüentemente ao longo do ciclo de vida
    - Conteúdo: data dos milestones, perfil da equipe
  - Plano de Iteração (refinado)
    - Plano corrente, usado para rastrear progresso
    - Plano da próxima iteração, em construção

Estrutura de Plano de Iteração

- Plano
  - Fluxos (macro): quais fluxos são abordados? Em qual intervalo de tempo serão executados?
  - Tarefas: quais atividades e tarefas serão realizadas? Quando serão realizadas? Quais recursos as executarão? Quando os milestones serão alcançados?
  - Lista de entregáveis e responsáveis
- Recursos
  - Staff
  - Financeiros
- Casos de uso
- Critérios de avaliação
- WBS – work Breakdown Structure
- OBS – Organizational Breakdown Structure

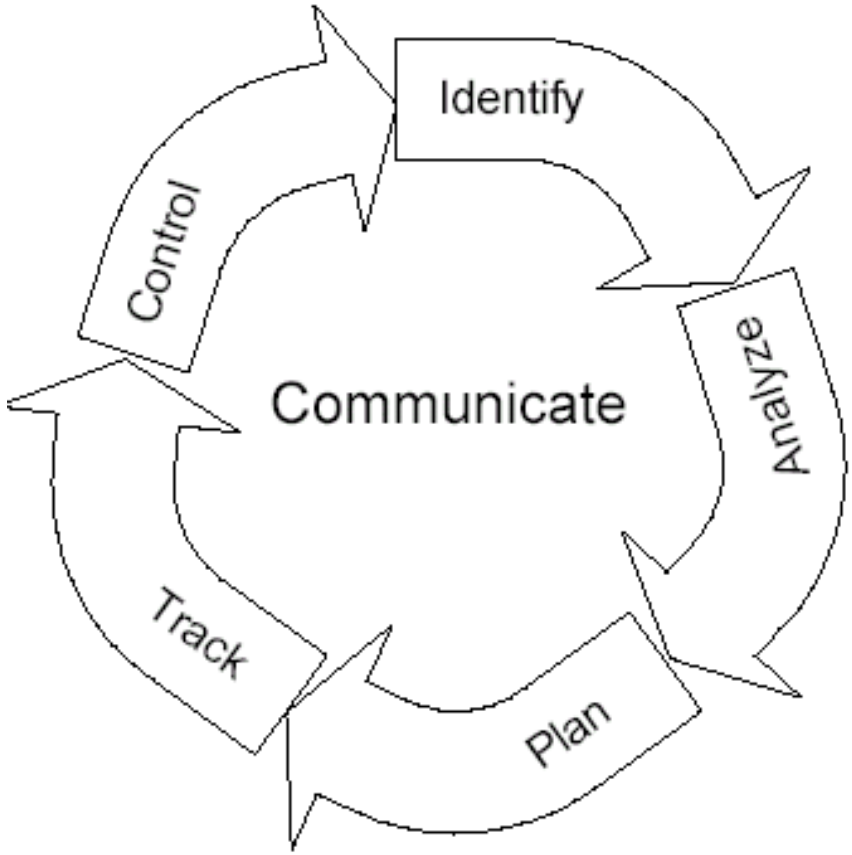
Conceitos de Risco

- Riscos são incertezas que podem conduzir a falhas em um projeto
  - Pessoas não são peças mecânicas intercambiáveis
  - Existem aspectos desconhecidos relativos ao software
- Riscos diretos e indiretos
  - Diretos: o projeto tem maior controle sobre ele
  - Indiretos: o projeto não tem controle sobre ele

Conceitos de Risco

- Atributos
  - Probabilidade
  - Severidade
- Tratamento de riscos
  - Identificação
  - Quantificação
  - Desenvolvimento de Respostas
    - Evitar
    - Transferir
    - Aceitar
      - Mitigar
      - Contingenciar (plano B)

Abordagem para Tratamento de Riscos do SEI – Software Engineering Institute



A. Product Engineering

1. Requirements
- a. Stability
- b. Completeness
- c. Clarity
- d. Validity
- e. Feasibility
- f. Precedent
- g. Scale

2. Design
- a. Functionality
- b. Difficulty
- c. Interfaces
- d. Performance
- e. Testability
- f. Hardware Constraints
- g. Non-Developmental Software

3. Code and Unit Test
- a. Feasibility
- b. Testing
- c. Coding/Implementation

4. Integration and Test
- a. Environment
- b. Product Integration
- c. System Integration

5. Engineering Specialties
- a. Maintainability
- b. Reliability
- c. Safety
- d. Security
- e. Human Factors
- f. Specifications

B. Development Environment

1. Development Process
- a. Formality
- b. Suitability
- c. Process Control
- d. Familiarity
- e. Product Control

2. Development System
- a. Capacity
- b. Suitability
- c. Usability
- d. Familiarity
- e. Reliability
- f. System Support
- g. Deliverability

3. Management Process
- a. Planning
- b. Project Organization
- c. Management Experience
- d. Program Interfaces

4. Management Methods
- a. Monitoring
- b. Personnel Management
- c. Quality Assurance
- d. Configuration Management

5. Work Environment
- a. Quality Attitude
- b. Cooperation
- c. Communication
- d. Morale

C. Program Constraints

1. Resources
- a. Schedule
- b. Staff
- c. Budget
- d. Facilities

2. Contract
- a. Type of contract
- b. Restrictions
- c. Dependencies

3. Program Interfaces
- a. Customer
- b. Associate Contractors
- c. Subcontractors
- d. Prime Contractor
- e. Corporate Management
- f. Vendors
- g. Politics

