Boletim Técnico da Escola Politécnica da USP
Departamento de Engenharia de Computação e
Sistemas Digitais

ISSN 1413-215X
BT/PCS/0515

Arquitetura de Software: Uma Abordagem para Minimização de Incertezas e Riscos no Processo de Teste Visando Garantia da Qualidade

Cristina Coelho de Abreu Pinna
Reginaldo Arakaki

São Paulo - 2005

O presente trabalho é parte da dissertação mestrado, apresentada por Cristina Coelho de Abreu Pinna, sob orientação do Prof. Dr. Reginaldo Arakaki: "Um Roteiro Centrado em Arquitetura para Minimização de Riscos e Incertezas em Projeto de Software", defendida em 13/07/2004, na EPUSP.

A íntegra da dissertação encontra-se à disposição com o autor e na Biblioteca de Engenharia de Eletricidade da Escola Politécnica da USP.

FICHA CATALOGRÁFICA

Pinna, Cristina Coelho de Abreu
Arquitetura de software: uma abordagem para minimização de incertezas e riscos no processo de teste visando garantia da qualidade / Cristina Coelho de Abreu Pinna, Reginaldo Arakaki. -- São Paulo : EPUSP, 2005.
12 p. – (Boletim Técnico da Escola Politécnica da USP, Departamento de Engenharia de Computação e Sistemas Digitais ; BT/PCS/0515)

1. Engenharia de programação 2. Arquitetura de software 3. Análise de risco I. Arakaki, Reginaldo II. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais III. Título IV. Série
ISSN 1413-215X

Arquitetura de *Software*: Uma Abordagem para Minimização de Incertezas e Riscos no Processo de Teste visando Garantia da Qualidade

Cristina Coelho de Abreu Pinna
cristina.abreu@poli.usp.br
Escola Politécnica da Universidade de São Paulo – Depto. de Engenharia de Computação e Sistemas Digitais

Prof. Dr. Reginaldo Arakaki
reginaldo.arakaki@poli.usp.br
Escola Politécnica da Universidade de São Paulo – Depto. de Engenharia de Computação e Sistemas Digitais

Resumo

A técnica de garantia da qualidade mais difundida é, de maneira indiscutível, o processo de teste, que consiste em verificar o funcionamento de um sistema após ele estar pronto frente ao que foi especificado ou requerido pelo cliente (McConnell, 1996). No entanto, apesar deste processo ser bastante eficiente na garantia da qualidade do produto de *software*, está sujeito a uma série de incertezas e riscos das mais variadas naturezas, tais como: estimativas de tempo e esforço de teste, cobertura das situações a serem testadas, disponibilidade de ambiente e infra-estrutura de teste adequados, entre outros. Tais incertezas e riscos devem ser gerenciados de forma adequada ao longo do processo de desenvolvimento de *software*.

O objetivo deste trabalho é apresentar uma abordagem especializada a partir do processo de Gestão de Riscos proposto pelo PMI, baseada em requisitos de Arquitetura de *software*, considerando a Norma NBR 13596 (ISO/IEC 9126). Tal abordagem visa minimizar os riscos e incertezas do processo de *software* e, conseqüentemente, garantir a qualidade do produto final.

1. Introdução

Segundo o SEI - *Software Engineering Institute*, a qualidade de um produto de *software* deve ser garantida através de processos, métodos e ferramentas que permitam identificar erros e problemas durante o processo de desenvolvimento, de forma a fornecer realimentação que promova ações corretivas. Esta abordagem também adotada pelo modelo de maturidade de processos de *software* do SEI, o CMM - *Capability Maturity Model*, estabelece que a qualidade obtida de forma oportunista, ou seja, de forma não planejada pode ser mais custosa devido à necessidade de constantes retrabalhos.

Segundo McConnell (1996), a técnica de garantia da qualidade mais difundida é, de maneira indiscutível, o processo de teste, que consiste em verificar o funcionamento de um sistema após ele estar pronto frente ao que foi especificado ou requerido pelo cliente. O processo de teste, entretanto, ocorre ao final do ciclo de desenvolvimento e exige que o sistema esteja finalizado e executando para se fazer a verificação da sua qualidade, realimentando o processo de desenvolvimento.

Convém ressaltar que os erros são intrínsecos dos projetos de *software*, estando sempre presentes. Se não detectados e corrigidos a tempo, tais erros podem se transformar em falhas na operação do sistema, podendo inclusive ocasionar prejuízos de negócio que são incalculáveis, tais como, a não abertura de uma rede de lojas de varejo, o não funcionamento de uma agência bancária ou dos trens metroviários, entre outros.

O processo de teste, apesar de ser eficiente para garantir a qualidade do processo de desenvolvimento e, consequentemente, do produto de *software* está sujeito a uma série de incertezas e riscos das mais variadas naturezas, tais como: estimativas do tempo e esforço de teste, cobertura das situações a serem testadas, disponibilidade de ambiente e infra-estrutura de teste adequados, entre outros.

Tais incertezas e riscos devem ser gerenciados de forma adequada ao longo do processo de desenvolvimento. Assim, a simples aplicação do processo de teste não basta para garantir a qualidade do produto. Um exemplo disso é a realização dos testes em ambientes de *hardware* e *software* que não representam o real ambiente operacional em que o sistema será executado. Neste caso, o sistema pode funcionar perfeitamente no ambiente de teste e apresentar falhas quando no ambiente de operação real.

O objetivo deste trabalho é apresentar uma abordagem para identificar e classificar as incertezas e riscos do processo de teste e, a partir disso, estabelecer requisitos a serem tratados na definição da Arquitetura do *software*. Desta forma, estes requisitos devem ser definidos e projetados nas fases iniciais do ciclo de vida de desenvolvimento, de forma a evitar que os riscos e incertezas do processo de teste ocorram e invalidem a realização dos testes e, conseqüentemente, a qualidade do produto final.

2. Caracterização do Processo de Teste

O processo de teste pode ser caracterizado como a busca de discrepâncias entre o quê o *software* pode fazer versus o quê o usuário ou o ambiente computacional espera que ele faça (Goel apud Ziv; Richardson; Klosch, 1997).

Para constatar que um sistema está adequadamente construído e pronto para ser colocado em produção, é necessário submetê-lo a uma série de atividades de testes de modo a validá-lo de acordo com vários aspectos como funcionalidade, desempenho e robustez. Tais procedimentos de testes aos quais o sistema é submetido são capazes de detectar situações inadequadas, registrá-las e repassá-las às equipes de desenvolvimento para que se processe a correção.

Adaptando o processo de teste descrito por Pressman (2002), a dinâmica do ciclo de vida dos testes pode ser modelada da seguinte maneira: uma espiral passando por quatro etapas, onde cada volta representa o processo de convergência da qualidade do sistema até atingir o nível esperado. Este modelo é apresentado na figura 1.

As etapas deste processo de teste podem ser caracterizadas da seguinte maneira:

1) Planejamento: Corresponde à identificação e caracterização das situações a serem testadas, identificação da Massa de Teste e dos pré-requisitos para a realização dos testes e especificação dos ambientes. O planejamento do teste se inicia nas fases iniciais do ciclo de desenvolvimento, ou seja, nas fases de Análise e Projeto e culmina com a elaboração e formalização de um documento que registra os procedimentos e situações a serem testadas: o Plano de Teste (Yourdon, 1992).

2) Preparação/ Inicialização: Corresponde ao "startup" ou inicialização dos testes, com a assistência da equipe de desenvolvimento, envolvendo preparação dos ambientes e das Massas de Teste, bem como o treinamento da equipe de teste no processo de negócio e no sistema que será testado.

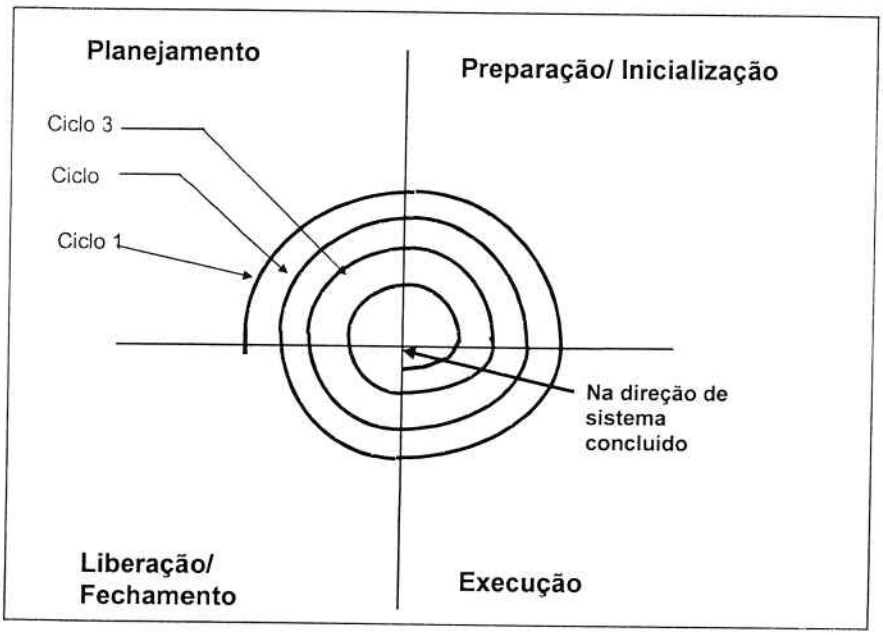

Figura 1. Ciclo de Vida do Processo de Teste.

3) Execução: Corresponde à realização dos testes propriamente dita, a partir das situações documentadas e procedimentadas no Plano de Teste. Iniciados os testes, esta etapa é a que executa e registra os resultados obtidos com os testes, detectando, documentando e repassando os problemas encontrados para a equipe de desenvolvimento, onde se classificam as gravidades dos defeitos.

