Qualidade no desenvolvimento de software

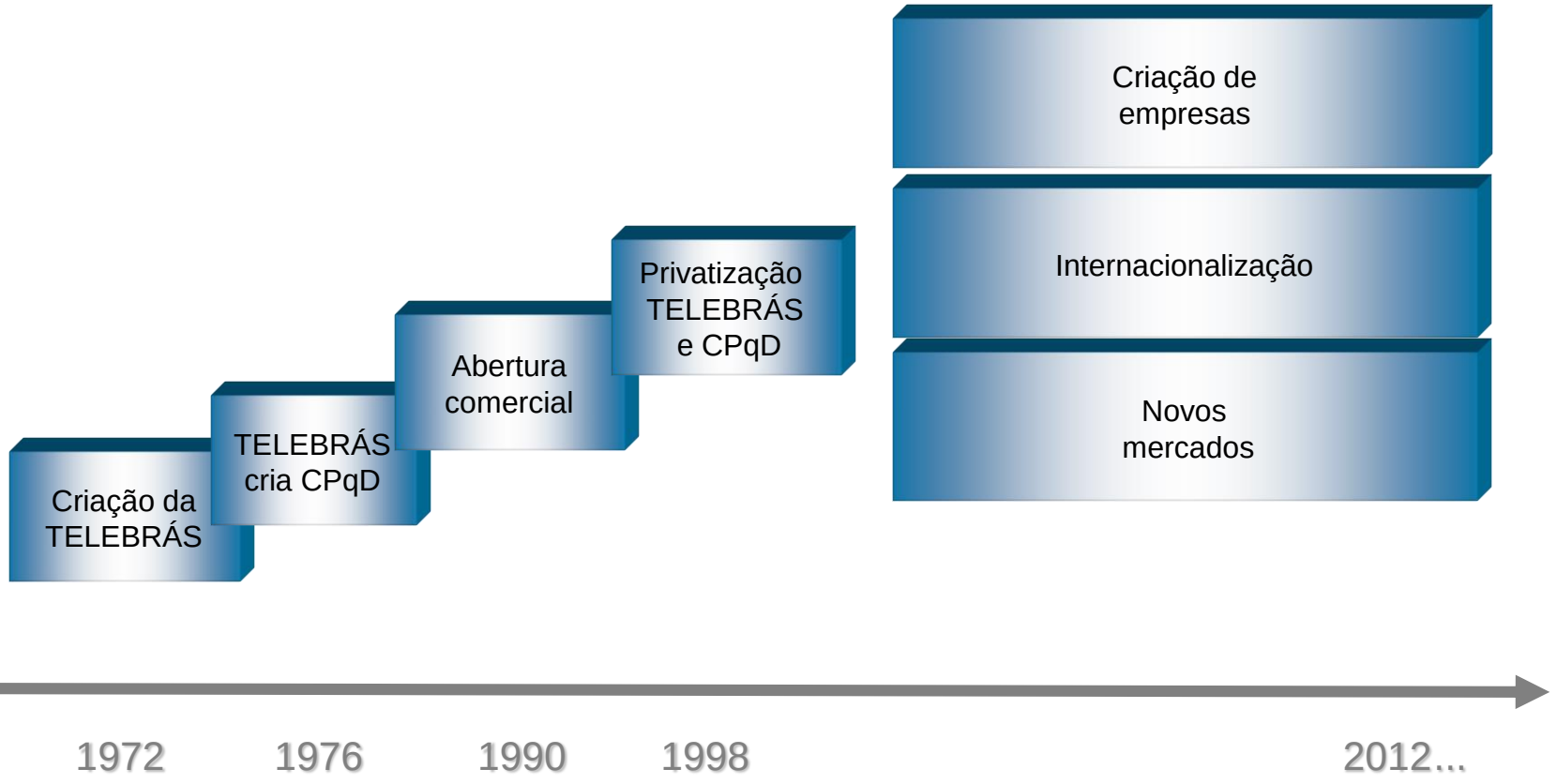
Processos e Ferramentas

TRANSFORMANDO
EM REALIDADE

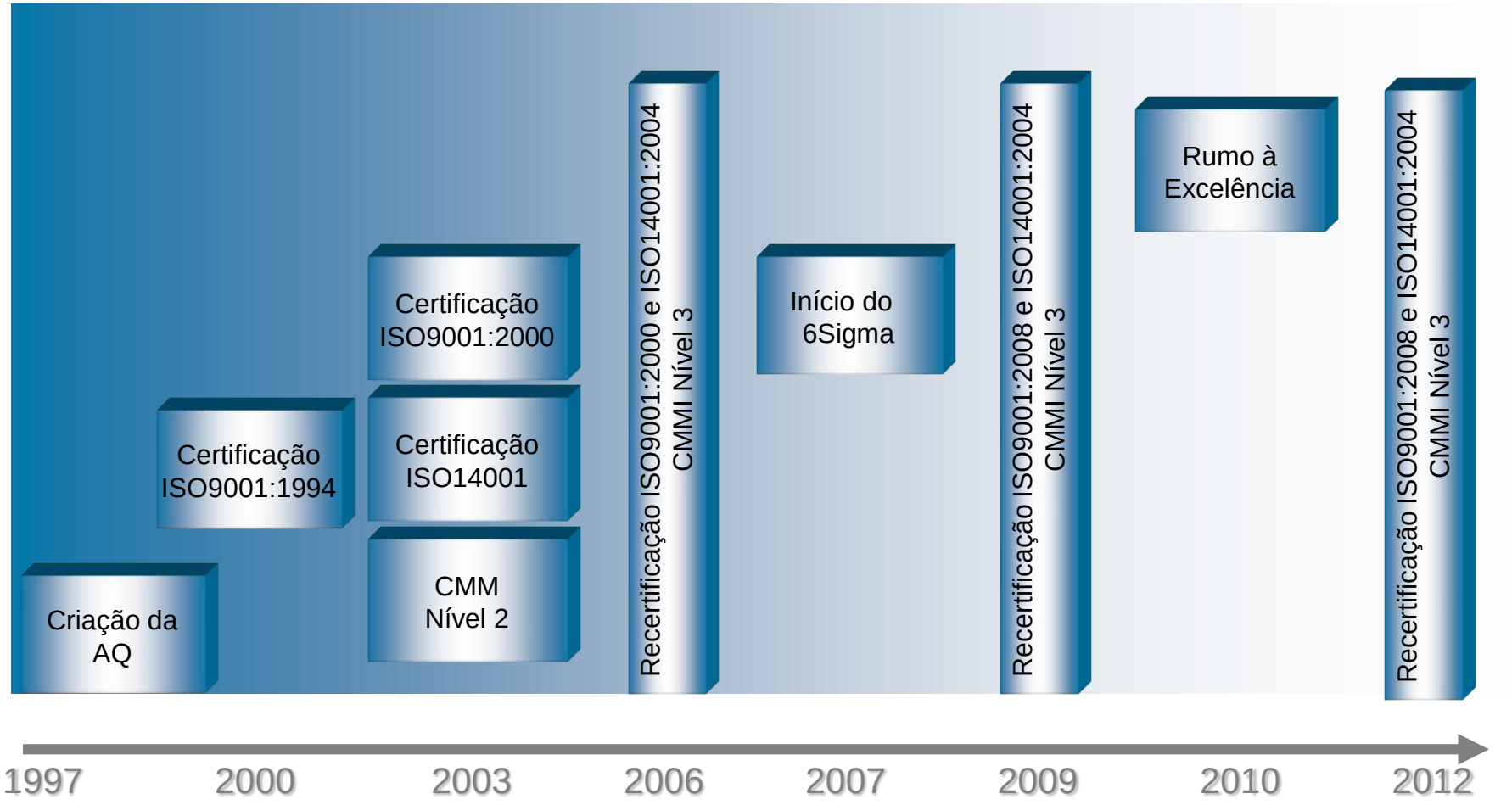
André Villas-Boas <villas@cpqd.com.br>

USP/SCar - Abr-2016

Evolução histórica do CPqD



Evolução na gestão de processos



Sistema de Gestão?