The SEI Risk Taxonomy

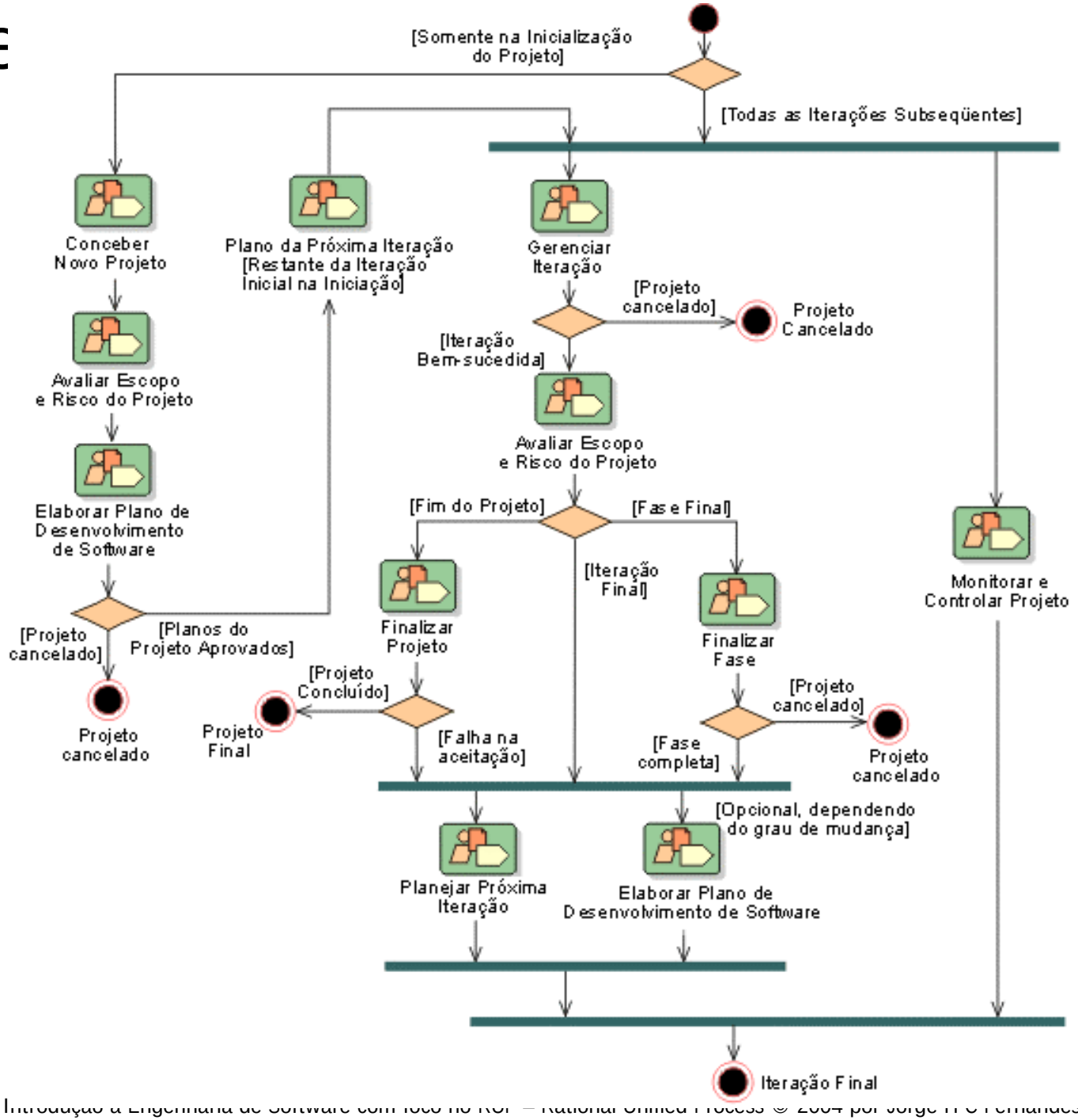
Conceito de Métricas

- Objetivos do uso de métricas
  - Conhecimento
    - Você não pode controlar o que não mede
    - Monitorar progresso
  - Mudança
    - Você não pode aperfeiçoar o que não controla
    - Melhorar satisfação do usuário, aumentar produtividade, melhorar previsibilidade, incrementar reuso

Conceito de Métricas

- Métricas e medições
  - Métrica (metrics): atributo mensurável de uma entidade
    - Exemplo: esforço para desenvolvimento de projeto
  - Medição (measurement): dato cru, usando para calcular métrica
    - Ex: registros de log de tempo
- Framework GQM (Goal-Question-Metric)
  - Goal: qual a meta a alcançar?
  - Question: que questões deve ser respondidas para avaliar se a meta foi alcançada?
  - Métrica: que métricas responderão à questão?

Ge s e


[Somente na Inicialização do Projeto]
[Todas as Iterações Subseqüentes]
Conceber Novo Projeto
Plano da Próxima Iteração [Restante da Iteração Inicial na Iniciação]
Gerenciar Iteração
[Projeto cancelado]
Projeto Cancelado
[Iteração Bem-sucedida]
Avaliar Escopo e Risco do Projeto
Avaliar Escopo e Risco do Projeto
Elaborar Plano de Desenvolvimento de Software
[Fim do Projeto]
[Fase Final]
[Iteração Final]
Monitorar e Controlar Projeto
[Projeto cancelado]
[Planos do Projeto Aprovados]
Finalizar Projeto
Finalizar Fase
[Projeto Concluído]
[Projeto cancelado]
Projeto cancelado
Projeto Final
[Falha na aceitação]
[Fase completa]
Projeto cancelado
[Opcional, dependendo do grau de mudança]
Planejar Próxima Iteração
Elaborar Plano de Desenvolvimento de Software
Iteração Final

Gestão de Projetos: Fluxo Típico

Gestão de Projetos: Detalhes de Workflow

Aula 8 - Fluxo do RUP: Modelagem de Negócios

- Propósito
- Porque modelagem de negócios?
- Engenharia de Software e Modelagem de Negócios
- Cenários de modelagem de negócios
- Trabalhadores e Artefatos
- Fluxo Típico
- Modelos de negócios e sistemas

Modelagem de Negócios: Propósito

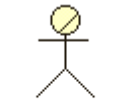
- Compreender a estrutura e comportamento (dinâmica) da organização para a qual o projeto será desenvolvido
- Compreender os problemas atuais da organização e identificar as possibilidades de melhoria
- Garantir que os stakeholders adquiram uma compreensão comum da organização
- Derivar requisitos do sistema (organização)

Porque modelagem de negócios?

- Ampliar os horizontes do software
- Facilitar a reestruturação dos negócios da organização, criando novas oportunidades
- C2B – Customer to business
- B2B – Business to business
- B2C – Business to Consumer
- C2C – Consumer to consumer

Engenharia de Software e Modelagem de Negócios

- Ator de negócios (business actor)
  - Interage com a organização
- Casos de uso de negócios
  - interações do ator de negócios
- Realização de casos de uso de negócios
  - Modelo refinado de caso de uso
- Trabalhador de negócios
  - Trabalha na organização
- Entidades de negócios
  - Coisas produzidas e consumidas
- Unidade Organizacional



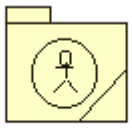
Business Actor

BusinessUseCase

BusinessUseCaseRealization

BusinessWorker

BusinessEntity



OrganizationalUnit

Cenários de modelagem de negócios

- Organograma
  - Estrutura organizacional simples
- Modelagem de domínio
  - Modelagem de informações da organização
- Vários projetos para uma organização
  - Facilitar levantamento de requisitos de vários projetos
- Modelo de negócios genérico
  - Para atender a várias organizações em um mesmo domínio
- Novos negócios
  - Definir viabilidade de novos modelos de negócios
- Reengenharia completa de negócios

Modelagem de Negócios: Trabalhadores e Artefatos

Modelagem de Negócios: Fluxo Típico

Modelos de negócios e sistemas

Aula 9 - Fluxo do RUP: Requisitos

- Propósito
- O que é um Requisito?
- Tipos de Requisitos
- Capturando e Gerenciando Requisitos
- Desenhando uma Interface Centrada no Usuário
- Fluxo Típico
- Papéis e Artefatos
- Suporte de Ferramentas

Requisitos: Propósito

- Estabelecer acordo entre envolvidos sobre o que o sistema deve realizar, e porque
- Prover bases para design do sistema
- Delimitar as funções a serem realizadas pelo sistema a desenvolver
- Permitir o planejamento técnico das iterações
- Permitir estimativas sobre custo e prazo do sistema
- Permitir a definição da interface do sistema com seus usuários

O que é um Requisito?

