APOIO À TOMADA DE DECISÃO NO PROCESSO DE SOLUÇÃO TÉCNICA EM AMBIENTES DE DESENVOLVIMENTO DE *SOFTWARE* ORIENTADOS À ORGANIZAÇÃO

Sávio Mendes de Figueiredo

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

Profa. Ana Regina Cavalcanti da Rocha, D. Sc.

Prof. Geraldo Bonorino Xexéo, D. Sc.

Profa. Káthia Marçal de Oliveira, D. Sc.

RIO DE JANEIRO, RJ-BRASIL

MARÇO DE 2006

FIGUEIREDO, SÁVIO MENDES DE

Apoio ao Processo de Solução Técnica em Ambientes de Desenvolvimento de *Software* Orientados à Organização [Rio de Janeiro] 2006

VIII, 137 p., 29,7 cm (COPPE/UFRJ, M. Sc., Engenharia de Sistemas e Computação, 2006)

Dissertação – Universidade Federal do Rio de Janeiro, COPPE

1. Solução Técnica

2. Ambientes de Desenvolvimento de *Software* Orientados à Organização

I. COPPE/UFRJ II. Título (série)

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M. Sc)

APOIO AO PROCESSO DE SOLUÇÃO TÉCNICA EM AMBIENTES DE DESENVOLVIMENTO DE *SOFTWARE* ORIENTADOS À ORGANIZAÇÃO

Sávio Mendes de Figueiredo

Março/2006

Orientadora: Ana Regina Cavalcanti da Rocha

Programa: Engenharia de Sistemas e Computação

Os produtos de software estão se tornando cada vez maiores e mais complexos. Além disso, o número de projetos de software que não conseguem ser terminados dentro dos prazos e custos estabelecidos e que não implementam um produto que satisfaça às necessidades do cliente também cresce. Neste contexto, se situam as pesquisas com foco em processos de software, pois se percebeu que a qualidade do produto a ser desenvolvido está fortemente relacionada com a qualidade do processo utilizado para desenvolvê-lo. O processo de Solução Técnica é um dos processos executados durante o desenvolvimento de software e tem por objetivo projetar e implementar uma solução para os requisitos desejados. Durante a seleção da solução a ser implementada para os requisitos deve-se investigar soluções alternativas e manter o registro das soluções investigadas e do porquê da escolha de uma solução específica.

Este trabalho apresenta uma abordagem de apoio à Solução Técnica em projetos de software, baseada nas normas ISO/IEC 12207, ISO/IEC 14102, nas áreas de processo Solução Técnica e Análise de Decisão e Resolução do CMMI (*Capability Maturity Model Integration*), nos processos de Solução Técnica e de Análise de Decisão e Resolução do MPS.BR e em abordagens existentes na literatura. Este apoio ocorre através da definição de processos e implementação de ferramentas nos Ambientes de Desenvolvimento de Software Orientados à Organização.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M. Sc)

SUPPORT TO THE TECHNICAL SOLUTION PROCESS IN ENTERPRISE-ORIENTED SOFTWARE DEVELOPMENT ENVIROMENT

Sávio Mendes de Figueiredo

March/2006

Advisor: Ana Regina Cavalcanti da Rocha

Department: Systems and Computing Engineering

The size and complexity of software products have been continuously increasing over the years. Moreover, the number of software projects that fails to finish inside the constraints of budget and schedule established and to deliver a product that addresses the needs of the customer, also increases significantly. In this context, the research with focus at software processes has began because people realized that the software product quality is extremely dependent of the software process that has been adopted in order to develop the product. The Technical Solution process is one of the processes that are executed during the software development and its purpose is to design and implement a solution for the requirements. During the selection of a solution that is going to be implemented in order to satisfy the requirements, many alternative solutions must be examined and the alternative solutions rejected must be kept in the project documentation, as well as the rationale that explains why a solution has been selected.

This work presents an approach to support the Technical Solution process in Enterprise-Oriented Software Development Environments, based on ISO/IEC 12207, ISO/IEC 14102, Technical Solution and Decision Analysis and Resolution process areas of CMMI (Capability Maturity Model Integration) and Technical Solution and Decision Analysis and Resolution processes of MPS.BR.

Agradecimentos

Ao meu pai, minha mãe, meu irmão e todos da minha família pelo apoio durante todas as etapas da minha vida, incluindo este trabalho.

À professora Ana Regina, por ter acreditado em mim, pelas oportunidades que me oferece e pela orientação desde a época da graduação.

Aos professores Geraldo Xexéo e Káthia Marçal por terem aceitado participar da minha banca de mestrado e pelas críticas para aprimorar o trabalho realizado.

Aos amigos de laboratório Sômulo e Gleison pela amizade, conselhos e por estarem sempre dispostos a ajudar.

Aos amigos de laboratório e trabalho David, Diogo, Mariano, Carlos Eduardo, Rafael, Carlos Eduardo e Sérgio pela convivência alegre durante todos os anos de mestrado e alguns da graduação.

Às amigas de trabalho e estudo Ana Cândida e Paulinha, pela amizade e por me tratarem com tanto carinho.

A todos os colegas do projeto Odyssey pela amizade e pelas conversas durante os almoços.

Aos alunos de mestrado e doutorado de engenharia de software que estudaram comigo durante estes anos. São tantas pessoas dessa lista que me ajudaram, que agradecer um por um seria impossível.

Ao amigo Fábio e aos colegas Sanderson, Pedro, Marina, Alan, Rodrigo, Natalina, Hélio, Fernando e Cristiano.

À Carolina, pela forma gentil como trata a todos do I-2000, pelos cafézinhos e pelas conversas agradáveis nos retornos para Niterói.

Ao amigo Rodrigo, sua esposa Mariana e sua filha Beatriz, por serem a minha principal forma de descontração nos momentos de *stress* e nos momentos difíceis e por estarem sempre dispostos a ajudar.

Ao pessoal da secretaria pela paciência que tiveram comigo durante estes anos.

À Daniella, pelo amor, carinho e amizade que me oferece e pela paciência nos meus momentos de ausência.

Conteúdo

CAPÍTULO 1 - INTRODUÇÃO

1.1 Motivação

O processo de Solução Técnica, um dos processos executados durante o desenvolvimento de um produto, é iniciado quando os requisitos para o problema a ser resolvido pelo software estiverem definidos, desenvolvidos e aprovados. Este processo pode ser executado tanto no contexto do software a ser desenvolvido quanto no contexto do sistema onde o software será integrado. O objetivo do processo de Solução Técnica é definir atividades que permitam a elaboração do projeto (*design*) do software e, também, possibilitem a implementação da solução de projeto para os requisitos em questão.

O processo de Solução Técnica é intenso em conhecimento e durante a sua execução é necessário tomar diversas decisões, pois podem existir diversas maneiras de solucionar um mesmo problema. Dessa forma, o conhecimento sobre Solução Técnica é de grande importância para auxiliar nas escolhas a serem tomadas. É importante armazenar o conhecimento organizacional adquirido durante a execução do processo mantendo o registro do raciocínio pelo qual uma determinada decisão foi tomada durante a etapa de projeto. Isso fica claro quando se tem um problema de rotatividade de pessoal na empresa. A alta rotatividade da equipe na indústria de software, associada com longos períodos de vida dos produtos, aumenta a probabilidade de que os projetistas originais não estejam presentes quando evoluções nos produtos forem necessárias e os problemas começarem a ocorrer (BURGE & BROWN, 2002).

Além disso, muito freqüentemente, o entendimento necessário para realizar manutenções depende da compreensão de quais decisões de projeto foram consideradas, que suposições foram feitas, que soluções alternativas foram rejeitadas e que critérios e requisitos foram satisfeitos no processo de deliberação (CONKLIN, 1989). Este tipo de conhecimento raramente é registrado e está acessível para consulta durante o período de manutenção do produto.

Outro problema comum nas organizações é a dificuldade de aproveitar o conhecimento de membros mais experientes durante o treinamento de novos membros das equipes, pois a dinâmica de trabalho não permite que os experientes parem a execução de suas atividades para compartilhar o seu conhecimento. Assim, em situações de tomada de decisão, os membros iniciantes tendem a repetir os mesmos erros cometidos por outros membros das equipes que já passaram por situações semelhantes (MONTONI, 2003). Manter o histórico das decisões ajuda que erros cometidos no passado não sejam cometidos novamente, pois se teria o conhecimento das desvantagens e problemas relacionados a uma determinada alternativa disponível para a organização.

Este tipo de documentação é chamado de *Design Rationale*. *Design Rationale*, ou simplesmente DR, é a explicação do porquê de um artefato ou alguma parte dele ter sido projetado do jeito que foi (LEE & LAI, 1991). Souza *et al.* (1998) definem DR como a documentação das decisões de projeto com suas respectivas justificativas, as opções consideradas, as avaliações e a argumentação que levaram a determinada decisão.

Um tipo mais específico de decisão que as organizações geralmente têm que tomar durante seus projetos é a decisão de desenvolver, comprar ou reutilizar um determinado componente do produto. Este tipo de decisão é muito importante para uma organização e deve levar em consideração, principalmente, dois fatores que são o custo de cada opção e o impacto nas competências centrais dessa organização que o componente possui.

É vital para as organizações que os custos de desenvolvimento dos produtos sejam reduzidos, assim como seu prazo de chegada ao mercado (*time to market*). Um meio de conseguir estes dois objetivos é delegar à outras empresas a confecção de componentes do produto, desde que estes componentes não façam parte da competência central da organização.

É necessário, portanto, a existência de uma abordagem que permita a uma organização decidir o que lhe é mais vantajoso: desenvolver um determinado componente internamente, contratar uma outra empresa para fazer este desenvolvimento ou reutilizar um componente já disponível na organização. As organizações devem ser capazes de

escolher as partes do produto que serão produzidas internamente e as partes dos produtos que serão produzidas por empresas contratadas.

1.2 Objetivos

O objetivo desta dissertação é apoiar a Solução Técnica através da definição e implementação de uma abordagem que apóie (i) a seleção de alternativas durante a etapa de projeto e (ii) a escolha entre desenvolver, comprar ou reutilizar um determinado componente. Para alcançar estes objetivos, foram definidos dois processos que permitem (i) manter o registro do raciocínio (*design rationale*) por trás das decisões tomadas durante a etapa de projeto e (ii) escolher entre as opções de desenvolver, comprar ou reutilizar.

Como parte do trabalho e como forma de mostrar a viabilidade dos processos definidos, foram desenvolvidas duas ferramentas denominadas TechSolution e MBR (***Make Buy or Reuse***). A ferramenta TechSolution tem como objetivo apoiar o projetista na seleção de alternativas e a ferramenta MBR possui a finalidade de apoiar a análise entre desenvolver, comprar ou reutilizar um determinado componente. As ferramentas possibilitam o registro das decisões tomadas, assim como o armazenamento e a posterior consulta deste registro. Estas ferramentas sugerem ao usuário critérios de seleção que podem ser utilizados na seleção da alternativa mais adequada. Além disso, a ferramenta TechSolution também fornece ao usuário conhecimento necessário para resolver determinados tipos de problemas, como a escolha do estilo arquitetural a ser utilizado.

As duas ferramentas estão integradas aos ambientes de Desenvolvimento e Manutenção instanciados a partir de Ambientes Configurados Taba para apoiar a execução de projetos de software de uma organização.

1.3 Estrutura da dissertação

Este trabalho está estruturado da seguinte forma:

- **Capítulo I - Introdução**: descreve a motivação, objetivos e organização da dissertação;

- **Capítulo II – Processos de Software:** apresenta uma revisão da literatura sobre processos de software, destacando o processo de Solução Técnica;

- **Capítulo III – *Design Rationale*:** apresenta uma revisão da literatura sobre *Design Rationale*;

- **Capítulo IV – Ambientes de Desenvolvimento de Software e a Estação Taba:** apresenta a Estação TABA e os Ambientes de Desenvolvimento de Software Orientados a Domínio e a Organização;

- **Capítulo V – Apoio a Escolha de Alternativas de Solução e a Decisão Fazer-Comprar-Reutilizar:** apresenta o processo de seleção de alternativas da etapa de projeto e o processo de apoio à decisão entre fazer, comprar ou reutilizar um determinado componente definidos neste trabalho. O capítulo também detalha as funcionalidades das ferramentas de apoio à execução destes processos que estão integradas à Estação TABA.

- **Capítulo VI – Considerações Finais**: apresenta as conclusões da dissertação, considerações finais e trabalhos futuros que podem ser realizados;

- **Referências Bibliográficas**: contém a listagem das referências bibliográficas utilizadas;

- **Anexo 1**: contém o questionário utilizado durante um *survey* (ROCHA *et al.*, 2005) para identificação das dificuldades e fatores de sucesso encontrados ou percebidos por implementadores do MPS.BR (Melhoria de Processo de Software Brasileiro) e/ou do CMMI (*Capability Maturity Model Integration*);

- **Anexo 2**: exibe a linguagem de modelagem de processos utilizada nesta dissertação;

- **Anexo 3**: apresenta a norma ISO/IEC 14102.

- **Anexo 4**: apresenta um resumo da pesquisa executada por XAVIER (2001).

CAPÍTULO 2 – PROCESSOS DE SOFTWARE

2.1 Introdução

Processo de software pode ser definido como o conjunto de atividades, métodos, práticas e transformações que as pessoas utilizam para desenvolver e manter software e produtos associados, como planos do projeto, documentos de projeto, código fonte, casos de teste e manuais do usuário (PAULK *et al.*, 1993). A norma ISO/IEC 12207 (2000) define processo como um conjunto de atividades inter-relacionadas que transformam entradas em saídas.

De acordo com PFLEEGER (2004) todo processo de software: (i) prescreve todas as suas principais atividades, (ii) utiliza recursos, está sujeito a restrições e gera produtos intermediários e finais, (iii) pode ser composto de sub-processos que estão relacionados de algum modo, (iv) possui na descrição das atividades os critérios de entrada e os critérios de saída das mesmas, (v) possui as atividades organizadas em uma seqüência, (vi) tem um conjunto de diretrizes que explicam os objetivos de cada atividade e (vii) possui restrições e controles que podem ser aplicados a uma atividade, recurso ou produto.

A atenção atual no estudo de processos de software é fruto da percepção de que a qualidade do produto de software desenvolvido está fortemente relacionada com a qualidade do processo de software utilizado para o seu desenvolvimento (FUGGETA, 2000, GOMES, 2001, GOLUBIÉ, 2005). Cada vez mais os desenvolvedores de software percebem que a sua habilidade em satisfazer os contratos está fortemente relacionada com os processos que utilizam para desenvolver o produto (SHEARD, 1997). No entanto, é essencial que o processo de software a ser utilizado definido esteja configurado de acordo com as características da organização e do produto a ser desenvolvido (GOLUBIÉ, 2005).

A importância da utilização do processo de software não se restringe às grandes organizações. Pelo contrário, as atividades relacionadas com o processo de software podem

ser executadas e têm sido executadas com grande sucesso por organizações pequenas (SWEBOK, 2004).

O objetivo dos estudos relacionados com processos é melhorar o desenvolvimento de software ao propor melhores maneiras de: (i) definir e modelar o desenvolvimento, (ii) descobrir as fraquezas da organização e (iii) melhorar a organização no nível dos processos e no nível da organização como um todo. Basicamente, estes estudos focam na definição, avaliação e melhoria do processo de software (KAWALEK & WASTELL, 1996, DERMIANE *et al.*, 1999, FUGGETA, 2000).

Neste contexto surgiram normas e modelos de maturidade para avaliar a qualidade dos processos de *software* existentes em uma organização e apoiar a melhoria destes processos. A melhoria dos processos consiste de um conjunto de ações que uma organização executa de modo a alterar os seus processos, tornando-os mais adequados para satisfazer as necessidades e os objetivos de negócio da organização. Os objetivos da organização podem ser, por exemplo, aumentar a satisfação do cliente, criar produtos e serviços de alta qualidade, aumentar a produtividade, diminuir o re-trabalho, aumentar a precisão das estimativas e/ou diminuir o *time-to-market* (IBRAHIM & PYSTER, 2004).

Este capítulo faz uma revisão da literatura sobre processos de software, descrevendo a norma ISO/IEC 12207 e suas evoluções, a norma ISO/IEC 15504 e os modelos de maturidade CMMI (*Capability Maturity Model Integration*) e MPS.BR (Melhoria de Processo de Software Brasileiro). Em seguida, o processo de Solução Técnica é apresentado, mostrando-se como este é tratado nos modelos de maturidade e normas descritos neste capítulo. Finalmente, é apresentado um mapeamento mostrando como os modelos e as normas apresentados são equivalentes em relação à Solução Técnica.

2.2 Fatores de Sucesso e Dificuldades encontrados durante a Implementação de Processos de Software

Observa-se na indústria várias iniciativas para a melhoria dos processos (SOUZA & OLIVEIRA, 2005, VIANA & VASCONCELLOS, 2005, FERREIRA *et al.*, 2005,

ANTONIOL *et al.*, 2004, TUFFLEY *et al.*, 2004, MOREAU *et al.*, 2003, KOMIYAMA *et al.*, 2000, DEBOU & KUNTZMANN-COMBELLES, 2000). Apesar do esforço, muitas destas tentativas fracassam devido a diversos fatores, dentre os quais pode-se destacar (ROCHA *et al.*, 2005, DEBOU & KUNTZMANN-COMBELLES, 2000):

- Falta de apoio da gerência.
- Falta de infra-estrutura para a implantação dos processos.
- Estabelecimento de cronogramas impossíveis de serem atingidos.
- Mudança da cultura organizacional ocasionada pela implantação dos processos.
- Falta de conhecimento em engenharia de software por parte dos membros da organização.
- Pressão no cronograma dos projetos que provocava o abandono do processo por parte dos membros da equipe (isto também é reflexo da falta de apoio da gerência).
- Demora em visualizar os resultados.

ROCHA *et al.* (2005) apresentam o resultado de um *survey* realizado com o objetivo de identificar fatores de sucesso e dificuldades relacionados à implementação de processos de software utilizando o modelo de referência do MPS.BR (MR-MPS.BR) e o CMMI. Como participantes do *survey*, foram selecionados 15 implementadores de processos de software participantes de projetos coordenados pela COPPE/UFRJ em empresas públicas e privadas de diversos portes na implantação do MPS.BR, do CMMI ou da implementação conjunta destes dois modelos.

Estes participantes receberam dois formulários para preencher. No primeiro formulário, os participantes deveriam informar as dificuldades encontradas ou percebidas pelos mesmos durante a implementação de processos de software. No segundo formulário, os participantes deveriam informar os fatores que possibilitaram o sucesso na implementação dos processos de software. O formulário utilizado para o preenchimento das

dificuldades encontradas e dos fatores de sucesso percebidos pelos implementadores pode ser visualizado no anexo 1 desta dissertação.

Após a devolução dos questionários e a análise dos dados, foram identificadas 12 categorias de achados relacionados aos fatores de sucesso e 16 categorias de achados relacionados às dificuldades na implementação de processos de software utilizando o MR-MPS.BR e o CMMI. Estes resultados podem ser observados nas Figuras 2.1 e 2.2. Os achados são divididos de acordo com os dois modelos de negócio descritos na guia geral do MR-MPS.BR: Modelo de Negócio Cooperado em grupo de empresa (pacote) e Modelo de Negócio Específico para cada empresa (personalizado).

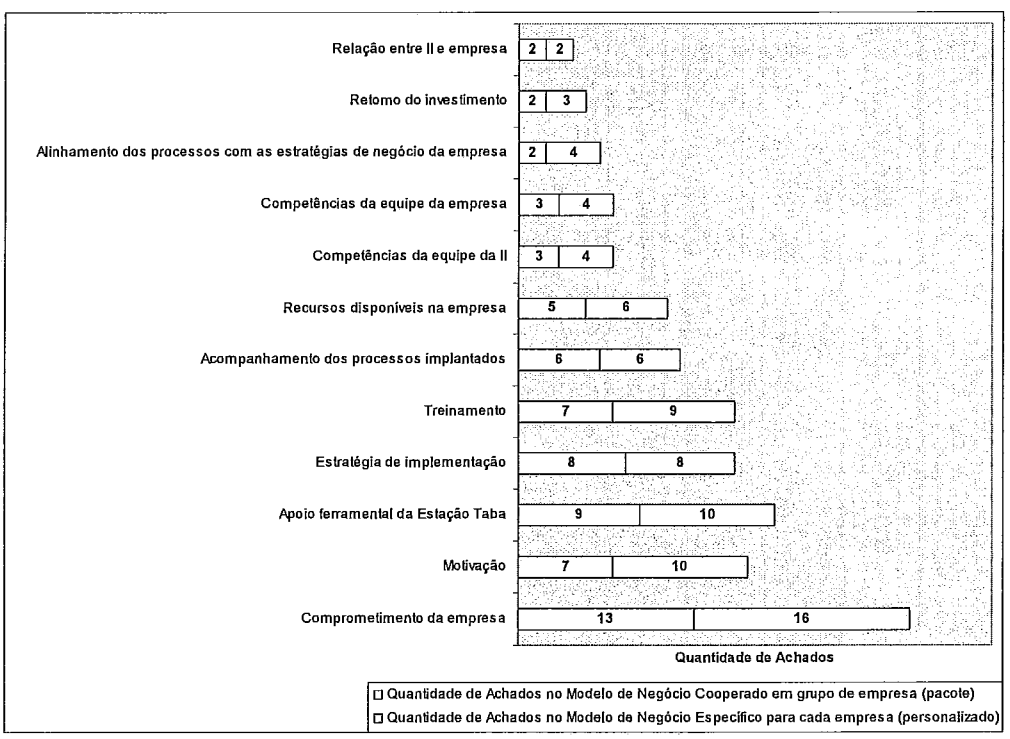


Figura 2.1 - Quantidade de achados relacionados aos fatores de sucesso na implantação de processos de software utilizando o MR-MPS.BR e o CMMI (ROCHA *et al.*, 2005)

Conforme pode ser visto na Figura 2.1, o maior fator de sucesso na implementação de processos de softwares está relacionado com o comprometimento dos membros da empresa com o projeto de implantação de processos na empresa. Este comprometimento engloba a gerência e os colaboradores da empresa. Os resultados também ressaltam a

importância de ter as pessoas da empresa motivadas e de possuir um apoio ferramental, que neste caso foi provido através da Estação Taba.

Em relação às dificuldades na implantação de processos de software utilizando o MR-MPS.BR e o CMMI, as maiores dificuldades estão relacionadas à competência da equipe da empresa. Percebeu-se um baixo conhecimento em engenharia de software entre as pessoas da empresa, o que dificulta a implantação de processos de software. Também foi apresentada uma grande dificuldade em relação à mudança da cultura organizacional durante a implantação de processos. Houve grande dificuldade em customizar os processos padrões de acordo com as necessidades da organização quando já existia uma cultura com problemas sobre os procedimentos de engenharia de software.

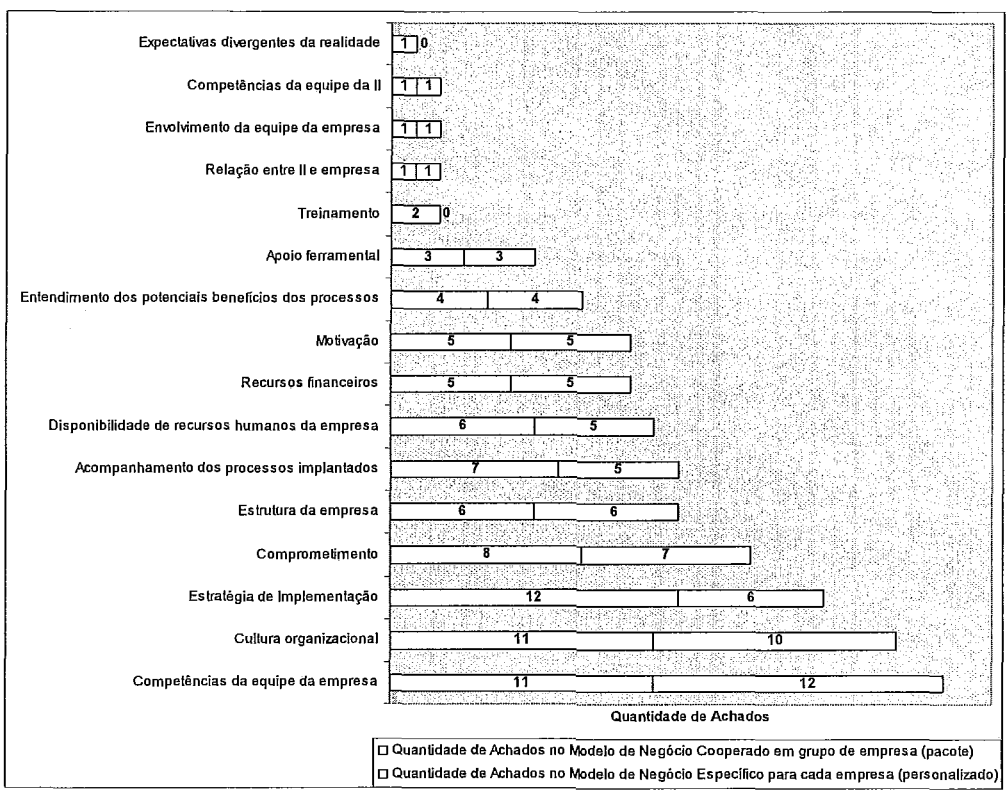


Figura 2.2 - Quantidade de achados relacionados às dificuldades na implantação de processos de software utilizando o MR-MPS.BR e o CMMI (ROCHA *et al.*, 2005)

Em relação à estratégia de implementação, as dificuldades foram encontradas, por exemplo, na demora para a tomada de decisões por precisar levar em consideração opiniões

de diversas fontes. Outra dificuldade relacionada a esta categoria foi a parceria com grupos locais inexperientes quando a empresa está em outra cidade. A estratégia de implementação também pode ser bastante dificultada quando não existe um processo de desenvolvimento e manutenção integrados aos processos.