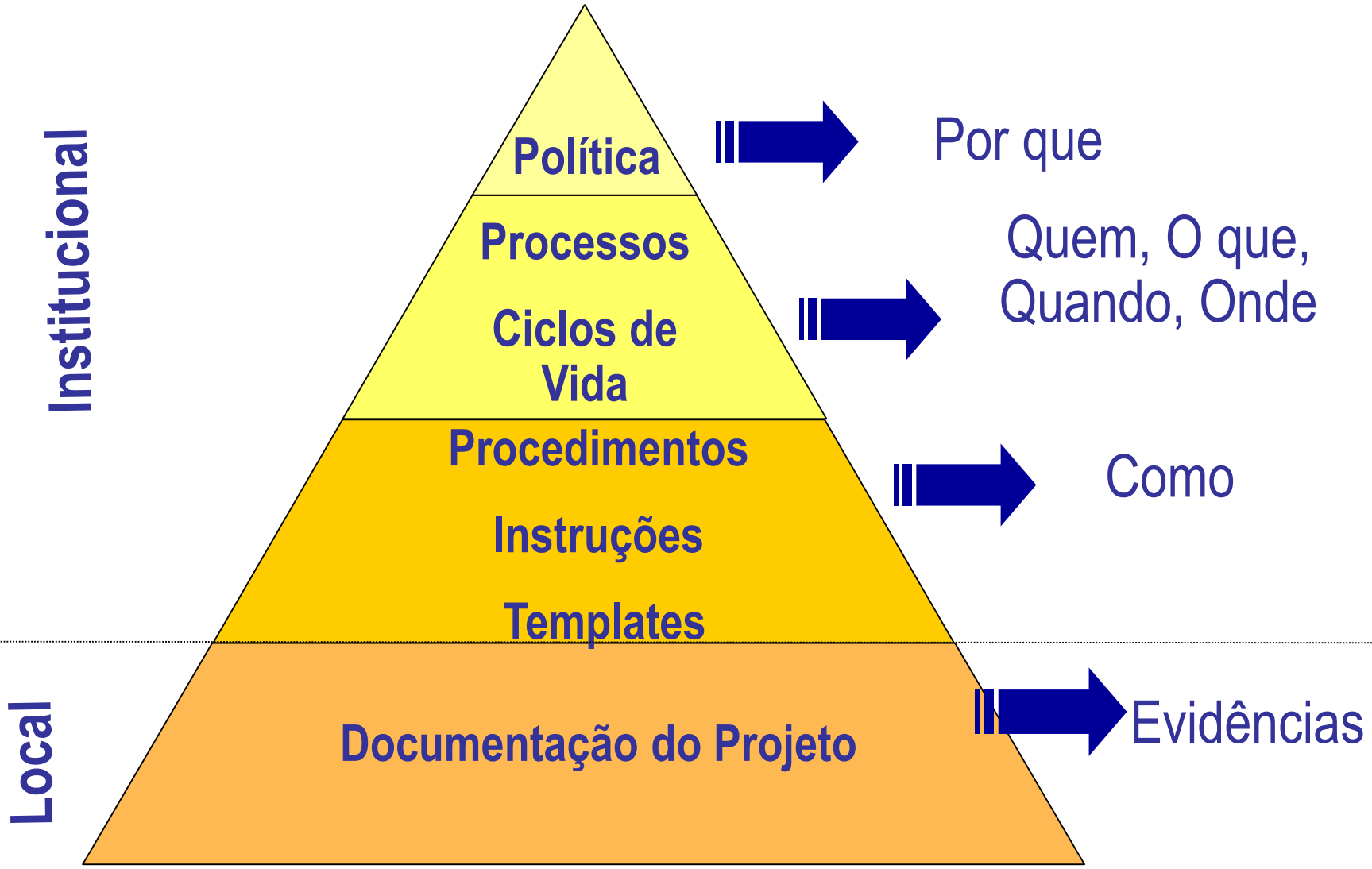
Um conjunto de documentos do tipo processos, procedimentos, práticas, orientações, instruções, nos quais encontraremos a descrição das atividades das pessoas envolvidas e das ferramentas necessárias para realizarem quaisquer atividades dentro de uma determinada organização.

ISO9001:2008 e ISO14001:2004

Ensaio e Calibração ISO17025 - CGCRE

OCD – Organismo Certificador Designado

Estrutura de Documentação do Sistema de Gestão



Política de Gestão

O CPqD busca a **inovação** e a **excelência tecnológica** objetivando atender às **necessidades** e **expectativas** dos seus **clientes** e da **sociedade brasileira**.

Atua com base:

- na **melhoria contínua** de seus **processos**;
- no estímulo à **criatividade** individual e coletiva;
- na relação **harmoniosa** entre seus **colaboradores** e com as **partes interessadas**;
- na responsabilidade **socioambiental**; e
- na permanente melhoria do seu **desempenho financeiro**, visando a sua **perpetuidade**.

Garantia da Qualidade de Processo

Política da qualidade com o objetivo de:

- **Avaliar objetivamente processos e produtos de trabalho em relação aos padrões aplicáveis**, fornecendo informações que podem ser utilizadas para melhoria dos processos – Foco em garantia da qualidade de processo

- Fornecer apoio aos projetos durante o seu desenvolvimento

- Fornecer informação à gerência do projeto e outros envolvidos sobre as atividades de garantia da qualidade