- Condição ou capacidade que um sistema deve desempenhar
- Qualidade de software
  - Funcionalidade: requisitos funcionais
  - Requisitos não funcionais
    - Usabilidade
    - Confiabilidade
    - Performance
    - Suportabilidade – manutenabilidade

Tipos de Requisitos

- Serviços (features)
  - Expressões de comportamento do sistema em alto nível (o quê)
- Solicitações dos stakeholders
- Requisitos do software
- Requisitos de casos de usos

Capturando e Gerenciando Requisitos

- Pirâmide de Requisitos e Rastreabilidade
  - Features
  - Requisitos de software
  - Requisitos de design
  - Requisitos de testes
  - Requisitos de documentação

Desenhando uma Interface Centrada no Usuário

Requisitos: Fluxo Típico

Requisitos: Papéis e Artefatos

Requisitos: Suporte de Ferramentas

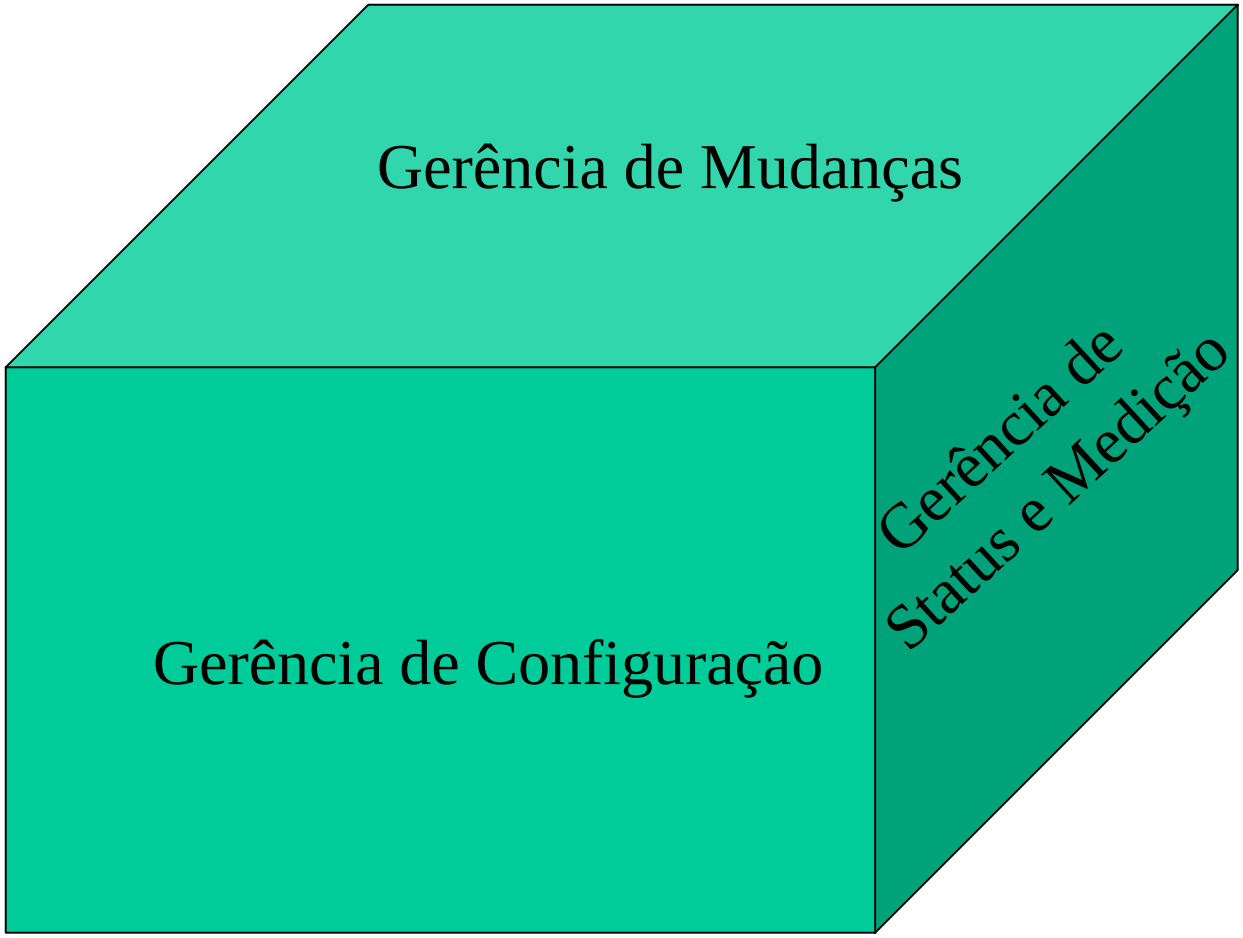
Aula 10 - Fluxo do RUP: Gestão de Configuração e Mudanças

- Propósito
- Cubo de Gerência de Configuração
- Trabalhadores e artefatos
- Fluxo típico
- Suporte de ferramentas

Gestão de Configuração e Mudanças: Propósito

- Rastrear e manter a integridade dos artefatos do projeto ao longo do tempo
- Metas
  - Disponibilidade
  - Reuso

Cubo de Gerência de Configuração e Mudanças



Cubo de Gerência de Configuração

- Gerência de Configuração
  - Preocupa-se com a identificação, versionamento, interdependência e consistência entre os artefatos
  - Integração de workspaces
  - Gerência de builds

Cubo de Gerência de Mudanças

- Gerência de Mudanças
  - Change Control Board - CCB
  - Captura e gestão de solicitações de mudanças
    - Mudança de status das solicitações:
      - Histórico
      - Submetida, registrada, aprovada, atribuída, concluída, abortada
  - Análise de impacto provocado por mudanças
- Gerência de Status e Medição
  - Auditoria
  - Avaliação de progresso e pendências por time, severidade, camada, causas

Gestão de Configuração e Mudanças: Trabalhadores e artefatos

Gestão de Configuração e Mudanças: Fluxo típico

Gestão de Configuração e Mudanças: Suporte de ferramentas

- ClearQuest
  - Gerência de Configuração
- ClearCase
  - Gerência de Solicitações de Mudanças

Aula 12 - Fluxo do RUP: Análise e Projeto

- Propósito
- Análise versus design
- Até onde aprofundar o design?
- Trabalhadores e artefatos
- Modelo de design
- Modelo de análise
- Papel das interfaces (contratos)
- Design baseado em componentes
- Fluxo Típico
- Suporte de Ferramentas

Análise e Projeto: Propósito

- Traduzir uma especificação de requisitos, que descreve o que deve ser feito para solucionar o problema, em uma descrição (especificação) suficientemente detalhada de como esta solução (o sistema) deve ser construída
- Selecionar as melhores estratégias para implementar soluções
- Selecionar uma arquitetura que atenda aos requisitos
- Ajustar a solução para atender aos requisitos não funcionais, como desempenho, robustez, escalabilidade, testabilidade, manutenabilidade, etc.