2.3 Normas e Modelos de Maturidade para Processos de Software

Motivados pela preocupação com a qualidade do software e pelo entendimento de que um bom processo conduz a melhores softwares, foram definidos normas e modelos nacionais e internacionais com o objetivo de apoiar a definição, avaliação e melhoria da qualidade dos processos de software. A seguir são descritos os principais modelos e normas relacionados a processos de software.

2.3.1 Norma ISO / IEC 12207

A norma ISO/IEC 12207 (2000) estabelece uma estrutura comum para os processos de ciclo de vida de software. A estrutura apresentada na norma é composta por processos, atividades e tarefas que servem para ser aplicadas durante a aquisição de um sistema que contém software, de um produto de software independente ou de um serviço de software e durante o fornecimento, desenvolvimento, operação e manutenção de produtos de software. Os processos, atividades e tarefas apresentados na norma podem ser adaptados de acordo com os projetos de software.

Apesar de descrever a arquitetura dos processos de ciclo de vida do software, a norma não especifica os detalhes de como implementar ou executar as atividades e tarefas incluídas nos processos.

A norma agrupa as atividades que podem ser executadas durante o ciclo de vida de software em processos fundamentais, processos de apoio e processos organizacionais conforme descrito a seguir:

- *Processos Fundamentais*: São os processos que atendem as partes fundamentais do ciclo de vida de software. Uma parte fundamental é aquela que inicia ou executa o desenvolvimento, operação ou manutenção dos produtos de software.

- *Processos de Apoio*: São os processos que auxiliam um outro processo como uma parte integrante, com um propósito distinto e contribuem para o sucesso e qualidade do projeto de software. Um processo de apoio é empregado e executado, quando necessário, por outro processo.

- *Processos Organizacionais*: São os processos empregados por uma organização para estabelecer e implementar uma estrutura subjacente, constituída de processos de ciclo de vida e pessoal associados, e melhorar continuamente a estrutura e os processos.

Duas evoluções da norma ISO/IEC 12207 foram publicadas em 2002 (ISO/IEC, 2002) e 2004 (ISO/IEC, 2004). Estas revisões foram definidas devido à evolução natural da Engenharia de Software ao longo do tempo, às observações de seus usuários e, também, à necessidade de sua adequação à norma ISO/IEC 15504 (ISO/IEC 15504, 2003) que trata da avaliação de processos de software. Com as duas revisões, novos processos foram definidos (Gerência de Ativos, Gerência de Pedidos de Mudança, Engenharia de Domínio, Recursos Humanos, Avaliação de Produto, Gerência de Programa de Reuso, Usabilidade) e outros foram expandidos.

Durante o desenvolvimento do relatório técnico ISO/IEC TR 15504-2 foram levantadas questões em relação à granularidade dos processos na norma ISO/IEC 12207 devido à dificuldade em pontuar os processos com a finalidade de avaliá-los e melhorá-los. Para resolver este problema de granularidade, os processos foram definidos na evolução da norma através de um propósito e uma lista de resultados esperados.

2.3.2 CMMI

O CMMI (*Capability Maturity Model Integration*) (CHRISSIS *et al.*, 2003) foi criado pelo SEI (*Software Engineering Institute*) como uma integração e evolução dos

modelos: SW-CMM (*Capability Maturity Model for Software*), SECM - EIA 731 (*System Engineering Capability Model*) e IPD-CMM (*Integrated Product Development* CMM). O CMMI é um modelo alinhado com a Norma ISO/IEC 15504 e é apresentado em duas representações: uma por estágio (como o CMM) e outra contínua (semelhante à ISO/IEC 15504).

O CMMI é uma abordagem de melhoria de processo que fornece às organizações os elementos essenciais de processos efetivos. O modelo pode ser usado para guiar a melhoria de processos em projetos, em uma divisão ou em uma organização por inteira. O CMMI consiste de boas práticas que tratam do desenvolvimento e manutenção de produtos e serviços cobrindo o ciclo de vida de um produto desde a concepção até a entrega e a manutenção.

O modelo é composto de 24 áreas de processo. Cada área contém objetivos que por sua vez são formados por um conjunto de práticas relacionadas que, quando implementadas coletivamente, satisfazem um conjunto de objetivos considerados importantes para fazer melhorias significativas nesta área.

A descrição de cada área de processo está dividida em objetivos específicos e objetivos genéricos. A função dos objetivos específicos é especificar as características únicas que devem estar presentes para que uma determinada área de processo seja satisfeita. Por outro lado, os objetivos genéricos estão associados a mais de uma área de processo e especificam as características que devem estar presentes para institucionalizar os processos que implementam a área de processo.

Os objetivos específicos possuem um conjunto de práticas específicas que são as descrições de atividades consideradas importantes para que o objetivo específico seja satisfeito. De forma análoga, uma prática genérica é a descrição de uma atividade considerada importante para a satisfação de um objetivo genérico.

Como citado anteriormente, existem dois tipos de representação no CMMI: em estágios e contínua. A representação em estágios oferece um caminho sistemático e estruturado para obter a melhoria de processo gradualmente. Esta representação define um

conjunto de áreas de processo para especificar um caminho de melhoria para a organização, descrito em termos de níveis de maturidade. Os possíveis níveis de maturidade da organização na representação por estágios são: 1 – Inicial; 2 – Gerenciado; 3 – Definido; 4 - Gerenciado Quantitativamente; 5 – Otimizado.

A representação contínua oferece uma abordagem flexível para a melhoria de processos. Esta representação (conforme é utilizada na norma ISO/IEC 15504) permite que uma organização selecione uma área de processo específica e melhore com relação a esta área. A representação contínua usa níveis de capacidade para caracterizar melhoria relacionada a uma área de processo. Os possíveis níveis de capacidade de uma área de processo são: 0 – Incompleto; 1 – Desempenhado; 2 – Gerenciado; 3 – Definido; 4 - Gerenciado Quantitativamente; 5 – Otimizado.

A Tabela 2.1 apresenta as áreas de processo com suas categorias e níveis de maturidade associados.

Tabela 2.1 – Áreas de Processo do CMMI

Área de Processo	Categoria	Nível de Maturidade
Resolução e Análise Causal	Apoio	5
Gerência de Configuração	Apoio	2
Análise de Decisão e Resolução	Apoio	3
Análise e Medição	Apoio	2
Ambiente Organizacional para Integração	Apoio	3
Garantia da Qualidade do Processo e do Produto	Apoio	2
Equipe Integrada	Gerência de Projeto	3
Gerência de Riscos	Gerência de Projeto	3
Gerência de Acordo com o Fornecedor	Gerência de Projeto	2
Gerência Quantitativa do Projeto	Gerência de Projeto	4
Monitoração e Controle do Projeto	Gerência de Projeto	2
Planejamento do Projeto	Gerência de Projeto	2
Gerência de Projeto Integrada	Gerência de Projeto	3
Gerência de Fornecedor Integrada	Gerência de Projeto	3
Performance do Processo Organizacional	Gerência de Processo	4
Treinamento Organizacional	Gerência de Processo	3
Inovação Organizacional e Posicionamento Estratégico	Gerência de Processo	5
Definição do Processo Organizacional	Gerência de Processo	3
Foco no Processo Organizacional	Gerência de Processo	3
Desenvolvimento de Requisitos	Engenharia	3
Gerência de Requisitos	Engenharia	2
Integração do Produto	Engenharia	3
Solução Técnica	Engenharia	3
Validação	Engenharia	3
Verificação	Engenharia	3

A representação contínua tem o foco na capacidade das áreas de processo medida em níveis de capacidade e a representação em estágios tem o foco na maturidade da organização medida em níveis de maturidade. Essas dimensões do CMMI são usadas para guiar a organização nos seus esforços de melhoria e, também, para atividades de *benchmarking* e avaliação.

A Tabela 2.2 compara os seis níveis de capacidade com os cinco níveis de maturidade. É importante se observar que o ponto de partida é diferente nas duas representações.

Tabela 2.2 – Comparação dos Níveis de Capacidade e Maturidade (CHRISSIS *et al.*, 2003)

Nível	**Representação Contínua Níveis de Capacidade**	**Representação em Estágios Níveis de Maturidade**
0	Incompleto	Não existe
1	Realizado	Inicial
2	Gerenciado	Gerenciado
3	Definido	Definido
4	Gerenciado Quantitativamente	Gerenciado Quantitativamente
5	Em Otimização	Em Otimização

As seguintes regras resumem a equivalência entre os estágios nas duas representações:

- Para atingir o nível de maturidade 2, todas as áreas de processo relacionadas ao nível de maturidade 2 devem atingir pelo menos o nível de capacidade 2.
- Para atingir o nível de maturidade 3, todas as áreas de processo relacionadas aos níveis de maturidade 2 e 3 devem atingir pelo menos o nível de capacidade 3.
- Para atingir o nível de maturidade 4, todas as áreas de processo relacionadas aos níveis de maturidade 2, 3 e 4 devem atingir pelo menos o nível de capacidade 3.
- Para atingir o nível de maturidade 5, todas as áreas de processo relacionadas aos níveis de maturidade 2, 3, 4 e 5 devem atingir pelo menos o nível de capacidade 3.

2.2.3 ISO/IEC 15504

O SPICE (Software Process Improvement and Capability Determination), também conhecido como ISO/IEC 15504 (2003), foi publicado pela primeira vez em 1998 com a designação de Relatório Técnico do Tipo 2, sendo composto de nove partes. Um relatório do Tipo 2 é criado quando o assunto está ainda em desenvolvimento técnico ou quando existe a possibilidade de elaboração de uma Norma Internacional.

Como resultado da revisão do relatório foi desenvolvida a norma ISO/IEC 15504. A nova versão da norma está organizada em cinco partes sob o título genérico Tecnologias de Informação – Avaliação de Processos. As cinco partes que compõem a norma são:

- Parte 1: Conceitos e vocabulário (ISO/IEC 15504-1, 2004).
- Parte 2: Execução de uma avaliação (ISO/IEC 15504-2, 2004)..
- Parte 3: Guia sobre como executar uma avaliação (ISO/IEC 15504-3, 2004).
- Parte 4: Guia para utilização em processos de melhoria e na determinação da capacidade de processos (ISO/IEC 15504-4, 2004).
- Parte 5: Um exemplo de um modelo de avaliação de processos (ISO/IEC 15504-5, 2004).

O modelo de referência define um modelo bidimensional de capacidade de processos. Em uma dimensão, a dimensão de processos, os processos associados com o software são definidos e classificados em três categorias de processos conforme a ISO/IEC 12207: processos primários, processo de apoio e processos organizacionais. Associado a cada categoria de processos, existe um conjunto de processos. Cada processo do modelo de avaliação é descrito em termos de um propósito e práticas base. Uma prática base é uma atividade que ajuda a satisfazer o propósito de um processo em particular.

A dimensão de processos está baseada nos propósitos e saídas dos processos descritos nas emendas 1 e 2 da norma ISO/IEC 12207 e nas atividades e tarefas descritas na norma ISO/IEC 12207.

Na segunda dimensão, uma série de atributos de processos agrupados em níveis de capacidade é definida. Estes atributos fornecem características mensuráveis da capacidade dos processos.

Os níveis de capacidade existentes no modelo são:

- *Otimizado*: O processo previsível é melhorado continuamente com a finalidade de satisfazer as necessidades atuais e projetadas de negócio da organização.

- *Previsível*: O processo estabelecido opera dentro de limites bem definidos a fim de alcançar os resultados do processo.

- *Estabelecido*: O processo gerenciado é implementado através de um processo definido capaz de alcançar os resultados do processo.

- *Gerenciado*: O processo executado é gerenciado (planejado, monitorado e corrigido) e seus produtos de trabalho são estabelecidos, controlados e mantidos.

- *Executado*: O processo é implementado e alcança o propósito do processo.

- *Incompleto*: Neste nível o processo não é implementado ou falha em alcançar o propósito do processo.

2.2.4 MPS.BR

Em dezembro de 2003 houve o início da elaboração do MPS.BR (MPS.BR, 2005a) (Melhoria de Processo de Software Brasileiro) pelas instituições listadas a seguir: Organizações Integrantes do Sistema SOFTEX (SOFTEX: Associação para Promoção da

Excelência do Software Brasileiro (coordenadora do projeto); Núcleo SOFTEX Campinas: Sociedade Núcleo SOFTEX 2000; RIOSOFT: Sociedade Núcleo de Apoio a Produção e Exportação de Software do Rio de Janeiro) e Instituições de Ensino, Pesquisa e Centros Tecnológicos (COPPE/UFRJ: Programa de Engenharia de Sistemas e Computação da Universidade Federal do Rio de Janeiro; CESAR: Centro de Estudos e Sistemas Avançados do Recife; CenPRA: Centro de Pesquisas Renato Archer; Sociedade de Economia Mista; CELEPAR: Companhia de Informática do Paraná).

O MPS.BR tem como objetivo definir um modelo de melhoria e avaliação de processo de software visando preferencialmente, mas não exclusivamente, as micro, pequenas e médias empresas, de forma a atender às suas necessidades de negócio e ser reconhecido nacional e internacionalmente como um modelo aplicável à indústria de software.

A base técnica utilizada para a construção do modelo de referência MPS.BR são as normas NBR ISO/IEC 12207 e suas emendas 1 e 2 e a ISO/IEC 15504 e seu Modelo de Avaliação de Processo de Software ISO/IEC 15504-5, estando o modelo totalmente aderente a essas normas. O MPS.BR também cobre o conteúdo do CMMI-SE/SW, sendo compatível com este modelo.

O Modelo de Referência de Melhoria de Processo de Software (MR-MPS) contém os requisitos que as organizações deverão atender para estar em conformidade com o modelo. O modelo de referência contém as definições dos níveis de maturidade, da capacidade de processos e dos processos em si. O MPS.BR está descrito através dos seguintes documentos:

- *Guia Geral*: Esta guia contém a descrição geral do MPS.BR e detalha o Modelo de Referência de Melhoria de Processo de Software (MR-MPS), seus componentes e as definições comuns necessárias para o seu entendimento (MPS.BR, 2005a).

- *Guia de Aquisição*: Esta guia contém as recomendações para a condução de compras de software e serviços correlatos (MPS.BR, 2005b).

- *Guia de Aquisição*: Esta guia contém as recomendações para a condução de compras de software e serviços correlatos (MPS.BR, 2005b).

- *Guia de Avaliação*: Esta guia, ainda em fase de revisão antes de sua publicação, contém a descrição do processo de avaliação, os requisitos para o avaliador, os requisitos para a avaliação, o método e os formulários para apoiar a avaliação.

A Figura 2.3 exibe a estrutura do MR-MPS. Para alcançar um determinado nível de maturidade, uma organização deve implementar os processos necessários para o nível em questão e também demonstrar que os seus processos possuem a capacidade esperada para aquele nível. A definição dos processos no MR-MPS segue a definição de processos da emenda 1 da norma ISO/IEC 12207, declarando o propósito e os resultados esperados de sua execução.

A capacidade do processo demonstra a sua habilidade para alcançar os objetivos de negócio, atuais e futuros e está relacionada com o atendimento dos atributos do processo associados aos processos propriamente dito de cada nível de maturidade.

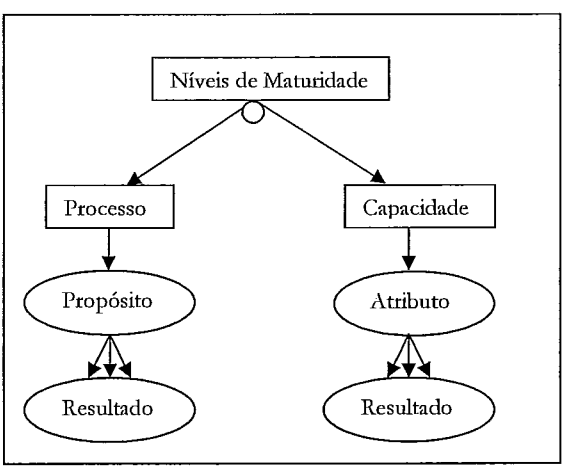


Figura 2.3 – Estrutura do MR-MPS (MPS.BR, 2005a)

O MR-MPS define sete níveis de maturidade: A (Em Otimização), B (Gerenciado Quantitativamente), C (Definido), D (Largamente Definido), E (Parcialmente Definido), F (Gerenciado) e G (Parcialmente Gerenciado). A escala de maturidade se inicia no nível G e progride até o nível A. A divisão em estágios, embora baseada nos níveis de maturidade do

CMMI-SE/SW tem uma graduação diferente, com o objetivo de possibilitar uma implementação e avaliação mais gradual e adequada às pequenas e médias empresas. A possibilidade de se realizar avaliações considerando mais níveis permite uma visibilidade dos resultados de melhoria de processos com prazos mais curtos.

A Tabela 2.3 mostra os processos que devem ser implementados de acordo com o nível de maturidade desejado, assim como os atributos de processo.

Tabela 2.3 – Níveis de Maturidade do modelo MR-MPS (MPS.BR, 2005a)

Nível	Processo	Capacidade
A	Inovação e Implantação na Organização	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
	Análise e Resolução de Causas	
B	Desempenho do Processo Organizacional	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
	Gerência Quantitativa do Projeto	
C	Análise de Decisão e Resolução	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
	Gerência de Riscos	
D	Desenvolvimento de Requisitos	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
	Solução Técnica	
	Integração do Produto	
	Instalação do Produto	
	Liberação do Produto	
	Verificação	
	Validação	
E	Treinamento	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
	Avaliação e Melhoria do Processo Organizacional	
	Definição do Processo Organizacional	
	Adaptação do Processo para Gerência de Projeto	
F	Medição	AP 1.1, AP 2.1 e AP 2.2
	Gerência de Configuração	
	Aquisição	
	Garantia da Qualidade	
G	Gerência de Requisitos	AP 1.1 e AP 2.1
	Gerência de Projeto	

* AP: Atributo de Processo

Neste momento o MPS.BR, embora com apenas 2 anos de existência contados desde o início dos trabalhos para sua definição, já exibe resultados significativos:

- Mais de 200 implementadores credenciados pela SOFTEX através de cursos e provas.

- 6 instituições implementadoras credenciadas pela SOFTEX em diversas regiões do país.

- 6 empresas avaliadas: In forma, Recife (Nível G), BL Informática, Niterói (Nível F), Compera, Campinas (Nível F), Programmer's, Campinas (Nível F), Relacional, Rio de Janeiro (Nível E) e Fábrica do TRE (Nível G).

- Apoio do BID (Banco Interamericano de Desenvolvimento) para a implantação em grupos de empresas brasileiras e em dois países latino-americanos.

- Pontuação do MPS.BR em licitações.

2.4 Processo de Solução Técnica

O processo de Solução Técnica, presente na ISO/IEC 12207, no ISO/IEC 15504, no CMMI e no MPS.BR, é um dos processos executados durante o desenvolvimento de um produto e tem início quando os requisitos para o problema a ser resolvido pelo software estão definidos, desenvolvidos e aprovados. O objetivo do processo de solução técnica é projetar (*design*) e implementar uma solução para os requisitos em questão. Este processo pode ser executado tanto para desenvolvimento de software quanto de sistemas.

Processos associados com a área de processo Solução Técnica recebem os requisitos do produto ou dos componentes de produto dos processos de gerência de requisitos. Os processos de gerência de requisitos colocam os requisitos, que se originaram nos processos de desenvolvimento de requisitos, sobre a apropriada gerência de configuração e mantém sua rastreabilidade com os requisitos anteriores (CHRISSIS *et al.*, 2003).

Projeto é o processo criativo de transformar o problema em uma solução, sendo que a descrição da solução também é chamada de projeto (PFLEEGER, 2004). PRESSMAN (2001) define projeto como a representação significativa de alguma coisa a ser construída. Os clientes sabem o que o sistema deve fazer e os construtores precisam saber como o sistema funcionará. Por isso, o projeto é um processo iterativo constituído de duas partes: o projeto conceitual ou projeto do sistema mostra ao cliente exatamente o que o sistema fará e o projeto técnico, que é uma tradução detalhada do projeto conceitual, permite que os

construtores do sistema saibam quais são o hardware e software necessários para resolver o problema do cliente (PFLEEGER, 2004).

De acordo com SHAW e GARLAN (1996) a definição da arquitetura do software é o primeiro passo a ser tomado durante o projeto de software, sendo possível identificar três níveis de projeto:

- *Projeto da Arquitetura*: Este nível de projeto associa as capacidades do sistema identificadas na especificação de requisitos com os componentes do sistema que irão implementá-las e a interconexão entre estes componentes. Os componentes da arquitetura são, geralmente, os módulos do sistema.

- *Projeto do Código*: Este nível de projeto envolve algoritmos e estruturas de dados. Os componentes são primitivas de linguagens de programação, como por exemplo, números, caracteres, ponteiros e estruturas de controle.

- *Projeto Executável*: Este nível de projeto detalha ainda mais o código, discutindo detalhes, por exemplo, de alocação de memória, de formato dos dados e de padrões de bits.

Durante o projeto da arquitetura, também é definido o estilo arquitetural a ser utilizado na construção do sistema. O estilo arquitetural envolve os componentes do sistema, os conectores e as restrições sobre a combinação dos componentes. Como exemplos de estilos arquiteturais pode-se citar: *pipe and filter*, objetos, chamada implícita, formação de camadas, repositórios, interpretadores e controle de processos (SHAW & GARLAN, 1996, PFLEEGER, 2004).

Em relação à avaliação do projeto, é necessário estabelecer guias que orientem a avaliação. PRESSMAN (2001) e PFLEEGER (2004) definem características de um bom projeto:

- Deve possuir alternativas de solução e selecionar a alternativa mais adequada com base nos requisitos, recursos disponíveis e nos conceitos de projeto.

- Deve ser rastreável em relação ao modelo de análise.
- Não deve "reinventar a roda", ou seja, deve observar padrões que possa reutilizar.
- Deve possuir a mesma estrutura que o domínio do problema a ser resolvido pelo projeto.
- Deve exibir uniformidade e integração.
- Deve permitir mudanças no mesmo.
- Deve acomodar eventos não esperados de maneira "sutil".
- Deve estar em um nível de abstração maior do que o código.
- Deve ser avaliado em relação à qualidade enquanto é criado, e não apenas depois de ter sido totalmente definido.
- Deve ser revisado para identificar erros de semântica.

A figura 2.4 exibe o relacionamento entre a área de processo Solução Técnica com as demais áreas de processo relacionadas com a engenharia no modelo CMMI. A área de processo Desenvolvimento de Requisitos fornece os requisitos para a área de processo de Solução Técnica, aonde estes requisitos são convertidos em uma arquitetura do produto, projeto dos componentes e no produto propriamente dito. A área de processo Integração do Produto também recebe os requisitos, combina os componentes do produto e verifica as interfaces para assegurar que as interfaces satisfazem os requisitos de interface estabelecidos no desenvolvimento de requisitos (CHRISSIS *et al.*, 2003).

A área de processo Gerência de Requisitos descreve as atividades para obter e controlar as mudanças nos requisitos e assegurar que os planos e dados relevantes estão atualizados em relação aos requisitos. Esta área fornece rastreabilidade entre os requisitos do cliente e os produtos e componentes de produtos (CHRISSIS *et al.*, 2003).

A área de processo Solução Técnica também desenvolve pacotes de dados técnicos para os componentes que serão utilizados pelas áreas de processo Integração do Produto e Gerência de Acordo com os Fornecedores. Durante a sua execução, soluções alternativas são examinadas com a finalidade de selecionar o projeto mais adequado com base em critérios estabelecidos. A área de processo Verificação verifica o projeto especificado e a área de processo Validação valida de forma incremental o produto em relação às necessidades do cliente (CHRISSIS *et al.*, 2003). A área de processo Integração do Produto identifica a melhor seqüência de integração possível, integra os componentes do produto e entrega o produto ao cliente. Durante a sua execução, as áreas de processo Verificação e Validação para garantir a qualidade (CHRISSIS *et al.*, 2003).

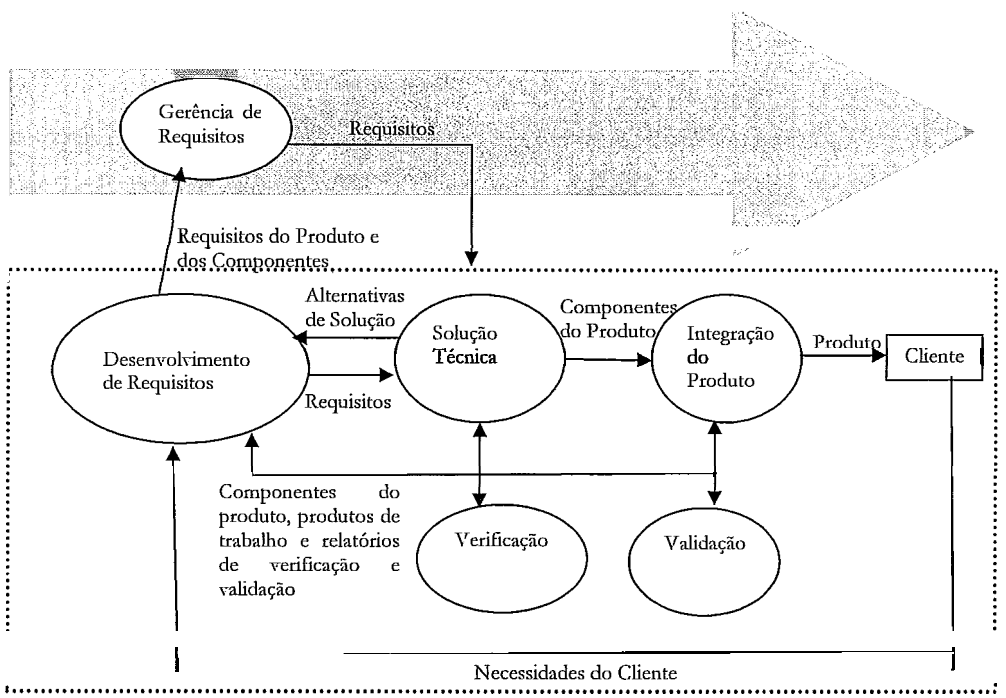


Figura 2.4 – Áreas de Processo de Engenharia (CHRISSIS *et al.*, 2003)

As próximas seções fornecerão maiores detalhes sobre como o processo de Solução Técnica é descrito nos modelos e normas apresentados.