- Identificar e documentar as não-conformidades

- Garantir que as não-conformidades sejam tratadas

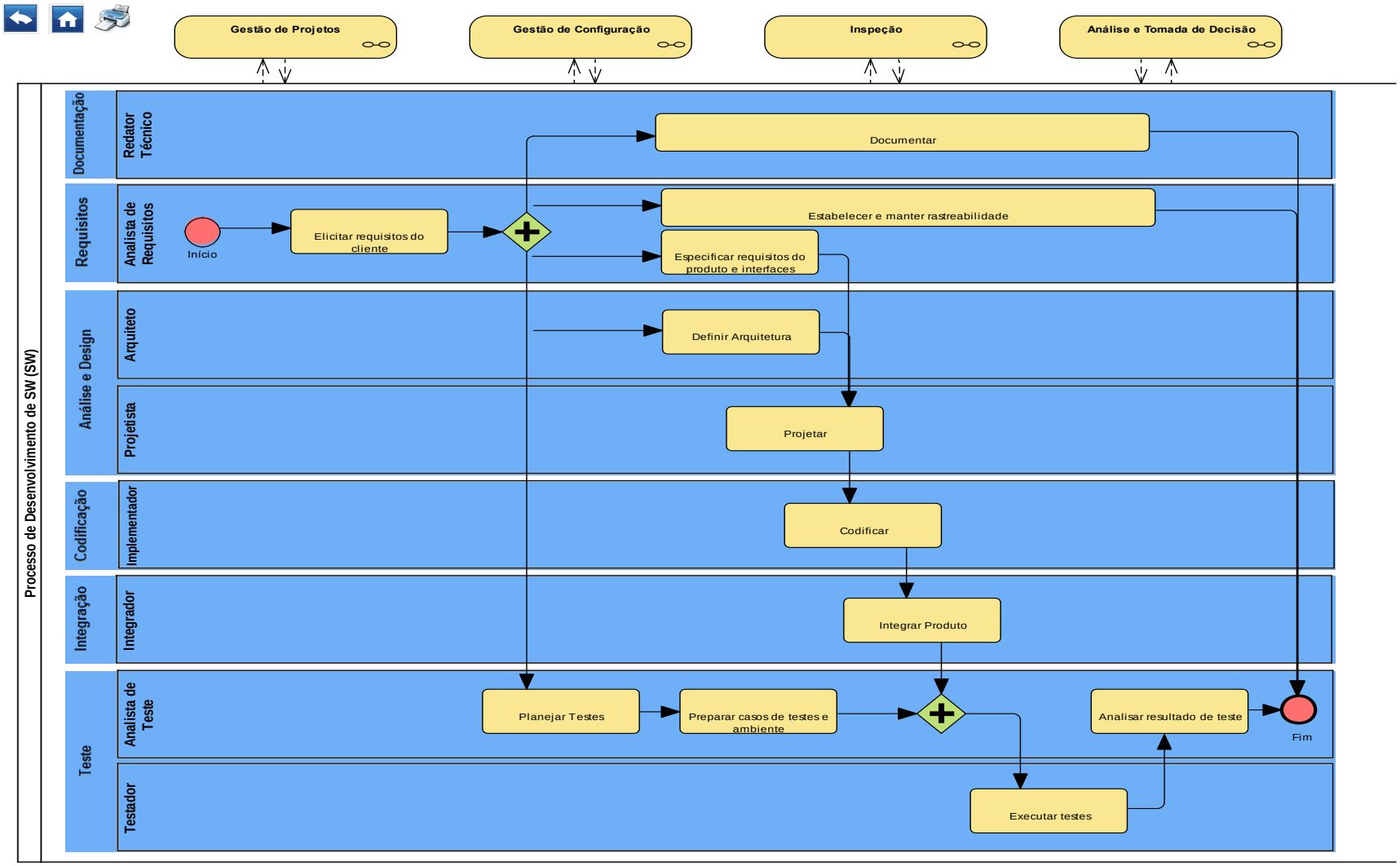
Sistema de Gestão do CPqD

Macroprocesso

Sistemas de Software

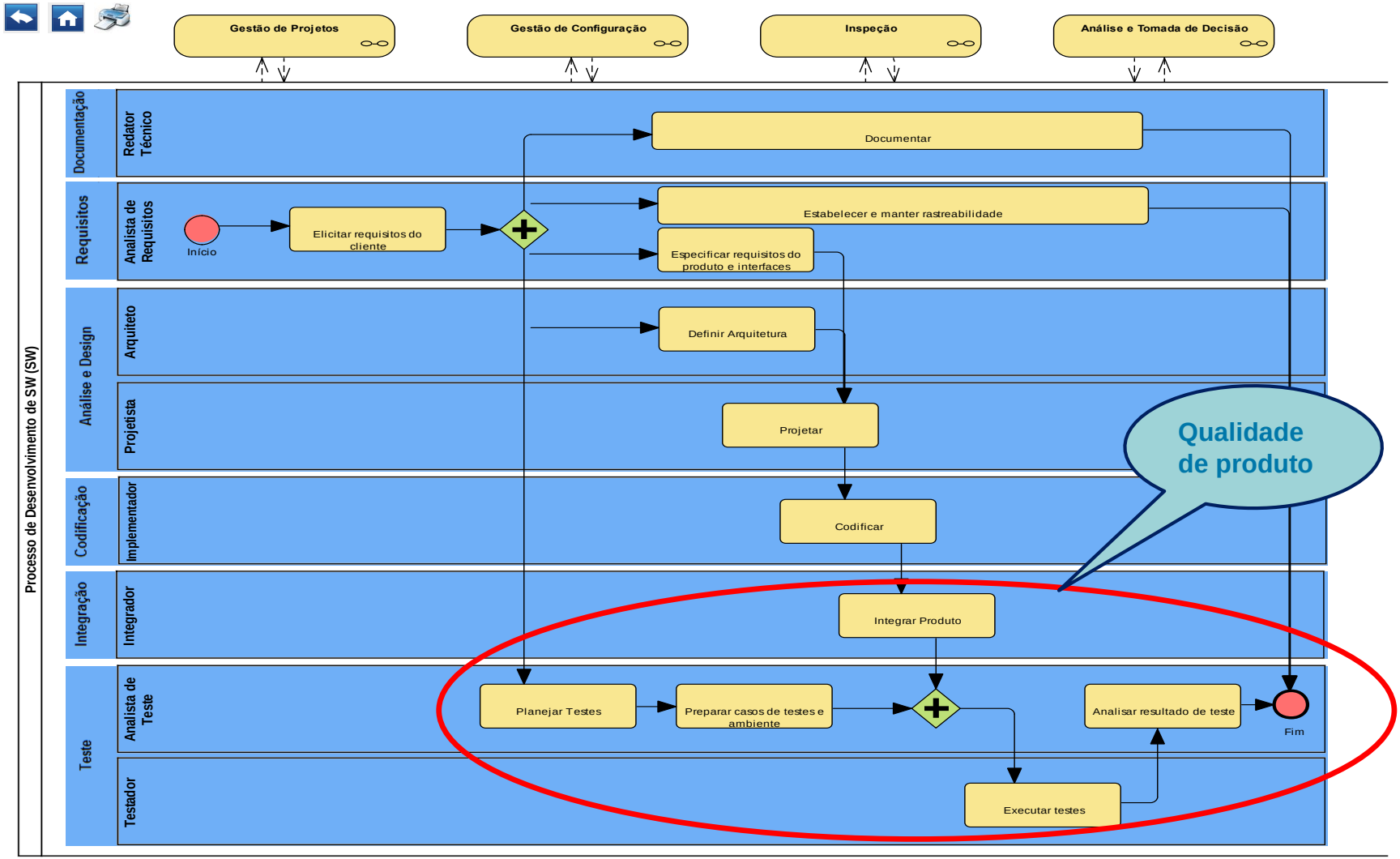
- Foco de atuação:

Desenvolvimento de software [BPMN]



Instruções, Orientações, Boas Práticas *Templates* *Indicadores*

Desenvolvimento de software



Garantia da Qualidade de Software

TRANSFORMANDO
EM REALIDADE

Garantia de Qualidade de Software

É um conjunto de **atividades técnicas** aplicadas durante **todo** o processo de desenvolvimento.

- O objetivo é garantir que tanto o **processo de desenvolvimento** quanto o **produto de software** atinjam níveis de qualidade especificados.

Qualidade de Software

Conformidade com requisitos funcionais e de desempenho, padrões de desenvolvimento documentados, e características implícitas esperadas de todo software profissionalmente desenvolvido.

Correção

- Confiabilidade
- Testabilidade
- Manutenabilidade
- Usabilidade

Atividades de Garantia de Qualidade

Validação: Assegurar que o produto final corresponda aos requisitos do software. “Estamos construindo o produto certo?”.

Verificação: Assegurar consistência, completitude e correção do produto em cada fase e entre fases consecutivas do ciclo de vida do software. “Estamos construindo corretamente o produto?”.

Teste: Examina o comportamento do produto através de sua execução.

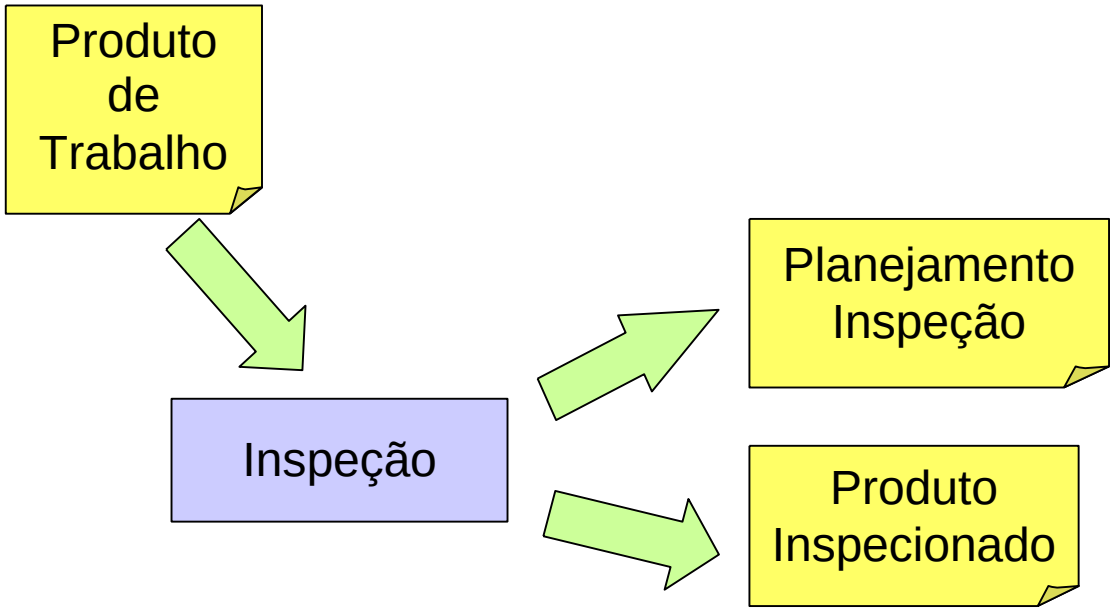
Defeitos no Processo de Desenvolvimento

- Quanto antes a presença do defeito for revelada, menor o custo de correção do defeito e maior a probabilidade de corrigi-lo corretamente.
- Principal causa: **tradução incorreta de informações**.

Solução: Introduzir atividades de VV&T ao longo de todo o ciclo de desenvolvimento.

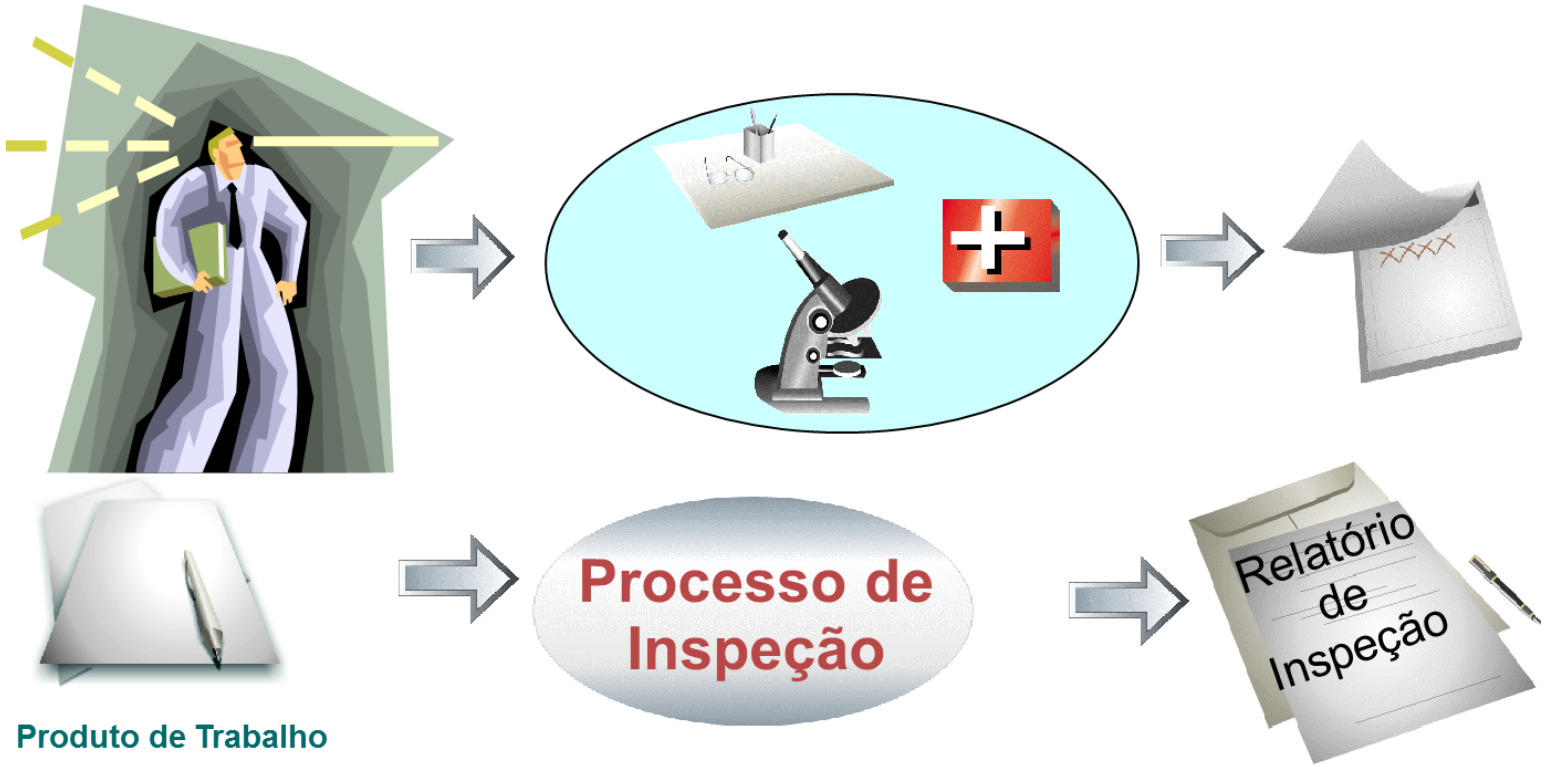
Aferição da qualidade do produto no CPqD