Análise versus design

- Análise
  - Mapeia requisitos de casos de uso em um design (abstrato) que preserva os mesmos conceitos presentes no domínio de aplicação do problema
  - Satisfaz (idealmente) requisitos funcionais
- Design
  - Adapta o design abstrato para satisfação dos requisitos não funcionais, ambiente de execução, implementação, testes, etc
  - Otimização dos resultados da análise

Até onde aprofundar o design?

- Que faz o design?
- Quem implementa?
- Qual a maturidade da equipe?
- Qual a maturidade do domínio?
- Round-Trip Engineering

Análise e Projeto: Trabalhadores e artefatos

- Arquiteto
  - Líder técnico
  - Modelos de análise e design
  - Módulos e interfaces
  - Visões arquiteturais
- Designer
  - Projetista de classes (classes, responsabilidades, colaboradores, atributos, operações, packages e subsistemas)
- Designer de base de dados / cápsulas
- Revisores

Modelo de análise

- Esboço da implementação das funções do sistema
- Orientada ao domínio de aplicação
- Serve para compreensão mais aprofundada do domínio da aplicação
- Modelo de Análise de Jacobson
  - Classes de fronteira, entidade e controle
- Padrões de Análise, de Martin Fowler

Modelo de design

- Classe
  - Objetos que apresentam o mesmo conjunto de responsabilidades, colaboradores, operações e atributos
- Package
  - Agregação de classes que contém uma coesão conceitual
- Subsistema
  - Agregação de packages cujas classes atuam como uma unidade, e que provê serviços através de uma interface bem definida (componente)

Refinamentos do Modelo de Design

- Padrões Arquiteturais de Design
- Mecanismos de Design (tecnologias)
- Padrões de design (design patterns)

Padrões Arquiteturais

- Persistência
- Comunicação entre Processos
- Concorrência e Sincronização
  - Multitarefa, multiprocesso, multithreading
- Messaging
- Gerência de Transação
- Segurança
- Redundância

Design baseado em interfaces e componentes

- Interfaces facilitam a evolução de subsistemas
- Componentes implementam interfaces
- O que é um componente?
  - Unidade funcional físicamente distribuível
- Estudo de caso
  - Interfaces, stubs e fábricas em Java
- Como os componentes são usados hoje?
  - DLLs, Jars
- Modelo de componentes de software JavaBeans
- Modelo de dados

Análise e Projeto: Fluxo Típico

- Executar Síntese Arquitetural
- Definir arquitetura candidata
- Refinar arquitetura
- Analisar comportamento
- Desenhar componentes

Análise e Projeto: Suporte de Ferramentas

Aula 13 - Fluxo do RUP: Implementação

- Propósito
- Builds
- Integração
- Protótipos
- Trabalhadores e artefatos
- Fluxo típico
- Suporte de ferramentas

Implementação: Propósito

- Definir a organização do código em subsistemas e camadas
- Implementar classes e interfaces, usando programas, compiladores, etc
- Testar componentes unitariamente desenvolvidos
- Integrar componentes implementados e unitariamente testados

Builds

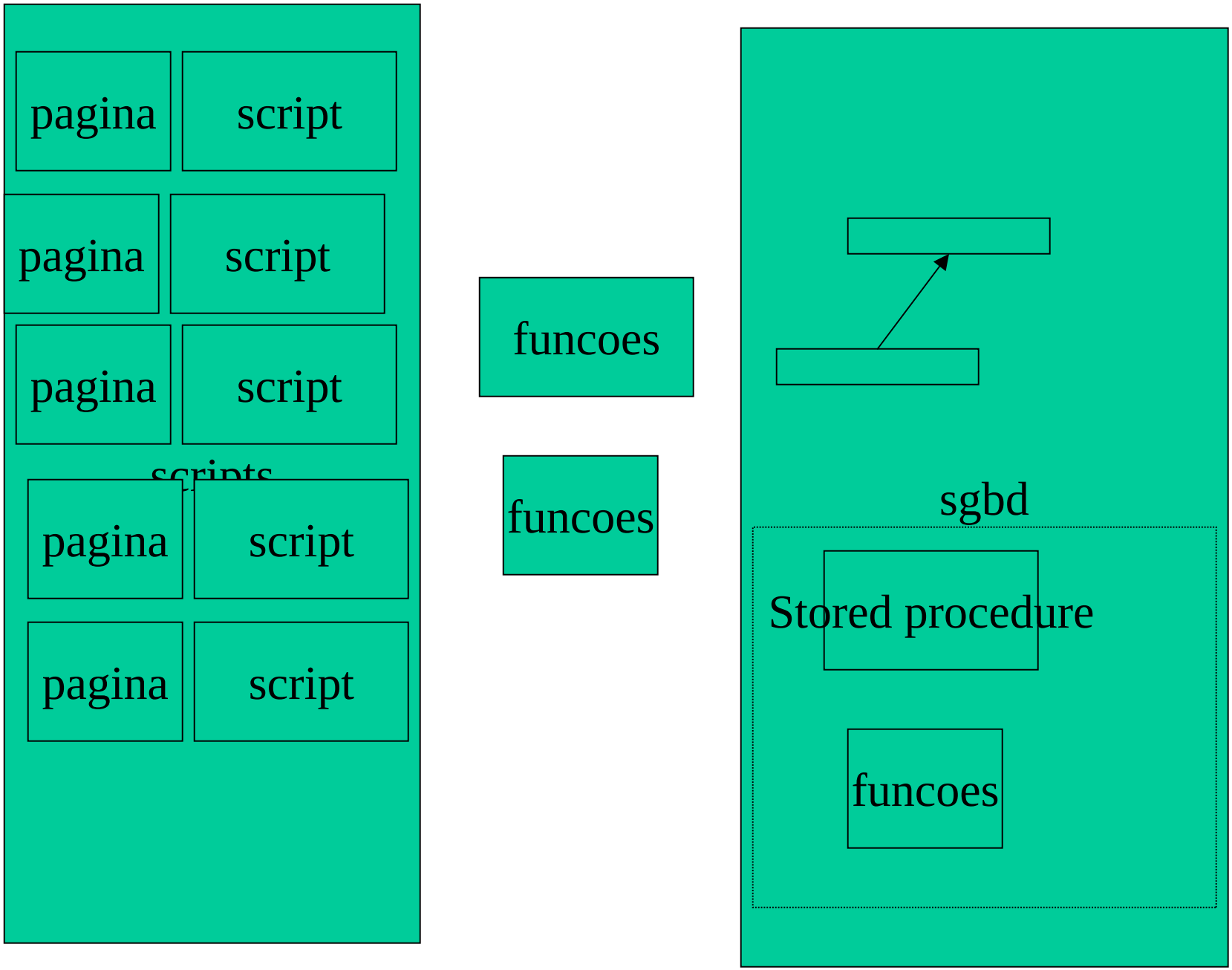
- Versão operacional do software
- Vários builds gerados