2.4.1 A área de processo Solução Técnica no CMMI

O propósito da área de processo Solução Técnica no CMMI é projetar, desenvolver e implementar soluções para os requisitos. As soluções, os projetos, e as implementações englobam produtos, componentes dos produtos e processos de ciclo de vida relativos aos produtos tanto de forma única quanto em combinações conforme for apropriado. Esta área de processo foca nos seguintes aspectos:

- Avaliar e selecionar soluções (algumas vezes referidas como "abordagens do projeto", "conceitos do projeto" ou "projetos preliminares") que potencialmente satisfazem um conjunto apropriado de requisitos alocados.

- Desenvolver projetos detalhados para as soluções selecionadas (detalhados no contexto de conter toda a informação necessária para a manufatura, codificação, ou outra forma de implementar o projeto como um produto ou componente de produto).

- Implementar os projetos na forma de produtos ou componentes do produto.

Tipicamente, estas atividades se apóiam iterativamente. Algum nível de projeto pode ser necessário para selecionar soluções. Protótipos dos componentes de produto podem ser usados como uma forma de ganhar conhecimento suficiente para desenvolver um pacote de dados técnico ou um conjunto completo de requisitos.

Durante a execução do processo de solução técnica soluções alternativas e seus méritos são considerados antes de se selecionar uma determinada solução para um projeto. Deste modo, requisitos chave, questões do projeto e restrições são estabelecidas para serem utilizadas em análises de soluções alternativas. As características da arquitetura que fornecem uma base para a melhoria e evolução do produto são consideradas, assim como o

uso de COTS[2] é considerado em relação ao custo, cronograma, desempenho e riscos (CHRISSIS *et al.*, 2003).

Para implementar a área de processo Solução Técnica no CMMI uma organização necessita satisfazer os três objetivos específicos que esta área de processo possui, além de satisfazer os objetivos genéricos. Cada objetivo possui um conjunto de práticas que devem ser implementadas pelo processo de Solução Técnica que a organização executa para que a mesma possa satisfazer os objetivos. Os objetivos e práticas específicos da área de processo Solução Técnica no CMMI são:

- **Objetivo Específico 1:** Selecionar soluções para os componentes dos produtos - Deve-se selecionar soluções para os produtos ou componentes dos produtos dentre um conjunto de soluções alternativas.
    - **Prática Específica 1.1:** Desenvolver soluções detalhadas e critérios de seleção – Deve-se gerar uma lista de possíveis soluções para que em seguida essas soluções sejam detalhadas, os requisitos sejam mapeados nas mesmas e desenvolver critérios de seleção que ajudem a selecionar a solução mais adequada.
    - **Prática Específica 1.2:** Evoluir os cenários e conceitos operacionais – Deve-se evoluir os cenários, conceitos e os ambientes para descrever as condições, modos de operação e estados de operação específicos para cada componente do produto.
    - **Prática Específica 1.3:** Selecionar soluções para os componentes de produtos – Deve-se selecionar as soluções para os componentes de produto que melhor satisfazem os critérios estabelecidos.

[2] COTS: (Commercial Off-the-Shelf) Software que pode ser comprador de vendedores comerciais (CHRISSIS *et al.*, 2003).

- **Objetivo Específico 2:** Desenvolver o Projeto – Deve-se desenvolver os projetos dos produtos ou dos componentes dos produtos.

    - **Prática Específica 2.1:** Projetar o produto ou componente do produto – Deve-se desenvolver o projeto para o produto ou componente do produto.

    - **Prática Específica 2.2:** Estabelecer um pacote de dados técnico – Deve-se estabelecer e manter um pacote de dados técnico. O pacote de dados técnico fornece ao desenvolvedor uma descrição completa do produto ou componente de produto conforme ele for sendo desenvolvido e inclui todos os dados técnicos aplicáveis como desenhos, listas associadas, especificações, descrições do projeto, banco de dados de projeto, padrões, requisitos de performance, provisões de garantia da qualidade, e detalhes de empacotamento.

    - **Prática Específica 2.3:** Projetar as interfaces usando critérios: Deve-se projetar as interfaces completas dos componentes do produto em termos dos critérios mantidos e estabelecidos.

    - **Prática Específica 2.4:** Realizar a Análise Fazer/Comprar/Reutilizar – Deve-se avaliar se os componentes dos produtos devem ser desenvolvidos, comprados ou reutilizados baseado em critérios estabelecidos.

- **Objetivo Específico 3:** Implementar o Projeto do Produto – Deve-se implementar os componentes do produto e a documentação de suporte associada a partir dos seus projetos.

    - **Prática Específica 3.1:** Implementar o projeto: Deve-se implementar os projetos dos componentes dos produtos.

    - **Prática Específica 3.2:** Desenvolver a documentação de suporte do produto: Deve-se desenvolver e manter a documentação do usuário final.

2.4.2 A Solução Técnica na norma ISO/IEC 12207 e na norma ISO/IEC 15504

A Emenda 1 da ISO/IEC 12207 define as atividades através de um propósito e uma lista de resultados esperados para aquela atividade. Em relação à solução técnica, pode-se identificar três atividades relacionadas com o processo de solução técnica: projeto da arquitetura do sistema, projeto do software e construção do software. Estas atividades são descritas a seguir conforme são apresentadas na emenda, através de um propósito e das saídas.

- *Projeto da Arquitetura do Sistema*: Seu propósito é identificar que requisitos do sistema devem ser alocados a que elementos do sistema. Os resultados esperados são:
    - Um projeto da arquitetura do sistema que identifica os elementos do sistema e satisfaz os requisitos estabelecidos é definido.
    - Os requisitos funcionais e não funcionais do sistema são definidos.
    - Os requisitos são alocados aos elementos do sistema.
    - Interfaces internas e externas de cada elemento do sistema são definidas.
    - É executada uma verificação entre os requisitos do sistema e a arquitetura do sistema.
    - Os requisitos alocados aos elementos do sistema e suas interfaces são rastreáveis em relação a *baseline* de requisitos do cliente.
    - É mantida a consistência e a rastreabilidade entre os requisitos do sistema e o projeto da arquitetura do sistema.
    - Os requisitos do sistema, o projeto da arquitetura do sistema e seus relacionamentos são colocados em uma *baseline* e as partes afetadas por estes itens são comunicadas.

- *Projeto do Software*: Seu propósito é fornecer um projeto para o software que implemente e possa ser verificado em relação aos requisitos. Os resultados esperados são:

    - Um projeto da arquitetura do software que descreve os elementos do software que implementarão os requisitos é desenvolvido e colocado em uma *baseline*.

    - As interfaces internas e externas de cada elemento do software são definidas.

    - Um projeto detalhado que descreve as unidades do software e que pode ser construído e testado é desenvolvido.

    - A consistência e a rastreabilidade entre os requisitos do software e o projeto do software são estabelecidas.

- *Construção do Software*: Seu propósito é produzir unidades de software executáveis que reflitam apropriadamente o projeto do software. Os resultados esperados são:

    - Critérios de verificação são definidos para todas as unidades do software em relação aos requisitos.

    - As unidades de software definidas no projeto são produzidas.

    - A consistência e a rastreabilidade são estabelecidas entre os requisitos do software, o projeto e as unidades do software.

    - É realizada a verificação das unidades de software em relação aos requisitos e o projeto.

A norma ISO/IEC 15504, assim como a emenda 1 da ISO/IEC 12207, também define os processos através de um propósito e resultados esperados. A seguir são apresentados os processos associados com a área de processo Solução Técnica:

- *Processo de Projeto do Software*: O propósito é definir um projeto para o software que implemente os requisitos e que possa ser testado em relação aos mesmos. Suas práticas base são: (i) desenvolver o Projeto Arquitetural do Software, (ii) projetar Interfaces, (iii) verificar o Projeto do Software, (iv) desenvolver o Projeto Detalhado e (v) garantir a consistência.

- *Processo de Construção do Software*: O propósito é construir unidades de software executáveis e verificar que estas unidades de software refletem apropriadamente o projeto do software. Suas práticas base são: (i) desenvolver procedimentos de verificação das unidades, (ii) desenvolver as unidades do software, (iii) garantir a consistência e (iv) verificar as unidades de software.

2.4.3 O Processo Solução Técnica no MPS.BR

O processo Solução Técnica (STE) é um dos processos que uma organização precisa executar durante os seus projetos para atingir o nível de maturidade D (Largamente Definido) do modelo. O propósito da Solução Técnica, de acordo com o modelo, é projetar, desenvolver e implementar soluções para atender os requisitos.

Os resultados esperados do processo de solução técnica são:

- *STE 1*: Alternativas de solução são desenvolvidas para atenderem aos requisitos definidos e soluções são selecionadas para o produto ou componentes do produto de acordo com critérios identificados.

- *STE 2*: A documentação das soluções, da avaliação das alternativas e das razões que levaram a adotar as soluções são estabelecidas e mantidas.

- *STE 3*: O produto ou componente do produto é projetado e documentado.

- *STE 4*: As interfaces entre os componentes do produto são projetadas baseando-se em critérios pré-definidos.

- *STE 5*: Uma análise em relação aos componentes do produto deve ser conduzida para verificar se é necessária sua construção, compra ou reuso.

- *STE 6*: Critérios de avaliação para o projeto são definidos.

- *STE 7*: Os componentes do produto e a sua documentação associada são implementados de acordo com o projeto.

- *STE 8*: Verificações são conduzidas para os componentes do produto selecionados.

- *STE 9*: Testes unitários são conduzidos para os componentes do produto, quando apropriado.

- *STE 10*: Uma estratégia é desenvolvida para identificar a documentação a ser produzida durante o ciclo de vida do produto ou serviço de software.

- *STE 11*: A documentação é desenvolvida e disponibilizada de acordo com os padrões identificados.

- *STE 12*: A documentação é mantida de acordo com os critérios definidos.

2.4 Mapeamento entre os modelos MR-MPS, CMMI e as normas ISO/IEC 12207 e ISO/IEC 15504 em relação à Solução Técnica

O modelo CMMI, o MR-MPS e as normas ISO/IEC 12207 e ISO/IEC 15504 possuem formas diferentes para representar o que esperam que as organizações implementem em seus processos. O CMMI define boas práticas que as organizações devem implementar em seus projetos para que possam satisfazer uma determinada área de processo. De modo análogo, o modelo MR-MPS, assim como as normas ISO/IEC 12207 e ISO/IEC 15504, definem um conjunto de resultados esperados para um determinado processo. Assim, uma organização que deseja implementar um processo definido no

modelo MR-MPS deve produzir os resultados esperados definidos no modelo para o processo em questão.

Tendo como foco o processo de solução técnica, os modelos podem ser mapeados, apesar da forma distinta de estruturação dos mesmos. Este mapeamento, exibido na tabela 2.4, mostra:

(I). A aderência do MPS.BR à ISO/IEC 15504. Note-se que os dois resultados esperados pela ISO/IEC 12207 e pela ISO/IEC 15504 e que não estão presentes no processo Solução Técnica do MPS.BR, estão presentes no processo Gerência de Requisitos.

(II). A compatibilidade quase total entre o MPS.BR e o CMMI no que se refere a este processo, o que permite a implementação concomitante dos dois modelos nas empresas.

Tabela 2.4 – Mapeamento entre o MPS.BR, CMMI, ISO/IEC 12207 e a ISO/IEC 15504 em relação à Solução Técnica

MPS.BR	ISO/IEC 12207 – ISO/IEC 15504	CMMI
STE1- Alternativas de solução são desenvolvidas para atenderem aos requisitos definidos e soluções são selecionadas para o produto ou componentes do produto de acordo com critérios identificados.		PE1.1- Desenvolver soluções detalhadas e critérios de seleção.
		PE1.3- Selecionar soluções para os componentes de produto
STE 2- A documentação das soluções, da avaliação das alternativas e das razões que levaram a adotar as soluções são estabelecidas e mantidas.		PE1.3- Selecionar soluções para os componentes de produto
STE 3- O produto ou componente do produto é projetado e documentado.	Desenvolver o projeto arquitetural do software	PE2.1- Projetar o produto ou componente do produto
	Desenvolver o projeto detalhado	PE2.2- Estabelecer um pacote de dados técnico
STE 4- As interfaces entre os componentes do produto são projetadas baseando-se em critérios pré-definidos.	Projetar Interfaces	PE2.3- Projetar as interfaces usando critérios
STE 5- Uma análise em relação aos componentes do produto deve ser conduzida para verificar se é necessária sua construção, compra ou reuso.		PE2.4- Realizar a análise fazer, comprar ou reutilizar
STE 6 - Critérios de avaliação para o projeto são definidos		PE2.1- Projetar o produto ou componente do produto
STE 7- Os componentes do produto e a sua documentação associada são implementados de acordo com o projeto.	Desenvolver as unidades de software	PE3.1- Implementar o Projeto
STE 8- Verificações são conduzidas para os componentes do produto selecionados.	Verificar o projeto do software	PE3.1- Implementar o Projeto
	Desenvolver procedimentos de verificação das unidades	
	Verificar as unidades de software	
STE 9- Testes unitários são conduzidos para os componentes do produto, quando apropriado.		PE3.1- Implementar o Projeto
STE 10- Uma estratégia é desenvolvida para identificar a documentação a ser produzida durante o ciclo de vida do produto ou serviço de software.		PE3.2- Desenvolver a documentação de suporte do produto
STE 11- A documentação é desenvolvida e disponibilizada de acordo com os padrões identificados.		PE3.2- Desenvolver a documentação de suporte do produto
STE 12- A documentação é mantida de acordo com os critérios definidos.		PE3.2- Desenvolver a documentação de suporte do produto.
	Garantir a consistência e a rastreabilidade entre os requisitos e o projeto de software	
	Garantir a consistência e a rastreabilidade entre o projeto de software e a construção.	
		PE1.2- Evoluir os cenários e conceitos operacionais

2.5 Considerações Finais

Um processo de software define o modo pelo qual o desenvolvimento de software é organizado, gerenciado, medido, apoiado e melhorado (independente do tipo da tecnologia utilizada neste desenvolvimento). Este capítulo apresentou uma revisão da literatura sobre processos de software. Também foram discutidos os modelos de maturidade CMMI e MPS.BR e as normas ISO/IEC 12207 e ISO/IEC 15504.

O capítulo destacou a forma como o processo de Solução Técnica é abordado pelos modelos e normas apresentados, além de discutir como é a execução deste processo durante o processo de desenvolvimento de software. Além disso, foi apresentado um mapeamento entre os modelos CMMI e MR-MPS e as normas ISO/IEC 12207 e ISO/IEC 15504 em relação à Solução Técnica.

O próximo capítulo faz uma revisão da literatura sobre *Design Rationale* mostrando definições existentes, ferramentas construídas para o seu manuseio, formas de representação, captura e recuperação.

CAPÍTULO 3 – DESIGN RATIONALE

3.1 Introdução

As informações relativas a decisões que ocorrem durante o desenvolvimento de software muitas vezes são perdidas devido à falta de seu registro. A ausência desses registros pode levar a uma repetição de erros anteriormente cometidos. O registro das informações sobre decisões se mostra útil para a tomada de decisões de outros projetos, pois erros cometidos e alternativas já investigadas serviriam como aprendizado, uma vez que muitas soluções discutidas e adotadas em um projeto podem ser relevantes a outros (SOUZA *et al.*, 1998).

Um indicador de um bom processo de projeto é que o projeto foi escolhido após terem sido comparados e avaliados com diversas alternativas de solução. Decisões sobre a arquitetura, desenvolvimento customizado *versus* de prateleira e modularização do componente do produto são escolhas típicas de projeto (CHRISSIS *et al.*, 2003).

De acordo com FRANCISCO (2004), as informações sobre as opções disponíveis e as decisões que levaram a uma determinada escolha se registradas, como é feito com a decisão final, tornariam o software mais fácil de entender. Além disso, facilitaria a sua manipulação, seja para manutenção, reuso ou quando da entrada de novos integrantes na equipe responsável pelo software.

Design Rationale (DR) é a explicação do porquê de um artefato, ou alguma parte dele, ter sido projetado do modo como foi (LEE & LAI, 1991). SOUZA *et al.* (1998) definem DR como a documentação das decisões de projeto com suas respectivas justificativas, as opções consideradas, as avaliações e a argumentação que levaram a determinada decisão.

A presença de DR poder ser muito valiosa para o desenvolvimento de software por diversas razões. Uma é que a mutabilidade inerente ao software aumenta a probabilidade

dele ser modificado durante o seu ciclo de vida. Outra é que o ciclo de vida do software tende a ser longo, geralmente mais longo do que o esperado, e isto requer que o software seja alterado na medida em que o mundo ao seu redor sofre mudanças. Manutenção de software é uma parte bastante cara do processo de software e torna-se mais difícil porque os projetistas originais, em geral, não estão disponíveis. Esta é, portanto, uma área onde o DR poderia ser útil. O raciocínio poderia prover indicações de porquê do sistema ser como é. Ao fornecer as razões por trás das decisões de projeto, poderia ajudar a indicar onde as mudanças são necessárias durante a manutenção se os objetivos do projeto mudarem e ajudar o mantenedor a evitar a repetição de erros anteriores ao documentar explicitamente as alternativas que foram tentadas anteriormente mas que não funcionaram (BURGE, 2005a).

Outros autores como CONKLIN (1989) também ressaltam a importância de DR para a manutenção. Segundo CONKLIN (1989), DR é especialmente útil durante a manutenção tendo em vista que, muito freqüentemente, o tipo de entendimento necessário para a manutenção depende da compreensão de quais decisões de projeto foram consideradas, que suposições foram feitas, que soluções alternativas foram rejeitadas e que critérios e requisitos foram considerados no processo de deliberação.

A alta rotatividade das equipes na indústria do software e a longa duração do ciclo de vida dos projetos aumentam a probabilidade dos projetistas originais não estarem disponíveis para serem consultados quando os problemas aparecerem (BURGE & BROWN, 2001), acentuando a importância de se manter o registro do DR.

A limitação da memória humana é outro motivo para se fazer o armazenamento do DR. Esse armazenamento permite que, depois de certo tempo, informações sobre as decisões feitas, o porquê delas e quais alternativas foram consideradas, sejam relembradas, possibilitando que erros já cometidos sejam evitados e a limitação da memória humana não seja causa de problemas (FRANCISCO, 2004).

Também é comum que soluções discutidas e adotadas em um projeto possam ser relevantes para adoção em outro (FRANCISCO, 2004). Assim, se as razões de projeto

gravadas puderem ser facilmente acessadas, os desenvolvedores de novos projetos poderão ser beneficiados das discussões e das decisões tomadas em projetos anteriores (SOUZA *et al.*, 1998).

Neste capítulo são tratadas algumas questões relacionadas a DR. A seção 3.2 apresenta a definição de DR adotada no trabalho, além de apresentar outras definições existentes na literatura. Na seção 3.3 são apresentados os potenciais benefícios a serem obtidos com a utilização de DR, as dificuldades encontradas atualmente para a sua utilização e, também, possíveis formas de utilização do DR. Na seção 3.4 são definidas as formas de representação de DR que foram encontradas na literatura. Na seção 3.5 são definidas as formas de captura de DR e na seção 3.6 são apresentados meios de recuperação do DR. A seção 3.7 descreve sistemas de DR existentes. Finalmente, na seção 3.8 são apresentadas as considerações finais do capítulo.

3.2 Definição de DR

A documentação padrão de projeto consiste, geralmente, de uma descrição do projeto final: uma fotografia das decisões finais. *Design Rationale* oferece mais: não somente as decisões, mas também as razões por trás de cada decisão, incluindo sua justificativa, alternativas consideradas e a argumentação que levou a decisão (LEE, 1997).

Existem muitas definições para DR, sendo que a essência de todas, como será visto adiante, é a possibilidade de manter o raciocínio por trás das decisões de projeto. A seguir são listadas algumas das definições de DR encontradas na literatura:

- DR é o conjunto de explicações que responde a questões sobre um projeto. Essas explicações são geradas a partir do conhecimento dos artefatos e das atividades de projeto (GRUBER & RUSSEL, 1990).

- DR é a explicação do porquê de um artefato ou alguma parte dele ter sido projetado da maneira que foi (LEE & LAI, 1991).

- DR é uma representação para documentar explicitamente o raciocínio e a argumentação que dão sentido a um artefato específico (MCLEAN *et al.*, 1991).

- DR é a documentação das decisões de projeto com suas respectivas justificativas, as opções consideradas, as avaliações e a argumentação que levaram a uma determinada decisão (SOUZA *et al.*, 1998).

- DR consiste das decisões elaboradas durante o processo de projeto e das razões que levaram a essas decisões (BURGE & BROWN, 2000).

- DR são as informações obtidas durante a elaboração de cada artefato de projeto, indicando todos os passos tomados, as discussões e as alternativas investigadas, justificando o artefato resultante (FRANCISCO, 2004).

Para este trabalho utilizaremos a definição de DR de HU *et al.*, (2000). Segundo estes autores, DR é a justificativa da decisão tomada para a elaboração de um artefato ou parte dele, incluindo deliberações, razões, alterações e todas as decisões intermediárias tomadas no processo de construção do projeto desse artefato.

3.3 Benefícios e Dificuldades relacionados à utilização de DR

Existem muitas utilizações possíveis para o DR, sendo diversos os benefícios a serem obtidos. Entre as possíveis utilizações, pode-se citar (BURGE & BROWN, 2002, LEE, 1997):

- *Verificação do Projeto*: o que significa usar o DR para verificar se o projeto satisfaz os requisitos e a intenção do projetista.

- *Avaliação do Projeto*: o que significa usar o DR para avaliar projetos e escolhas de projeto buscando detectar inconsistências.

- *Manutenção do Projeto*: o que significa usar o DR para localizar fontes de problemas no projeto para indicar onde são necessárias modificações no projeto e para assegurar que as opções rejeitadas não serão reimplementadas inadvertidamente.

- *Gerência de Dependências*: o que significa usar o DR para tornar explícitas as relações de dependências entre as partes do projeto, decisões, argumentos e alternativas de modo que funcionem corretamente juntas. Esta necessidade existe pelo fato de o projeto poder ser visto como um processo de gerência de dependências para construir um produto que honre todas as dependências entre os requisitos e os componentes que os implementam.

- *Gerência de Projeto/Colaboração*: o que significa usar o DR para fornecer uma base comum quando múltiplas partes estão envolvidas. Raciocínios representados explicitamente podem fornecer um vocabulário comum e manter a memória dos projetos, facilitando as negociações e o consenso.

- *Reuso do Projeto*: o que significa usar o DR para determinar partes do projeto que podem ser reutilizadas e, eventualmente, sugerir onde e como serão modificadas para satisfazer um novo conjunto de requisitos.

- *Ensino*: o que significa usar o DR para treinar novos membros da equipe sobre o projeto.

- *Comunicação*: o que significa usar o DR para comunicar as razões para as decisões de projeto a outros membros da equipe.

- *Apoio ao Projeto*: usar o DR para apoiar as discussões, checar o impacto das modificações no projeto, realizar checagem de consistência e ajudar na mitigação de conflitos ao olhar para as violações de restrições entre os múltiplos projetistas.

- *Documentação do Projeto*: usar o DR para documentar o projeto ao oferecer uma fotografia da história do projeto e as razões para as escolhas assim como uma visão do produto final.

Os serviços que um sistema de DR fornece irão determinar quase todos os outros aspectos do seu projeto, como por exemplo o que representar e como representar os raciocínios. A Figura 3.1 apresenta os serviços mais comuns, classificados em quatro grupos de acordo com o perfil dos usuários que serão beneficiados pelo DR: melhor projeto (projetistas), melhor manutenção (mantenedores do sistema), aprendizagem (novos *trainees*, estudantes e programas de ensino) e documentação (futuros projetistas e mantenedores). As setas entre os serviços indicam que o serviço no fim da seta apóia o serviço no início da seta (ponta da linha onde está a seta) (LEE, 1997).