Garantia da qualidade do produto



- **Planejamento:**
  - O que (escopo)
  - Qual o método (par, técnica, etc)
  - Critérios a serem utilizados (amostragem)
  - Ambiente
  - Equipe
- **Execução:**
  - JIRA: registro das observações e tratamento
  - Análise das revisões nas reuniões de marco

Inspeção



- O projeto pode optar pelo uso de uma ferramenta de inspeção para registro das observações.
  - Nesse caso, o Relatório de Inspeção fica armazenado na própria ferramenta

Visão Geral do Processo de Inspeção

- Planejamento das inspeções
  - Plano de Inspeção (seleção de produtos de trabalho, métodos, critérios, recursos, cronograma etc.)
- Execução das inspeções
  - Coleta das observações
  - Tratamento das observações
- Avaliação das inspeções
  - Verificação dos critérios e de qualidade dos produtos e processos

Planejamento da Inspeção

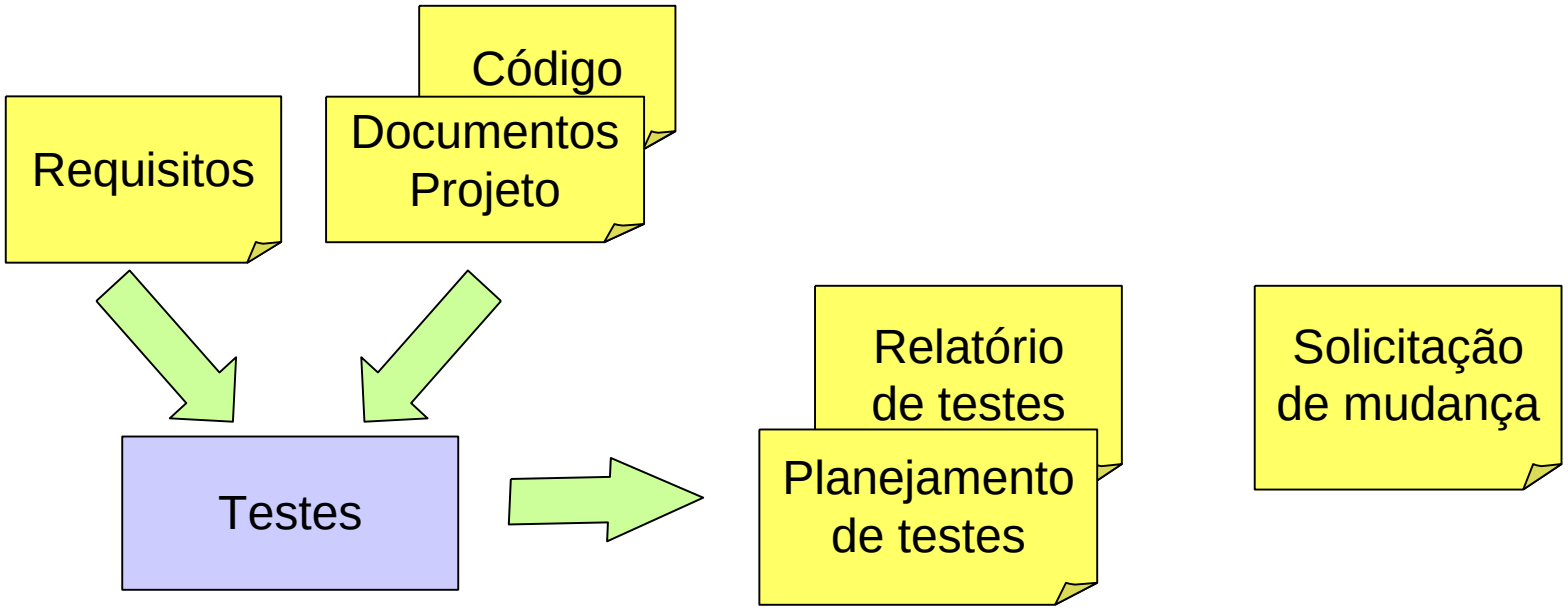
- Selecionar **métodos de inspeção** para cada produto de trabalho a ser inspecionado
  - Walkthrough
  - Revisão por Pares
  - Revisão Técnica

- Definir os seguintes **critérios da inspeção** para cada produto de trabalho
  - Início
  - Interrupção
  - Término

Planejamento da Inspeção

- Definir **critérios de amostragem** para cada produto de trabalho
  - Amostragem vertical – quais versões dos artefatos serão inspecionadas
  - Amostragem horizontal – quais artefatos de um mesmo tipo serão inspecionados
- Definir **listas de inspeção** a serem utilizadas nas inspeções
  - **Por produto de trabalho**

Garantia da qualidade do produto



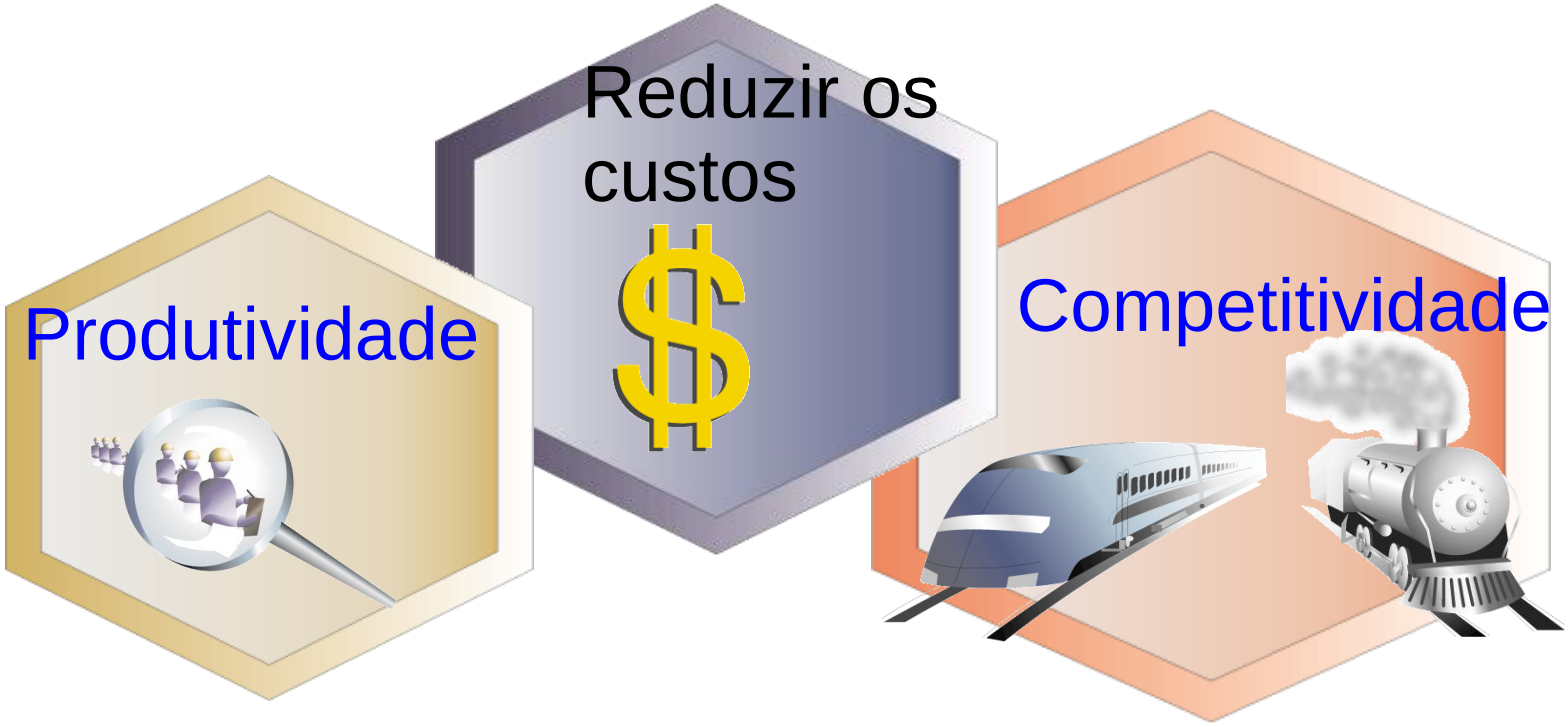
- **Planejamento:** estratégia, escopo, tipos de testes (integração, sistemas, homologação), ambiente, critérios, equipe