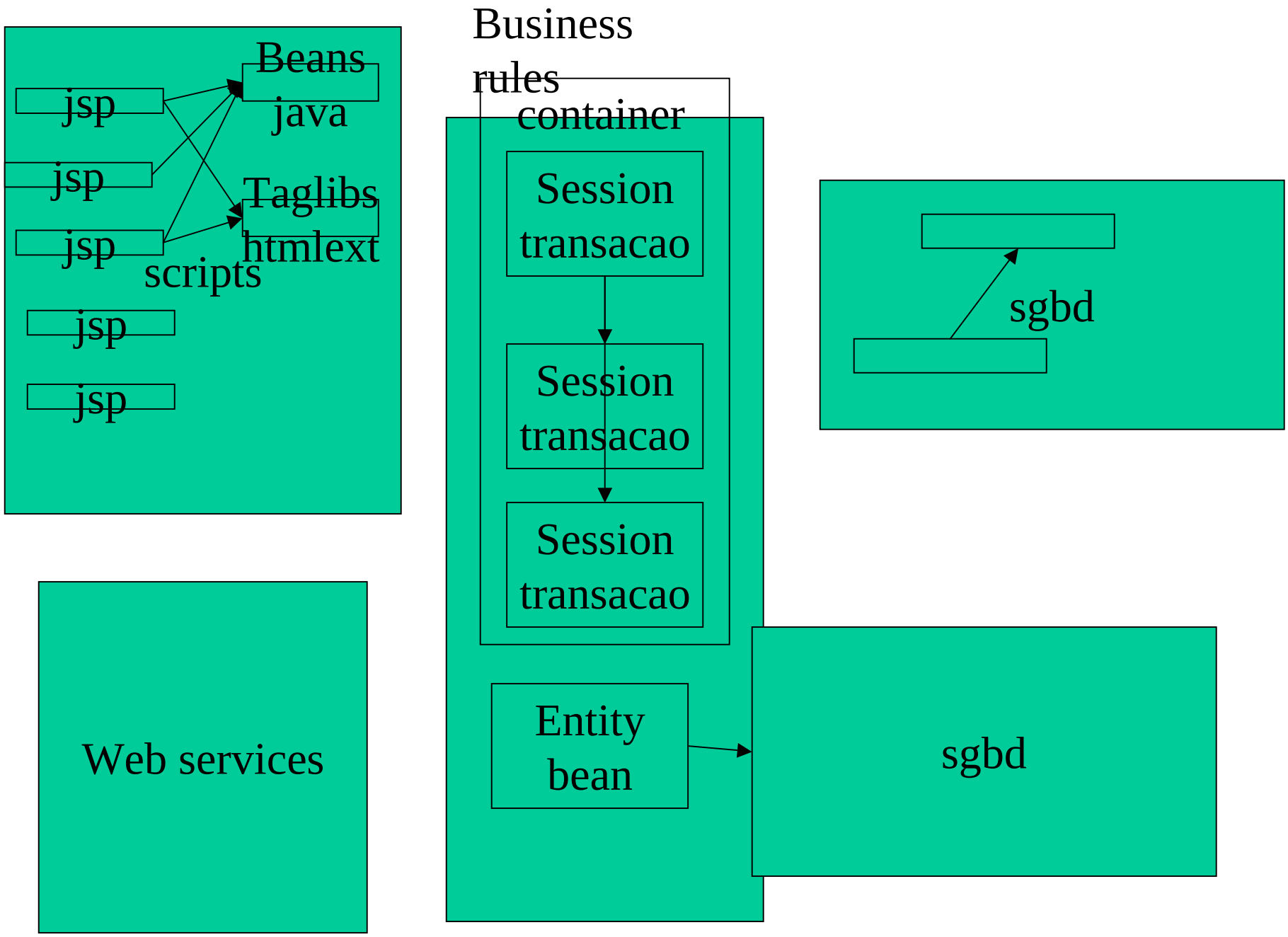
Integração

- Integrar trabalho de times e subsistemas
- Integração
  - Em fases (periodicamente)
  - Incremental (quase todo dia)
  - contínua (várias vezes por dia - Martin Fowler)