4) Liberação e Fechamento: Corresponde ao término de um ciclo de testes com os possíveis resultados: pronto para ser liberado ou com erros para serem corrigidos. Neste último caso, o sistema deve ser corrigido e percorrer novamente o ciclo até a convergência de especificações. Nesta etapa também são gerados indicadores sobre a qualidade do produto de *software*, tais como, quantidade de testes satisfatórios, insatisfatórios, reincidências de erros, entre outros.

O processo de teste aqui descrito permite a realimentação do processo de desenvolvimento do ponto de vista de aferição da sua qualidade. Quanto mais ciclos um sistema percorrer nesta espiral, menor a qualidade do produto de *software* e a maturidade do processo de desenvolvimento.

3. As Incertezas e Riscos do Processo de Teste e a Gestão de Riscos Tradicional

Pode-se caracterizar de maneira indistinta como risco ou incerteza todos os aspectos que de alguma forma impactam o planejamento de um projeto, considerando as dimensões prazo, custo e qualidade (Pressman, 2002); (Boehm, 1989).

No entanto, alguns autores diferenciam risco de incerteza: risco corresponde à ameaça que pode ser mensurada em termos de probabilidade de ocorrência e impactos, ao passo que a incerteza não pode ser mensurada (Hirshleifer; Riley 1992). Para o trabalho em questão, esta distinção não tem relevância.

O trabalho elaborado por Ziv; Richardson; Klosch (1997) define o Princípio da Incerteza na Engenharia de *Software*. Segundo este princípio, as incertezas e riscos são inerentes e inevitáveis ao processo de desenvolvimento de *software* e aos produtos gerados. Isto significa que tais ameaças estão sempre presentes e, cabe ao gerente de projeto e sua equipe, identificar e definir o que será feito com cada uma delas.

Como uma fase do processo de desenvolvimento, o processo de teste também está sujeito a uma série de incertezas e riscos. Caso tais riscos e incertezas não sejam adequadamente monitorados e gerenciados, podem ocasionar impactos na qualidade do produto final. Assim, o processo de teste que primariamente deve garantir qualidade, nestas situações, pode mascarar eventuais problemas.

Ziv; Richardson; Klosch (1997) destacam, como exemplo, algumas incertezas que estão presentes no processo de teste, entre elas:

- O sistema de *software* em teste inclui especificações de requisitos, representações de projeto técnico, módulos de códigos-fontes entre outros. Tais artefatos de *software* são produzidos pelas atividades de análise de requisitos, projeto de arquitetura e codificação. Segundo o Princípio da Incerteza em Engenharia de *Software*, a incerteza permeia estes processos e, conseqüentemente, os artefatos resultantes. O planejamento de teste que se baseia nestes artefatos também carrega estas incertezas;
- O teste de um sistema, assim como outras atividades do projeto, são intensamente realizadas por atividades humanas, o que introduz incertezas;
- A cobertura de um plano de teste é inerentemente incerta – só um teste exaustivo de *stress* de todos os caminhos de um *software* em um ambiente ideal garante a absoluta certeza do processo de teste e seus resultados;
- As diferenças existentes entre o ambiente de teste e o ambiente real de operação do sistema também introduzem incertezas no processo.

Como os riscos constituem constantes ameaças aos resultados dos projetos de *software*, tornam-se necessárias medidas de minimização e eliminação dos riscos e incertezas.

Segundo o PMI – *Project Management Institute*, a Gestão de Riscos em projetos inclui os processos de identificação, análise e resposta ao risco através do estabelecimento de planos de ação e mecanismos de controle que visam maximizar os resultados dos eventos positivos e minimizar as consequências de eventos adversos em decorrência de riscos. Esta abordagem coincide com os modelos de Gestão de Riscos apresentados por outros autores, como Boehm (1989) e Carr et al. (1993), denominada para efeitos deste trabalho de Gestão de Riscos Tradicional.

A figura 2 resume os processos que compõem a Gestão de Riscos Tradicional. Tais processos estão descritos utilizando a notação SADT/IDEF0 (Marca; McGowan, 1988), que consiste em uma notação para modelagem de processos.

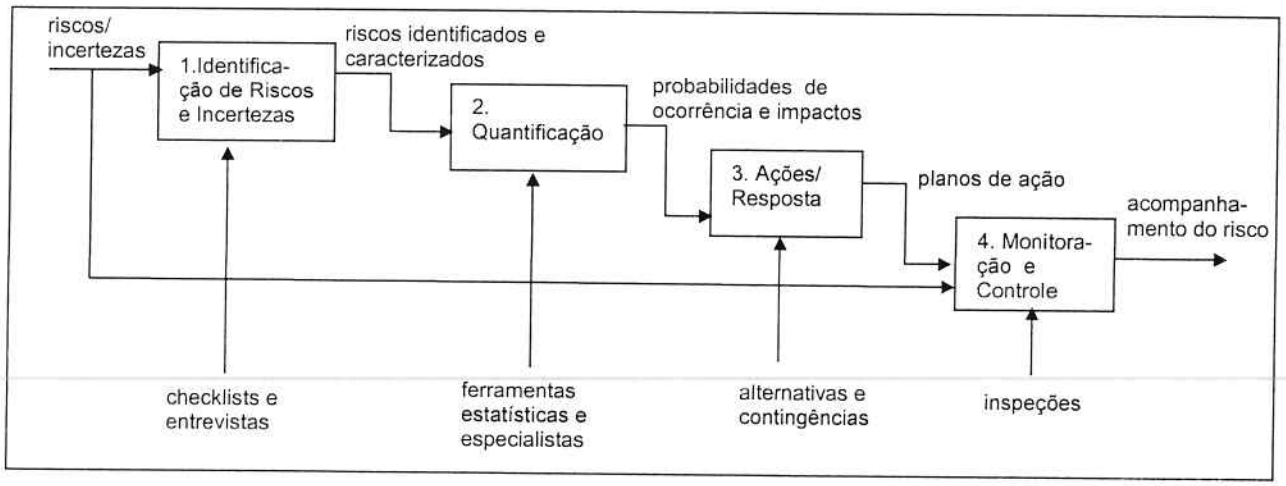

Figura 2. Processos da Gestão de Riscos Tradicional.

4. Arquitetura de *Software*, Requisitos e Visões de Arquitetura

O Processo de Desenvolvimento de *Software* Unificado (UP – *Unified Process*) desenvolvido por Jacobson, Booch e Rumbaugh, considerando as inúmeras contribuições da *Rational Software Corporation* e outras organizações, utiliza como conceito de Arquitetura de *software* o conjunto de decisões significantes sobre:

1) A organização do sistema de *software*;
2) A seleção dos elementos estruturais e suas interfaces que compõem o sistema, junto com os comportamentos especificados pelas colaborações entre estes elementos;
3) A composição destes elementos estruturais e comportamentais em sub-sistemas maiores;
4) O estilo arquitetural que guia esta organização: elementos, interfaces, colaborações e composições.

Segundo Jacobson; Booch; Rumbaugh (1999), a Arquitetura de *software* está relacionada não somente com a estrutura e comportamento, mas também com aspectos como uso, funcionalidade, desempenho, reuso, restrições econômicas e tecnológicas, entre outros.

Assim, a Arquitetura de *software* descreve sua estrutura geral, os elementos e a forma como estes elementos interagem e se comunicam. Segundo diversos autores, entre eles Garlan (2000) e Bass; Clements; Kazman (1998) a Arquitetura é uma característica do sistema de *software* e que afeta diretamente sua qualidade.

Quando o usuário estabelece os requisitos de um software, em geral, ele se preocupa com os requisitos e funcionalidades de negócio, pois são estes que atendem diretamente suas necessidades. No entanto, outros requisitos são de fundamental importância para garantir a qualidade do produto de *software* como um todo, tais como infra-estrutura de *hardware*, *software*, comunicação e segurança. Tais requisitos são muitas vezes denominados requisitos não-funcionais.

Segundo Mendes (2002), os requisitos não-funcionais abordam aspectos de qualidade importantes nos sistemas de *software*. Se tais requisitos não forem levados em consideração, o sistema resultará inconsistente e de baixa qualidade. Além disso, os usuários ficarão insatisfeitos e o prazo e orçamento do projeto não serão cumpridos.

Para efeitos deste trabalho, a expressão requisito de Arquitetura refere-se aos elementos que direcionam a implementação da Arquitetura de *software*..

Alguns exemplos de requisitos de Arquitetura são:

- Desacoplamento técnico entre as funções de negócio e as funções de infra-estrutura de um sistema de *software*;
- Centralização das funções de comunicação com sistemas externos em componentes modularizados, de forma a evitar possíveis impactos decorrentes de mudanças nas interfaces com os sistemas externos;
- Arquitetura que possibilite a utilização de simuladores para a realização de testes, visando diminuir a dependência de disponibilidade de sistemas externos, entre outros.

Convém destacar que, para cada um dos requisitos de Arquitetura citados acima, a Arquitetura do *software* deve ser projetada e preparada para suportá-los.

Com base na classificação de requisitos entre funcionais e não-funcionais e na Norma NBR 13596 (ISO/IEC 9126), neste trabalho serão consideradas as visões de Arquitetura de *software* apresentadas na figura 3.