- **Preparação:** massa de dados, roteiros de testes

- **Execução:**
  - Solicitação de mudança
  - Relatório de testes: análise ao final de cada ciclo
  - Liberação do produto: conforme critério de aceite

Motivação para o teste

- Por que testar ?
  - Assegurar a qualidade do produto e satisfação do cliente
  - Reduzir os custos com retrabalho
  - Aumentar a produtividade e competitividade



Teste de Software

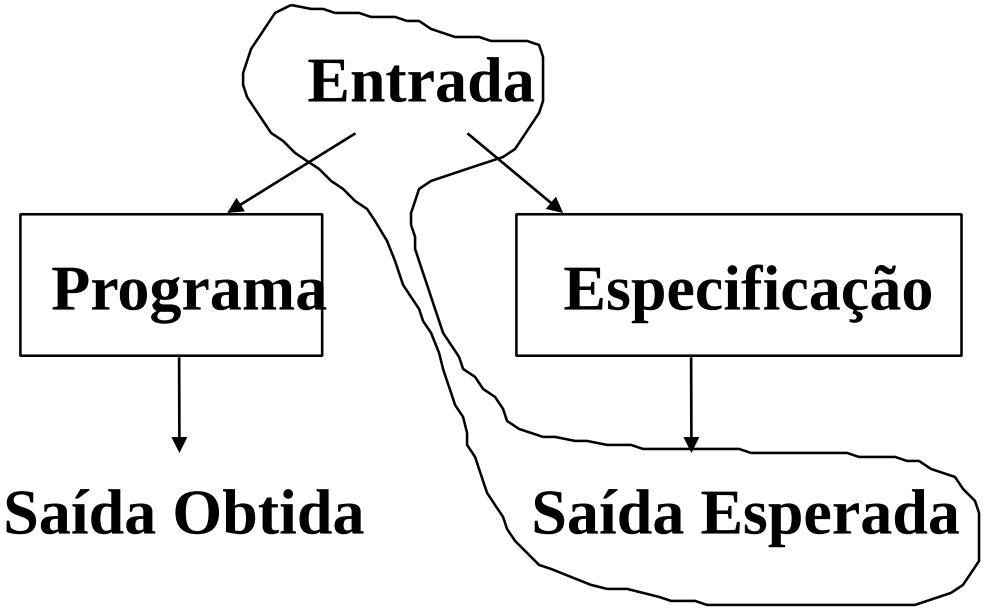
Principal Objetivo: refutar a assertiva de que o produto está correto

- Determinar entradas que façam as saídas obtidas diferirem das saídas esperadas segundo a especificação (busca de um contra-exemplo).
- É um *processo destrutivo*, sob o ponto de vista psicológico, contrariamente às demais fases da Engenharia de Software, onde constrói-se um produto.

Teste de Software

Um **teste bem sucedido** é aquele que revela a presença de um defeito ainda não descoberto.

Caso de Teste:



Por que testar?

Objetivos:

- O teste é a última revisão da especificação, do projeto e da codificação. (Pressman)
- Certificar aderência aos requisistos.
- Revelar presença de erros.

Limitações:

- Não garante ausência de erros.
- Alto custo.
- 50% tempo/custo de desenvolvimento = TESTE

Benefícios:

- Confiança na qualidade do software.
- Reduz manutenção, aumenta satisfação do cliente.
- Melhoria da qualidade.

Como testar?

"Formas" para projetar os casos de teste

Técnicas de teste

- Funcional
- Estrutural
- Baseado em Erros

São técnicas complementares

A questão não está em qual deles utilizar e sim como utilizá-las de forma complementar.

O que testar?

Teste de unidade.

- verificação léxico-sintática, lógica do programa.

Teste de integração.

- interfaces, chamadas, parâmetros.

Teste de sistema.

- Funcionalidade.

Teste de aceitação.

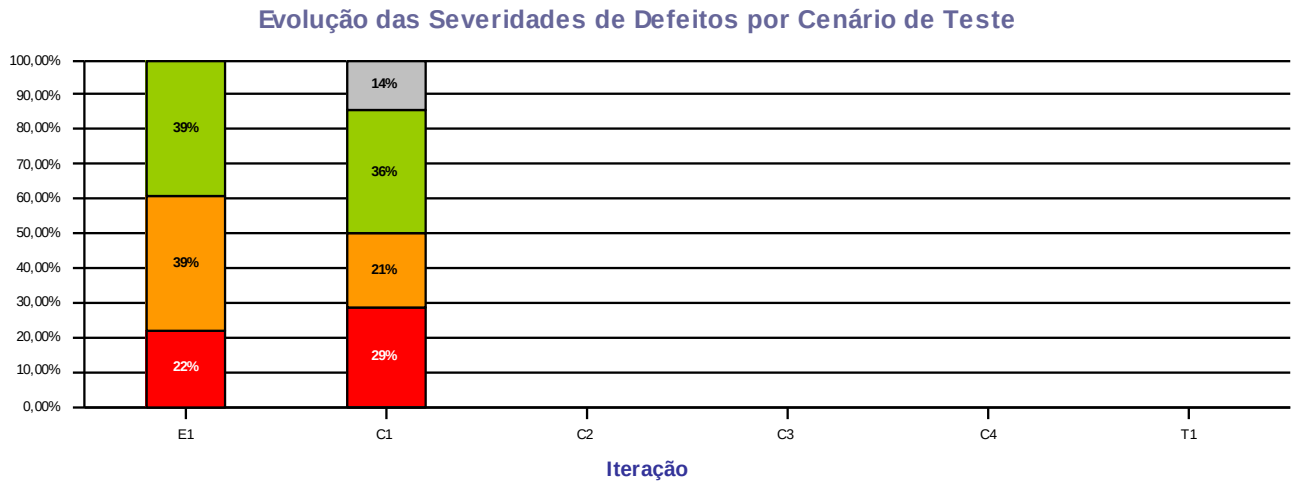
- Performance no ambiente real.

Visão Geral do Processo de Testes

- Planejamento dos testes
  - Plano de Testes (escopo, estratégia, pessoal, recursos, cronograma etc.)
- Projeto dos testes
  - Casos de teste (entradas e saídas esperadas)
  - Roteiros de teste (agrupamento e seqüência de casos de teste)
- Execução dos testes
  - Coleta das saídas reais
- Avaliação dos testes
  - Verificação dos critérios de término de teste e de qualidade de produto

Relatório de Testes (exemplo)

Cenários	Iterações			
	E1	C1	C2	T1
Severidade alta	6	6	2	0
Severidade média	8	6	1	0
Severidade baixa	10	8	5	0
Corretos	0	0	0	0
Não realizados	0	3	0	0