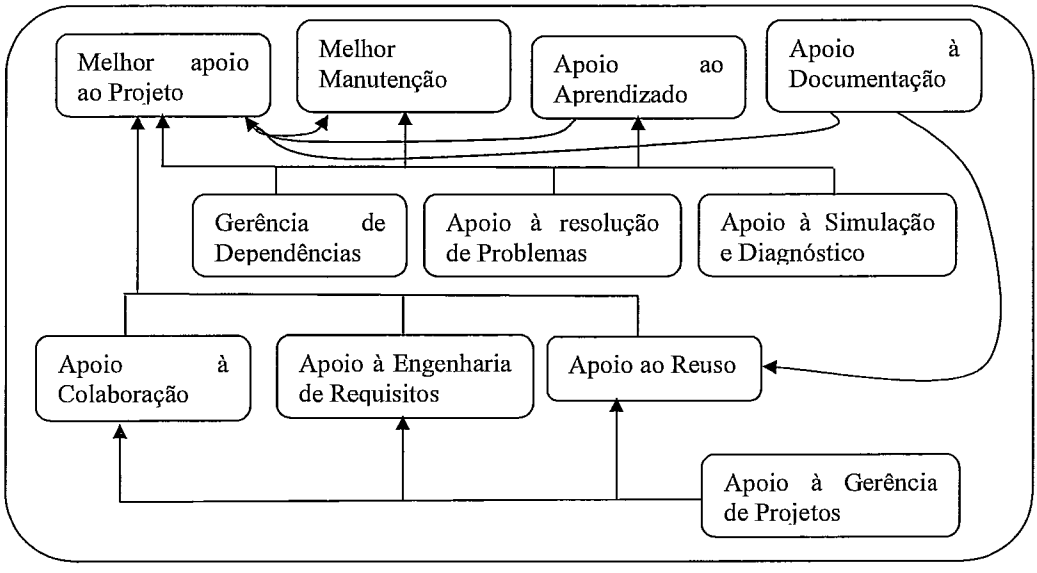


FIGURA 3.1 – Serviços fornecidos por muitos sistemas de DR. As setas indicam relações de apoio. (LEE, 1997)

Apesar de todos estes benefícios e potenciais formas de utilização, DR não é muito utilizado na prática. Existem algumas razões que nos ajudam a entender o porquê desta não utilização do DR nas organizações.

Em relação à captura, o registro das decisões tomadas assim como daquelas rejeitadas, pode consumir bastante tempo e ser custoso. Quanto mais intrusivo for o processo de captura, maior será a resistência do projetista. Por ser um processo que consome tempo e visto como documentação, a captura de DR pode ser considerada dispensável caso o prazo final para a realização do projeto esteja se aproximando (CONKLIN & BURGESS-YAKEMOVIC, 1995).

Documentar as decisões pode impedir o processo de projeto, se o registro das decisões é visto como um processo separado do processo de construção do artefato (FISCHER *et al.*, 1995). Os projetistas são relutantes em usar seu tempo para as decisões que não foram tomadas ou que foram tomadas e depois rejeitadas (CONKLIN & BURGESS-YAKEMOVIC, 1995). Também existem problemas em relação à responsabilidade pois, no caso da ocorrência de um erro, o DR que documentou as razões para o problema poderia ser utilizado contra o projetista (CONKLIN & BURGESS-YAKEMOVIC, 1995).

Existe ainda outra questão referente ao risco de que o *overhead* da captura de DR possa ter impacto negativo no cronograma do projeto, fazendo o projeto ultrapassar os prazos previstos (GRUDIN, 1995).

Outro aspecto que diminui o interesse dos projetistas em registrar suas decisões de projeto é o fato de que aqueles que se esforçam registrando as decisões, provavelmente não serão os mesmos que se beneficiarão pelo uso do DR registrado. Esta possibilidade não incentiva os profissionais a utilizarem e se esforçarem para registrar o DR, especialmente após as questões já citadas de responsabilidade pelos problemas e impacto no cronograma (GRUDIN, 1995).

HU *et al.* (2000) ressaltam, ainda, o fato de que os sistemas de DR disponíveis não suprem as necessidades específicas de cada empresa. Por mais que o sistema envolva as mais diversas necessidades, sempre existem algumas necessidades particulares a cada empresa que não são atendidas.

3.4 Representação de DR

É impossível representar um DR inteiro explicitamente. Raciocínios estão embutidos não apenas em documentos formais como as especificações de projeto, resumo de reuniões, documentos de interface, mas também em mídias informais como conversas de telefone, desenhos no quadro negro e discussões na hora do almoço. O que for escolhido para ser representado, no entanto, deve ser acessível e estar em alguma estrutura (LEE, 1997). A escolha do esquema de representação é uma questão crítica, porque determinará como organizar a enorme quantidade e diversidade de materiais e construir uma estrutura útil que determinará como capturar e recuperar o DR (BUCKINGHAM-SHUM & HAMMOND, 1994).

3.4.1 Informação a ser Representada

Quase todo tipo de informação relacionada com o projeto é potencialmente útil e pode ser classificada de alguma forma em relação ao raciocínio. Uma estrutura genérica para DR é descrita em LEE (1997) e consiste de três camadas: (i) camada de decisão, (ii) camada de projeto do artefato, (iii) camada da intenção do projeto.

A *camada da intenção* é a camada mais alta de abstração do DR. Esta camada contém as informações que orientam as decisões tomadas durante o processo de projeto e contém as intenções, estratégias, objetivos e requisitos. É importante capturar esta informação de modo que o projeto, e posteriores alterações, possam ser verificados.

A *camada de decisão* fornece a informação detalhada que descreve as decisões feitas durante o processo de projeto. Esta é a informação consultada para determinar porque foi feita uma determinada escolha de projeto. Esta camada possui cinco sub-camadas: (i) *questão*, com as questões individuais e suas relações (gera, depende de, substitui); (ii) *argumento*, com os argumentos por trás de uma decisão e suas relações (apóia, refuta, qualifica); (iii) *alternativa*, explicitando as alternativas individuais e suas relações (componente de, incompatível, especializa); (iv) *avaliações*, com as avaliações usadas para

priorizar as alternativas e, (v) *critério*, com os critérios para agrupar as avaliações e argumentos e suas relações (mutuamente exclusivos, *tradeoffs*, especializa).

A *camada de projeto* do artefato contém informações sobre o que está sendo projetado, por exemplo, os componentes e como estes componentes se relacionam uns com os outros.

3.4.2 Esquemas de Representação

A representação do DR pode implementar uma abordagem formal, uma abordagem informal ou uma abordagem semiformal. Uma abordagem formal permite que o computador use o dado mas nem sempre produz a informação num formato que um ser humano possa entender. Além disso, requer que o dado seja fornecido ao computador em um formato mais rígido. Uma abordagem informal fornece o dado em um formato que é facilmente gerado e entendido pelas pessoas, mas que não pode ser utilizado pelo computador. As abordagens semiformais buscam um meio termo, a fim de obter um pouco das vantagens da abordagem formal e um pouco das vantagens da abordagem informal (BURGE, 2005a).

Quanto mais formalmente os DR forem representados, maior será a quantidade de serviços que o sistema poderá prover ao projetista. No entanto, formalizar o conhecimento é uma tarefa custosa. Uma forma de reduzir este custo é formalizá-lo de maneira incremental – essencialmente transformando uma representação semiformal em uma formal. Assim, os DRs podem ser capturados com menos *overhead*, mas uma vez formalizados podem ser usados para apoiar uma quantidade maior de serviços computacionais (HU *et al.*, 2000).

O DR pode ser representado através de três perspectivas: a argumentação, a comunicação e a documentação (SHIPMAN e MCALL, 1997). Argumentação e documentação focam nas decisões de projeto e nas razões por trás das mesmas. A diferença é que o objetivo da documentação é fornecer conhecimento sobre o projeto a pessoas

externas ao projeto, enquanto que a argumentação tem o objetivo adicional de estruturar como o projetista abordou o problema. A perspectiva da comunicação é uma tentativa de capturar naturalmente a comunicação do projeto enquanto a mesma ocorre, como e-mails, minuta de reunião, etc (BURGE e BROWN, 2001). SHIPMAN & MCALL (1997) argumentam que a comunicação é o método mais efetivo para a captura de todo o DR envolvido em um projeto, mas é difícil de ser indexada para permitir a recuperação do DR relevante. A argumentação por outro lado, possui maiores problemas em relação à captura efetiva do DR, mas é bastante útil para a recuperação do DR relevante. A tabela 3.1 resume as diferenças entre estas perspectivas (FRANCISCO, 2004).

TABELA 3.1 – Diferenças entre as perspectivas de representação do DR (FRANSCISCO, 2004)

Perspectivas / **Características das Informações**	**Argumentação**	**Comunicação**	**Documentação**
Conteúdo	Semi-estruturado	Não Estruturado	Estruturado
Nível de detalhamento da informação armazenada em relação à produzida durante as discussões	Médio – analítica e sintetizada	Alto - Analítica	Baixo – Sintetizada
Momento da Captura	Durante a discussão	Durante a discussão	Após a discussão
Esforço/atividade requerido(a) no momento da discussão	Classificar os nós e links	Não possui	Não possui
Esquemas de representação	Nós e *links*	Diversas mídias	Linear

Documentação - A documentação objetiva registrar a história das atividades de projeto: que decisões os projetistas tomaram, quando estas decisões foram tomadas, que pessoas foram responsáveis pelas mesmas e o porquê (HU *et al.*, 2000). Existem dois itens que motivam a criação de tal documentação. Um é o objetivo de permitir que as pessoas de fora do grupo do projeto possam entender, supervisionar, e regular, se necessário, o que está sendo feito pelo grupo do projeto. A outra motivação é identificar e assegurar as propriedades intelectuais geradas no projeto – com os propósitos de obter e defender as patentes (SHIPMAN & MCALL, 1997). É uma característica da documentação registrar menos informação do que a argumentação e a comunicação (SHIPMAN & MCALL, 1997).

Comunicação - A argumentação registra o DR naturalmente enquanto o mesmo é gerado durante o projeto. É a perspectiva que menos interfere nas atividades do projetista e que possui a maior facilidade de obtenção do DR, pois a informação é capturada na forma em que é produzida (e-mails, gravação de voz, filmagem de reuniões etc). No entanto, pode ser muito custoso recuperar o DR relevante quando o projetista necessitar, pois a informação não está representada em um formato que facilite a busca. De modo oposto, a documentação e a argumentação buscam impor algum tipo de ordem ao DR capturado (SHIPMAN & MCALL, 1997).

Argumentação - Esta perspectiva é característica de projetistas e metodologias que buscam melhorar a qualidade do projeto ao melhorar o raciocínio dos projetistas - em particular, ao melhorar os argumentos que eles usam. Para a perspectiva da argumentação, o propósito de registrar o DR é chamar a atenção para as deficiências no raciocínio e portanto, levar a correção dessas deficiências. Esta abordagem possui dificuldade para capturar a informação, mas possui uma rica coleção de técnicas e tecnologias para a recuperação do DR relevante (SHIPMAN e MCALL, 1997). No contexto do trabalho descrito nesta dissertação, foi utilizada a argumentação para representar o DR.

Entre as notações para representação da argumentação pode-se destacar IBIS (*Issue Based Information System*) (CONKLIN & BEGEMAN, 1988), PHI (*Procedural Hierarchy of Issues*) (MCALL, 1991), QOC (*Question Option Criteria*) (MCLEAN *et al.*, 1995) e DRL (*Decision Rationale Language*) (LEE e LAI, 1991).

IBIS é um modelo com base em questões no qual a representação da argumentação é feita por meio de três abstrações: questões (problemas), posições (respostas) e argumentos (prós e contras). O modelo PHI estende IBIS, representando a informação por meio de Questão, Resposta e Argumento, além de Sub-Questão, Sub-Resposta e Sub-Argumento. Os tipos de nós no modelo QOC são Questão, Opção e Critério. O nó Questão é equivalente ao nó Questão dos modelos IBIS e PHI e o nó Opção é equivalente ao nó Posição e Resposta dos modelos IBIS e PHI respectivamente. O nó Critério é a vantagem do modelo QOC em relação aos modelos IBIS e PHI, pois possibilita a captura explícita dos critérios, os quais são importantes para uma melhor e mais precisa tomada de decisão.

DRL também é uma extensão do modelo IBIS que envolve um número maior de abstrações e relacionamentos. O modelo DRL é composto pelos seguintes nós: Problema de Decisão, Objetivo, Alternativa, Alegação, Questão, Procedimento e Grupo (FRANCISCO, 2004).

3.5 Captura de DR

A principal preocupação das pesquisas relacionadas à DR é como capturar o conhecimento do processo com um overhead mínimo, com a menor interferência possível na progressão natural das atividades de projeto (GARCIA & SOUZA, 1997, RAMESH & SENGUPTA, 1995).

A determinação de quais informações capturar durante o projeto e de como capturá-las depende de quais informações deseja-se obter na recuperação. Por exemplo, algumas capturas são feitas com a finalidade de responder às futuras questões dos desenvolvedores (HU *et al.*, 2000).

Os problemas mais comuns durante a captura são (HU *et al.*, 2000): (i) a pouca informação ou a informação incorreta impossibilita a criação de uma representação de DR; (ii) se as atividades dos desenvolvedores forem desviadas por causa da captura da informação, eles certamente vão desistir de fornecê-la.

De acordo com HU *et al.* (2000) os métodos de captura podem ser classificados em métodos de captura automática e métodos com base na intervenção do usuário.

A captura do DR com intervenção do usuário tem sido, geralmente, utilizada pelo método da documentação que objetiva registrar a história das atividades de projeto: que decisões os projetistas tomaram, quando estas decisões foram tomadas, que pessoas foram responsáveis pelas mesmas e o porquê das decisões.

A captura automática do DR assume a existência de um método para capturar a comunicação entre os projetistas e a equipe de projetistas ou entre o projetista e o sistema de apoio ao projeto. Os registros da comunicação podem ser utilizados para extrair o DR

durante o processo de projeto (SHIPMAN & MCALL, 1997). Usualmente, a comunicação emprega ferramentas de apoio ao trabalho colaborativo (POLTROCK & GRUDIN, 1999) ou tecnologias de reunião. Isto inclui, por exemplo, telefone, gravadores de voz, câmeras de vídeo, aplicações compartilhadas e e-mail para capturar tanto discussões orais assim como textos e desenhos trocados entre os projetistas. Quando as comunicações são arquivadas digitalmente as atividades de projeto podem ser processadas e o DR determinado. Uma desvantagem é o que o material registrado durante a comunicação e a colaboração normalmente não tem um formato definido e está em desordem (SHIPMAN e MCALL, 1997). A comunicação "crua" é pouco estruturada e, como conseqüência do uso desta abordagem para capturar a informação do projeto, o processo de recuperação do conhecimento poderá ser ineficiente (HU *et al.*, 2000).

De acordo com a participação do usuário e do sistema, as possíveis abordagens de captura são:

- *Reconstrução* (LEE, 1997) – Nesta perspectiva os desenvolvedores produzem o DR sem o uso do sistema, utilizando apenas seus conhecimentos. A captura de informações pode ser feita por meio de entrevistas com os envolvidos no projeto ou por meio de filmadoras, gravadores de fitas cassetes etc. A vantagem desta abordagem é o fato de ser "não intrusiva" e a desvantagem é que a mesma pode não capturar o DR de forma exata e completa.

- *Metodologia* (LEE, 1997) – O DR emerge durante o processo de projeto, seguindo uma metodologia pré-definida. Os passos do método são essencialmente diferentes tipos de refinamento e reformulação que o sistema captura e usa para geração das explicações. O desafio dessa forma de captura é fornecer um método que realmente ajude o usuário sem impor uma sobrecarga excessiva de restrições.

- *Aprendizado* (LEE, 1997) – Nesta abordagem, o sistema observa as ações tomadas pelo projetista e faz perguntas quando não entende uma determinada ação. Nestes sistemas o DR é, até certo ponto, pré-gerado – se as ações do projetista forem iguais às ações embutidas no sistema, então o raciocínio gerado pelo sistema é salvo.

- *Geração Automática* (LEE, 1997) – Na geração automática os DRs são gerados automaticamente a partir do histórico de execução. O custo não é excessivamente alto, além de manter a consistência e a atualização da justificativa. Entretanto, há um alto custo inicial devido à compilação do conhecimento necessário para a construção da justificativa.

- *Histórico* (CHEN, 1990) – Nesta abordagem, o projetista ou computador registra o histórico das ações tomadas durante o processo de projeto. Este método é similar à abordagem de Aprendizado, porém o sistema não faz sugestões. Também é similar à abordagem de geração automática, mas o DR é registrado especificamente durante o processo de projeto e não gerado depois.

As Tabelas 3.2 e 3.3 apresentam várias maneiras nas quais as abordagens de captura podem ser agrupadas (BURGE, 2005b). A Tabela 3.2 ilustra o momento em que o DR é capturado.

TABELA 3.2 - Momento da Captura do DR (BURGE, 2005b)

Método de Captura	Após o Projeto	Durante o Projeto
Reconstrução	X	X
Metodologia de Produto		X
Aprendizado		X
Geração Automática	X	
Histórico		X

Outra distinção que pode ser feita é entre o DR e o histórico do projeto. Apesar do fato de que os termos são geralmente usados com o mesmo significado, um histórico de projeto é diferente do DR, porque o DR captura as escolhas feitas, as escolhas rejeitadas e as razões por trás de ambas. O histórico de projeto captura somente as escolhas feitas e não necessita capturar as razões. A Tabela 3.3 mostra que abordagens capturam o histórico e que abordagens capturam o DR. Para algumas abordagens, não fica claro se é capturado o histórico ou o DR, neste caso foi colocado o sinal de interrogação na célula.

TABELA 3.3 - DR vs. Histórico do Projeto (BURGE, 2005b)

Método de Captura	Histórico do Projeto	DR
Reconstrução	X	?
Metodologia de Produto	X	X

Aprendizado	X	X
Geração Automática	X	
Histórico	X	?

Outra forma de agrupar os métodos de captura é pela quantidade de interação com o projetista. Métodos de captura que interagem com o projetista são geralmente mais intrusivos do que os métodos que não interagem. A Tabela 3.4 mostra o volume de interação do projetista em relação aos métodos de captura.

TABELA 3.4 - Interação com o Projetista (BURGE, 2005b)

Método de Captura	Baixa Interação	Alta Interação
Reconstrução	X	
Metodologia de Produto		X
Aprendizado		X
Geração Automática	X	
Histórico	X	

3.6 Recuperação do DR

As estratégias de recuperação são determinadas pelo esquema de representação e pelos requisitos do processo de projeto (HU *et al.*, 2000). Os sistemas que permitem a recuperação do DR são classificados em dois tipos: (i) iniciativa do usuário e (ii) iniciativa do sistema (LEE, 1997, PENA-MORA *et al.*, 1995).

Em um sistema com iniciativa do usuário, o usuário decide as partes do DR que irá examinar e quando ou como olhará para estas partes. Estes sistemas devem ajudar o usuário a tornar-se ciente do que existe e a achar facilmente as partes desejadas. A maior parte dos sistemas permite que o usuário faça consultas ou adota um esquema de navegação para permitir a recuperação do DR (LEE, 1997).

Em um sistema com iniciativa do sistema, por outro lado, o sistema decide quando e como apresentar as partes do raciocínio. Tais sistemas devem possuir conhecimento suficiente para tomar decisões inteligentes e devem apresentar o raciocínio de forma não intrusiva para o usuário (LEE, 1997).

HU *et al.* (2000) definem três tipos de métodos para recuperação do DR capturado: navegação, consulta, acionamento automático e estratégia híbrida.

A *navegação* permite que os projetistas investiguem o DR ao caminhar de um nó para outro através dos *links* existentes. Para um artefato complexo, uma grande quantidade de informação é armazenada durante o processo do seu projeto, o que dificulta a busca por respostas específicas por meio da recuperação pela estratégia de navegação (HU *et al.*, 2000).

De acordo com os projetistas, estratégias de recuperação com base em *consulta* são mais eficientes do que percorrer os nós das estruturas do DR. As consultas podem ser na forma de perguntas do tipo "o que aconteceria se", que podem ser respondidas ao explorar diferentes opções; ou perguntas do tipo "por quê?", que poderiam ser respondidas ao retornar na rede de nós e links para encontrar a argumentação ou raciocínio por trás da decisão (HU *et al.*, 2000).

Muitos sistemas de DR possibilitam a captura do DR através do acionamento *automático*, tais sistemas detectam ou monitoram certas condições de acordo com o contexto do projeto. Este tipo de abordagem possui similaridades com o paradigma de programação baseada em eventos na engenharia de sistemas de software interativos e de tempo real (HU *et al.*, 2000).

Estratégias de recuperação *híbridas* fazem uso de uma combinação das estratégias de navegação, recuperação e acionamento automático, proporcionando uma recuperação conveniente e eficiente (HU *et al.*, 2000).

Os sistemas também diferem na forma como a informação é organizada. Uma organização comum é ter o DR organizado por componentes do artefato projetado. Esta é uma boa representação do sistema a ser usada por novos projetistas que necessitam entender como o projeto está organizado (GRUBER, 1990). Outros sistemas estão organizados em torno de um único processo que está modelado (BROWN & BANSAL, 1991). Isto é útil para usuários que estão abordando o problema em um alto nível e estão interessados no processo e não apenas no produto. Também é útil se o sistema de DR está

projetado para trabalhar com múltiplos domínios e tipos de projeto onde o processo é similar. Alguns sistemas modelam diferentes aspectos do processo de projeto e armazenam o raciocínio deste modo (KLEIN, 1992). Isto facilita a visão ou documentação do projeto sob diferentes perspectivas.

3.7 Sistemas de DR

Existem inúmeros sistemas de DR, mas uma queixa freqüente em relação a eles é que adicionam mais trabalho ao processo de projeto; utilizando-os, o projetista passa a possuir a tarefa adicional de criar o DR.

O sistema gIBIS (CONKLIN & BURGESS-YAKEMOVIC, 1995) utiliza a notação de argumentação IBIS para representar a atividade de argumentação dos desenvolvedores. O sistema gIBIS almeja facilitar, capturar, estruturar e gravar a discussão entre esses desenvolvedores, construindo uma rede de forma a representar tal discussão (FRANCISCO, 2004). A notação IBIS também é utilizada por itIBIS (CONKLIN & BURGESS-YAKEMOVIC, 1995).

SIBYL é um sistema de apoio à decisão em grupo, também não dedicado a um domínio específico, que conta com representação e gerenciamento de aspectos qualitativos do processo de decisão, tais como os objetivos, as alternativas e a avaliação dessas alternativas de acordo com os objetivos. O SYBIL é uma extensão do sistema de hipertexto simples gIBIS. Para a criação e gerência da rede de argumentação, o sistema SYBIL adota o modelo DRL (LEE, 1990).

JANUS é um sistema para o ambiente de projeto de cozinhas planejadas que utiliza a notação PHI (MCCALL, 1991). A notação PHI também é utilizada por PHIDIAS, um sistema hipermídia usado para projetos gráficos 2D e 3D. Uma representação simplificada da notação PHI é utilizada por DocRationale (FRANCISCO, 2004).

InfoRat utiliza um subconjunto significativo da notação DRL. Este sistema pode ser utilizado de forma integrada à plataforma *eclipse*[3] e fornece uma ontologia de critérios a serem utilizados durante a avaliação e seleção das alternativas. Através de InfoRat é possível realizar inferências sobre o DR para verificar a sua estrutura sintática e semântica (BURGE, 2005a). SYBIL também verifica o DR ao verificar se o raciocínio que levou a uma determinada decisão está completo Outros sistemas, como JANUS, criticam o projeto e fornecem aos projetistas um DR para apoiar as críticas.

CoMo-Kit (DELLEN *et al.*, 1996) utiliza um modelo de processo de software para obter as decisões de projeto e as dependências de causa entre as mesmas. Estas dependências de causa são, então, consideradas como sendo o DR. O sistema observa tarefas (objetivos a serem alcançados durante o processo), produtos (os produtos produzidos, como especificações), métodos (o modo como a tarefa é realizada) e agentes (a pessoa ou computador que executa a tarefa). Quando o modelo de processo é gerado, o fluxo da informação entre as tarefas é capturado e usado para deduzir a dependência entre as mesmas. O raciocínio consiste da justificativa pela escolha de um determinado método. CoMo-Kit (DELLEN et al., 1996) não é um sistema de argumentação – não registra as alternativas tentadas e rejeitadas e as razões por trás das mesmas. Por outro lado, usa as dependências para encontrar a justificativa para cada decisão.

ABCDE-DR (SOUZA *et al.*, 1998) é um editor de diagramas de objetos que implementa um modelo cooperativo para o registro do DR. O sistema apóia: (i) a representação semi-formal do DR, (ii) o registro do DR de uma maneira semi-automática através da reflexão na teoria da ação, e (iii) a integração entre o artefato e o DR. O modelo de DR foi desenvolvido baseado no modelo de cooperação de SOUZA *et al.* (1997), que utiliza as anotações como forma de cooperação entre os usuários. O modelo de DR também usa as anotações como forma de representar a informação de DR. Este modelo é chamado semi-automático porque coleta a informação do DR por fora da interação normal dos

[3] *Eclipse*: é uma comunidade de código aberto cujos projetos focam no fornecimento de plataformas de desenvolvimento e *framework* de aplicações para o desenvolvimento de software (ECLIPSE, 2006).

autores e revisores durante o processo de melhoria do projeto deles, ou seja, o DR é obtido quase que automaticamente a partir da interação entre os projetistas.

WinWin (BOEHM e BOSE, 1994) é uma abordagem que objetiva a coordenação das atividades de tomada de decisão feitas pelos vários *stakeholders* envolvidos no processo de desenvolvimento de software. BOSE (1995) definiu uma ontologia para o raciocínio de decisão com a finalidade de manter a estrutura de decisão. O objetivo era modelar o raciocínio da decisão para apoiar a manutenção ao permitir que o sistema determine o impacto de uma mudança e propague os efeitos da modificação.

3.8 Considerações Finais

Um indicador de um bom processo de projeto é que o projeto foi escolhido após ser comparado e avaliado com relação a diferentes alternativas de soluções. A documentação padrão do projeto consiste de uma descrição do projeto final: efetivamente uma fotografia das decisões finais. *Design Rationale* oferece mais: não somente as decisões, mas também as razões por trás de cada decisão, incluindo sua justificativa, alternativas consideradas e a argumentação que levou à decisão.