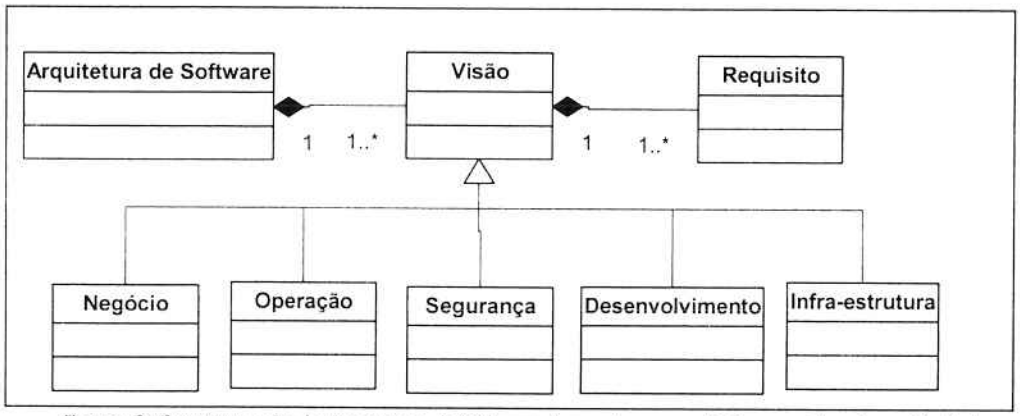

Figura 3. Caracterização da Arquitetura de Software de acordo com as Visões consideradas no Trabalho.

As cinco visões que compõem a Arquitetura de um sistema de *software* podem ser caracterizadas da seguinte forma:

- **Visão Negócio**: trata dos aspectos do sistema referente ao usuário, tais como funcionalidades disponibilizadas, aspectos da interação homem-máquina e modelos de processos de negócio relacionados com o sistema. Tais aspectos consistem a especificação e modelagem de requisitos de negócio;
- **Visão Operação**: trata dos processos básicos que sustentam o funcionamento do sistema, englobando também aspectos de monitoração e parametrização de regras de negócio pelo usuário, visando flexibilidade na operação do sistema;
- **Visão Segurança**: caracteriza a segurança do sistema em aspectos como integridade, disponibilidade, privacidade de acesso aos recursos do sistema, confiabilidade e rastreabilidade;
- **Visão Desenvolvimento**: caracteriza os aspectos relacionados com os desenvolvedores do sistema, tais como ambientes, processos de desenvolvimento, padrões, testes, homologação, implantação, ferramentas e procedimentos de controle de qualidade. Esta visão também incorpora ferramentas conceituais, métodos e técnicas conceituadas no mercado e que apóiam o processo de desenvolvimento, tal como a Orientação a Objetos (OMG);
- **Visão Infra-estrutura**: caracteriza os aspectos de *hardware*, *software*, Base de Dados e comunicação que sustentam os processos de desenvolvimento e operação do sistema.

A preocupação com estas cinco visões da Arquitetura no planejamento, modelagem e projeto visa garantir a completeza dos requisitos bem como a qualidade do processo de desenvolvimento e do produto final.

Esta abordagem permite estabelecer todos os requisitos de um sistema considerando os requisitos solicitados e explicitados pelo cliente - geralmente os de negócio - e também os demais requisitos fundamentais para a qualidade do produto final: os requisitos implícitos ou não-funcionais. Assim, a abordagem da Arquitetura de *software* permite garantir completeza dos requisitos do sistema de *software*.

5. A Abordagem de Arquitetura para Minimizar as Incertezas e Riscos do Processo de Teste

5.1. Gestão de Riscos centrada em Arquitetura

A Arquitetura de *software* pode ser considerada como um instrumento de planejamento e gestão, facilitando a tomada de decisões e minimizando riscos e incertezas de projetos de *software* (Garlan, 2000); (Mendes, 2002). Também, de acordo com os estudos realizados por Kozaczynski (2002), existe uma associação entre a Arquitetura de *software*, seus requisitos e os riscos do projeto dentro do ciclo de vida do *software*.

Apesar de diversos autores destacarem a Arquitetura de *software* como ferramenta mitigadora de riscos e incertezas, os mesmos não detalham os métodos e técnicas utilizadas.

A Gestão de Riscos centrada em Arquitetura proposta consiste em uma abordagem sistemática para minimização dos riscos e incertezas mais comuns em projetos de *software*, através do mapeamento em requisitos de Arquitetura de *software*. Esta abordagem é baseada na Gestão de Riscos Tradicional, endereçando mais diretamente as ações para minimização dos riscos e incertezas como requisitos de Arquitetura de *software*. Esta abordagem pretende também ser mais preventiva, à medida em que estabelece requisitos que servem como realimentação e controle ao processo de desenvolvimento, de forma a garantir qualidade do produto final.

Especializando o processo de Gestão de Riscos Tradicional apresentado na figura 2, a figura 4 apresenta o detalhamento dos processos que compreendem a abordagem proposta.

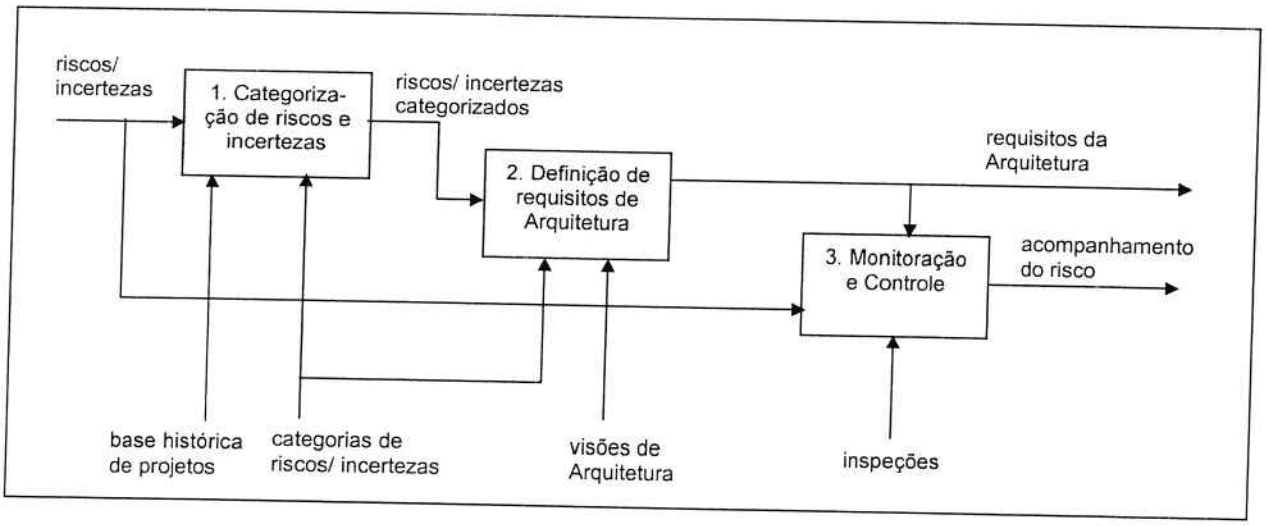

Figura 4. Processos da Gestão de Riscos centrada em Arquitetura.

Os processos da Gestão de Riscos centrada em Arquitetura são:

1) Categorização de Riscos e Incertezas

Os riscos e incertezas que ocorrem em um projeto de *software* são comuns a vários outros. Assim, é possível tratar categorias dos principais riscos e incertezas, levando-se em consideração o tipo de processo de negócio, padrões de Arquitetura de *software* existentes na empresa e *templates* corporativos.

Considerando o histórico de projetos de um determinado processo de negócio, este sub-processo pode ser suprimido, devido à experiência das pessoas que atuam neste contexto.

Outra interpretação seria considerar que tais riscos e incertezas já foram previamente identificados. Isto significa que para tais situações, assume-se que a probabilidade de ocorrência de determinados riscos é igual a 100 %, partindo-se direto para o estabelecimento de ações.

Este processo corresponde aos processos de Identificação, Quantificação e Priorização de riscos da Gestão de Riscos Tradicional apresentados na figura 2.

2) Definição de Requisitos da Arquitetura

A partir das categorias de riscos e incertezas é possível definir os requisitos de Arquitetura de *software* a serem aplicados para minimização destes riscos e incertezas. Este processo corresponde ao processo de Ações e Respostas aos riscos da Gestão de Riscos Tradicional apresentado na figura 2.

3) Monitoração e Controle

Da mesma forma que na Gestão de Riscos Tradicional, os riscos e incertezas devem ser monitorados e acompanhados. Neste caso, parâmetros adicionais a serem considerados na monitoração são os requisitos de Arquitetura de *software* estabelecidos, de forma a verificar a eficiência da minimização dos riscos e a possibilidade de surgimento de novos.

O roteiro centrado em Arquitetura apresenta a característica de, caso um certo risco não seja identificado e nem quantificado a tempo, estabelecer uma postura natural para buscar e estabelecer princípios e requisitos de Arquitetura de *software*, com base em categorias de riscos mais frequentes e pré-identificados, de forma a garantir a qualidade do produto final.

A especialização da Gestão de Riscos centrada em Arquitetura a partir da Gestão de Riscos Tradicional justifica-se em função da complexidade crescente dos projetos de *software* e de sua estratégia preventiva, que permite estabelecer ações, no caso requisitos de Arquitetura de *software*, que reduzem custos de erros e retrabalhos, aumentando a qualidade e produtividade. A abordagem proposta também se baseia no fato da existência de categorias de riscos comuns e reincidentes em projetos de *software*.

5.2. Um Exemplo

Um exemplo para ilustrar o que foi apresentado acima seria de um projeto de desenvolvimento de um sistema que depende do recebimento de um arquivo gerado por um sistema externo, que será desenvolvido por outra equipe. Segundo a Gestão de Riscos Tradicional, uma vez identificado como risco que o arquivo pode não ficar pronto a tempo para ser integrado e testado com o sistema, na fase de planejamento do projeto, este risco seria registrado, quantificado em termos de probabilidade de sua ocorrência e seus impactos e estabelecidos os planos de ação. Neste caso, uma ação possível seria planejar uma reunião com a equipe responsável pela geração do arquivo para definição do layout e estabelecimento das datas de entrega deste layout e do arquivo definitivo para realização dos testes integrados.