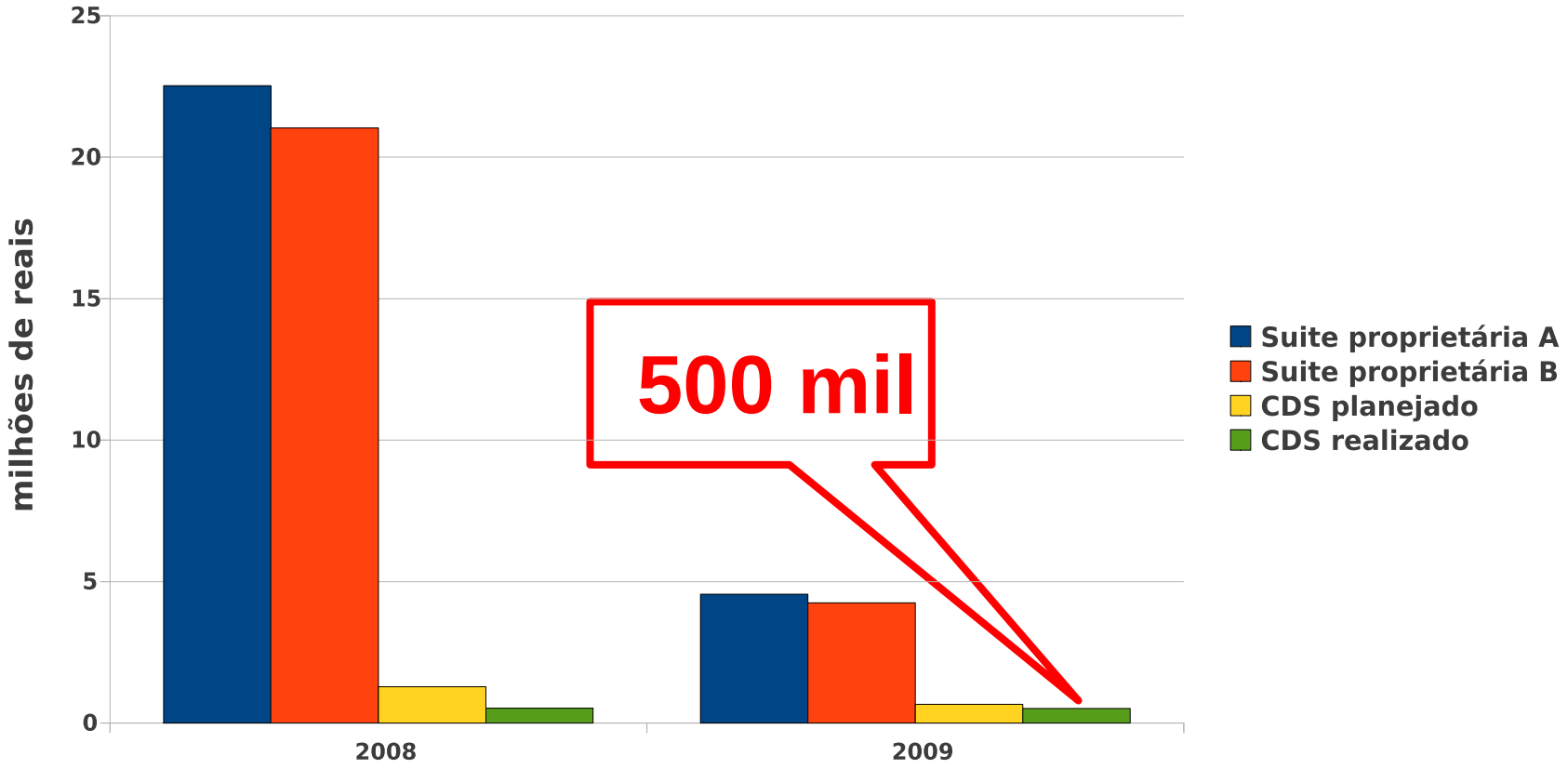
Projeto CPqD Developer Suite

- Em 2007 o CPqD usava uma suíte proprietária de ferramentas para desenvolvimento de software
  - com um número insuficiente de licenças
  - sem contrato de suporte
  - com versões desatualizadas

- Em 2008 iniciou o projeto **CPqD Developer Suite**

Objetivo	Implantar uma infraestrutura de ferramentas e um modelo de serviços que suportem os processos de desenvolvimento de software do CPqD e que garantam a sua sustentabilidade
Premissa	Adotar preferencialmente ferramentas livres ou de baixo-custo que estejam consolidadas no mercado.

Custo Planejado x Realizado



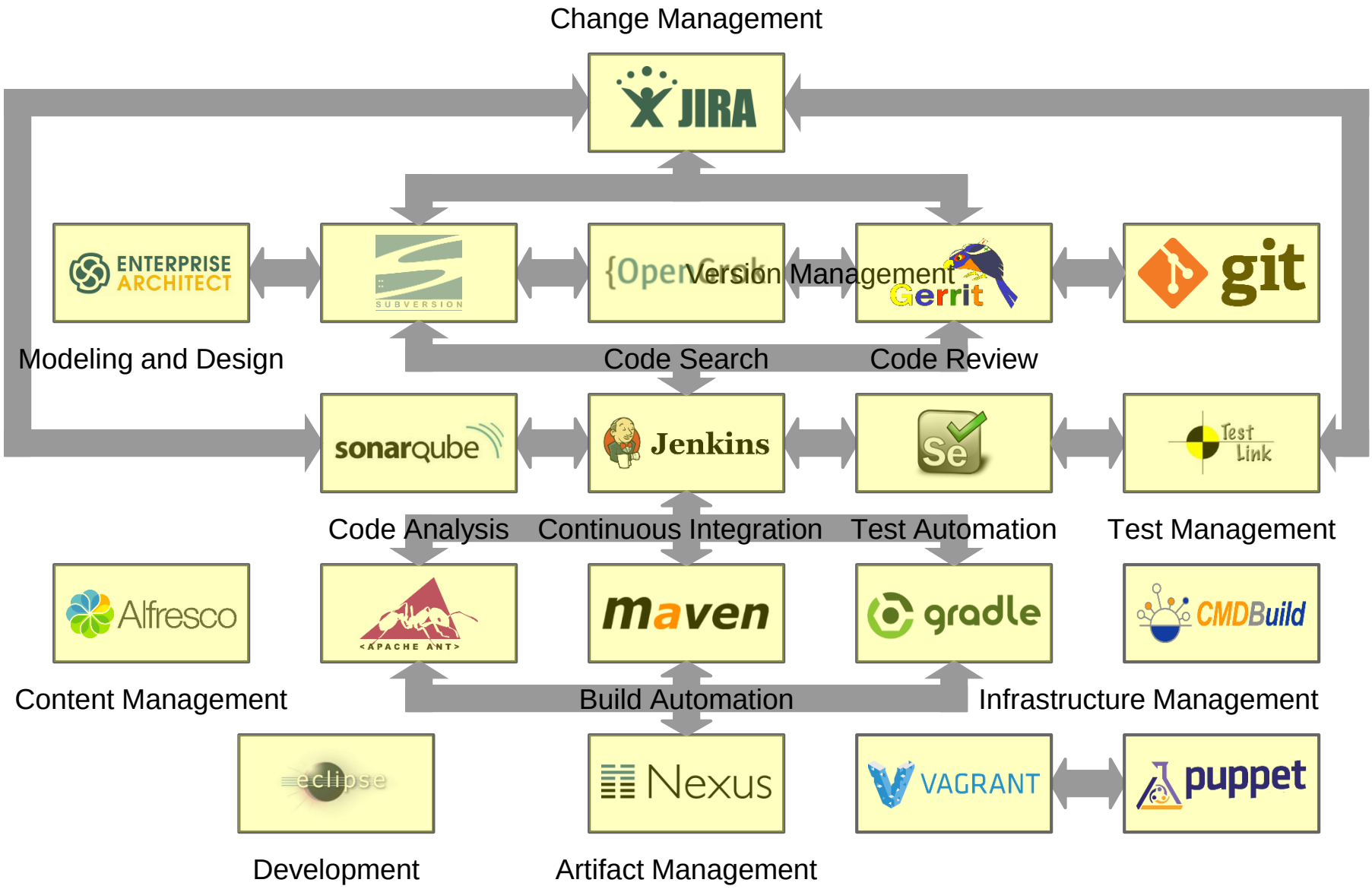
CPqD Developer Suite

Um conjunto de ferramentas

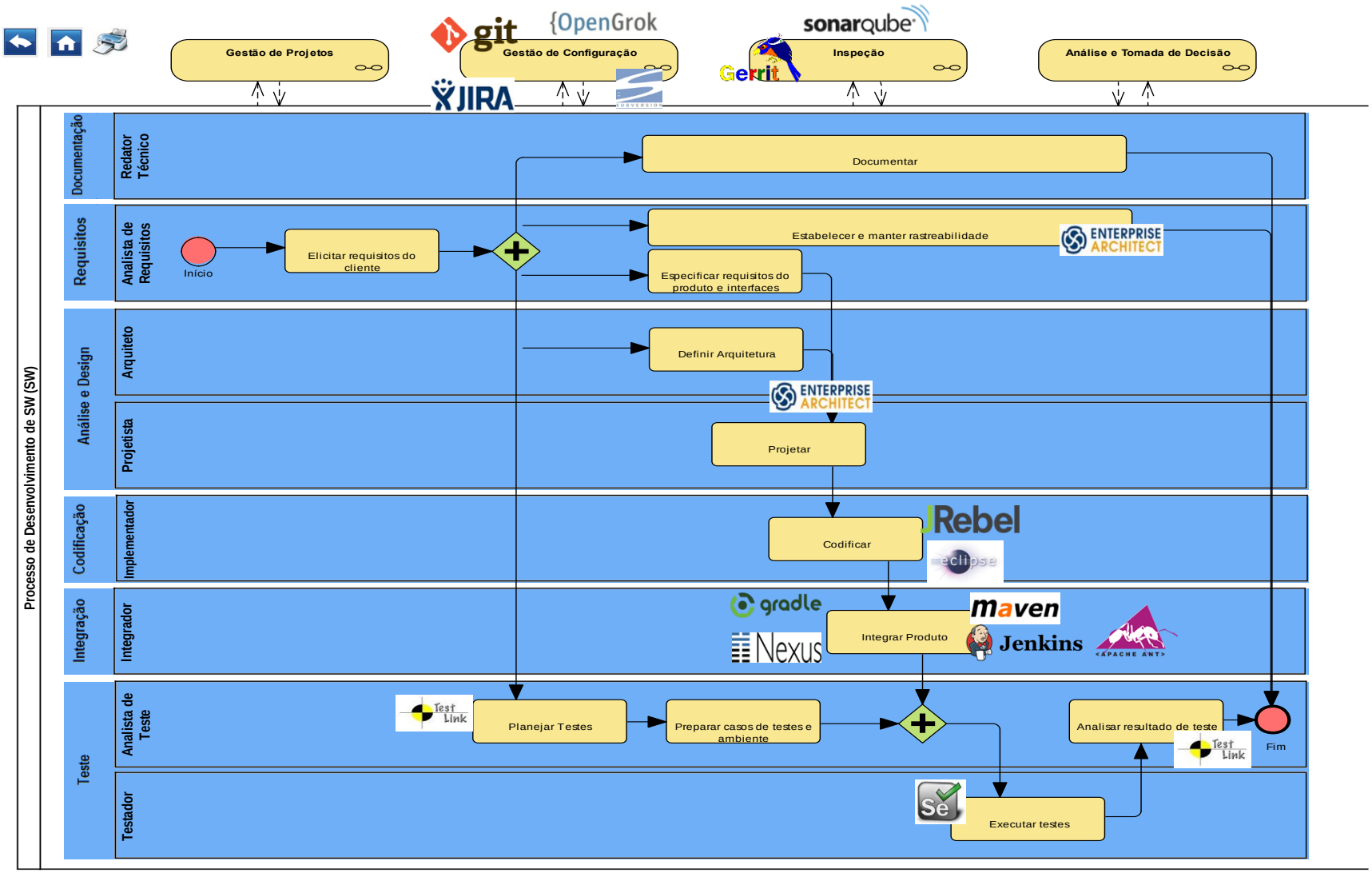
- de software livre e de baixo-custo,
- de qualidade comprovada,
- líderes de mercado em suas categorias,

- integradas para propiciar o
- desenvolvimento eficiente de
- software de qualidade.