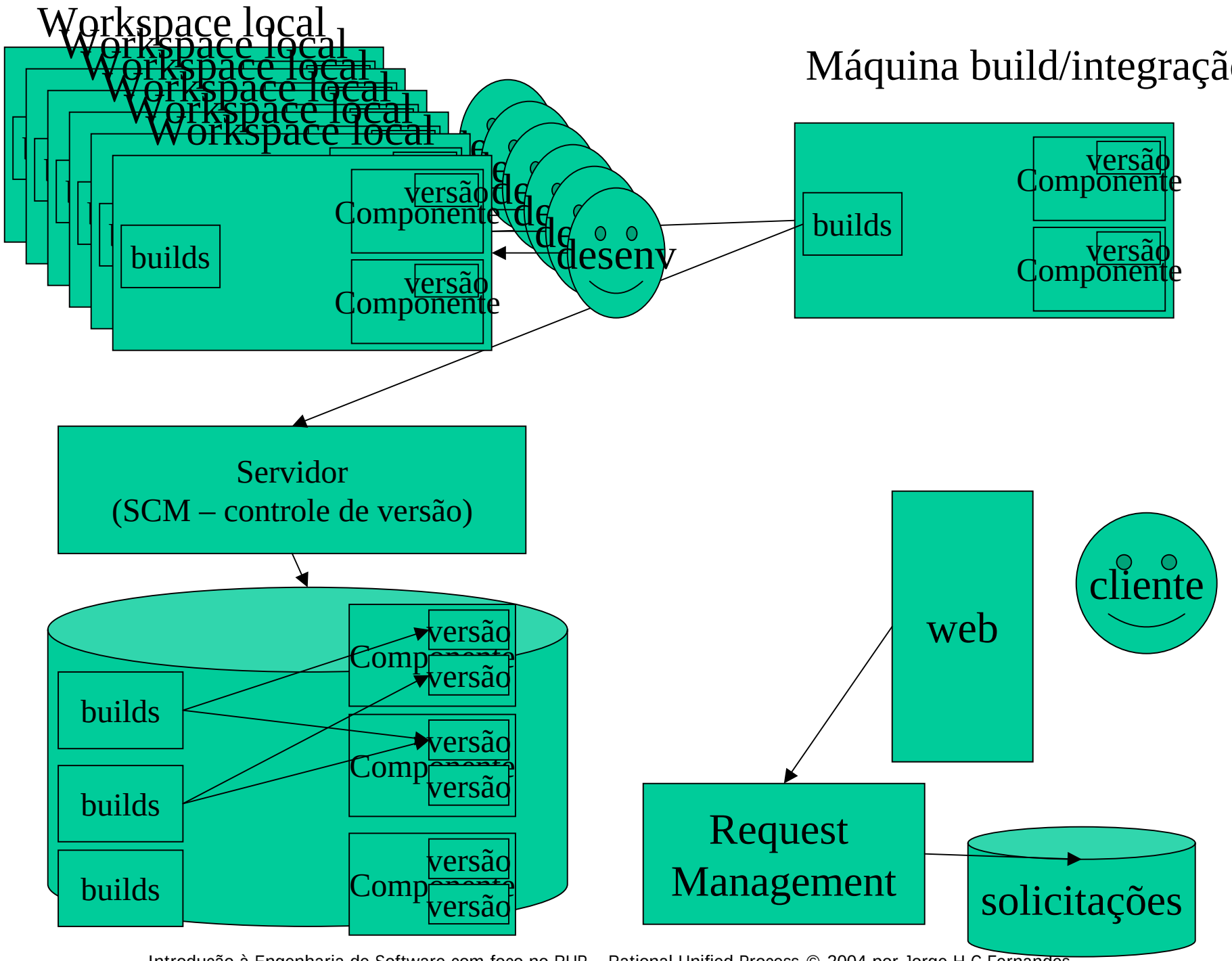
Protótipos

- Viabilidade
- Estabilidade e performance
- Prova de conceitos
- Compreensão dos requisitos
- Usabilidade
- Tipos
  - Comportamental
  - Estrutural
  - Exploratório
  - Evolucionário


pagina
script
pagina
script
pagina
script
scripts
pagina
script
pagina
script
funcoes
funcoes
sgbd
Stored procedure
funcoes


Beans
java
jsp
jsp
jsp
Taglibs
htmlext
scripts
jsp
jsp
Business
rules
container
Session
transacao
Session
transacao
Session
transacao
sgbd
Web services
Entity
bean
sgbd

Implementação: Trabalhadores e artefatos


Workspace local
Workspace local
Workspace local
Workspace local
Workspace local
Workspace local
builds
Componente
versão
Componente
versão
de desenv
Máquina build/integração
builds
Componente
versão
Componente
versão
Servidor
(SCM – controle de versão)
builds
builds
builds
Componente
versão
versão
Componente
versão
versão
Componente
versão
versão
web
cliente
Request
Management
solicitações

Implementação: Fluxo típico

Implementação: Suporte de ferramentas

- Editores
- Compiladores
- Ligadores
- Depuradores
- Rose (round-trip engineering)
- ClearCase/CVS/PVCS/VisualSourceSafe-VSS
- ClearQuest/Bugzilla
- Ant

Aula 14 - Fluxo do RUP: Testes

- Propósito
- O que é Qualidade?
- Teste no ciclo de vida iterativo
- Dimensões de teste
- Modelo de testes
- Trabalhadores e artefatos
- Fluxo típico
- Suporte de ferramentas

Testes: Propósito

- Atestar qualidade do que foi produzido
  - Verificar interações entre componentes
  - Verificar integração entre componentes
  - Verificar implementação correta dos requisitos
  - Identificar e corrigir defeitos

O que é Qualidade?

- Ausência de defeitos?
- Adequação ao propósito de uso
- Good-enough Testing
- Papel dos testes
  - Atestar a qualidade? Não!
  - Avaliar a qualidade? Sim!
    - Avaliar e prover feedback

Qualidade de Produto versus Qualidade de Processo

- Qualidade de produto
  - Testes
- Qualidade de processo
  - ISO-9000/9000-3
  - ISO-12207
  - CMM/CMMI

Teste no ciclo de vida iterativo

- Teste de Protótipos e produtos intermediários
  - Estabilidade
  - Cobertura
  - Desempenho
- Produto final
  - Adequação
  - Cobertura

Dimensões de teste: Focos de qualidade

- Confiabilidade
  - É resistente a falhas?
- Funcionalidade
  - Atende casos de uso?
- Desempenho
  - É aceitável nas condições previstas?

Dimensões de teste: Focos de qualidade

- Evolução
  - Manutenabilidade
    - É fácil de adaptar e corrigir?
  - Extensibilidade
    - nÉ fácil de estender funcionalmente?
- Flexibilidade
  - É flexível quando à aplicabilidade?

Dimensões de teste: Estágios de Teste

- Unitário
  - durante implementação de módulos
- Integração
  - Durante união de módulos
- Sistema/ Homologação Interna
  - Durante funcionamento em bancada
- Certificação/Aceitação/Homologação Externa
  - Durante funcionamento no ambiente do usuário

Dimensões de teste: Tipos de teste

- De beckmark
  - Comparativo
- Configuração
  - Nas várias plataformas
- Funcional
  - Executa os casos de uso (e variações) como esperado

Dimensões de teste: Tipos de teste

- Instalação
  - Software com instalador em diversas plataformas e condições de espaço

Dimensões de teste: Tipos de teste

- Integridade
  - Confiabilidade, robustez, tolerância a falhas
- Carga/Performance/Stress
  - Carga
    - Desempenho sobre várias condições operacionais (variações na quantidade de usuários, tipo de carga, tamanho da carga) fixando a configuração
  - Performance
    - Desempenho de várias configurações sobre uma condição operacional fixa (quantidade de usuários, tipo de carga, tamanho da carga)
  - Stress
    - Desempenho sobre condição operacional e de configuração extremas

Dimensões de teste

- Regressão
  - Retestar o sistema em novas versões do software
    - Verifica a manutenção ou degradação da qualidade

Reproducibilidade de Testes

Elementos de Design e Implementação de Testes

- O que será testado e como
- TestPlan
  - Planejamento de as atividades relativas a um teste, para alcance de objetivos específicos
- Procedimentos de teste
  - Instruções detalhadas para montagem, execução e avaliação de resultados
- TestCase
  - Conjunto de dados de teste, condições de execução, resultados esperados