Uma ação complementar focada em Arquitetura consistiria em planejar a organização do *software* de forma a desacoplar a parte das funcionalidades que fazem o recebimento e tratamento do arquivo externo, por exemplo, através de componentes de *software*, de forma a evitar que mudanças no layout definido para o arquivo impactem a lógica de negócio da aplicação em desenvolvimento. Também seria importante o planejamento para a construção de

um simulador que simule o recebimento de diversos arquivos com conteúdos diferentes, de acordo com o layout definido, para realização dos testes de stress da aplicação.

5.3. Mapeamento das Incertezas e Riscos em Requisitos de Arquitetura no Processo de Teste

Considerando as incertezas e riscos do processo de teste apresentados e aplicando o roteiro de Gestão de Riscos centrada em Arquitetura descrito no item 5.1, a abordagem proposta consiste em endereçar requisitos da Arquitetura para cada uma destas incertezas e riscos. A tabela 1 apresenta o mapeamento das incertezas e riscos do processo de teste nos requisitos da Arquitetura de *software* e suas respectivas visões

Convém ressaltar que os riscos e incertezas aqui apresentados não são exaustivos e que cada um dos requisitos apresentados na tabela 1 devem ser obtidos em tempo de modelagem e projeto, de forma a disponibilizar todas as condições favoráveis para a realização do processo de teste.

Tabela 1. Riscos e Incertezas do Processo de Teste e os respectivos Requisitos de Arquitetura.

Etapa	Categoria de Incerteza/Risco	Requisito da Arquitetura (0)	Visão da Arquitetura
1) Planejamento dos Testes	1.1) Escolha de um conjunto adequado de situações a serem testadas; 1.2) O produto de *software* será suficientemente testado?; 1.3) Escolha dos dados adequados para a realização dos testes (massa de teste).	• Modelos de Análise e Projeto centrados no Processo de Negócio (1); • Arquitetura e planejamento que contemple categorias de tipos de teste (2).	• Desenvolvimento.
	1.4) O *software* pode ser testado?; 1.5) O *software* apresentará problemas em produção?	• Projeto e Logs de teste; • Auto-diagnóstico (3).	• Desenvolvimento. • Operação.
	1.6) Resultado esperado do teste formalizado no Plano de Teste não ser condizente com o que o usuário espera (carrega incertezas das especificações e modelagens).	• Modelos, protótipos e especificações validados e produzidos em conjunto com o cliente (4).	• Negócio. • Desenvolvimento.
	1.7) Pontos de observação introduzidos para a realização dos testes podem afetar os resultados obtidos.	• Planos de testes centrados nos modelos de Análise e Projeto (1).	• Desenvolvimento.
	1.8) Elaboração, formalização e organização das situações a serem testadas no Plano de Testes do Sistema.	• Plano deve ser organizado de acordo com o processo de negócio e navegação (5).	• Desenvolvimento. • Negócio.
	1.9) Planejamento e estimativas em relação aos prazos, recursos e esforços para o planejamento e execução dos testes.	• Plano de Teste modularizado, de forma a permitir teste simultâneo (5); • Base Histórica e critérios (6).	• Desenvolvimento.
2) Preparação dos Testes	2.1) Ambiente de teste não condizente com o ambiente de operação real do sistema.	• Ambiente de teste semelhante ao ambiente real de operação (7); • Simuladores (8).	• Infra-estrutura.
	2.2) Indisponibilidade de sistemas externos dos quais o sistema em teste possui dependência.	• Simuladores (8).	• Infra-estrutura.
3) Execução dos Testes	3.1) Entendimento e interpretação dos Planos de Teste pelos testadores.	• Plano de Teste orientado pelo processo de negócio e navegação (5).	• Desenvolvimento.
	3.2) Tempo e esforço de execução dos testes não estar condizente com o planejamento.	• Plano de Teste modularizado, de forma a permitir teste simultâneo (5); • Ferramentas automatizadas (9).	• Desenvolvimento. • Infra-estrutura.
4) Liberação e Fechamento	4.1) Indicadores e estatísticas significativas da qualidade do produto.	• Indicadores representativos, identificados a partir de métodos como por exemplo, o GQM (10).	• Desenvolvimento.
	4.2) Erros humanos na consolidação dos resultados dos testes executados.	• Ferramentas automatizadas (9).	• Infra-estrutura.

(0) Os requisitos de Arquitetura apresentados permitem aumentar a testabilidade do produto de *software* e, conseqüentemente, a sua qualidade.

(1) **Modelos de Análise e Projeto**: modelos que permitem estabelecer os fluxos e informações de um processo de negócio, independente do grau de automação previsto. A formalização destes modelos pode ser feita utilizando-se diversas notações entre as quais destacam-se o SADT/IDEF0 (Marca; McGowan, 1988), UML Use Case, demais diagramas da UML (OMG) e o Mapa de Navegação (Pinna; Souza; Arakaki, 2003), que indica o fluxo de navegação do sistema.Tais modelos permitem a identificação dos casos a serem testados e que irão compor o Plano de Teste do Sistema, bem como na definição da estratégia de teste e dos pontos de observação.

(2) **Categorias de tipos de teste**: classificação dos tipos de teste que auxiliam na identificação dos casos de teste que irão compor o Plano de Teste. Categorias típicas de tipos de teste são: volume, desempenho, stress, funcionalidade de negócio, regressão, entre outros (Yourdon, 1992); (Pressman, 2002).

(3) **Projeto de Teste e Auto-diagnóstico**: as situações de teste devem ser projetadas visando como o teste será executado e quais mecanismos e ferramentas serão utilizadas pela equipe de teste para fazer a verificação da qualidade do *software*, como por exemplo, logs de teste. O *software* também deve ser projetado visando apresentar avisos e alertas nos casos de degeneração e mau funcionamento de operação, como rotinas de auto-diagnóstico.

(4) **Modelos validados e produzidos com o cliente**: permite que situações de divergências ou desvios entre o solicitado pelo cliente e o entendido ou projetado pelo analista sejam minimizados. Tal prática proposta por diversos autores, entre eles Pressman (2002), visa minimizar os custos decorrentes de retrabalhos.

(5) **Organização do Plano de Teste**: o Plano de Teste deve ser estruturado de forma a facilitar a realização dos testes pelo testador. Em geral, a organização dos casos de teste pelo fluxo do processo de negócio e pela navegação do sistema (Pinna; Souza; Arakaki, 2003) facilita a execução dos testes. Também o Plano de Teste deve ser construído de forma modular, permitindo que o mesmo seja desmembrado para a execução de testes por equipes simultâneas, o que permite reduzir os tempos de execução dos testes.

(6) **Base Histórica e Critérios**: utilização de critérios e dados históricos extraídos de testes de projetos anteriores. Tais informações são úteis para o dimensionamento dos esforços de teste para projetos futuros.

(7) **Ambientes de Teste e Operação**: as características do ambiente de teste, incluindo *hardware*, *software* e comunicação devem ser as mais semelhantes possíveis das características do ambiente de operação real do sistema.

(8) **Simuladores**: a utilização de simuladores de sistemas externos permite que os testes do sistema em questão prossigam, caso os sistemas externos estejam indisponíveis ou ainda não estejam prontos. Este desacoplamento permite minimizar os impactos de prazos causados por atrasos nos sistemas dos quais existem dependências. Convém ressaltar que a utilização de simuladores introduz novas incertezas e riscos no processo, uma vez que consistem em elementos externos. A Arquitetura de *software* deve ser projetada para trabalhar com simuladores.

(9) **Ferramentas automatizadas**: permitem a execução dos testes e a consolidação dos resultados de forma automática, a partir do planejamento das situações de teste. Tais ferramentas permitem reduzir os tempos de teste, bem como auxilia a equipe de teste nas atividades mais repetitivas, no entanto, constitui um elemento que introduz novas incertezas e riscos no processo.

(10) **Indicadores representativos**: os indicadores devem ser em quantidade pequena, de forma a permitir uma fácil monitoração, porém representativos do processo. Existem algumas técnicas para a identificação destes indicadores, entre as quais pode-se destacar o GQM – *Goal Questions and Metrics* (Basili et al., 1994 apud Kan, 1995) a qual permite definir, classificar e priorizar métricas e indicadores a partir de metas pré-estabelecidas. A Arquitetura de software deve ser preparada para permitir a extração de informações para tais indicadores.

6. Considerações Finais

O presente trabalho apresenta uma abordagem, dada uma incerteza ou risco identificado no processo de teste, de como o mesmo pode ser minimizado através de requisitos da Arquitetura de *software*, de forma que o processo de teste garanta a qualidade do produto de *software* como um todo, considerando prazos, custos e requisitos especificados.

A abordagem proposta é baseada e especializada a partir da Gestão de Riscos Tradicional. Nesta abordagem, mesmo que uma incerteza ou risco não seja identificado e quantificado antecipadamente, o que é uma situação bastante comum nos projetos de *software*, estabelece-se uma postura de pensar nos diversos requisitos da Arquitetura, e não somente nas funcionalidades de negócio que geralmente constituem os requisitos explicitados pelo usuários. Tais requisitos implícitos ou não-funcionais são fundamentais para garantir a qualidade do produto e do processo de *software*.