CDS - CPqD Developer Suite



Desenvolvimento de software



Instruções, Orientações, Boas Práticas *Templates* *Indicadores*

CDS em números

177 projetos
100 milhões de linhas de código analisadas por mês

150 projetos
3.000 testes por mês

17.000 jobs configurados
16 mil builds por mês

980 repositórios
50 mil commits por mês

519 projetos
5.500 tíquetes criados por mês

Benefícios percebidos

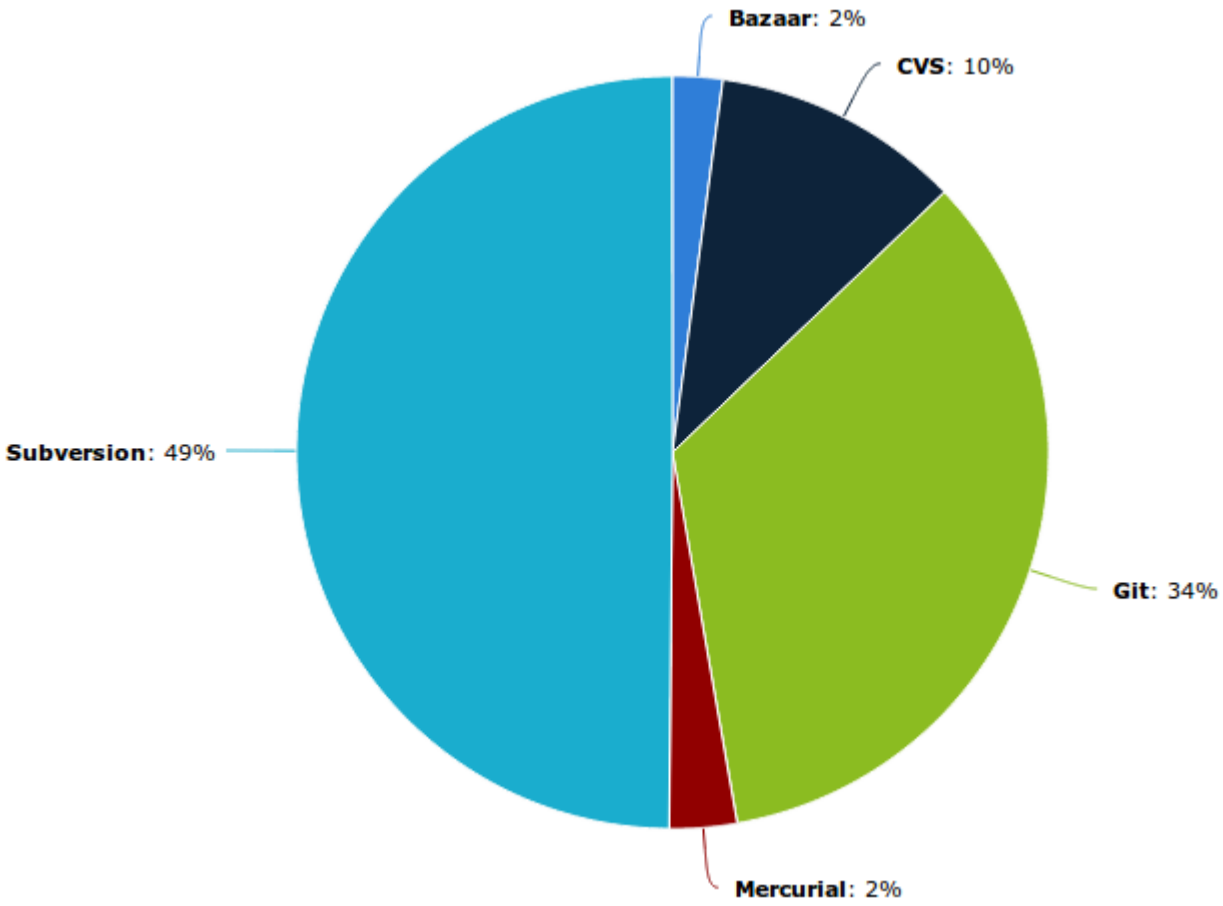
- Simplificação de processos e uniformização de procedimentos
- Ferramentas multi-plataforma, modernas e integradas
- Baixo custo
- Suporte centralizado de "alto nível"
- Evolução contínua
- Software livre
- Mobilidade de pessoal
  Economia de recursos
- Produtividade
- Disponibilidade
  Atualizações frequentes
- Desverticalização
  Especialização
- Novas funcionalidades
  Amadurecimento
- https://metacpan.org/release/SVN-Hooks
- https://metacpan.org/release/Git-Hooks
- https://metacpan.org/release/JIRA-REST

Resultados Gerais do Projeto

- Portal de ferramentas num site
- Suporte e manutenção centralizado
- Simplificação e uniformização de processos
  - GC/GM eliminaram 29 documentos de orientação!
- Novas funcionalidades
  - Jenkins, Selenium, Sonar, TestLink, Gerrit
- Treinamentos
- Contribuições à sociedade
  - *Frameworks* de *hooks* para o Subversion e Git
  - Módulo para automação do JIRA

subversion.apache.org

ohloh.net/repositories/compare



Controle de Versões Convencional

Centralizado

Robusto

Simples

Subversion no CPqD

+700 repositórios

+1.000 usuários

+50.000 commits por mês

atlassian.com/jira

Gestão de Mudanças

Gestão de Projetos Ágeis

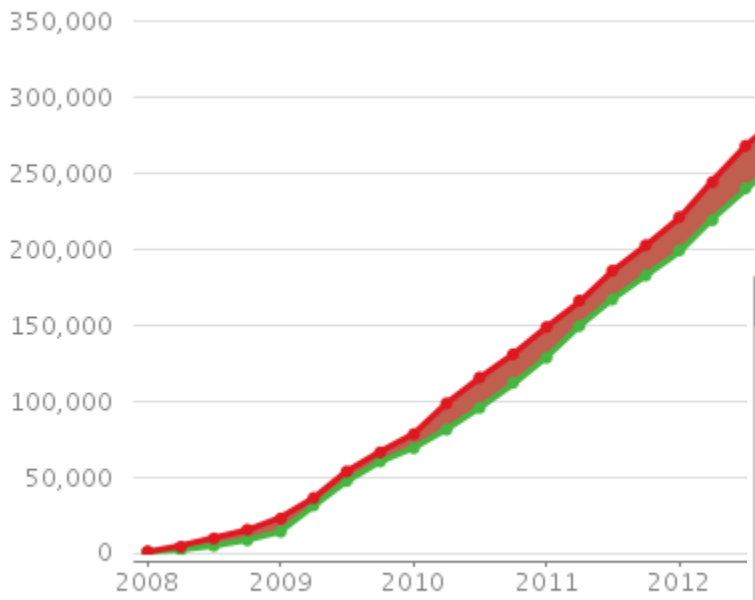
Ferramenta de Workflow

JIRA no CPqD

Database Statistics

Issues	363336
Produtos	411
Campos Personalizados	308
Workflows	34
Attachments	193274
Comentários	926573
Users	7594

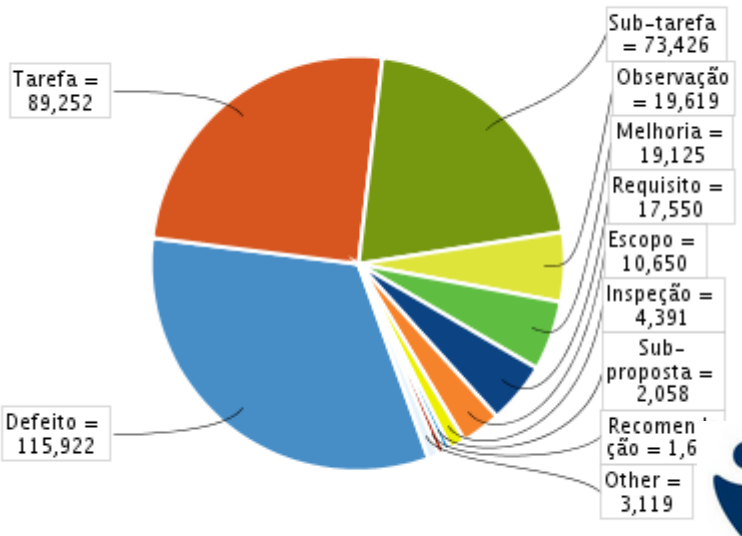
Gráfico - Criado vs Resolvidos: all-but-sandbox