Com o intuito de apoiar o registro, representação e recuperação do *Design Rationale*, foram criados diversos sistemas de DR cuja descrição pode ser encontrada na literatura. Neste capítulo foi feita uma revisão da literatura sobre *Design Rationale*. Foram apresentadas definições de DR, formas de representação, captura e recuperação do DR, assim como alguns sistemas de DR existentes na literatura. No próximo capítulo serão apresentados conceitos relacionados com Ambientes de Desenvolvimento de Software e com a Estação Taba.

CAPÍTULO 4 – AMBIENTES DE DESENVOLVIMENTO DE SOFTWARE E A ESTAÇÃO TABA

4.1 Introdução

Ambiente de Desenvolvimento de Software (ADS) pode ser entendido como sendo um sistema computacional que visa dar suporte para o desenvolvimento e manutenção de software e para o gerenciamento destas atividades (MOURA & ROCHA, 1992).

O princípio de ADS apareceu na década de 70 e desde então evoluiu rapidamente para fornecer apoio mais amplo e efetivo aos desenvolvedores de software, de forma que metas como aumento da produtividade, melhoria da qualidade, diminuição de custos e do tempo para introdução no mercado de novos produtos possam ser alcançadas. No início quando se pesquisava sobre ADS o intuito era desenvolver ferramentas de automação do processo de desenvolvimento de software. A intenção da pesquisa atual em ADS é prover ao desenvolvedor de software ferramentas integradas que ajudem na execução das atividades do processo de desenvolvimento (VILLELA, 2004).

Na COPPE/UFRJ existe um grupo de trabalho em ambientes de desenvolvimento de software que iniciou na década de 90 a definição e a construção da Estação TABA. A Estação TABA é um meta-ambiente de desenvolvimento de software capaz de gerar outros ambientes de desenvolvimento de software através de configuração e instanciação. Ao longo desses anos de trabalho o conceito de ADS na Estação Taba evoluiu para a definição de ADS com apoio à utilização de informações sobre o conhecimento de domínio da aplicação durante o desenvolvimento (ADSOD) (OLIVEIRA, 1999) e para os que armazenam e utilizam o conhecimento organizacional (ADSOrg), que descrevemos a seguir.

4.2 Ambientes de Desenvolvimento de Software Orientados a Organização

Em 2004, uma evolução da Estação TABA (VILLELA, 2004) deu origem aos Ambientes de Desenvolvimento de Software Orientados a Organização (ADSOrg), o que permitiu a configuração de ambientes para uma organização específica, denominados Ambientes Configurados. Estes ambientes possibilitam a instanciação do ambiente para projetos específicos e a aquisição e disseminação do conhecimento organizacional durante o desenvolvimento de software através dos ambientes instanciados para cada projeto.

A motivação para a construção de ADSOrg surgiu de duas constatações (VILELLA *et al.*, 2000):

- Duas ou mais organizações podem desenvolver software para um mesmo domínio com processos, interesses e características muito distintas; e,

- O conhecimento de domínio não é o único conhecimento importante para apoiar desenvolvedores de software em suas atividades. Outros conhecimentos também são extremamente importantes e úteis para os desenvolvedores, como: diretrizes e melhores práticas organizacionais, lições aprendidas com o uso de processos, métodos e técnicas de software etc.

Os ADSOrg possuem como objetivos (VILLELA *et al.*, 2000):

- Prover aos desenvolvedores de software o conhecimento acumulado pela organização e relevante no contexto do desenvolvimento de software; e

- Apoiar o aprendizado organizacional neste contexto.

A estratégia para instanciação e utilização dos ADSOrg é iniciada com a configuração do meta-ambiente para uma organização. Através dessa configuração são levantadas as principais características do desenvolvimento de software na organização e é disponibilizado um ambiente capaz de dar apoio às atividades de desenvolvimento e

manutenção de software. Este ambiente contém o processo padrão e processos especializados para o desenvolvimento e/ou manutenção de software, além de conhecimento para dar apoio às atividades envolvidas nestes processos. De posse deste ambiente, a organização pode instanciar novos ambientes, denominados ADSOrg, que apóiam a execução das atividades dos processos em projetos específicos, além de apoiar o aprendizado organizacional a partir dos projetos. Estes ambientes são gerados a partir da adaptação dos processos especializados para atender às características dos projetos específicos.

A Figura 4.1 exibe o esquema utilizado para a construção de ADS na Estação TABA.

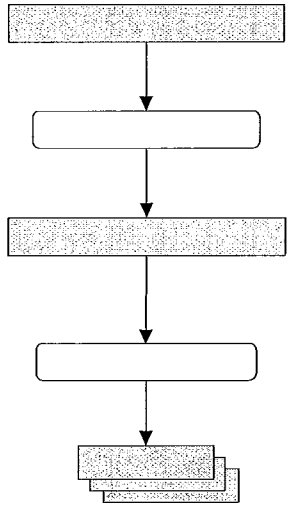


Figura 4.1: Esquema para Construção de ADS na Estação TABA

Estes ambientes podem ser definidos da seguinte maneira:

1. **Meta-Ambiente**: É utilizado em uma entidade externa à organização desenvolvedora (atualmente apenas a COPPE), sendo utilizado por profissionais especializados em engenharia de software que fazem a definição e manutenção dos ativos de processos e demais recursos disponíveis neste ambiente. Sua principal função é apoiar a configuração de ambientes para organizações específicas, gerando os Ambientes Configurados.

2. **Ambiente Configurado**: É utilizado por profissionais de nível gerencial e pelo grupo de processos da organização. Uma de suas principais funções é apoiar o processo de Instanciação, ou seja, a geração dos ambientes que serão utilizados pelos profissionais dos projetos, os Ambientes Instanciados ou ADSOrg.
3. **ADSOrg**: Ambiente de desenvolvimento de software instanciado a partir do Ambiente Configurado e específico a um projeto. É utilizado pelo gerente/líder do projeto e demais membros da equipe do projeto, dando apoio às várias atividades dos processos do ciclo de vida de um software através de ferramentas específicas, conhecimento organizacional, descrição de procedimentos e modelos de documentos, dentre outros.

4.3 Definição de Processos na Estação Taba

A definição dos Ambientes Configurados e Instanciados está intrinsecamente associada à definição de processos pela Estação TABA como pode ser visto na Figura 4.2.

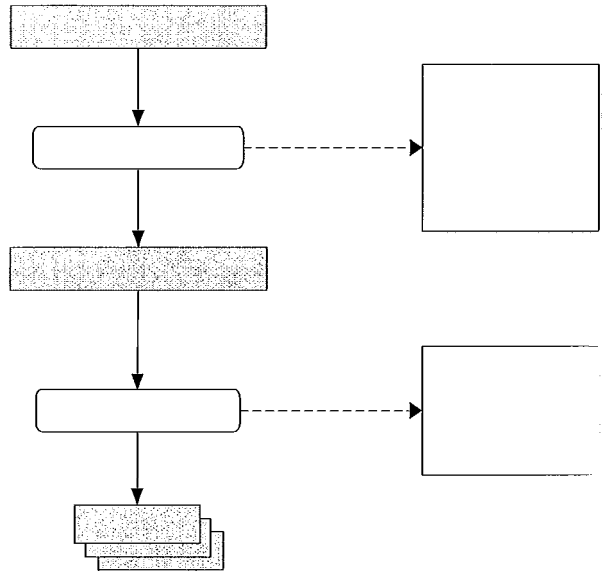


Figura 4.2 – Definição de Ambientes e Processos

A Figura 4.3 representa, de forma esquemática, a abordagem adotada para a definição dos processos. A definição dos processos padrão e especializados é realizada pela ferramenta Config (VILLELA, 2004, VILLELA *et al.*, 2004) e dá origem aos Ambientes Configurados. A definição dos processos instanciados é realizada pela ferramenta AdaptPro (BERGER, 2003) (VILLELA *et al.*, 2004) e dá origem aos Ambientes Instanciados.

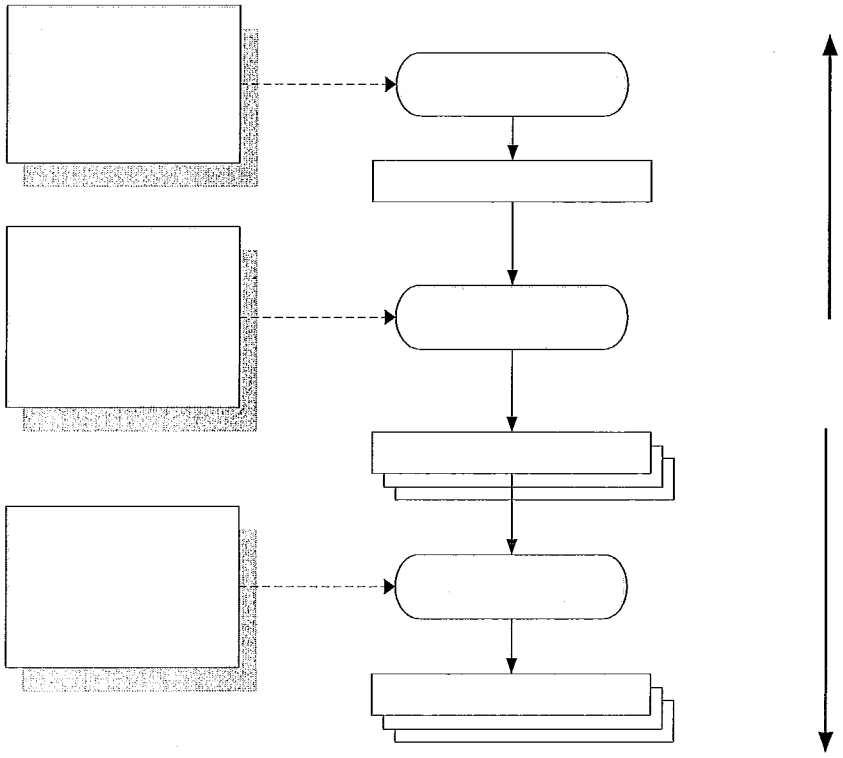


Figura 4.3 - Abordagem para Definição de Processos (VILLELA, 2004)

A norma ISO/IEC 12207 (ISO/IEC, 2000) é a base para a definição de qualquer Processo Padrão a partir da Estação TABA. Para a definição do Processo Padrão são, ainda, consideradas as características do desenvolvimento de software na organização, que são relacionadas ao ambiente de trabalho, conhecimento e experiência das equipes envolvidas e à própria cultura e experiência da organização no desenvolvimento de software. Também são considerados modelos de maturidade como o MPS.BR (Melhoria de Processo do Software Brasileiro) (MPS.BR, 2005a) e o CMMI (*Capability Maturity Model Integration*) (CHRISSIS *et al.*, 2003).

- Processos de Ciclo de
- Modelos de Maturidad Capacidade
- Características do Desenvolvimento de S

A partir do Processo Padrão, diferentes processos de software podem ser especializados de acordo com as características dos tipos de software desenvolvidos na organização (por exemplo: sistemas especialistas, sistemas de informação e sistemas de controle de processos) e dos paradigmas de desenvolvimento adotados (por exemplo: orientado a objetos e estruturado). Neste momento, novas atividades podem ser definidas e incluídas nos processos especializados e a descrição de atividades já definidas no Processo Padrão pode ser adequada. No entanto, todos os elementos básicos definidos no Processo Padrão sempre deverão estar presentes nos processos especializados.

Para ser utilizado em um projeto, o processo especializado mais adequado para um determinado tipo de software deve ser instanciado para atender às características do projeto específico, devendo-se considerar o tamanho e a complexidade do produto bem como as características de qualidade desejadas, a expectativa de vida útil, as características da equipe de desenvolvimento e demais características do projeto. Neste momento, são selecionados o modelo de ciclo de vida, os métodos e ferramentas (BERGER, 2003).

4.4 O Estágio Atual da Estação Taba

A partir de 2003, os ambientes TABA passaram a ser utilizados por empresas brasileiras desenvolvedoras de software, fato que vem fornecendo *feedback* ao projeto e criando novos desafios (MONTONI *et al.*, 2005, SANTOS *et al.*, 2005). Os ambientes TABA foram implantados em 10 empresas associadas à Riosoft (Núcleo Softex do Rio de Janeiro) através de um projeto Riosoft/COPPE-UFRJ, entre elas a BL Informática (MPS.BR Nível F obtido em outubro de 2005) e a Marlin (avaliação MPS.BR Nível D em andamento), apoiando a implantação do MPS.BR e do CMMI.

Além dessas empresas, o ambiente também foi implantado:

- na Relacional Consultoria apoiando a implantação do CMMI Nível 2 (obtido em março de 2005), MPS.BR Nível E (obtido em outubro de 2005) e atualmente do MPS.BR Nível C e CMMI Nível 3,

- no Tribunal Superior Eleitoral apoiando a implantação do CMMI Nível 2 (obtido em dezembro de 2005),
- na Maxtrack em Belo Horizonte apoiando a implantação do CMMI Nível 2 (obtido em janeiro de 2006),
- no CASNAV, órgão da Aeronáutica, visando o nível G do MPS.BR,
- na Synapsis, fábricas de software de Niterói e Fortaleza, visando o CMMI Nível 2,
- no Centro de Computação da Aeronáutica de São José dos Campos, visando o Nível E do MPS.BR.

A utilização da Estação TABA foi apontada como um dos fatores que favorece o sucesso de uma implantação de processos de software utilizando o MR-MPS.BR ou o CMMI em um *survey*, do qual participaram 15 implementadores. Estes implementadores participaram em projetos coordenados pela COPPE/UFRJ em empresas públicas e privadas de diversos portes na implantação do CMMI, MPS.BR ou implementação conjunta destes dois modelos (ROCHA *et al.*, 2005). De forma análoga, a ausência deste apoio ferramental provido pela Estação TABA aparece como dificuldade para a implantação de processo.

O comprometimento da empresa, grau de acompanhamento dos processos implantados, disponibilidade de recursos, motivação da empresa e treinamento também demonstraram ser fatores que influenciaram positivamente quando estavam fortemente presentes e quando eram fracos ou ausentes influenciaram negativamente na implantação de processos utilizando o MR-MPS e o CMMI.

4.4.1 Ferramentas da Estação Taba

As ferramentas TechSolution e MBR, desenvolvidas no contexto deste trabalho, apóiam as organizações na obtenção do Nível 3 e superiores do CMMI e do Nível D e superiores do MPS.BR. Assim como estas ferramentas, outras ferramentas foram construídas com a finalidade de tornar a Estação TABA capaz de apoiar a execução de

processos que estejam aderentes aos modelos CMMI e MPS.BR em seus diversos níveis. A Tabela 4.1 exibe o conjunto de ferramentas e serviços fornecidos atualmente pela Estação TABA para auxiliar os engenheiros de software na definição, execução, avaliação e melhoria dos processos de desenvolvimento e manutenção.

Tabela 4.1 – Ferramentas da Estação TABA

Ferramenta	Descrição
Acknowledge	Apóia o processo de captura de conhecimento ao longo dos processos de software, englobando registro, filtragem e empacotamento de conhecimento (MONTONI, 2003, MONTONI *et al*, 2004)
ActionPlanManager	Apóia a elaboração de planos de ação ao longo do projeto
AdaptPro	Apóia a instanciação de processos de software para projetos específicos a partir dos processos especializados da organização (BERGER, 2003)
Config	Apóia a configuração de ambientes para as organizações (VILLELA, 2004)
ControlManager	Apóia o planejamento do acompanhamento e controle do projeto
DocPlan e GeraDoc	Apóia, respectivamente, o planejamento da documentação a ser produzida em um projeto de software e a gestão de documentos a partir da agregação de outros documentos (MARTINS, 2004)
Editar	Apóia a definição de teorias de tarefas (OLIVEIRA, 1999, ZLOT, 2002, ZLOT *et al.*, 2002)
Edited	Apóia a definição de teorias de domínio (OLIVEIRA *et al.*, 2000)
GConf	Apóia a gerência de configuração dos artefatos produzidos em um projeto de software (FIGUEIREDO, 2004)
Genesis	Apóia a atividade de investigação do domínio (GALOTTA, 2000)
MedPlan	Apóia o planejamento das medições no contexto da organização e do projeto (SCHNAIDER *et al.*, 2004)
Metrics	Apóia a coleta de métricas baseada no plano de medição (SCHNAIDER *et al.*, 2004)
Navegue	Apóia a atividade de investigação do domínio (GALOTTA, 2000)
OrgPlan	Apóia a elaboração do plano de organização para um projeto
Planilha de Atividades	Apóia o registro das atividades do desenvolvedor no projeto
ProjectStatus	Apóia o registro e comunicação da situação dos projetos sendo conduzidos pela organização
QFuzzy	Apóia a identificação dos requisitos de qualidade de produtos (OLIVEIRA, 1999)
QualityPlan	Apóia o planejamento do controle da qualidade que deve ser efetuado no projeto
Regcon	Apóia a atividade de investigação do domínio (GALOTTA, 2000)
ReqManager	Apóia a gerência de requisitos com a construção da matriz de rastreabilidade.
RHPlan e RHManager	Apóia o planejamento, monitoração e avaliação da alocação de profissionais aos projetos de software, o que inclui a solicitação de contratação e capacitação, o acompanhamento das horas dedicadas a cada atividade, além da atualização, ao final do projeto, das competências por eles possuídas (SCHNAIDER, 2003)
RiscManager	Apóia o planejamento e a monitoração de riscos em projetos de software que baseia-se na reutilização do conhecimento organizacional sobre riscos (FARIAS, 2002, FARIAS *et al.*, 2003)
Sapiens	Permite a descrição e visualização de estruturas organizacionais, englobando os profissionais alocados e as competências requeridas e possuídas ao longo dessas estruturas (SANTOS, 2003, SANTOS *et al.*, 2003, SANTOS *et al.*, 2004)

Tabela 4.1 – Ferramentas da Estação TABA

Ferramenta	Descrição
TempPlan, TempManager, CustPlan e CustManager	Apóiam o planejamento e o controle de tempo e custo em projetos de software, baseadas na reutilização do conhecimento organizacional e nos modelos paramétricos COCOMO II e Análise de Pontos de Função (BARCELLOS, 2003, BARCELLOS *et al.*, 2003)
ProcKnow	Apóia a descrição dos processos organizacionais, sejam de software ou não (VILLELA, 2004)
VerificationManager	Apóia a Verificação de Software ao longo do processo de desenvolvimento de software
ValidationManager	Apóia a Validação de Software ao longo do processo de desenvolvimento de software
Biblioteca de Ativos	Permite a definição de uma Biblioteca de Ativos de Processo para organização integrada à ferramenta de Gerência de Configuração
AvalPro	Ferramenta de apoio à Avaliação de Processo Instanciado compatível com o CMMI Nível 3 (ANDRADE, 2005)
Pilot	Permite a realização de projetos-piloto para avaliar melhorias propostas nos processos da organização
TechSolution	Apóia a seleção de alternativas de solução na fase referente a Solução Técnica de um processo de desenvolvimento de software
MBR	Apóia a decisão sobre fazer, comprar ou reutilizar um determinado componente durante o processo de desenvolvimento de software

As ferramentas TechSolution e MBR, por serem objeto desta dissertação, são descritas com detalhes no capítulo 5.

4.5 Trabalhos Relacionados

Uma grande variedade de abordagens de ADS centrados em processo foi definida, projetada e implementada durante os últimos anos. Muitas dessas abordagens foram desenvolvidas para lidar com os ambientes dinâmicos de engenharia de software, tais como a evolução do processo de software. Algumas destas abordagens são descritas a seguir:

- EPOS (*Expert System for Program and ("og") System Development*) é um ADS com ênfase na modelagem de software, na gerência de configuração de software e no apoio ao trabalho cooperativo (MINH *et al.*, 1997). EPOS oferece suporte a uma linguagem de modelagem de processo de software orientada a objetos e reflexiva denominada SPELL. EPOS facilita mecanismos básicos para o (re) planejamento e execução incremental dos modelos de processo através de ferramentas como *Planner* e *Process Engine*. EPOSDB é uma evolução de EPOS construída para armazenar

produtos de software em versões, assim como os seus relativos modelos de software. EPOS também apóia transações cooperativas. EPOS é constituído de um meta-processo para gerenciar a evolução do modelo e de mecanismos para gerenciar a evolução do processo: recuperação da experiência de projeto, registro da experiência de projeto e a manipulação do *layout* da rede de tarefas. Apesar de apoiar eficientemente a modelagem, evolução e execução do processo, EPOS não contém mecanismos de gerência de conhecimento para fornecer conhecimento às pessoas que estão executando o processo como a Estação TABA faz. Além do mais, ao contrário da Estação TABA, que é baseada em uma ontologia de Engenharia de Software, EPOS fornece um meta-modelo apenas para o domínio de processo de software. O relacionamento entre este domínio e as outras áreas de engenharia de software não é permitido.

- Oz é um ADS centrado em processo que implementa os requisitos para um ADS centrado em processo descentralizado baseado em um projeto para a descentralização da modelagem e execução do processo (BEN-SHAUL *et al.*, 1994). O ambiente Oz apóia a modelagem de processo usando duas famílias populares de linguagem de modelagem de processo, regras e *Petri-Nets*. Apesar de implementar uma arquitetura descentralizada de ambiente centrado em processo, o ambiente Oz não possui integração entre os modelos de processo e outras ferramentas de engenharia de software. Além do mais, os formalismos para a modelagem de processo podem se tornar incômodos para o modelador de processos devido à falta de intuição e tolerância. Isto cria barreiras significativas e, conseqüentemente, limita a possibilidade para as linguagens de modelagem de processo serem adotadas na prática (FUGGETTA, 2000). Para lidar com este problema, a Estação TABA foi projetada com uma linguagem de modelagem de processos simples e eficiente, que apóia a modelagem e a evolução do processo de uma forma eficiente e eficaz.

- SPADE é um projeto de pesquisa com o objetivo de desenvolver um ambiente para a análise, projeto e execução do processo de software (BANDINELLI, *et al.*, 1996). SPADE é centrado em uma linguagem de modelagem de processo denominada SLANG. SPADE-1 evoluiu para apoiar a cooperação no desenvolvimento de software. Apesar de SPADE-1 ter demonstrado ser eficiente para lidar com atividades síncronas e

assíncronas, não há evidência da aplicação de tal abordagem na indústria em larga escala. Uma das principais contribuições da Estação TABA é que a sua arquitetura e ferramentas de apoio ao desenvolvimento de software foram modificadas ao longo do tempo para se adequarem melhor às necessidades das organizações de software que executam projetos reais em ambientes dinâmicos e em constante evolução.

- MILOS (*Minimally Invasive Long Term Organizational Support*) tem o objetivo de apoiar processos ágeis através do fornecimento de tecnologia de colaboração e coordenação para desenvolvimento de software distribuído (BOWEN & MAURER, 2002). MILOS também oferece suporte ao planejamento de projeto e à gerência de conhecimento, sendo constituído pelos seguintes componentes: uma máquina de processo, uma base de experiência e um conjunto de recursos. MILOS também apóia o desenvolvimento de software ágil através do uso de práticas de *Extreme Programming* (XP). Ao contrário da Estação TABA, o ambiente MILOS não apóia as atividades de gerência de conhecimento, como a consulta dos conhecimentos e habilidades dos membros do projeto para preencher as necessidades específicas do projeto. Como MILOS não modela o domínio de engenharia de software, a integração eficiente das ferramentas de MILOS à outras ferramentas para apoiar outras áreas de engenharia de software não é garantida.

- Artemis 7 é um software WEB desenvolvido para fornecer suporte aos processos de negócio e às regras associadas e para permitir o desenvolvimento estratégico de múltiplas soluções em uma plataforma comum (ARTEMIS 7, 2006) Artemis 7 permite a configuração de níveis de acesso baseado em regras e direitos garantidos que permite que os usuários acessem os vários módulos e características da solução baseado em suas necessidades individuais. Esta abordagem assegura que cada usuário precisa enxergar apenas as necessidades, as funcionalidades e as informações necessárias para realizar as suas responsabilidades, tornando a aplicação mais fácil de ser utilizada pelos *stakeholders*. Visto que Artemis 7 foi desenvolvido para ser usado em diversos domínios, a definição, projeto, implementação e integração de ferramentas para apoiar as necessidades específicas da área de engenharia de software não são possíveis.

4.6 Considerações Finais

Neste capítulo foram apresentados os conceitos de Ambientes de Desenvolvimento de Software e os principais trabalhos nesta área. Também foram apresentados os conceitos necessários ao entendimento da Estação TABA. Sendo assim, foi discutida a forma como os processos são definidos, especializados e instanciados e a forma como os ambientes são configurados e instanciados no contexto da Estação TABA. Alguns dados sobre a utilização de ambientes TABA por empresas foram fornecidos e discutidos.

Além disso, o conjunto de ferramentas existentes na Estação TABA foi apresentado, fornecendo-se uma visão geral de seu estágio de desenvolvimento e utilização.

No próximo capítulo serão apresentados os processos definidos para apoiar a seleção de alternativas durante a etapa de projeto e para apoiar a análise entre fazer, comprar ou reutilizar um determinado componente. As ferramentas implementadas para apoiar estes processos também serão apresentadas.

CAPÍTULO 5 - APOIO À ESCOLHA DE ALTERNATIVAS DE SOLUÇÃO E À DECISÃO FAZER-COMPRAR-REUTILIZAR

5.1 Introdução

Com o objetivo de apoiar a Solução Técnica em projetos de software, foram definidos dois processos: um processo que apóia a seleção de alternativas de projeto, mantendo assim o registro das decisões tomadas e o porquê de cada decisão; e um processo que apóia a decisão entre desenvolver, comprar ou reutilizar um determinado componente do produto.

Após a definição dos processos, duas ferramentas foram implementadas para dar suporte à execução dos mesmos: a ferramenta TechSolution que implementa as atividades do processo de seleção de alternativas de projeto e a ferramenta MBR que implementa as atividades do processo de análise entre fazer, comprar ou reutilizar um determinado componente. Estas ferramentas foram implementadas utilizando a plataforma de desenvolvimento Microsoft Visual C++ e encontram-se integradas à Estação TABA, podendo ser executadas a partir de um ADSOrg instanciado para um projeto específico.