Os pontos positivos da abordagem de Arquitetura discutidas neste trabalho se resumem a:

- Estabelecer requisitos da Arquitetura de *software* e, indiretamente, uma postura para lidar com estas incertezas e riscos de projetos de *software*;
- A abordagem pode ser aplicada nas fases iniciais do ciclo de vida do *software*, o que permite a redução de custos decorrentes de riscos e incertezas;
- Os principais riscos e incertezas podem ser categorizados por processos de negócio e/ou base histórica e, para estas categorias, podem ser pré-estabelecidos requisitos de Arquitetura de *software* que minimizem os seus efeitos. Isto permite o reuso de Arquitetura com base em perfis de risco conhecidos;
- Estabelecer uma gestão preventiva considerando que para categorias de riscos comuns ou genéricos as ações de minimização dos riscos são estabelecidas de forma objetiva através de requisitos de Arquitetura de *software*;
- Permite categorizar padrões de riscos e incertezas mais comuns que afetam o processo de desenvolvimento de *software*.

Naturalmente, um ponto crítico da implantação desta abordagem consiste em vencer as barreiras culturais das equipes de projeto em lidar com as incertezas e riscos e aceita-los como presentes e inerentes ao processo. Para os projetistas, o caminho mais natural é possuir todas as definições, requisitos e certezas a respeito de um sistema que deverá ser desenvolvido.

O trabalho aqui apresentado deve ser expandido para outras fases do ciclo de desenvolvimento de *software*, adicionais à fase de teste. Também, para que a abordagem de Arquitetura se torne mais abrangente, devem ser previstos requisitos não técnicos, incluindo fatores humanos e motivacionais que também constituem riscos e incertezas bastante comuns do processo de desenvolvimento. Tais aspectos podem constituir escopo de trabalhos posteriores.

7. Referências

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **Tecnologia de Informação: Avaliação de Produto de Software – Características de Qualidade e Diretrizes para o seu Uso – NBR 13596 (ISO/IEC 9126)**. Junho, 2003.

BASS, L.; CLEMENTS, P.; KAZMAN, R. **Software Architecture in Practice**. Massachusetts: Addison Wesley, 1998.

BOEHM, B. W. **Software Risk Management**. Washington: IEEE Computer Society Press, 1989.

CARR. M. J. et al. **Taxonomy-Based Risk Identification**. Pittsburh: Software Engineering Institute, 1993. (Technical Reports CMU/ SEI-93-TR-6 and ADA 266992). Disponível em <http:// www.sei.cmu.edu>. Acesso em: 14 de maio de 2004.

GARLAN, D. Software Architecture: A Roadmap. In: International Conference on Software Engineering, Ireland, 2000. **Anais**. New York: ACM Press, 2000. p.91 – 101.

HIRSHLEIFER, J.; RILEY, J. G. **The Analytics of Uncertainty and Information**. United Kingdom: Cambridge University Press, 1992.

JACOBSON, I.; BOOCH, G.; RUMBAUGH, J. **The Unified Software Development Process**. Massachusetts: Addison Wesley, 1999.

KAN, S. H. **Metrics and Models in Software Quality Engineering**. Massachusetts: Addison Wesley, 1995.

KOZACZYNSKI, W. Requirements, Architectures and Risks. In: Proceedings of the IEEE Joint International Conference on Requirements Engineering, Germany, Setembro, 2002. **Anais**. Germany, 2002.

MARCA, D. A.; McGOWAN, C. L. **IDEF0/SADT: Business Process and Enterprise Modeling**. California: Eclectic Solutions, 1988.

McCONNELL, S. **Rapid Development: Taning Wild Software Schedules**. Washington: Microsoft Press, 1996.

MENDES, A. **Arquitetura de Software: Desenvolvimento Orientado para Arquitetura**. Rio de Janeiro: Campus, 2002.

OMG WEB SITE – Object Management Group. **Apresenta processos e padrões sobre Orientação a Objetos e UML (Unified Modeling Language)**. Disponível em <http://: www.omg.org>. Acesso em: 14 de maio de 2004.

PINNA, C. C. A.; SOUZA, A. A.; ARAKAKI, R. Uma Abordagem para Projeto da Interface e da Navegação de Aplicativos Web orientada pelo Processo de Negócio. In: SUCESU – 1 Congresso Nacional de Tecnologia da Informação e Comunicação. Salvador, 2003. **Anais**. Salvador, 2003.

PMI WEB SITE - Project Management Institute. **Apresenta processos, técnicas e padrões para gerenciamento de projetos**. Disponível em <http:// www.pmi.org>. Acesso em: 14 de maio de 2004.

PRESSMAN, R. S. **Engenharia de Software**. 5 ed. Trad. Mônica Maria G. Traviesso. Rio de Janeiro: McGraw-Hill, 2002.

SEI WEB SITE – Software Engineering Institute. Pittsburgh. **Apresenta padrões, modelos, processos e conceitos da Engenharia de Software**. Disponível em <http:// www.sei.cmu.edu>. Acesso em: 14 de maio de 2004.

YOURDON, E. **Análise Estruturada Moderna**. Trad. Dalton Conde de Alencar. Rio de Janeiro: Campus, 1992.

ZIV, H.; RICHARDSON, D. J.; KLOSCH, R. The Uncertainty Principle in Software Engineering. In: 19th International Conference on Software Engineering, Massachusetts, 1997. **Anais**. Massachusetts, 1997.

BOLETINS TÉCNICOS - TEXTOS PUBLICADOS

BT/PCS/9301 - Interligação de Processadores através de Chaves Ômicron - GERALDO LINO DE CAMPOS, DEMI GETSCHKO

BT/PCS/9302 - Implementação de Transparência em Sistema Distribuído - LUÍSA YUMIKO AKAO, JOÃO JOSÉ NETO

BT/PCS/9303 - Desenvolvimento de Sistemas Especificados em SDL - SIDNEI H. TANO, SELMA S. S. MELNIKOFF

BT/PCS/9304 - Um Modelo Formal para Sistemas Digitais à Nível de Transferência de Registradores - JOSÉ EDUARDO MOREIRA, WILSON VICENTE RUGGIERO

BT/PCS/9305 - Uma Ferramenta para o Desenvolvimento de Protótipos de Programas Concorrentes - JORGE KINOSHITA, JOÃO JOSÉ NETO

BT/PCS/9306 - Uma Ferramenta de Monitoração para um Núcleo de Resolução Distribuída de Problemas Orientado a Objetos - JAIME SIMÃO SICHMAN, ELERI CARDOSO

BT/PCS/9307 - Uma Análise das Técnicas Reversíveis de Compressão de Dados - MÁRIO CESAR GOMES SEGURA, EDIT GRASSIANI LINO DE CAMPOS

BT/PCS/9308 - Proposta de Rede Digital de Sistemas Integrados para Navio - CESAR DE ALVARENGA JACOBY, MOACYR MARTUCCI JR.

BT/PCS/9309 - Sistemas UNIX para Tempo Real - PAULO CESAR CORIGLIANO, JOÃO JOSÉ NETO

BT/PCS/9310 - Projeto de uma Unidade de Matching Store baseada em Memória Paginada para uma Máquina Fluxo de Dados Distribuído - EDUARDO MARQUES, CLAUDIO KIRNER

BT/PCS/9401 - Implementação de Arquiteturas Abertas: Uma Aplicação na Automação da Manufatura - JORGE LUIS RISCO BECERRA, MOACYR MARTUCCI JR.

BT/PCS/9402 - Modelamento Geométrico usando do Operadores Topológicos de Euler - GERALDO MACIEL DA FONSECA, MARIA ALICE GRIGAS VARELLA FERREIRA

BT/PCS/9403 - Segmentação de Imagens aplicada a Reconhecimento Automático de Alvos - LEONCIO CLARO DE BARROS NETO, ANTONIO MARCOS DE AGUIRRA MASSOLA

BT/PCS/9404 - Metodologia e Ambiente para Reutilização de Software Baseado em Composição - LEONARDO PUJATTI, MARIA ALICE GRIGAS VARELLA FERREIRA

BT/PCS/9405 - Desenvolvimento de uma Solução para a Supervisão e Integração de Células de Manufatura Discreta - JOSÉ BENEDITO DE ALMEIDA, JOSÉ SIDNEI COLOMBO MARTINI

BT/PCS/9406 - Método de Teste de Sincronização para Programas em ADA - EDUARDO T. MATSUDA, SELMA SHIN SHIMIZU MELNIKOFF

BT/PCS/9407 - Um Compilador Paralelizante com Detecção de Paralelismo na Linguagem Intermediária - HSUEH TSUNG HSIANG, LÍRIA MATSUMOTO SAITO

BT/PCS/9408 - Modelamento de Sistemas com Redes de Petri Interpretadas - CARLOS ALBERTO SANGIORGIO, WILSON V. RUGGIERO

BT/PCS/9501 - Sintese de Voz com Qualidade - EVANDRO BACCI GOUVÊA, GERALDO LINO DE CAMPOS

BT/PCS/9502 - Um Simulador de Arquiteturas de Computadores "A Computer Architecture Simulator" - CLAUDIO A. PRADO, WILSON V. RUGGIERO

BT/PCS/9503 - Simulador para Avaliação da Confiabilidade de Sistemas Redundantes com Reparo - ANDRÉA LUCIA BRAGA, FRANCISCO JOSÉ DE OLIVEIRA DIAS

BT/PCS/9504 - Projeto Conceitual e Projeto Básico do Nível de Coordenação de um Sistema Aberto de Automação, Utilizando Conceitos de Orientação a Objetos - NELSON TANOMARU, MOACYR MARTUCCI JUNIOR

BT/PCS/9505 - Uma Experiência no Gerenciamento da Produção de Software - RICARDO LUIS DE AZEVEDO DA ROCHA, JOÃO JOSÉ NETO