Elementos de Design e Implementação de Testes

- Scripts de teste
  - Passo a passo para execução de testes
  - Manual ou automatizado
- Classes e componentes de teste
  - Drivers, stubs
- Colaborações e dependências
  - Seqüência de comunicações e dependências entre módulos no momento do teste
- Observações e Notas
- Resultados de teste

Testando sistemas na Web
Nguyen

Making Your Web Application Test Report More Reproducible

- Check if the client operating system, versions, and patches meet system requirements
- Check if the correct version of the browser is installed on the client machine
- Check if the browser is properly installed on the machine (for example, the JVM is also successfully installed)
- Check the browser settings
- Try the same set of steps with different browsers (e.g., Netscape Navigator versus Internet Explorer)
- Try the same set of steps with different supported versions of the same browsers (e.g., 3.1, 3.2, 4.2, 4.3, etc.)
- Check to ensure that all servers are running
- Check to ensure that all service-based components have been started
- Check to ensure that application access privileges are properly set up
- Check for missing components on the server (DLLs, scripts, etc.)
- Check for proper registration of components (COMs, Java, etc.)
- Check to ensure that DNS is properly configured
- Check if firewall configuration is causing packets to drop or blocking access
- Check if a slow connection is causing the application to time-out
- Check for potential race or time-related conditions
- Check for potential network inaccessibility issues on the client machines
- Check for potential network inaccessibility issues on the server machines
- Check if the server operating system version and patches meet system requirements
- Check if the proper versions of the server software such as Web server, SQL database, and other middle-ware packages are installed
- Check server configurations for proper settings

Testes: Trabalhadores e artefatos

Testes: Fluxo típico

Testes: Suporte de ferramentas

Aula 15 - Fluxo do RUP: Instalação

- Propósito
- Modos de instalação
- Trabalhadores e artefatos
- Fluxo Típico

Instalação: Propósito

- Testar o software em sua condição operacional final
- Empacotar o software para entrega
- Distribuir o software
- Instalar o software
- Treinar usuários finais e força de vendas
- Migrar software
- Converter bases de dados

Instalação: Modos de Instalação

- Sistemas customizados
- Pacotes (shrink-wrapped)
- Download via web

Instalação: Trabalhadores e artefatos

Instalação: Fluxo típico

Aula 11 - Fluxo do RUP: Ambiente

- Propósito
- Trabalhadores e artefatos
- Fluxo típico

Ambiente: Propósito

- Prover atividades de suporte à organização, com processos e ferramentas
- Seleção e aquisição de ferramentas
- Instalação e configuração de ferramentas
- Configuração de processos
- Melhoria de processos
- Suporte técnico
  - IT, contabilização, backup, etc

Ambiente: Trabalhadores e artefatos

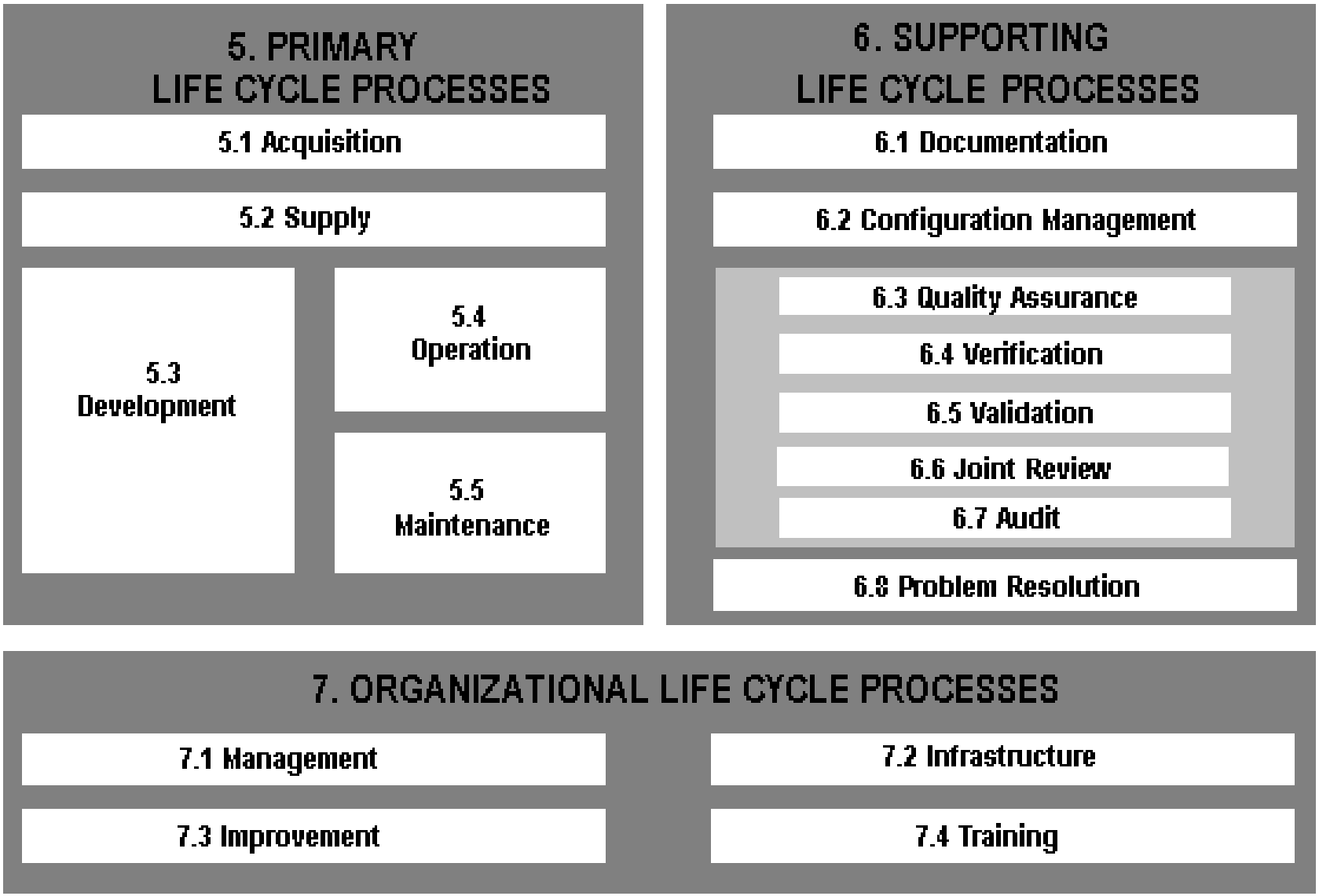
- Engenheiro de processos
- Documentador técnico
- Analista de sistema
- Analista de processos de negócios
- Projetista de testes
- Especialista de ferramentas
- Projetista de interface com o usuário
- Administrador do sistema
- Arquiteto