Algumas atividades do processo de Solução Técnica, como o projeto da solução selecionada e a implementação desta solução, não são apoiadas diretamente pelos processos definidos. Optamos por fornecer suporte através dos processos definidos e das ferramentas implementadas às atividades do processo de Solução Técnica que geralmente não são executadas pelas organizações em seus projetos. As atividades que não são diretamente apoiadas pelos processos definidos de seleção de alternativas de projeto e de análise entre desenvolver, comprar ou reutilizar são apoiadas na Estação TABA através de roteiros específicos associados às atividades do processo de desenvolvimento e/ou manutenção previstas para gerá-los.

Este capítulo apresenta a descrição dos dois processos e as funcionalidades das ferramentas de apoio à execução destes processos.

5.2 Apoio a Seleção de Alternativas

Durante a execução do processo de Solução Técnica, em muitos projetos de software, o único tipo de documentação que é mantida é a documentação final do artefato. Com a finalidade de apoiar as empresas de software a registrar, armazenar e recuperar o DR, foi definido um processo e implementada uma ferramenta denominada TechSolution. Espera-se que com a utilização da ferramenta, os projetos de software possam ser beneficiados pela utilização do DR, conforme discutido no capítulo 3 desta dissertação.

As próximas seções detalham o processo de seleção de alternativas de projeto e a ferramenta TechSolution.

5.2.1 Processo de Seleção de Alternativas

Um processo foi definido para apoiar a seleção de alternativas de projeto, mantendo desta forma o raciocínio por trás das decisões tomadas, ou seja, o DR. Este processo foi definido baseado principalmente nas áreas de processo Solução Técnica e Análise de Decisão e Resolução do CMMI, nos processos de Solução Técnica e de Análise de Decisão e Resolução do MPS.BR e na norma ISO/IEC 14102 (ISO/IEC 14102, 1995). Além disso, outros sistemas de DR encontrados na literatura e apresentados no capítulo 3 desta dissertação também influenciaram na elaboração do processo.

A norma ISO/IEC 14102 descreve um conjunto de processos que possuem como objetivo avaliar ferramentas CASE e selecionar a mais apropriada. As descrições das atividades destes processos da norma influenciaram a definição das atividades do processo proposto. O Anexo 2 apresenta um resumo desta norma.

Em relação à área de processo Análise de Decisão e Resolução do CMMI, esta área apresenta a descrição de práticas a serem executadas por um processo de avaliação de

alternativas e seleção da mais apropriada. No MPS.BR, são descritos os resultados esperados da execução do processo de Análise de Decisão e Resolução. O objetivo do processo de Análise de Decisão e Resolução é analisar possíveis decisões através de um processo formal avaliando alternativas em relação a critérios pré-estabelecidos de modo a identificar a alternativa mais adequada.

Para a representação gráfica do processo, foi utilizada a linguagem de modelagem de processos desenvolvida por VILLELA (2004). A descrição dos elementos da linguagem utilizados nesta dissertação encontra-se no Anexo 3.

A seguir descrevemos as atividades do processo de seleção de alternativas de projeto:

(I) Atividade: Definir Objetivos e Restrições da Seleção

O objetivo desta atividade é definir os objetivos, expectativas e as restrições relativas à avaliação e seleção que serão realizadas nas atividades seguintes. Devem ser identificadas, por exemplo, restrições de cronograma, recursos e custos (ISO/IEC 14102, 1995). Estes aspectos deverão ser levados em consideração durante a avaliação das soluções alternativas e seleção da solução mais apropriada.

(II) Atividade: Estabelecer Critérios de Seleção

Baseado nas expectativas, restrições e objetivos desenvolvidos na atividade Definir Objetivos e Restrições da Seleção, é necessário estabelecer os critérios de seleção. Estes critérios de seleção serão utilizados na avaliação do conjunto de soluções alternativas a serem consideradas e na seleção da solução alternativa mais adequada.

Os critérios devem ser ponderados de modo que o critério de maior peso exerça a maior influência, ou seja, deve-se atribuir um valor a estes critérios que indique a importância em se satisfazer o critério para o projeto em questão. Também é necessário definir as faixas e escalas associadas a cada critério para que se possa, posteriormente, avaliar os critérios de seleção em relação às soluções alternativas ao selecionar um valor na

faixa ou escala estabelecida. Esta atividade é composta pelas sub-atividades: Definir Critérios de Seleção, Definir Faixas e Escalas dos Critérios de Seleção e Ponderar Critérios de Seleção.

Sub-Atividade: Definir Critérios de Seleção

Esta atividade visa definir os critérios para avaliar as soluções alternativas e selecionar a(s) mais apropriada(s). Os critérios devem ser rastreados em relação aos requisitos, cenários, objetivos do negócio e outras fontes externas. CHRISSIS *et al.* (2003) mencionam possíveis tipos de critérios: Limitações Tecnológicas, Impacto no Ambiente, Riscos, Custo do Ciclo de Vida. Durante a execução desta atividade são sugeridas como critérios de seleção ao projetista as características de qualidade existentes na norma ISO/IEC 9126 (2001).

Sub-Atividade: Definir Faixas e Escalas dos Critérios de Seleção

O objetivo desta sub-atividade é definir as faixas e escalas para a posterior avaliação das soluções alternativas em relação aos critérios de seleção. Escalas de importância relativa para os critérios de seleção podem ser estabelecidas com valores não numéricos ou com fórmulas que relacionem o parâmetro de avaliação com um peso numérico (CHRISSIS *et al.*, 2003).

Sub-Atividade: Ponderar Critérios de Seleção

Nesta sub-atividade deve-se ponderar os critérios de seleção de modo que os critérios com maior peso possuam uma maior importância na decisão sobre a seleção de uma alternativa. Esta ponderação deve refletir as necessidades, objetivos e prioridades dos *stakeholders* relevantes em relação à avaliação.

(III) Atividade: Desenvolver Soluções Alternativas

Durante a execução desta atividade deve-se desenvolver soluções alternativas detalhadas para o problema em questão. Uma ampla quantidade de alternativas aparece ao

se solicitar ao maior número de *stakeholders* possível, soluções para a questão. Sessões de *brainstorming* também podem estimular alternativas inovadoras através de uma rápida interação e *feedback* (CHRISSIS *et al.*, 2003). As soluções alternativas identificadas devem ser documentadas.

(IV) Mapear Requisitos às Soluções Alternativas

O objetivo desta atividade é mapear os requisitos com as soluções alternativas geradas. É necessário mapear cada solução alternativa com os requisitos que estão sendo atendidos pela mesma. Desta forma o projetista poderá visualizar como os requisitos estão sendo satisfeitos por cada solução e quais requisitos não estão sendo considerados por uma determinada solução alternativa.

(V) Evoluir Cenários e Conceitos Operacionais

Nesta atividade o projetista deve evoluir os conceitos operacionais, cenários e ambientes para descrever as condições, modos de operação e estados de operação específicos de cada componente do produto. Os conceitos operacionais e cenários são evoluídos para facilitar a seleção de soluções para os componentes de produto. Conceitos operacionais e cenários documentam a interação dos componentes do produto com o ambiente, usuários e outros componentes do produto, sem levar em conta a disciplina de engenharia (CHRISSIS *et al.*, 2003).

A elaboração dos cenários e conceitos operacionais tem início no processo de desenvolvimento de requisitos, sendo neste momento detalhada. Exemplos de cenários podem ser encontrados em BASS *et al.* (2003), onde os autores apresentam *templates* para apoiar a construção de cenários de qualidade, importantes para a avaliação de arquiteturas.

(VI) Avaliar Alternativas

O propósito da avaliação é produzir um relatório técnico que será utilizado como entrada durante a atividade de seleção. Cada solução alternativa deve ser avaliada em relação aos critérios de seleção estabelecidos. A atividade de avaliação deve ser

documentada e esta documentação deve conter a razão pela qual uma determinada solução alternativa recebeu uma determinada pontuação em relação a um critério de seleção.

Sub-Atividade: Avaliar Soluções Alternativas em relação aos Critérios

Esta sub-atividade visa avaliar cada solução alternativa detalhada em relação aos critérios estabelecidos. Essas avaliações devem ser documentadas para que se possa manter o raciocínio que levou uma determinada solução ter sido recomendada.

Sub-Atividade: Relatar Avaliação

No fim da avaliação é gerado um relatório sobre a avaliação, que além de descrever o processo de avaliação executado, também deve apresentar os critérios de seleção identificados, as soluções alternativas com os requisitos mapeados e a avaliação das soluções alternativas em relação aos critérios pré-estabelecidos. O relatório deve discutir as atividades do processo de avaliação no nível de detalhe necessário para permitir ao leitor tanto entender o escopo e a profundidade da avaliação, quanto repetir a avaliação.

(VII) Selecionar Solução

Para selecionar a solução mais adequada, o projetista deve aplicar um algoritmo de seleção aos resultados dos esforços de avaliação das soluções alternativas. Esta atividade tem início assim que o relatório de avaliação estiver completo. Com a aplicação do algoritmo de seleção, uma solução alternativa é recomendada. Os resultados da avaliação podem ser avaliados e então se indicar a necessidade de informações adicionais, o que implicará em re-executar algumas das atividades anteriores do processo.

A figura 5.1 exibe o processo de seleção de alternativas de projeto definido neste trabalho e as figuras 5.2 e 5.3 exibem, respectivamente, as sub-atividades das atividades Estabelecer Critérios de Seleção e Avaliar Alternativas, existentes no processo.

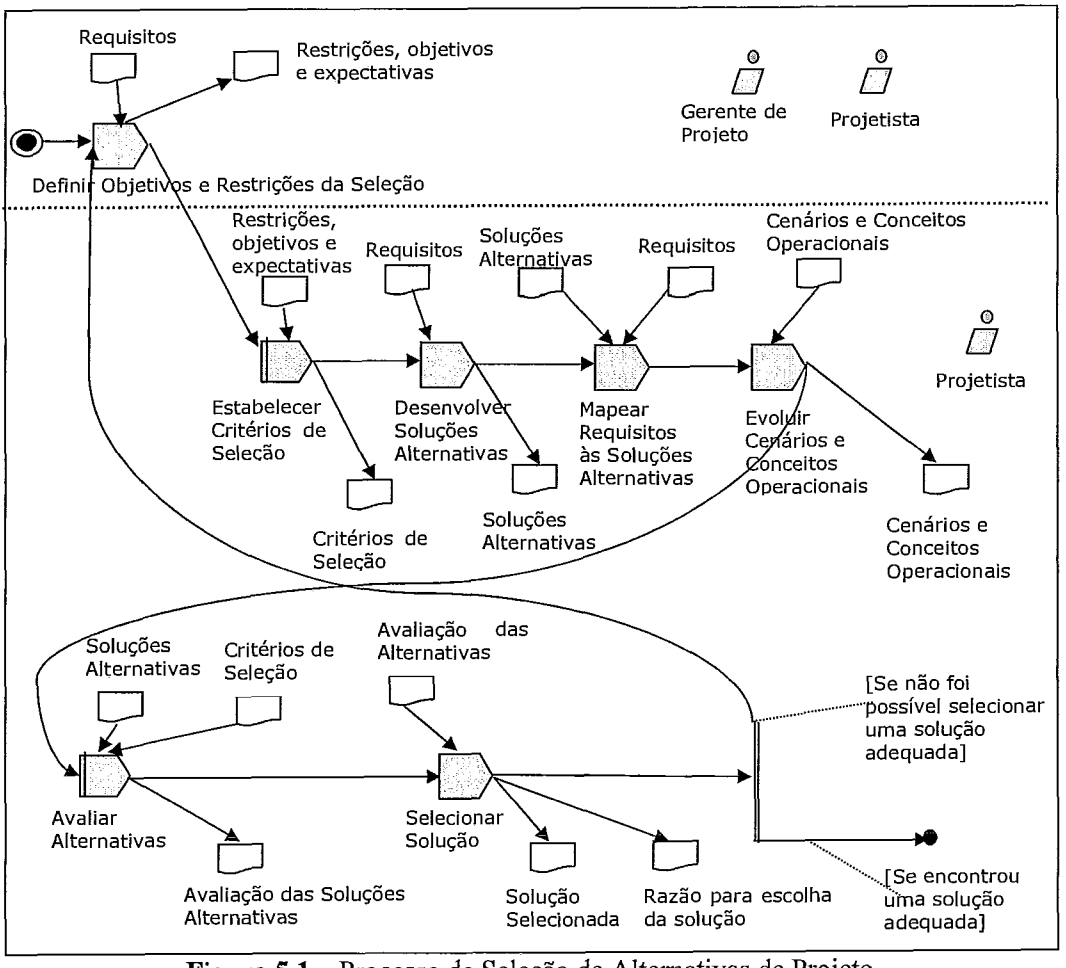


Figura 5.1 – Processo de Seleção de Alternativas de Projeto

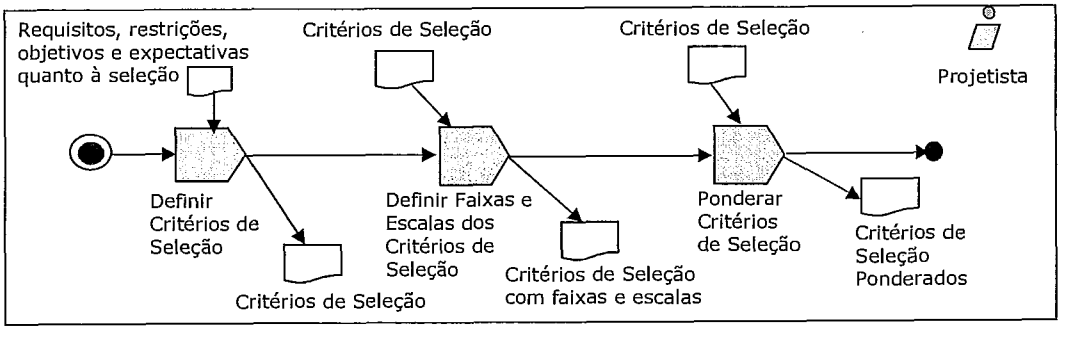


Figura 5.2 – Detalhamento da Atividade "Estabelecer Critérios de Seleção"

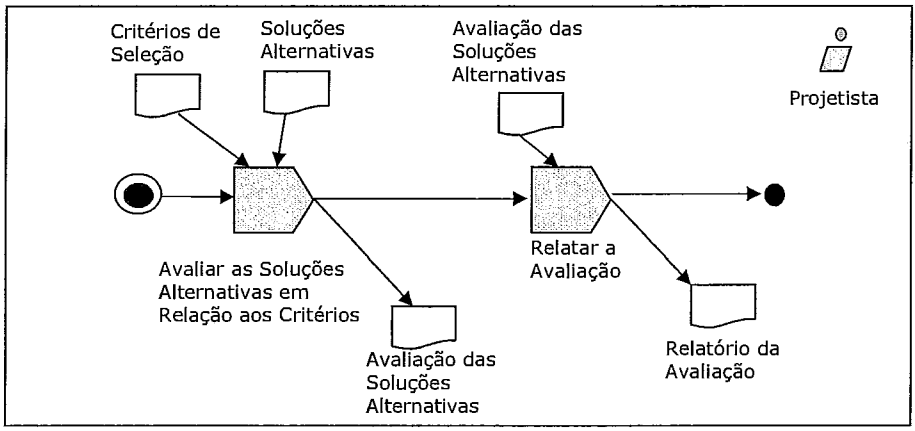


Figura 5.3 – Detalhamento da Atividade "Avaliar Alternativas"

5.2.2 Ferramenta TechSolution

A ferramenta TechSolution está disponível para ser executada nos ADSOrgs instanciados TABA. Em relação à representação do DR, pode-se dizer que a abordagem implementada utiliza a argumentação, tipo de representação explicada no capítulo 3 desta dissertação, e dispõe dos seguintes nós:

- *Avaliação*: Problema para o qual se busca a melhor solução dentre uma lista de possíveis soluções. Uma avaliação possui objetivos, uma lista de possíveis soluções alternativas e critérios que serão utilizados para que seja possível selecionar a solução mais adequada.

- *Objetivo*: Objetivos da avaliação. Este nó poderá conter informações sobre possíveis restrições, como por exemplo, restrições de tempo ou esforço da avaliação em questão. Os objetivos também podem indicar os requisitos do projeto que devem ser atendidos pela solução a ser selecionada.

- *Critério*: As soluções alternativas identificadas serão avaliadas em relação a uma lista de critérios. Cada critério possui um peso que indicará a importância do critério em questão para o projeto. Este peso é atribuído livremente pelo projetista, sendo depois normalizado pelo somatório dos pesos. Um critério pode conter uma lista de sub-critérios quando houver necessidade de refinamento. Neste caso, apenas os sub-critérios serão avaliados.

- *Solução Alternativa*: Soluções para o problema sob avaliação. Estas soluções serão avaliadas em relação aos critérios a fim de que se possa selecionar a solução mais indicada.
- *Avaliação do Critério*: Os critérios da avaliação serão avaliados em relação às soluções alternativas. Esta avaliação compreende a atribuição de um valor que indica o quanto a solução alternativa satisfaz o critério em questão. Além disso, deve-se justificar o porquê da atribuição de um valor a um determinado critério durante a avaliação de uma solução alternativa.

Este esquema de representação foi construído com base nos outros modelos de representação de DR já apresentados nesta dissertação. Comparando-o com o modelo DRL (LEE & LAI, 1991), por exemplo, pode-se dizer que o nó "Avaliação" armazenaria as mesmas informações que o nó "Problema de Decisão", "Solução Alternativa" representaria o nó "Alternativa", e "Avaliação dos Critérios" teria como nó correspondente no outro modelo o nó "Alegação".

O objetivo é manter a documentação das decisões tomadas, das soluções alternativas avaliadas, dos critérios utilizados durante a avaliação e das avaliações propriamente ditas destas alternativas em relação aos critérios.

Para a escolha do estilo arquitetural a ser utilizado (XAVIER, 2001), a ferramenta possui grande parte do conhecimento necessário para a seleção da solução mais apropriada. Neste caso, *TechSolution* fornece ao usuário uma lista de soluções alternativas (estilos arquiteturais), critérios para avaliar estas soluções (características de qualidade), e além disso, a avaliação destes critérios em relação às soluções alternativas (fruto da pesquisa executada por XAVIER (2001), na qual identificou-se a importância dos estilos arquiteturais identificados quando existe a necessidade de satisfazer uma determinada característica de qualidade). O usuário necessita apenas indicar a importância de cada critério para o projeto atual, através da atribuição de um peso. Um resumo da pesquisa realizada por XAVIER (2001) é apresentado no anexo 4.

O projetista também recebe suporte para a identificação dos critérios a serem utilizados para avaliar as soluções alternativas, mesmo na situação em que ainda não foi realizada uma decisão semelhante e em que a ferramenta não possui conhecimento sobre a avaliação a ser realizada. Para isto, a ferramenta apresenta ao usuário uma lista de critérios que poderão ser utilizados durante a avaliação em questão. O projetista poderá partir desta lista de critérios para adicionar novos critérios, excluir critérios sugeridos ou então refinar os critérios sugeridos de acordo com o problema. Uma ajuda similar a esta é encontrada em InfoRat (BURGE, 2005a), na qual foi definida uma ontologia de critérios a serem utilizados. A lista de critérios utilizada inicialmente para apoiar o projetista corresponde às características e sub-características de qualidade que podem ser encontradas na norma ISO/IEC 9126 (2001).

A figura 5.4 exibe a tela principal de um ADSOrg instanciado para apoiar a execução do processo de um projeto. Para executar a ferramenta TechSolution é preciso selecionar a atividade do processo que está sendo apoiada pela ferramenta e dar um duplo clique no ícone da ferramenta, que aparece na guia Ferramentas Associadas.

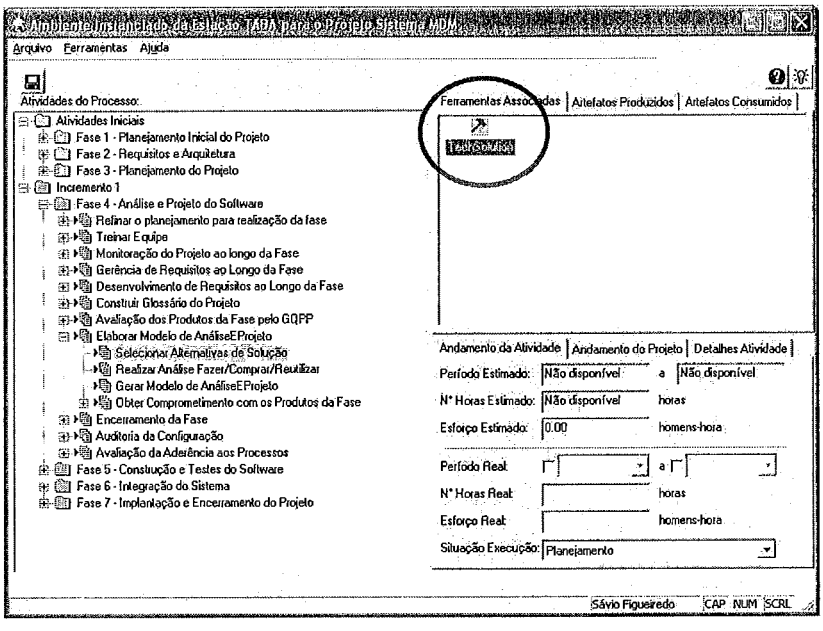


Figura 5.4 – Tela Principal do ADSOrg com o ícone para executar a ferramenta TechSolution em destaque

Ao executar a ferramenta, é exibida a tela ilustrada pela figura 5.5. Esta tela representa a atividade "Definir Objetivos e Restrições da Seleção" e através dela é possível criar uma nova avaliação. Existem dois modos distintos para criar uma nova avaliação: (i) selecionando o *checkbox* de uma das avaliações que são listadas e que estão com o *checkbox* desmarcado ou (ii) clicando na lista de avaliações com o botão direito do mouse e em seguida clicando no botão inserir que é exibido em uma janela *popup*.

No primeiro caso, a avaliação é criada com base no conhecimento existente sobre uma determinada questão. No exemplo exibido, a avaliação é criada com base no conhecimento existente sobre estilos arquiteturais disponibilizado na ferramenta. Sendo assim, o projetista obterá informações sobre: (i) os critérios de seleção utilizados para a avaliação de estilos arquiteturais, (ii) as soluções alternativas que poderiam ser utilizadas para esta avaliação, no caso, uma lista de possíveis estilos arquiteturais e (iii) a avaliação das soluções alternativas sugeridas em relação aos critérios de seleção sugeridos.

No segundo caso, o conhecimento fornecido para a realização da avaliação, além da descrição do processo a ser executado, é uma lista de critérios de seleção. A lista de critérios sugerida inicialmente são as características de qualidade da norma ISO/IEC 9126.

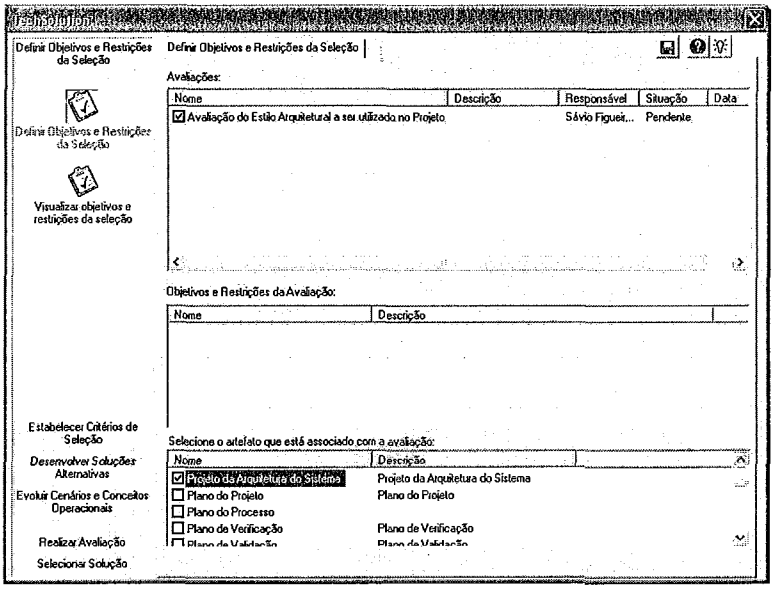


Figura 5.5 – Atividade "Definir Objetivos e Restrições da Seleção"

Após criar a avaliação, o projetista deve indicar quais são os objetivos, expectativas e restrições da avaliação e selecionar o artefato do processo sobre o qual a avaliação será realizada. Desta forma, outras pessoas poderão consultar, posteriormente, que DRs existem para aquele artefato.

Após executar a atividade Definir Objetivos e Restrições da Seleção, o projetista poderá visualizar um relatório que contém as informações fornecidas durante a execução da mesma. Para isto, é necessário clicar no ícone da atividade "Visualizar Objetivos e Restrições da Seleção". A figura 5.6 apresenta a tela referente a esta atividade.