BT/PCS/9506 - MétodOO - Método de Desenvolvimento de Sistemas Orientado a Objetos: Uma Abordagem Integrada à Análise Estruturada e Redes de Petri - KECHI HIRAMA, SELMA SHIN SHIMIZU MELNIKOFF

BT/PCS/9601 - MOOPP: Uma Metodologia Orientada a Objetos para Desenvolvimento de Software para Processamento Paralelo - ELISA HATSUE MORIYA HUZITA, LÍRIA MATSUMOTO SATO

BT/PCS/9602 - Estudo do Espalhamento Brillouin Estimulado em Fibras Ópticas Monomodo - LUIS MEREGE SANCHES, CHARLES ARTUR SANTOS DE OLIVEIRA

BT/PCS/9603 - Programação Paralela com Variáveis Compartilhadas para Sistemas Distribuídos - LUCIANA BEZERRA ARANTES, LIRIA MATSUMOTO SATO

BT/PCS/9604 - Uma Metodologia de Projeto de Redes Locais - TEREZA CRISTINA MELO DE BRITO CARVALHO, WILSON VICENTE RUGGIERO

BT/PCS/9605 - Desenvolvimento de Sistema para Conversão de Textos em Fonemas no Idioma Português - DIMAS TREVIZAN CHBANE, GERALDO LINO DE CAMPOS

BT/PCS/9606 - Sincronização de Fluxos Multimídia em um Sistema de Videoconferência - EDUARDO S. C. TAKAHASHI, STEFANIA STIUBIENER

BT/PCS/9607 - A importância da Completeza na Especificação de Sistemas de Segurança - JOÃO BATISTA CAMARGO JÚNIOR, BENÍCIO JOSÉ DE SOUZA

BT/PCS/9608 - Uma Abordagem Paraconsistente Baseada em Lógica Evidencial para Tratar Exceções em Sistemas de Frames com Múltipla Herança - BRÁULIO COELHO ÁVILA, MÁRCIO RILLO

BT/PCS/9609 - Implementação de Engenharia Simultânea - MARCIO MOREIRA DA SILVA, MOACYR MARTUCCI JÚNIOR

BT/PCS/9610 - Statecharts Adaptativos - Um Exemplo de Aplicação do STAD - JORGE RADY DE ALMEIDA JUNIOR, JOÃO JOSÉ NETO

BT/PCS/9611 - Um Meta-Editor Dirigido por Sintaxe - MARGARETE KEIKO IWAI, JOÃO JOSÉ NETO

BT/PCS/9612 - Reutilização em Software Orientado a Objetos: Um Estudo Empírico para Analisar a Dificuldade de Localização e Entendimento de Classes - SELMA SHIN SHIMIZU MELNIKOFF, PEDRO ALEXANDRE DE OLIVEIRA GIOVANI

BT/PCS/9613 - Representação de Estruturas de Conhecimento em Sistemas de Banco de Dados - JUDITH PAVÓN MENDONZA, EDIT GRASSIANI LINO DE CAMPOS

BT/PCS/9701 - Uma Experiência na Construção de um Tradutor Inglês - Português - JORGE KINOSHITA, JOÃO JOSÉ NETO

BT/PCS/9702 - Combinando Análise de "Wavelet" e Análise Entrópica para Avaliar os Fenômenos de Difusão e Correlação - RUI CHUO HUEI CHIOU, MARIA ALICE G. V. FERREIRA

BT/PCS/9703 - Um Método para Desenvolvimento de Sistemas de Computacionais de Apoio a Projetos de Engenharia - JOSÉ EDUARDO ZINDEL DEBONI, JOSÉ SIDNEI COLOMBO MARTINI

BT/PCS/9704 - O Sistema de Posicionamento Global (GPS) e suas Aplicações - SÉRGIO MIRANDA PAZ, CARLOS EDUARDO CUGNASCA

BT/PCS/9705 - METAMBI-OO - Um Ambiente de Apoio ao Aprendizado da Técnica Orientada a Objetos - JOÃO UMBERTO FURQUIM DE SOUZA, SELMA S. S. MELNIKOFF

BT/PCS/9706 - Um Ambiente Interativo para Visualização do Comportamento Dinâmico de Algoritmos - IZAURA CRISTINA ARAÚJO, JOÃO JOSÉ NETO

BT/PCS/9707 - Metodologia Orientada a Objetos e sua Aplicação em Sistemas de CAD Baseado em "Features" - CARLOS CÉSAR TANAKA, MARIA ALICE GRIGAS VARELLA FERREIRA

BT/PCS/9708 - Um Tutor Inteligente para Análise Orientada a Objetos - MARIA EMÍLIA GOMES SOBRAL, MARIA ALICE GRIGAS VARELLA FERREIRA

BT/PCS/9709 - Metodologia para Seleção de Solução de Sistema de Aquisição de Dados para Aplicações de Pequeno Porte - MARCELO FINGUERMAN, JOSÉ SIDNEI COLOMBO MARTINI

BT/PCS/9801 - Conexões Virtuais em Redes ATM e Escalabilidade de Sistemas de Transmissão de Dados sem Conexão - WAGNER LUIZ ZUCCHI, WILSON VICENTE RUGGIERO

BT/PCS/9802 - Estudo Comparativo dos Sistemas da Qualidade - EDISON SPINA, MOACYR MARTUCCI JR.

BT/PCS/9803 – The VIBRA Multi-Agent Architecture: Integrating Purposive Vision With Deliberative and Reactive Planning – REINALDO A. C. BIANCHI , ANNA H. REALI C. RILLO, LELIANE N. BARROS

BT/PCS/9901 – Metodologia ODP para o Desenvolvimento de Sistemas Abertos de Automação – JORGE LUIS RISCO BECCERRA, MOACYR MARTUCCI JUNIOR

BT/PCS/9902 – Especificação de Um Modelo de Dados Bitemporal Orientado a Objetos – SOLANGE NICE ALVES DE SOUZA, EDIT GRASSIANI LINO DE CAMPOS

BT/PCS/9903 – Implementação Paralela Distribuída da Dissecação Cartesiana Aninhada – HILTON GARCIA FERNANDES, LIRIA MATSUMOTO SATO

BT/PCS/9904 – Metodologia para Especificação e Implementação de Solução de Gerenciamento – SERGIO CLEMENTE, TEREZA CRISTINA MELO DE BRITO CARVALHO

BT/PCS/9905 – Modelagem de Ferramenta Hipermídia Aberta para a Produção de Tutoriais Interativos – LEILA HYODO, ROMERO TORI

BT/PCS/9906 – Métodos de Aplicações da Lógica Paraconsistente Anotada de Anotação com Dois Valores-LPA2v com Construção de Algoritmo e Implementação de Circuitos Eletrônicos – JOÃO I. DA SILVA FILHO, JAIR MINORO ABE

BT/PCS/9907 – Modelo Nebuloso de Confiabilidade Baseado no Modelo de Markov – PAULO SÉRGIO CUGNASCA, MARCO TÚLIO CARVALHO DE ANDRADE

BT/PCS/9908 – Uma Análise Comparativa do Fluxo de Mensagens entre os Modelos da Rede Contractual (RC) e Colisões Baseada em Dependências (CBD) – MÁRCIA ITO, JAIME SIMÃO SICHMAN

BT/PCS/9909 – Otimização de Processo de Inserção Automática de Componentes Eletrônicos Empregando a Técnica de Times Assíncronos – CESAR SCARPINI RABAK, JAIME SIMÃO SICHMAN

BT/PCS/9910 – MIISA – Uma Metodologia para Integração da Informação em Sistemas Abertos – HILDA CARVALHO DE OLIVEIRA, SELMA S. S. MELNICOFF

BT/PCS/9911 – Metodologia para Utilização de Componentes de Software: um estudo de Caso – KAZUTOSI TAKATA, SELMA S. S. MELNIKOFF

BT/PCS/0001 – Método para Engenharia de Requisitos Norteado por Necessidades de Informação – ARISTIDES NOVELLI FILHO, MARIA ALICE GRIGAS VARELLA FERREIRA

BT/PCS/0002 – Um Método de Escolha Automática de Soluções Usando Tecnologia Adaptativa – RICARDO LUIS DE AZEVEDO DA ROCHA, JOÃO JOSÉ NETO

BT/PCS/0101 – Gerenciamento Hierárquico de Falhas – JAMIL KALIL NAUFAL JR., JOÃO BATISTA CAMARGO JR.