Ambiente: Fluxo típico

RUP e Modelos de Qualidade de Processo

- ISO-12.207
- RUP e CMM-2 [Cognence, 2002]

RUP e Modelos de Qualidade de Processo ISO 12.207



CMMI:
Níveis de Maturidade de Processos

- Nível 1: Resultados imprevisíveis

Maturity Level	Staged Representation Maturity Levels
1	Initial
2	Managed
3	Defined
4	Quantitatively Managed
5	Optimizing

CMMI Nível 2: Desempenho repetido

- Gerencia de requisitos: gerenciar requisitos e identificar inconsistências entre planos e artefatos
- Planejamento de projetos: estabelecer e manter planos que definem atividades do projeto
- Monitoramento e controle de projetos: compreender o processo e tomar ações corretivas
- Gestão de contrato com fornecedores
- Garantia de qualidade de produto e processo
- Gerencia de configuração: estabelecer e manter integridade dos produtos
- Medição e análise: desenvolver e sustentar capacidade de medição que apóie a gerência

CMMI Nível 3: Melhoria de desempenho de projetos

Desenvolvimento de requisitos: evolução de requisitos com multiplos envolvidos (multi-stakeholder)

Solução técnica: evolução de design e engenharia de qualidade

Integração de produto: integração contínua, controle de interfaces, gerência de mudanças

Verificação: técnicas de avaliação para garantir que o produto está sendo construído corretamente

Validação: técnicas de avaliação para garantir que o produto correto está sendo construído

Gerência de riscos: detecção, priorização e resolução de questões e contigências relevantes

Treinamento organizacional: mecanismos para desenvolvimento de pessoas mais produtivas

Foco organizacional em processos: estabelecer um 'framework' organizacional para definição de processos para projetos

Análise e resolução de decisões: avaliação sistemática de alternativas

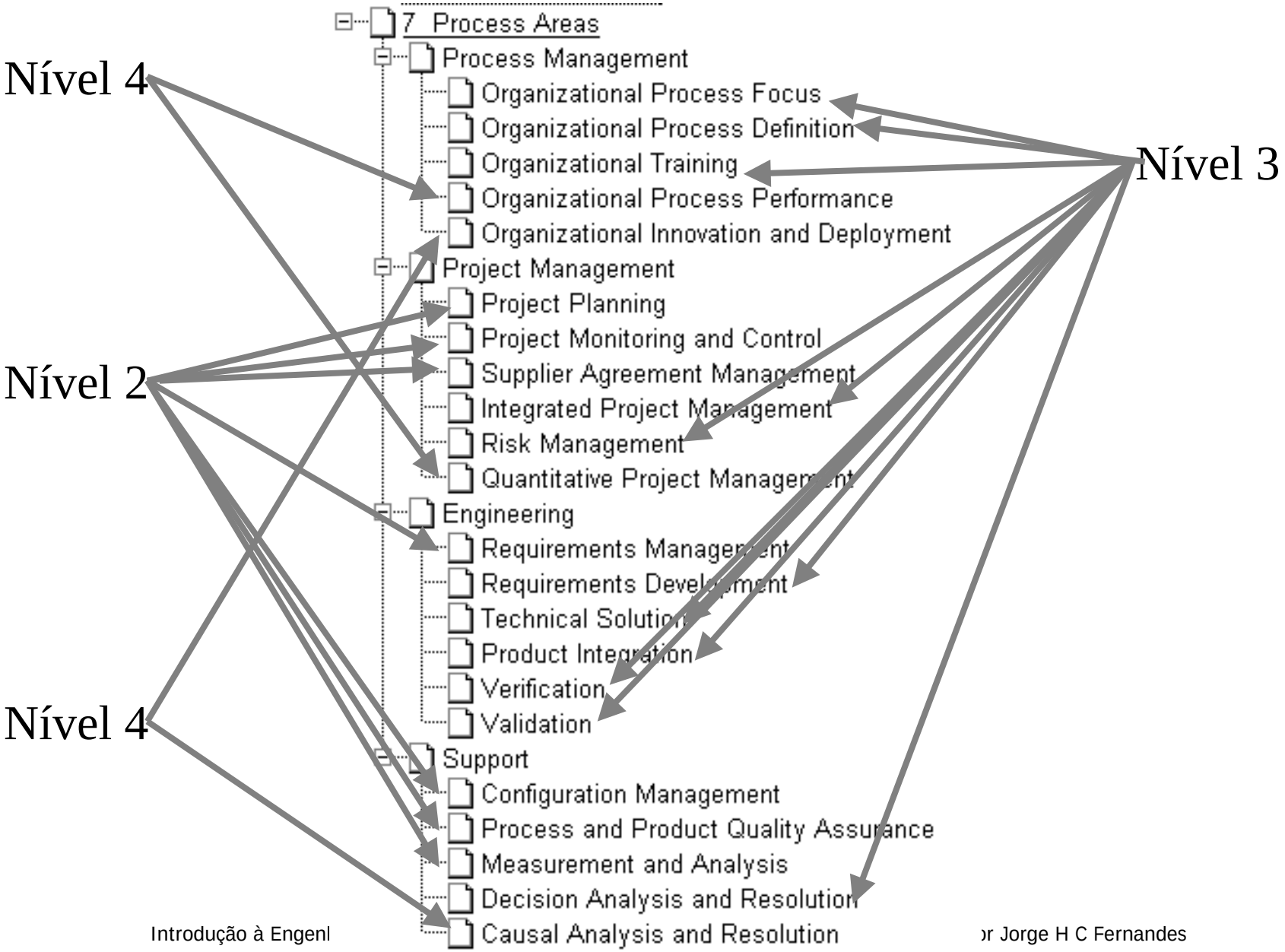
Definição organizacional de processos: tratamento de processos como um ativo persistente e evolutivo da organização

Gerência integrada de projetos: métodos para unificar os vários times e envolvidos em um projeto

CMMI: Níveis 4 e 5

- Nível 4: quantitativamente gerenciado
  - Performance de processos organizacionais: definir normas e comparativos para desempenho de processos
  - Gerência quantitativa de projeto: executar projetos baseados em controle estatístico de qualidade
- Nível 5: contínua melhoria de processos
  - Resolução e análise causal: evitar falhas proativamente e reforçar melhores práticas
  - Inovação Organizacional: criar uma organização que aprende, adapta-se e aperfeiçoa-se organicamente

Áreas de Processo do CMMI



Introdução À Engenharia De Software Com Foco No RUP: Rational Unified Process

Prof. Dr. Jorge Henrique C Fernandes
(jorge@dimap.ufrn.br)
POTI – Pólo De Tecnologia Da Informação
Departamento De Informática E Mat. Aplicada
Universidade Federal Do Rio Grande Do Norte