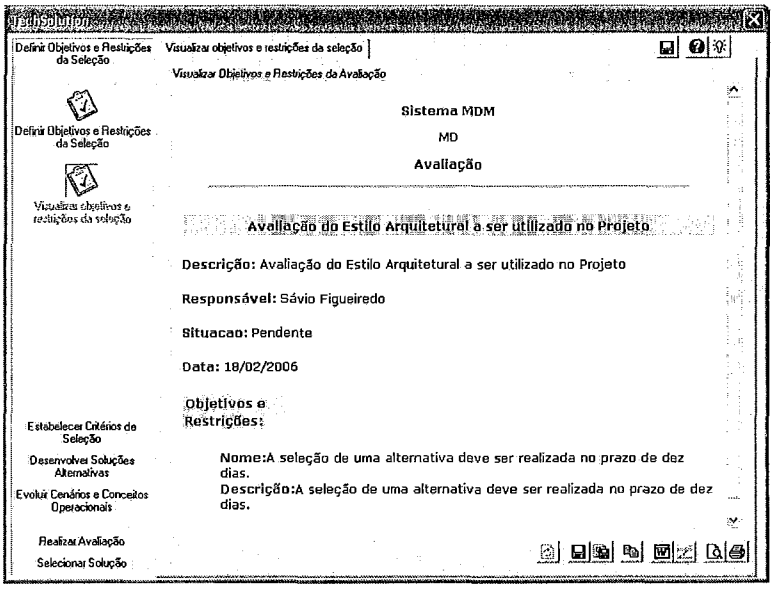


Figura 5.6 – Atividade "Visualizar Objetivos e Restrições da Seleção"

A próxima atividade a ser executada é a definição dos critérios de seleção. Esta atividade é uma sub-atividade da atividade "Estabelecer Critérios de Seleção" e é exibida na figura 5.7. Durante esta atividade, o projetista deve identificar os critérios que serão utilizados para selecionar a solução mais adequada. Neste momento, será possível selecionar os critérios sugeridos ou incluir novos critérios de seleção.

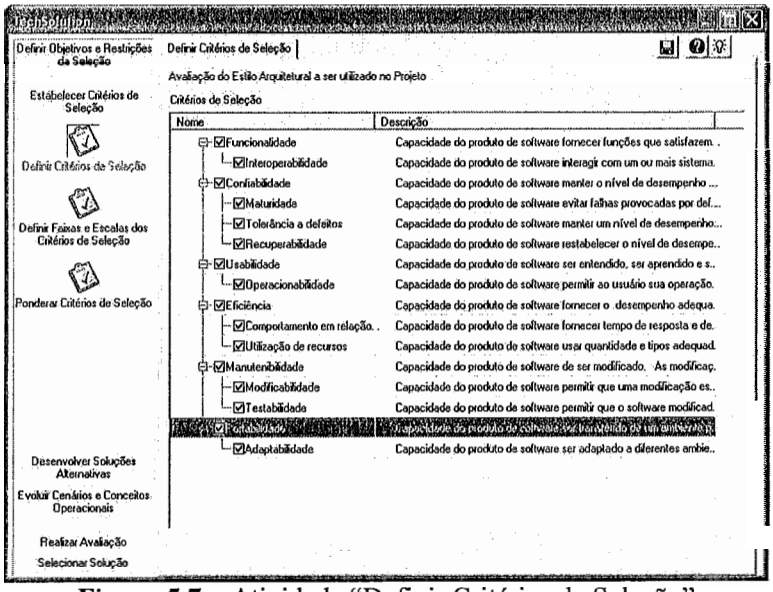


Figura 5.7 – Atividade "Definir Critérios de Seleção"

O próximo passo é atribuir uma faixa ou escala para os critérios selecionados. A Figura 5.8 exibe a tela da atividade "Definir Faixas e Escalas dos Critérios de Seleção". Nesta atividade, para cada critério elementar, ou seja, para cada critério que não possui um sub-critério, deve-se selecionar uma escala que represente o universo de valores que poderá ser atribuído a este critério durante a avaliação do mesmo em relação às soluções alternativas. Estas escalas são definidas no Ambiente Configurado.

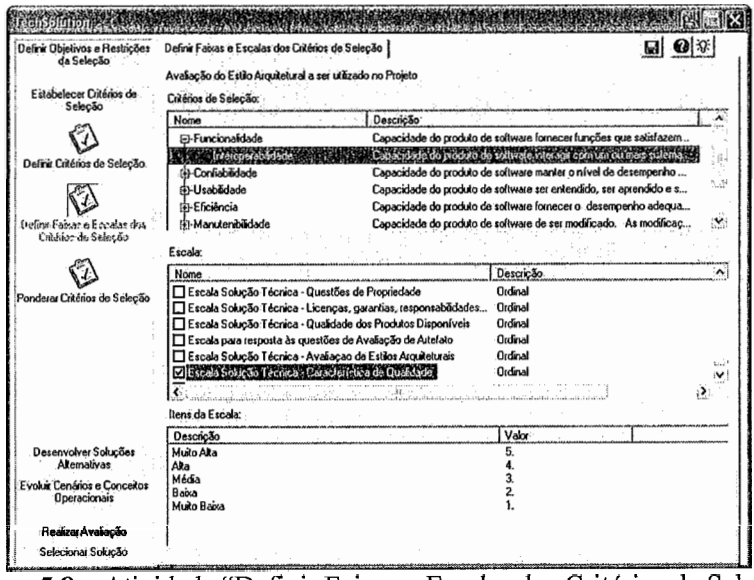


Figura 5.8 – Atividade "Definir Faixas e Escalas dos Critérios de Seleção"

No exemplo exibido, para o critério "Interoperabilidade" foi atribuída a escala "Escala Solução Técnica – Características de Qualidade" que contém os valores "Muito Alta", "Alta", "Média", "Baixa" e "Muito Baixa". Deste modo, durante a avaliação de uma solução alternativa, o projetista deverá selecionar um destes itens da escala para indicar o grau em que aquela solução alternativa satisfaz o critério.

Durante a atividade "Ponderar Critérios de Seleção", representada pela figura 5.9, o projetista deve indicar qual peso dos critérios selecionados para o projeto, de modo a refletir a importância em satisfazer os critérios para o projeto em questão. Deste modo, critérios considerados mais importantes exercerão um maior peso na avaliação. Após atribuir valor para o peso, a coluna contribuição é atualizada automaticamente exibindo a contribuição do critério em questão. Esta contribuição é calculada ao dividir o peso em questão pelo somatório dos pesos.

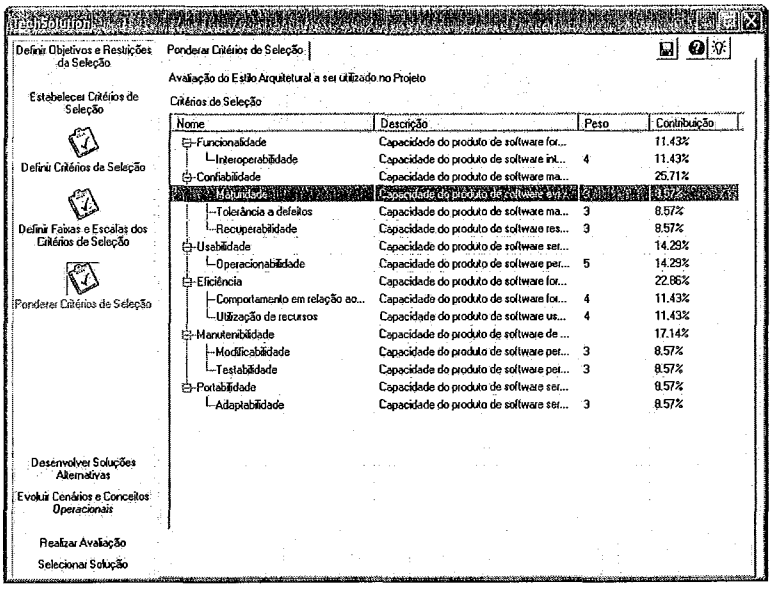


Figura 5.9 – Atividade "Priorizar Critérios de Seleção"

A macro-atividade "Desenvolver Soluções Alternativas" contém as atividades de especificação das soluções alternativas, de mapeamento das soluções alternativas em relação aos requisitos e de visualização das soluções alternativas.

Ao executar a atividade "Especificar Soluções Alternativas", ilustrada na figura 5.10, o projetista deve definir que soluções alternativas devem ser avaliadas. No caso da avaliação do estilo arquitetural a ser utilizado, algumas soluções são sugeridas, visto que a ferramenta possui conhecimento sobre este tipo de questão. Este conhecimento foi obtido de uma pesquisa descrita em XAVIER (2001), na qual o autor buscou relacionar o quanto às características de qualidade definidas na norma ISO/IEC 9126 satisfazem determinados estilos arquiteturais.

Para cada solução alternativa inserida na lista de soluções alternativas, deve-se incluir uma descrição para a mesma e associar um artefato que a explique com maiores detalhes. O registro das soluções alternativas é uma importante parte do DR, pois permite que um projetista saiba que soluções foram analisadas e recusadas para uma decisão específica.

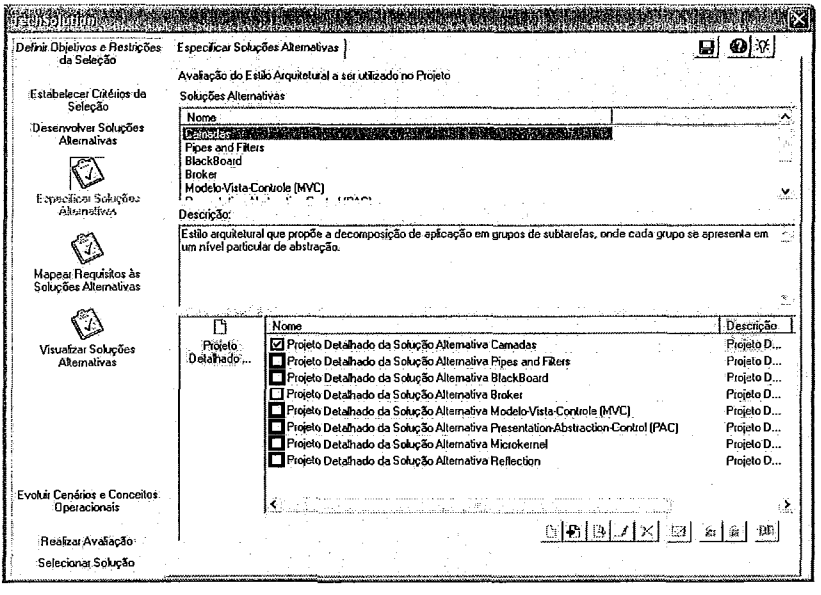


Figura 5.10 – Atividade "Especificar Soluções Alternativas"

Durante a execução da atividade "Mapear Requisitos às Soluções Alternativas", exibida na figura 5.11, deve-se indicar os requisitos que são atendidos por uma solução alternativa. Assim, após selecionar uma solução alternativa na lista de soluções alternativas

existente na parte superior da tela, o projetista deve marcar os *checkbox* dos requisitos que são satisfeitos pela solução alternativa em questão.

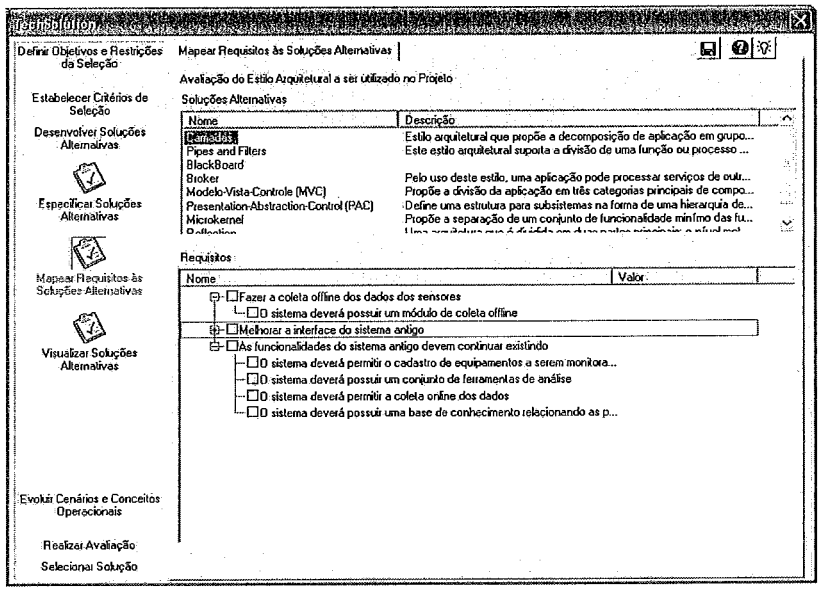


Figura 5.11 – Atividade "Mapear Requisitos às Soluções Alternativas"

Para finalizar esta macro-atividade, basta visualizar o relatório que descreve as soluções alternativas, gerado pela atividade "Visualizar Soluções Alternativas", que é ilustrado pela figura 5.12.

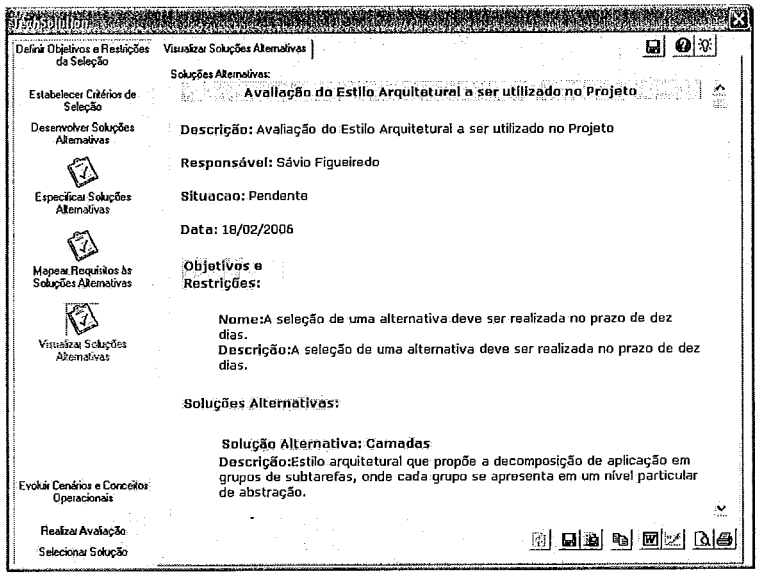


Figura 5.12 – Atividade "Mapear Requisitos às Soluções Alternativas"

Com o objetivo de apoiar a avaliação das soluções alternativas, é necessário evoluir os cenários e conceitos operacionais. O projetista deve avaliar quais soluções alternativas estão satisfazendo os cenários de forma apropriada. A figura 5.13 exibe a tela da ferramenta TechSolution que apóia a execução desta atividade.

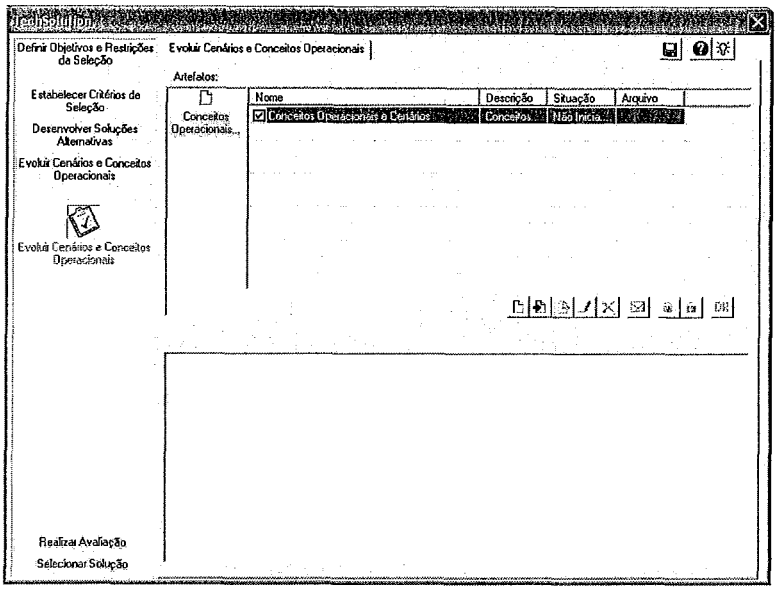


Figura 5.13 – Atividade "Evoluir Cenários e Conceitos Operacionais"

Na macro-atividade "Realizar Avaliação", as soluções alternativas são avaliadas em relação aos critérios pré-estabelecidos e um relatório é gerado com a finalidade de apoiar o projetista na seleção da solução mais adequada. A avaliação das soluções alternativas ocorre durante a atividade "Avaliar Soluções Alternativas em relação aos Critérios", ilustrada pela figura 5.14.

Ao selecionar uma solução alternativa na lista de soluções alternativas, na parte superior da tela, são exibidos os critérios que precisam ser avaliados. Então, o projetista precisa dar um duplo clique na coluna "Valor" da lista "Avaliação" para cada critério elementar, ou seja, para cada critério que não possuir sub-critério. Em seguida, o projetista deve selecionar um valor na faixa de valores disponível para o critério em questão. Além disso, também é necessário indicar o porquê da atribuição daquele valor para o critério através do preenchimento do campo "Justificativa". Assim, o raciocínio pelo qual uma

determinada solução alternativa recebeu um valor em relação a um critério é mantido, fato que contribui para a formação do DR.

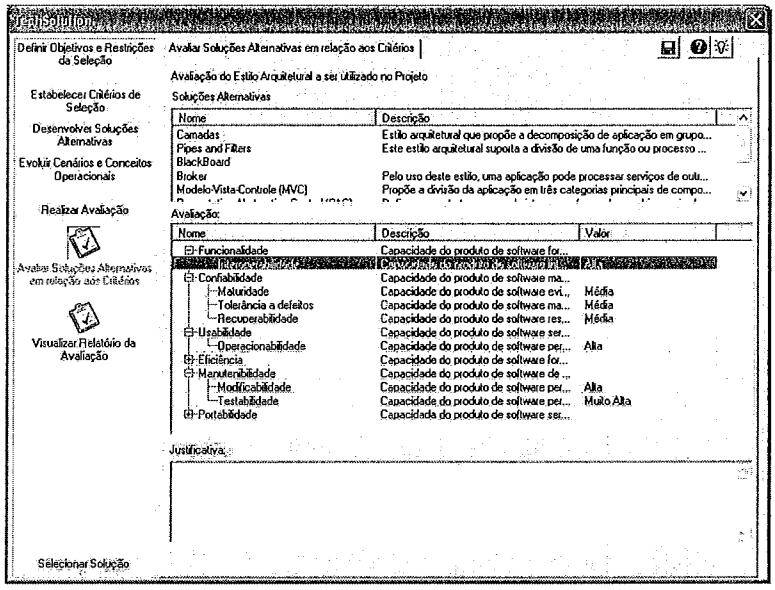


Figura 5.14 – Atividade "Evoluir Cenários e Conceitos Operacionais"

Tendo avaliado as soluções alternativas é necessário gerar um relatório que sirva de entrada para a atividade de seleção da alternativa mais adequada. Este relatório deve conter, além da descrição do processo de avaliação, todas as informações geradas durante o mesmo. O relatório também sugere a seleção de uma determinada solução alternativa, com base no peso dos critérios e na avaliação das soluções alternativas em relação aos critérios. A figura 5.15 exibe a atividade da ferramenta TechSolution que gera este relatório.

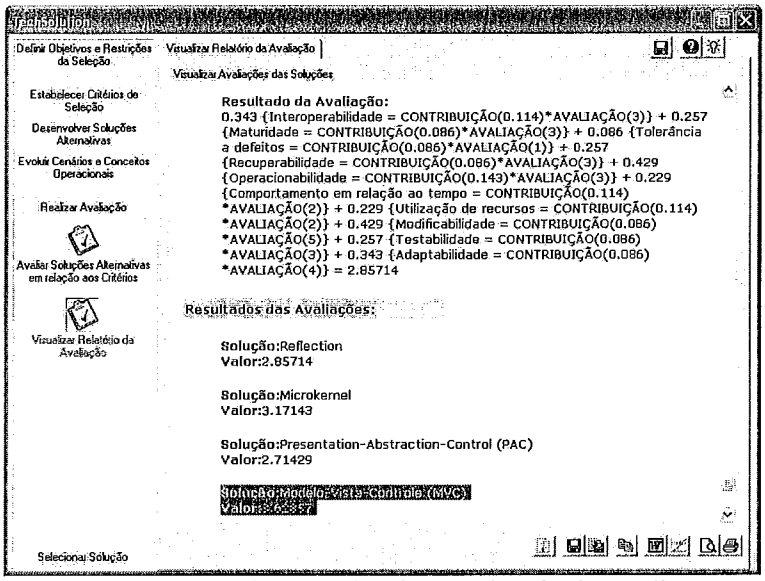


Figura 5.15 – Atividade "Visualizar Relatório da Avaliação"

Por fim, o projetista deve selecionar a solução mais adequada tendo em vista o relatório de avaliação gerado. Caso não consiga chegar a uma solução, pode ser necessário re-executar alguma das atividades do processo para refinar as informações. A figura 5.16 exibe a atividade da ferramenta TechSolution através da qual o projetista seleciona uma solução. Após selecionar a solução mais adequada na lista de soluções alternativas da parte superior da tela, é necessário justificar a decisão através do preenchimento do campo "Justificativa". Deste modo, qualquer pessoa que precise saber o porquê da escolha de um determinado estilo arquitetural para o projeto, poderá consultar o DR que foi gerado. A figura 5.17 exibe o relatório que descreve toda a avaliação, incluindo a decisão tomada.

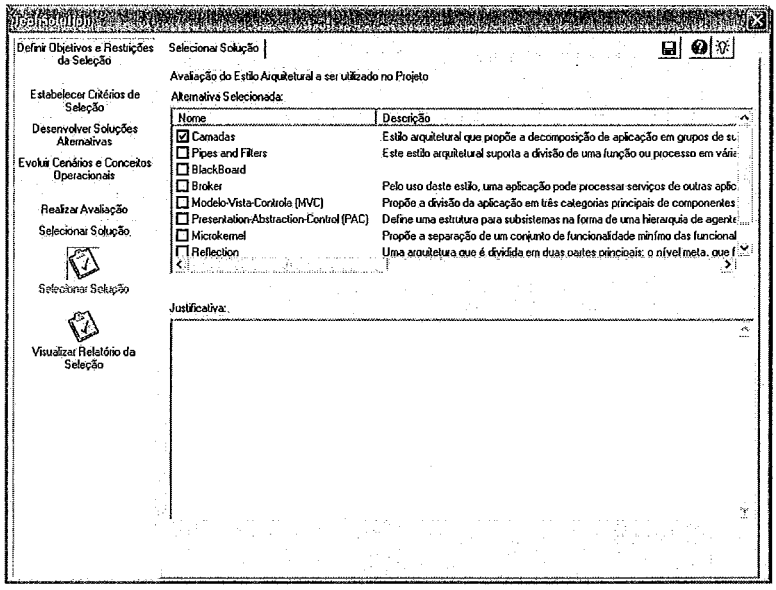


Figura 5.16 – Atividade "Selecionar Solução"

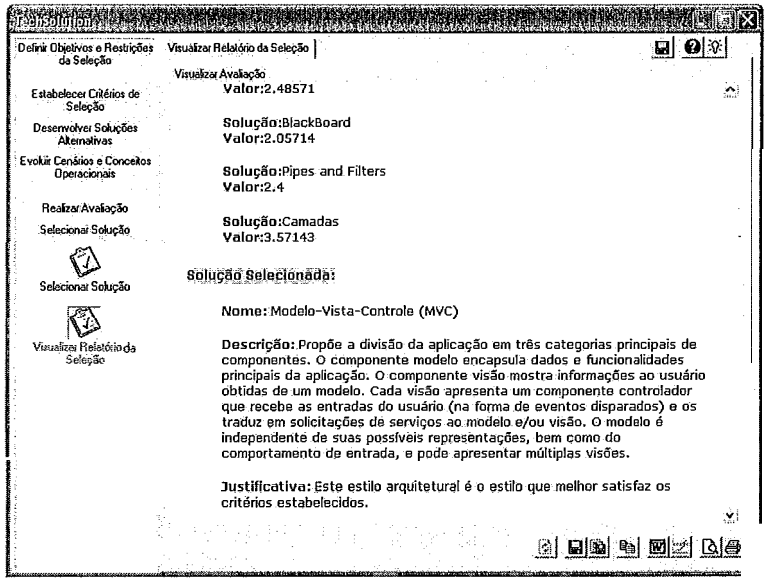


Figura 5.17 – Atividade "Visualizar Relatório da Seleção"

5.3 Apoio a Análise Fazer-Comprar-Reutilizar

Em virtude da atual globalização, da forte competitividade e do crescimento da complexidade dos produtos de software, as empresas buscam melhores formas de se organizarem. As organizações tentam centralizar-se em suas atividades principais, a fim de melhorar a gerência dos custos. Isto levanta questões como: Que recursos deveriam ser desenvolvidos para aumentar as competências da organização? Que atividades deveriam ser feitas por outras empresas e a que parceiros em potencial deveriam ser contratados para executar estas atividades? Que atividades internas deveriam ser preservadas e desenvolvidas pela própria organização? Como os recursos da organização deveriam ser alocados em relação às atividades (BOUCHRIHA *et al.*, 2002)?

As abordagens existentes na literatura para responder a estas questões possuem duas correntes principais de pensamento: a primeira busca responder a pergunta sob um ponto de vista econômico e a segunda foca no ponto de vista estratégico (CÁNEZ *et al.*, 2001). CÁNEZ *et al.* (2001) descrevem as principais abordagens criadas na década passada para apoiar a decisão de fazer ou comprar.

Além disso, assim como as tecnologias evoluem, o raciocínio para fazer ou comprar um componente do produto também evolui. Por um lado, esforços de desenvolvimento complexos favorecem a compra de um componente de produto de prateleira, mas por outro, avanços na produtividade e ferramentas favorecem a opção de desenvolvimento. Durante essa escolha, deve-se levar em consideração que produtos de prateleira podem ter documentação incompleta ou inexata e as organizações que os fornecem podem não dar suporte para os mesmos no futuro (CHRISSIS *et al.*, 2003). Estes fatores devem ser levados em consideração durante o processo de escolha. A lista a seguir apresenta fatores que podem ser levados em consideração durante o processo de escolha (CHRISSIS *et al.*, 2003):

- Funções que os produtos ou serviços irão fornecer e como essas funções se encaixarão no projeto.
- Recursos e habilidades disponíveis no projeto.
- Datas de entrega e integração críticas.
- Uniões de negócio estratégicas, incluindo requisitos de negócio de alto nível.