BT/PCS/0102 – Um Método para a Construção de Analisadores Morfológicos, Aplicado à Língua Portuguesa, Baseado em Autômatos Adaptativos – CARLOS EDUARDO DANTAS DE MENEZES, JOÃO JOSÉ NETO

BT/PCS/0103 – Educação pela Web: Metodologia e Ferramenta de Elaboração de Cursos com Navegação Dinâmica – LUISA ALEYDA GARCIA GONZÁLEZ, WILSON VICENTE RUGGIERO

BT/PCS/0104 – O Desenvolvimento de Sistemas Baseados em Componentes a Partir da Visão de Objetos – RENATA EVANGELISTA ROMARIZ RECCO, JOÃO BATISTA CAMARGO JÚNIOR

BT/PCS/0105 – Introdução às Gramáticas Adaptativas – MARGARETE KEIKO IWAI, JOÃO JOSÉ NETO

BT/PCS/0106 – Automação dos Processos de Controle de Qualidade da Água e Esgoto em Laboratório de Controle Sanitário – JOSÉ BENEDITO DE ALMEIDA, JOSÉ SIDNEI COLOMBO MARTINI

BT/PCS/01/07 – Um Mecanismo para Distribuição Segura de Vídeo MPEG – CÍNTIA BORGES MARGI, GRAÇA BESSAN, WILSON VICENTE RUGGIERO

BT/PCS/0108 – A Dependence-Based Model for Social Reasoning in Multi-Agent Systems – JAIME SIMÃO SICHMAN

BT/PCS/0109 – Ambiente Multilinguagem de Programação – Aspectos do Projeto e Implementação – APARECIDO VALDEMIR DE FREITAS, JOÃO JOSÉ NETO

BT/PCS/0110 – LETAC: Técnica para Análise de Tarefas e Especificação de Fluxo de Trabalho Cooperativo – MARCOS ROBERTO GREINER, LUCIA VILELA LEITE FILGUEIRAS

BT/PCS/0111 – Modelagem ODP para o Planejamento de Sistemas de Potência – ANIRIO SALLES FILHO, JOSÉ SIDNEI COLOMBO MARTINI

BT/PCS/0112 – Técnica para Ajuste dos Coeficientes de Quantização do Padrão MPEG em Tempo Real – REGINA M. SILVEIRA, WILSON V. RUGGIERO

BT/PCS/0113 – Segmentação de Imagens por Classificação de Cores: Uma Abordagem Neural – ALEXANDRE S. SIMÕES, ANNA REALI COSTA

BT/PCS/0114 - Uma Avaliação do Sistema DSM Nautilus –MARIO DONATO MARINO, GERALDO LINO DE CAMPOS

BT/PCS/0115 – Utilização de Redes Neurais Artificiais para Construção de Imagem em Câmara de Cintilação – LUIZ SÉRGIO DE SOUZA, EDITH RANZINI

BT/PCS/0116 – Simulação de Redes ATM – HSU CHIH WANG CHANG, WILSON VICENTE RUGGIERO

BT/PCS/0117 – Application of Monoprocessed Architecture for Safety Critical Control Systems – JOSÉ ANTONIO FONSECA, JORGE RADY DE ALMEIDA JR.

BT/PCS/0118 – WebBee – Um Sistema de Informação via WEB para Pesquisa de Abelhas sem Ferrão – RENATO SOUSA DA CUNHA, ANTONIO MOURA SARAIVA

BT/PCS/0119 – Parallel Processing Applied to Robot Manipulator Trajectory Planning – DENIS HAMILTON NOMIYAMA, LÍRIA MATSUMOTO SATO, ANDRÉ RIYUITI HIRAKAWA

BT/PCS/0120 – Utilização de Padrão de Arquitetura de Software para a Fase de Projeto Orientado a Objetos – CRISITINA MARIA FERREIRA DA SILVA, SELMA SHIN SHIMIZU MELNIKOFF

BT/PCS/0121 – Agilizando Aprendizagem por Reforço Através do uso de Conhecimento sobre o Domínio – RENÊ PEGORARO, ANNA H. REALI COSTA

BT/PCS/0122 – Modelo de Segurança da Linguagem Java Problemas e Soluções – CLAUDIO MASSANORI MATAYOSHI, WILSON VICENTE RUGGIERO

BT/PCS/0123 – Proposta de um Agente CNM para o Gerenciamento Web de um Backbone ATM – FERNANDO FROTA REDÍGOLO, TEREZA CRISTINA MELO DE BRITO CARVALHO

BT/PCS/0124 – Um Método de Teste de software Baseado em Casos Teste – SÉRGIO RICARDO ROTTA, KECHI HIRAMA

BT/PCS/0201 – A Teoria Nebulosa Aplicada a uma Bicicleta Ergométrica para Fisioterapia – MARCO ANTONIO GARMS, MARCO TÚLIO CARVALHO DE ANDRADE

BT/PCS/0202 – Synchronization Constraints in a Concurrent Object Oriented Programming Model – LAÍS DO NASCIMENTO SALVADOR, LIRIA MATSUMOTO SATO

BT/PCS/0203 – Construção de um Ambiente de Dados sobre um Sistema de Arquivos Paralelos – JOSÉ CRAVEIRO DA COSTA NETO, LIRIA MATSUMOTO SATO

BT/PCS/0204 – Maestro: Um Middleware para Suporte a Aplicações Distribuídas Baseadas em Componentes de Software – CLÁUDIO LUÍS PEREIRA FERREIRA, JORGE LUÍS RISCO BECERRA

BT/PCS/0205 - Sistemas de Automação dos Transportes (ITS) Descritos Através das Técnicas de Modelagem RM-OPD (ITU-T) e UML (OMG) – CLÁUDIO LUIZ MARTE, JORGE LUÍS RISCO BECERRA, JOSÉ SIDNEI COLOMBO

BT/PCS/0206 – Comparação de Perfis de Usuários Coletados Através do Agente de Interface PersonalSearcher – GUSTAVO A. GIMÉNEZ LUGO, ANALÍA AMANDI, JAIME SIMÃO SICHMAN

BT/PCS/0207 – Arquitetura Reutilizáveis para a Criação de Sistemas de Tutorização Inteligentes – MARCO ANTONIO FURLAN DE SOUZA, MARIA ALICE GRIGAS VARELLA FERREIRA

BT/PCS/0208 – Análise e Predição de Desempenho de Programas Paralelos em Redes de Estações de Trabalho – LIN KUAN CHING, LIRIA MATSUMOTO SATO

BT/PCS/0209 – Previsões Financeiras Através de Sistemas Neuronebulosos – DANIEL DE SOUZA GOMES, MARCO TÚLIO CARVALHO DE ANDRADE

BT/PCS/0210 – Proposta de Arquitetura Aberta de Central de Atendimento – ANA PAULA GONÇALVES SERRA, MOACYR MARTUCCI JÚNIOR

BT/PCS/0211 – Alternativas de Implementação de Sistemas Nebulosos em Hardware – MARCOS ALVES PREDEBON, MARCO TÚLIO CA.RVALHO DE ANDRADE

BT/PCS/0212 – Registro de Imagens de Documentos Antigos – VALGUIMA VICTORIA VIANA ODAKURA MARTINEZ, GERALDO LINO DE CAMPOS

BT/PCS/0213 – Um Modelo de Dados Multidimensional – PEDRO WILLEMSENS, JORGE RADY DE ALMEIDA JUNIOR

BT/PCS/0214 – Autômatos Adaptativos no Tratamento Sintático de Linguagem Natural – CÉLIA YUMI OKANO TANIWAKI, JOÃO JOSÉ NETO

BT/PCS/0215 – Fatores e Subfatores para Avaliação da Segurança em Software de Sistemas Críticos – JOÃO EDUARDO PROENÇA PÁSCOA, JOÃO BATISTA CAMARGO JÚNIOR

BT/PCS/0216 – Derivando um Modelo de Projeto a Partir de um Modelo de Análise, com Base em Design Patterns J2EE – SERGIO MARTINS FERNANDES, SELMA SHIN SHIMIZU MELNIKOFF

BT/PCS/0217 – Domínios Virtuais para Redes Móveis Ad Hoc: Um Mecanismo de Segurança – LEONARDO AUGUSTO MARTUCCI, TEREZA CRISTINA DE MELO BRITO CARVALHO

BT/PCS/0218 – Uma Ferramenta para a Formulação de Consultas Baseadas em Entidades e Papéis – ANDRÉ ROBERTO DORETO SANTOS, EDIT GRASSIANI LINO CAMPOS

BT/PCS/0219 – Avaliação de Performance de Arquiteturas para Computação de Alto Desempenho – KARIN STRAUSS, WILSON VICENTE RUGGIERO

BT/PCS/0220 – BGLsim: Simulador de Sistema Completo para o Blue Gene/L – LUÍS HENRIQUE DE BARROS CEZE, WILSON VICENTE RUGGIERO

BT/PCS/0221 - μP: Uma Solução de Micropagamentos – PEDRO ANCONA LOPEZ MINDLIN, TEREZA CRISTINA MELO DE BRITO CARVALHO

BT/PCS/0222 - Modelamento de Roteadores IP para Análise de Atraso – MARCELO BLANES, GRAÇA BRESSAN

BT/PCS/0223 - Uma Biblioteca de Classes Utilizando Java 3D para o Desenvolvimento de Ambientes Virtuais Multi-Usuários - RICARDO NAKAMURA, ROMERO TORI

BT/PCS/0224 – Interactive 3D Physics Experiments Through the Internet – ALEXANDRE CARDOSO, ROMERO TORI

BT/PCS/0225 – Avaliação do Desempenho de Aplicações Distribuídas sob Duas Velocidades de Rede – AMILCAR ROSA PEREIRA, GERALDO LINO DE CAMPOS

BT/PCS/0226 – Acompanhamento do Aprendizado do Aluno em Cursos a Distância através da WEB: Metodologias e Ferramentas – LUCIANA APARECIDA MARTINEZ ZAINA, GRAÇA BRESSAN

BT/PCS/0227 – Um Ambiente Colaborativo para Simulação de Redes de Computadores - OSCAR DANTAS VILCACHAGUA, GRAÇA BRESSAN

BT/PCS/0301 – Diretrizes para o Projeto de Base de Dados Distribuídas – PEDRO LUIZ PIZZIGATTI CORRÊA, JORGE RADY DE ALMEIDA JR.