Pendências: **345689** criadas e **310753** resolvidas

Período: última **2000** dias (agrupados **Trimestral**)

Gráfico de Pizza: all-but-sandbox

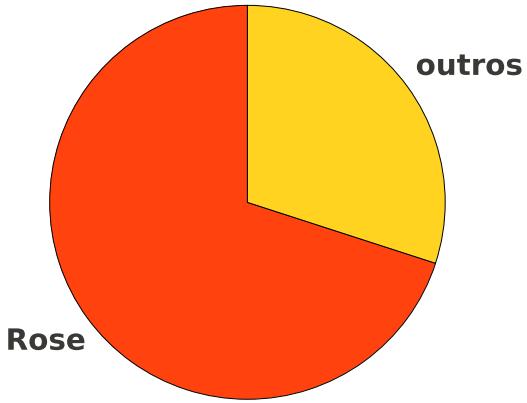


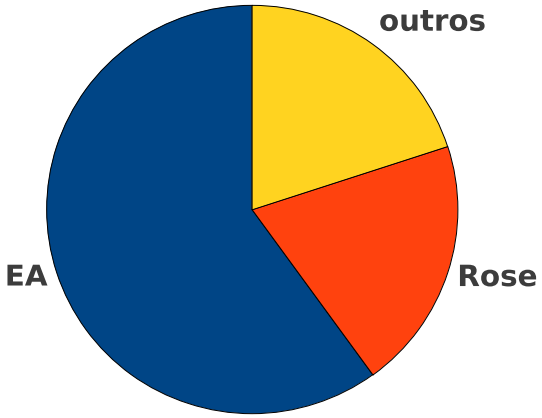
Total de Pendências: **356783** Tipo de Estatística: **Tipo de Pendê**

900 usuários ativos!

sparxsystems.com/products/ea

Enterprise Architect

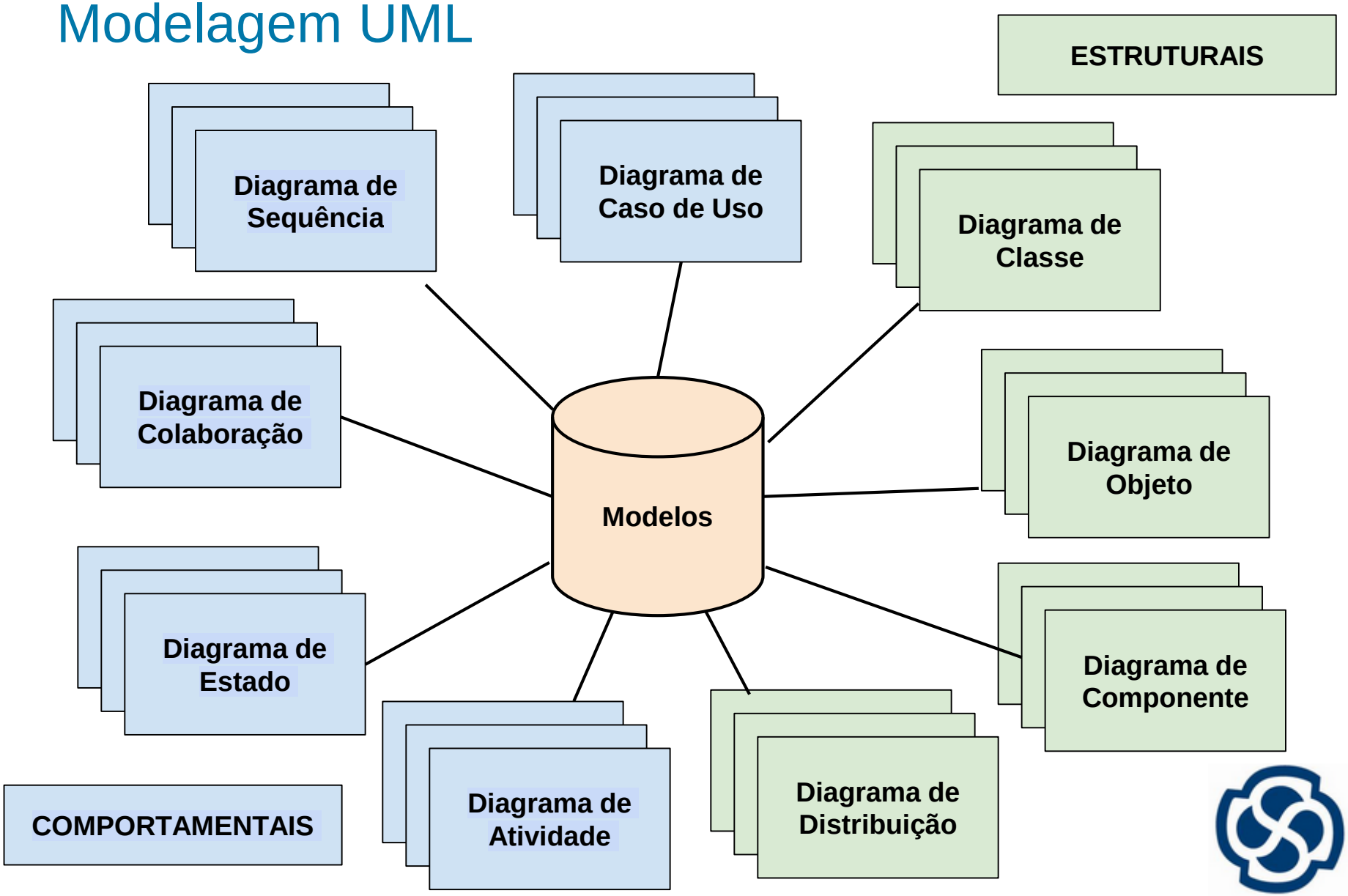




- Modelagem UML e de dados
- Migração semi-automática do Rose para o EA
- Monitoração do uso de licenças
- Edição conjunta de modelos
- 200+ instalações
- 50+ projetos em uso simultâneo

ENTERPRISE ARCHITECT

Modelagem UML



seleniumhq.org

Selenium

- Ferramenta para teste funcional de aplicações web
- Selenium IDE: extensão do Firefox para geração de roteiros de teste
- Selenium RC: automatiza qualquer *browser* para executar os testes
- Testes escritos em HTML, Java, C#, Perl, PHP, Python e Ruby
- Google utiliza e contribui para o projeto

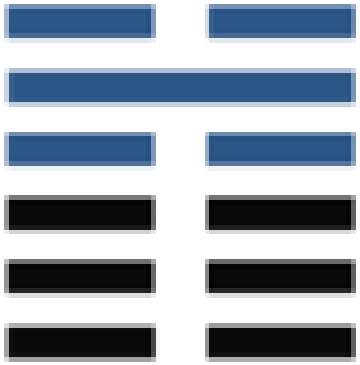
Maven

- Automação de *build* de aplicações Java
  - O Maven é o sucessor do Ant
- Integração com Jenkins, Nexus e Sonar
- Já são usados informalmente pela maioria dos usuários
- Uniformização das versões e do modo de uso

sonatype.org/nexus

Nexus

- Gerenciador de repositórios de artefatos do Maven
- Instância corporativa
- Catálogo automático de produtos entregues
- Gestão de configuração das dependências externas



Jenkins

- Gerenciador de integração contínua
- Uma instância por diretoria
- Integrado a Maven, Subversion, Git, TestLink e JIRA
- Geração contínua de versões de desenvolvimento com testes automatizados
- Administradores de sistemas estão trocando o crontab pelo Jenkins!

Jenkins no CPqD

11 servidores

90 slaves

2.500 jobs

24 mil builds por mês

SonarQube

- Inspeção contínua de software
- Acompanhamento da evolução de métricas de qualidade (*kwalitee?*)
- Integrado ao Maven
- Suporte a Java, C, C#, Flex, Natural, PHP, PL/SQL, Cobol e VB6

nemo.sonarsource.org

SonarQube no CPqD

144 projetos ativos

11 milhões de linhas de código

Git

Muito melhor desempenho

Merges mais fáceis e inteligentes

Melhor visualização da história

Workflows muito mais flexíveis

git

Gerrit

- Sistema web para revisão de código integrado ao Git
- Garante a inspeção de todo código novo, mantendo registros das revisões realizadas
- Integrado ao Jenkins para validação automática
- Integrado ao JIRA

Pontos importantes para o CDS

- Critério para seleção de ferramentas (baixo-custo e maturidade)
- Equipe dedicada para manutenção, suporte e evolução
- Acompanhamento contínuo dos clientes e dos fornecedores
- Apoio dos grupos de processo
- Patrocínio forte da gerência

Muito obrigado pela atenção!

www.**cpqd**.com.br