BT/PCS/0302 – Análise e Predição de Desempenho de Programas MPI em Redes de Estações de Trabalho – JEAN MARCOS LAINE, EDSON T. MIDORIKAWA

BT/PCS/0303 – Padrões de Software para Tutores Inteligentes Cooperativos em Engenharia de Requisitos – MARIA EMILIA GOMES SOBRAL, MARIA ALICE GRIGAS VARELLA FERREIRA

BT/PCS/0304 – Performance Analysis and Prediction of Some MPI Communication Primitives – HÉLIO MARCI DE OLIVEIRA , EDSON TOSHIMI MIDORIKAWA

BT/PCS/0305 – RM-ODP para Expressar o Licenciamento Nuclear – EDILSON DE ANDRADE BARBOSA, MOACYR MARTUCCI JUNIOR

BT/PCS/0306 – Modelo de Avaliação para Métricas de Software – VINICIUS DA SILVA ALMENDRA, KECHI HIRAMA

BT/PCS/0307 – Análise de Confiabilidade de Sistemas Redundantes de Armazenamento em Discos Magnéticos – ENDERSON FERREIRA, JORGE RADY DE ALMEIDA JUNIOR

BT/PCS/0308 – Utilizando Realidade Virtual e Objetos Distribuídos na Construção de uma Ferramenta de Aprendizagem Colaborativa – O Projeto Piaget – ISMAR FRANGO SILVEIRA, MARIA ALICE GRIGAS VARELLA FERREIRA

BT/PCS/0309 – Construção de Base de Conhecimento em Prolog a partir de Páginas HTML – WAGNER TOSCANO, EDSON SATOSHI GOMI

BT/PCS/0310 – Verificação de Segurança em Confluência de Trajetórias de Aeronaves Utilizando Autômatos Híbridos – ÍTALO ROMANI DE OLIVEIRA, PAULO SÉRGIO CUGNASCA

BT/PCS/0311 – Sistemas de Reconhecimento Biométrico Aplicados à Segurança de Sistemas de Informação – VILMAR DE SOUZA MACHADO, JORGE RADY DE ALMEIDA JUNIOR

BT/PCS/0312 – Análise Comparativa de Arquiteturas Híbridas Intserv-Diffserv Utilizadas para Obtenção de QoS Fim-a-Fim em Redes IP – CARLOS A. A. BENITES, GRAÇA BRESSAN

BT/PCS/0313 – Proposta para Otimização de Desempenho do Protocolo TCP em Redes Wireless 802.11 – ANDRÉ AGUIAR SANTANA, TEREZA CRISTINA DE MELO BRITO CARVALHO

BT/PCS/0314 – Using the Moise + Model for a Cooperative Framework of MAS Reorganization – JOMI FRED HUBENER, JAIME SIMÃO SICHMAN

BT/PCS/0315 – Ferramenta para Acompanhamento da Participação do Aluno em Sessões de Fórum Aplicada no Ensino a Distância via Web – GUSTAVO BIANCHI CINELLI, GRAÇA BRESSAN

BT/PCS/0316 – Uma Infra-Estrutura para Agentes Arrematantes em Múltiplos Leilões Simultâneos – PAULO ANDRÉ LIMA DE CASTRO, JAIME SIMÃO SICHMAN

BT/PCS/0317 – Reutilização de Software Através de Geração de Código e de Desenvolvimento de Componentes – Estudo de Caso – FÁBIO FÚRIA SILVA, MARIA ALICE GRIGAS VARELLA FERREIRA

BT/PCS/0318 – Detecção Automática das Transições de Corte e Fades – IZAURA CRISTINA ARAÚJO, MARIA ALICE GRIGAS VARELLA FERREIRA

BT/PCS/0319 – Educação a Distância e a Web Semântica: Modelagem Ontológica de Materiais e Objetos de Aprendizagem para Plataforma CoL – MOYSÉS DE ARAÚJO, MARIA ALICE GRIGAS VARELLA FERREIRA

BT/PCS/0320 – Análise da Aplicação dos Padrões TMN no Gerenciamento de Sistemas de CRM – SANDRO ANTÔNIO VICENTE, MOACYR MARTUCCI JR

BT/PCS/0321 – Alinhamento de Corpus Bilíngües: Modelos e Aplicações – JOSÉ FONTEBASSO, JORGE KINOSHITA

BT/PCS/0322 – Arquitetura de Integração dos Sistemas CORBA e Fieldbus: Aplicação do Padrão ODP – DANTE LINCOLN CAROAJULCA TANTALEÁN, JORGE LUIS RISCO BECERRA

BT/PCS/0401 – Resultados Obtidos com a Implantação de um Ambiente para o Desenvolvimento de uma Maturidade em Engenharia de Software – LUIZ RICARDO BEGOSSO, LUCIA VILELA FILGUEIRAS

BT/PCS/0402 – Procedimentos para Elaboração do Modelo de Análise UML com Características de Testabilidade – ROGÉRIA CRISTIANE GRATÃO DE SOUZA, SELMA SHIN SHIMIZU MELNIKOFF

BT/PCS/0403 – INTEREXPO3D – Uma Ferramenta para Geração de Exposições Virtuais 3D Interativas – ANDRÉA ZOTOVICI, ROMERO TORI

BT/PCS/0404 – Metodologia para Especificação e Implementação de Solução de Gerenciamento – SÉRGIO CLEMENTI, TEREZA CRISTINA MELO DE BRITO CARVALHO

BT/PCS/0405 – CPAR - Cluster: Um Ambiente de Execução para Clusters de Nós Mono e Multiprocessadores – GISELE DA SILVA CRAVEIRO, LIRIA MATSUMOTO SATO

BT/PCS/0406 – Monitores de Execução de Software para Sistemas Similares de Mesma Funcionalidade – SÉRGIO RICARDO ROTA, JORGE RADY DE ALMEIDA JR.

BT/PCS/0407 – Avaliação do Perigo de Colisão entre Aeronaves em Operação de Aproximação em Pistas de Aterrissagem Paralelas – PAULO HIDESHI OGATA, JOÃO BATISTA CAMARGO JR.

BT/PCS/0408 – Integration of Ontologies and Organization Models: A MAS View – GUSTAVO GIMENEZ LUGO, JAIME SIMÃO SICHMAN

BT/PCS/0409 – The use Heuristics to Speedup Reinforcement Learning – REINALDO AUGUSTO DA COSTA BIANCHI, ANNA HELENA REALI COSTA

BT/PCS/0410 – Gestão de QOS na Visão ODP: Uma Aplicação na Arquitetura SLM – CRISTINA MORI MIYATA, JORGE LUIS RISCO BECERRA

BT/PCS/0411 – Proposta de um Modelo Simplificado de Aquisição de Software para Pequenas Empresas – PAULO SÉRGIO BRANDÃO LIMA, LÚCIA VILELA LEITE FILGUEIRAS

BT/PCS/0412 – Avaliação da Carga Mental de Trabalho na Operação de Interfaces Homem - Computador de Sistemas de Controle de Processo – JOSÉ LUIZ LOPES ALVES, LÚCIA VILELA LEITE FILGUEIRAS

BT/PCS/0413 – Identificação de Aspectos de Desenho de Interface de Documentos Hipermídia Educacionais que Influenciam na aprendizagem e Propostas de Utilização – RENATO JOSUÉ DE CARVALHO, MARIA ALICE GRIGAS VARELLA FERREIRA

BT/PCS/0501 – Representação Gramatical Adaptativa com Verificação de Aparência de Linguagens Dependentes de Contexto – CÉSAR ALBERTO BRAVO PARIENTE, JOÃO JOSÉ NETO

BT/PCS/0502 – PartNet+: Simulando Parcerias entre Múltiplos Agentes – JULIO DE LIMA DO RÊGO MONTEIRO, JAIME SIMÃO SICHMAN

BT/PCS/0503 – Um Processo de Transformação de Arquiteturas de Sistemas Legados Baseado em Reengenharia – RODRIGO ALVARES DE SOUZA, REGINALDO ARAKAKI

BT/PCS/0504 – Uma Proposta de Ontologia para Plano de Projeto – LUIS ALVES FERREIRA FILHO, EDSON SATOSHI GOMI

BT/PCS/0505 – Ensino de Arquitetura de Computadores Utilizando Simuladores Completos – ANDRÉ EVANDRO LOURENÇO, EDSON TOSHIMI MIDORIKAWA

BT/PCS/0506 – Armazenamento de Documentos XML "Text-Centric" e "Data-Centric" em Sistemas Relacionais e Objeto-Relacionais – CLÓVIS KIYOHIDE HANASHIRO, EDIT GRASSIANI LINO DE CAMPOS

BT/PCS/0507 – Integração ODP – CIM para Definição de Arquitetura de Sistema de Informação: Uma Aplicação na Qualidade de Distribuição de Energia Elétrica – PEDRO SOUZA ROSA, JORGE LUIS RISCO BECERRA

BT/PCS/0508 – Uma Ferramenta de Apoio para a Gestão do Conhecimento com Base em Representação por Mapas – SIDNEI NICOLI, EDISON SPINA

BT/PCS/0509 – Modelo de Serviços e Arquitetura para Alocação Dinâmica de Recursos com MPLS-TE – Vivian Bastos Dias, Tereza Cristina Melo de Brito Carvalho

BT/PCS/0510 – Atena: Um Sistema para Suporte ao Planejamento na Área de Gestão de Projetos – FABRÍCIO JAILSON BARTH, EDSON SATOSHI GOMI

BT/PCS/0511 – A Floor Controlled Centralized Voice Conference System for Distance Learning – GEORGE MARCEL M. A. SMETANA, TEREZA CRISTINA MELO DE BRITO CARVALHO

BT/PCS/0512 – Avaliação da Representação do Componente de Software na FARCSoft – Ferramenta de Apoio à Reutilização de Componentes de Software – ANA CLAUDIA ROSSI, SELMA SHIN SHIMIZU MELNIKOFF

BT/PCS/0513 – AdapTools: Aspectos de Implementação e Utilização – HERMESON PISTORI, JOÃO JOSÉ NETO

BT/PCS/0514 – Algoritmo Adaptativo de Substituição de Páginas LRU-WAR – HUGO HENRIQUE CASSETTARI, EDSON TOSHIMI MIDORIKAWA