- Pesquisa do mercado sobre produtos disponíveis, incluindo produtos COTS (*Commercial Off-the-Shelf*).
- Funcionalidade e qualidade dos produtos disponíveis.
- Habilidades e capacidade dos potenciais fornecedores.
- Impacto do produto ou serviço nas competências centrais da organização.
- Licenças, garantias, e limitações associadas com os produtos sendo adquiridos.
- Disponibilidade do produto.
- Questões proprietárias.
- Redução de riscos.

5.3.1 Processo de Análise entre Fazer, Comprar ou Reutilizar

A decisão entre fazer, comprar ou reutilizar um determinado componente do produto também é uma escolha de alternativa de projeto, entretanto, pode ser mais bem apoiada através da utilização de um processo que possua atividades específicas para este tipo de decisão. Neste sentido, foi definido um processo, com base no processo de seleção de alternativas de projeto definido, para apoiar a escolha entre fazer, comprar ou reutilizar um componente do produto.

Este processo é composto pelas seguintes atividades:

(I) Caracterizar Produto/Componente

Durante a execução desta atividade, deve-se identificar os requisitos que o produto/componente necessita satisfazer. Estes requisitos serão fundamentais durante todas as atividades do processo de seleção de uma alternativa de compra, reutilização ou desenvolvimento. Tais requisitos já foram desenvolvidos no processo de desenvolvimento de requisitos, mas podem não estar agrupados de forma a indicar que estarão sendo atendidos por um produto/componente.

(II) Estabelecer Critérios de Seleção

Baseado nos requisitos identificados para o produto/componente, é necessário estabelecer critérios de seleção. Estes critérios de seleção serão utilizados na avaliação e seleção de uma das alternativas que são: Comprar, Fazer ou Reutilizar. Os critérios devem ser ponderados de modo que o critério de maior peso exerça a maior influência. Esta atividade também é executada no processo de seleção de alternativas de projeto apresentado e é composta pelas mesmas sub-atividades: Definir Critérios de Seleção, Definir Faixas e Escalas dos Critérios de Seleção e Ponderar Critérios de Seleção.

Sub-Atividade: Definir Critérios de Seleção

Esta atividade visa definir os critérios para avaliar as alternativas de compra, desenvolvimento ou reuso e selecionar a mais apropriada. Possíveis tipos de critérios são: Recursos e habilidades disponíveis no projeto, custo de adquirir versus custo de desenvolver internamente, datas criticas de entrega e integração, alianças estratégicas de negócio, funcionalidade e qualidade dos produtos disponíveis, habilidades e capacidades dos potenciais fornecedores, redução de riscos etc.

Sub-Atividade: Definir Faixas e Escalas dos Critérios de Seleção

O objetivo desta sub-atividade é definir as faixas e escalas para a posterior avaliação das soluções alternativas em relação aos critérios de seleção. Escalas de importância relativa para os critérios de seleção podem ser estabelecidas com valores não numéricos ou com fórmulas que relacionem o parâmetro de avaliação com um peso numérico (CHRISSIS *et al.*, 2003).

Sub-Atividade: Ponderar Critérios de Seleção

Nesta sub-atividade deve-se ponderar os critérios de seleção de modo que os critérios com maior peso possuam uma maior importância na decisão sobre a seleção de uma alternativa.

(III) Identificar Fornecedores e Produtos\Componentes Potenciais

Esta atividade tem por objetivo identificar potenciais fornecedores do produto/ componente em questão e localizar produtos\componentes capazes de satisfazer aos requisitos desenvolvidos. Devem ser identificados os fornecedores que poderiam entregar um produto\componente que atenda aos requisitos levantados, assim como os produtos\componentes já existentes que preencham a estes requisitos. Os produtos\componentes existentes na organização e que poderiam ser reutilizados devem ser levados em consideração.

Após identificar potenciais fornecedores e produtos\componentes, o projetista deve decidir entre continuar o processo de avaliação e seleção das alternativas, ou então, selecionar a decisão de desenvolver o produto\componente internamente.

Caso após ter sido feita uma pesquisa por potenciais fornecedores e produtos\componentes, a busca não tenha encontrado nenhum fornecedor ou produto\componente potencial e o projetista não possua a expectativa de encontrar um produto\componente que satisfaça aos requisitos e nem um fornecedor capaz de desenvolvê-lo, então a decisão deve ser por construir o produto\componente internamente e não é necessário continuar com o processo de avaliação e seleção da alternativa.

Se, apesar de não ter encontrado, o projetista possua a expectativa de que poderá encontrar um fornecedor ou produto\componente adequado, então o projetista deve re-executar a atividade de caracterização do produto\componente para rever a lista de requisitos definida e também re-executar a busca. Esta lista pode não estar adequada e por isso a busca não foi bem sucedida.

Caso tenha encontrado um número de potenciais fornecedores e\ou produtos\componentes que considere suficiente, o projetista pode executar a próxima atividade do processo.

Sub-Atividade: Identificar Fornecedores e Produtos\Componentes Externos

Nesta sub-atividade, o projetista deve realizar uma busca por fornecedores e produtos\componentes que possam satisfazer aos requisitos levantados para o produto\componente em questão.

Sub-Atividade: Identificar Produtos\Componentes Reutilizáveis

O objetivo desta sub-atividade é identificar dentro da organização os produtos\componentes que podem ser reutilizados e que poderiam satisfazer aos requisitos do produto\componente.

(IV) Coletar Dados dos Produtos\Componentes e ou Fornecedores

Nesta atividade, o projetista deve coletar os dados dos produtos\componentes e fornecedores identificados que serão necessários para a execução da atividade de avaliação. Em relação aos produtos\componentes, o projetista deve coletar informações como, por exemplo: a versão a ser utilizada, tipos e preços das licenças existentes, *site* onde é possível localizá-lo, funções, público alvo e o ambiente de software onde é possível utilizá-lo

(V) Realizar Avaliação

O propósito da avaliação é produzir um relatório técnico que será utilizado como entrada durante a atividade de seleção de uma das alternativas. Cada alternativa deve ser avaliada em relação aos critérios de seleção estabelecidos. Toda a atividade de avaliação deve ser documentada e esta documentação deve conter, entre outras coisas, a razão pela qual uma determinada solução alternativa recebeu uma determinada pontuação em relação a um critério de seleção.

Sub-Atividade: Avaliar Soluções Alternativas em relação aos Critérios

Esta sub-atividade visa avaliar as alternativas de comprar de um dos fornecedores identificados, comprar um dos componentes\produtos identificados, reutilizar ou construir internamente em relação aos critérios estabelecidos. Essas avaliações devem ser documentadas para que se possa manter o raciocínio pelo qual uma determinada solução foi recomendada.

Sub-Atividade: Relatar Avaliação

No fim da avaliação é gerado um relatório sobre a avaliação, que além de descrever o processo de avaliação executado, também deve apresentar os critérios de seleção identificados, as soluções alternativas com os requisitos mapeados e a avaliação das soluções alternativas em relação aos critérios pré-estabelecidos. O relatório deve discutir as atividades do processo de avaliação no nível de detalhe necessário para permitir ao leitor tanto entender o escopo e a profundidade da avaliação, quanto repetir a avaliação.

(VI) Tomar Decisão

Decidir entre comprar, construir internamente ou reutilizar o produto\componente a partir do resultado da avaliação realizada. Esta decisão tem como base o resultado da avaliação das alternativas mencionadas em relação aos critérios pré-estabelecidos. Deve-se aplicar um algoritmo de seleção no resultado da avaliação para obter uma solução recomendada. Caso não consiga chegar a uma conclusão, o projetista pode re-executar alguma das atividades do processo com a finalidade de refinar as informações.

A representação gráfica das atividades do processo apresentado pode ser vista através das figuras 5.18, 5.19, 5.20 e 5.21. A figura 5.18 apresenta uma visão geral do processo, a figura 5.19 exibe o detalhamento da atividade "Estabelecer Critérios de Seleção", a figura 5.20 ilustra o detalhamento da atividade "Identificar Fornecedores e Produtos\Componentes Potenciais" e a figura 5.21 apresenta o detalhamento da atividade "Realizar Avaliação".

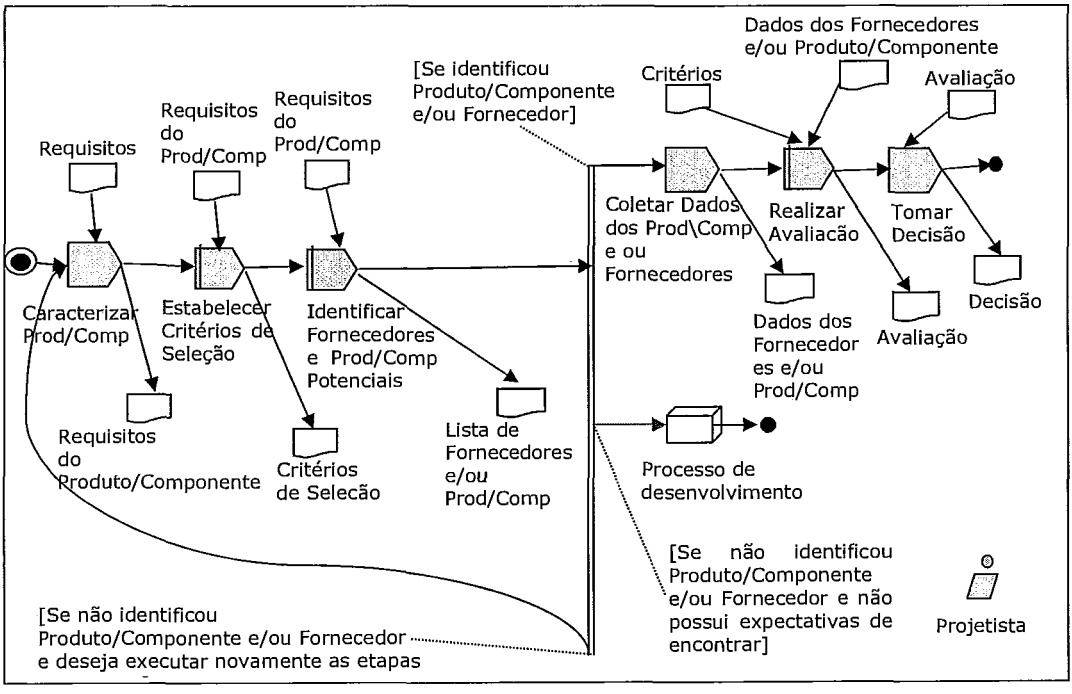


Figura 5.18 – Processo de Análise Fazer/Comprar/Reutilizar

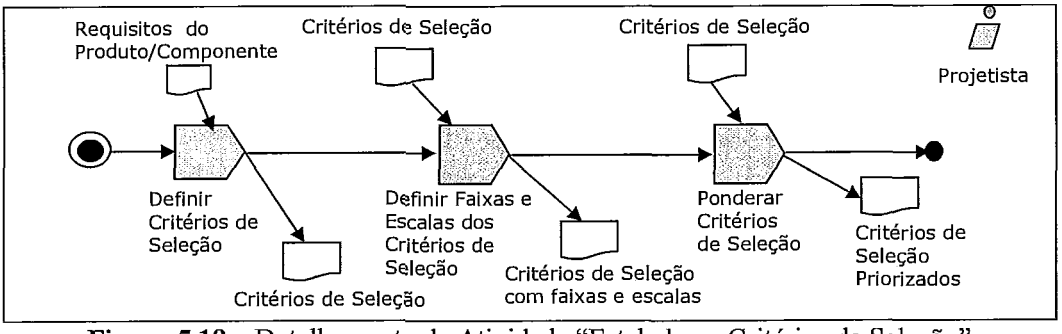


Figura 5.19 – Detalhamento da Atividade "Estabelecer Critérios de Seleção"

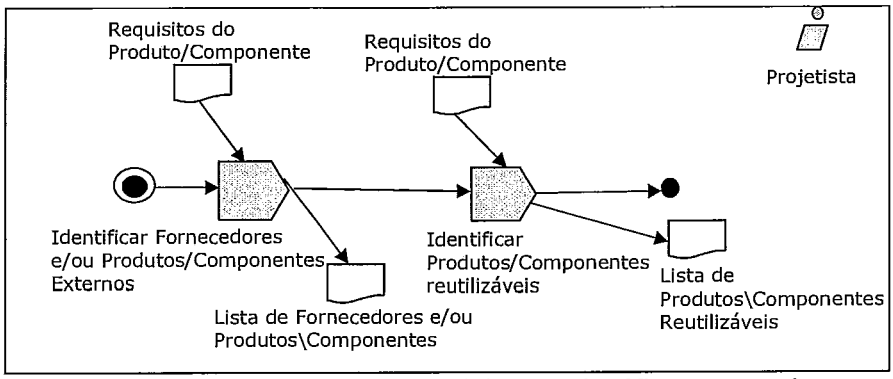


Figura 5.20 – Detalhamento da Atividade "Identificar Fornecedores e Produtos/Componentes Potenciais"

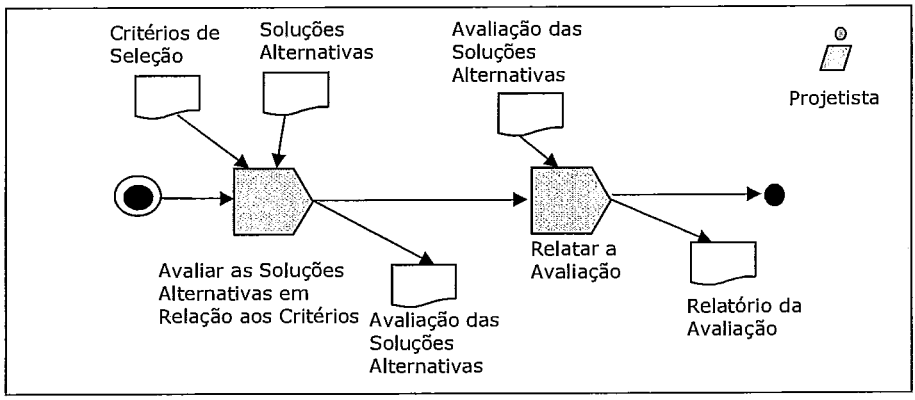


Figura 5.21 – Detalhamento da Atividade "Realizar Avaliação"

5.3.2 Ferramenta MBR

Com o propósito de dar suporte a execução do processo decisório entre fazer, comprar ou reutilizar um componente do produto, apresentado na seção 5.3.1, foi implementada a ferramenta MBR. Esta ferramenta pode ser executada a partir dos ADSOrgs instanciados pela Estação TABA e, assim como a ferramenta MBR, tem sua execução iniciada pelo duplo clique no ícone da MBR disponível na guia de ferramentas associadas da tela principal de um ADSOrg. A figura 5.22 ilustra a tela principal de um ADSOrg instanciado, destacando o ícone da ferramenta MBR.

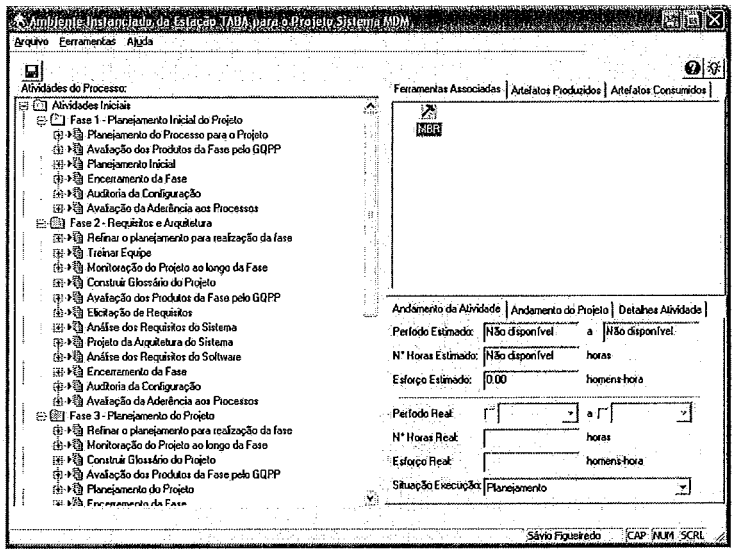


Figura 5.22 – Tela Principal do ADSOrg com o ícone para executar a ferramenta MBR em destaque

Ao executar a ferramenta MBR, a primeira tela a ser exibida é a tela apresentada na figura 5.23. Esta tela é referente à atividade de caracterização do produto/componente sob o qual a organização precisa decidir entre desenvolver, comprar ou reutilizar. Após inserir uma nova avaliação na lista presente na parte superior da tela, deve-se selecionar na lista existente na parte inferior da tela os requisitos que o componente ou produto em questão deve satisfazer e as características de qualidade importantes de serem atendidas pelo mesmo.

Em seguida, a figura 5.24 exibe a próxima atividade da ferramenta a ser executada, denominada de "Visualizar Caracterização do Produto/Componente", na qual é possível visualizar um relatório contendo as informações fornecidas durante a execução da atividade de caracterização do produto/componente.

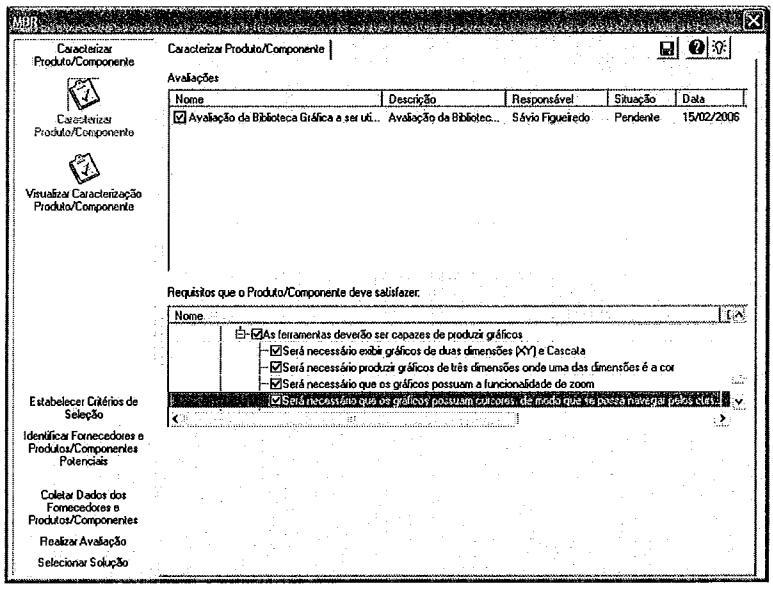


Figura 5.23 – Atividade "Caracterizar Produto\Componente"

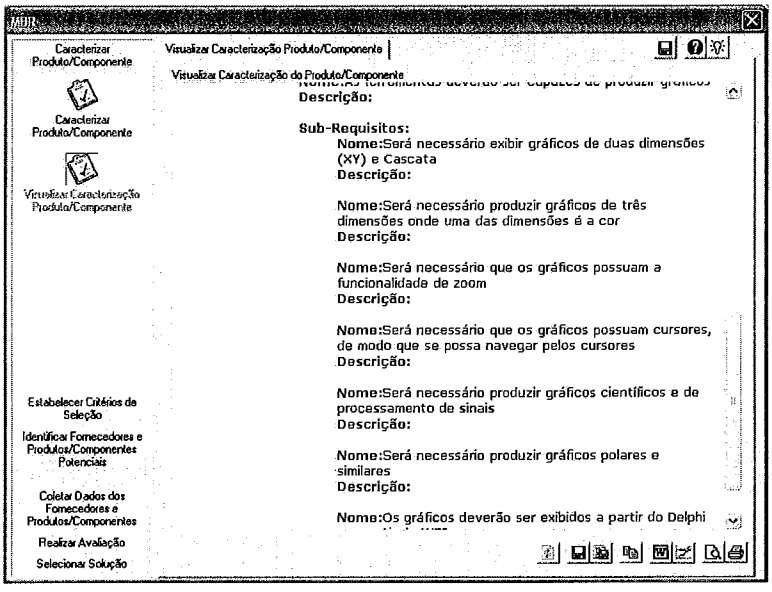


Figura 5.24 – Atividade "Visualizar Caracterização do Produto\Componente"

A macro-atividade "Estabelecer Critérios de Seleção" funciona da mesma forma que a atividade "Estabelecer Critérios de Seleção" da ferramenta TechSolution. A única diferença entre as mesmas é a lista de critérios de sugerida pela ferramenta, visto que a ferramenta MBR sugere critérios que estão relacionados com a análise de desenvolver,

comprar ou reutilizar. A figura 5.25 apresenta a atividade "Definir Critérios de Seleção" da Ferramenta MBR. Após a definição dos critérios, são definidas as faixas e escalas para estes critérios e, em seguida, estes critérios são ponderados, assim como na ferramenta TechSolution.

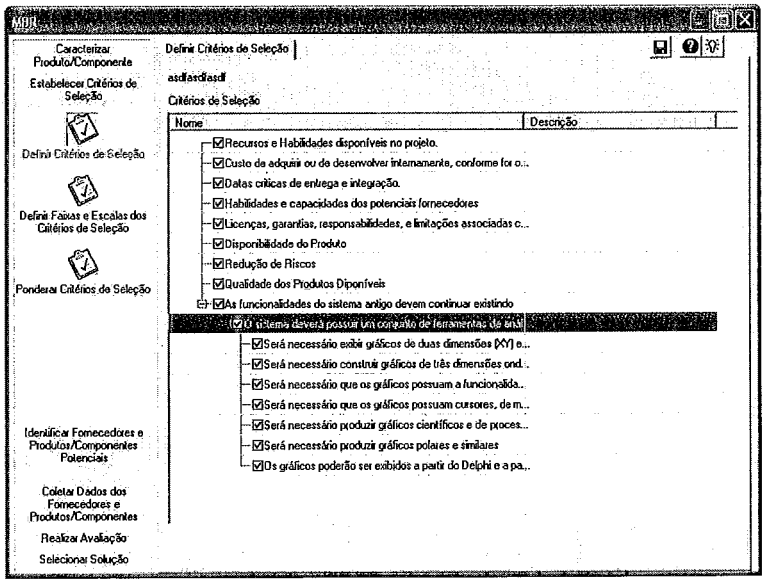


Figura 5.25 – Atividade "Definir Critérios de Seleção"

A macro-atividade "Identificar Fornecedores e Produtos\Componentes Potenciais" tem por objetivo identificar os produtos\componentes e os fornecedores que serão avaliados durante as próximas atividades. A figura 5.26 ilustra a atividade "Identificar Fornecedores e Componentes Externos", na qual são selecionadas as organizações que serão avaliadas como possíveis desenvolvedores do produto\componente e os produtos\componentes existentes no mercado que poderiam ser comprados. Após selecionar um fornecedor ou produto/componente na lista existente na parte inferior da tela, o projetista poderá inserir algumas informações sobre o mesmo na lista existente na parte inferior da tela.

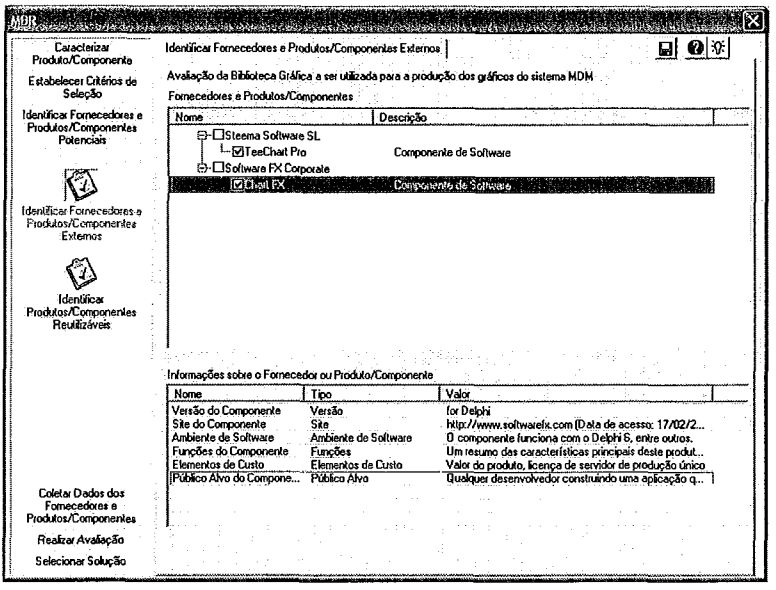


Figura 5.26 – Atividade "Identificar Fornecedores e Produtos\Componentes Externos"

Tendo selecionado os produtos\componentes existentes no mercado que poderiam ser comprados e as organizações existentes no mercado que poderiam desenvolver o produto/componente, o projetista deve identificar os produtos\componentes existentes na organização que poderiam ser reutilizados e, além disso, indicar se deseja avaliar a opção de desenvolvimento interno do produto/componente. A figura 5.27 exibe a tela referente a atividade "Identificar Produtos\Componentes Reutilizáveis", na qual essas tarefas são realizadas.

No exemplo apresentado não se selecionou nada nesta atividade, pois o desenvolvimento interno do componente desejado não era uma opção e a organização não possuía um componente que pudesse ser reutilizado a fim de satisfazer os requisitos estabelecidos. O desenvolvimento interno do componente não foi uma opção porque a organização possuía um prazo pequeno para a entrega do produto e a equipe de desenvolvimento não teria o tempo necessário para desenvolver o componente em questão.

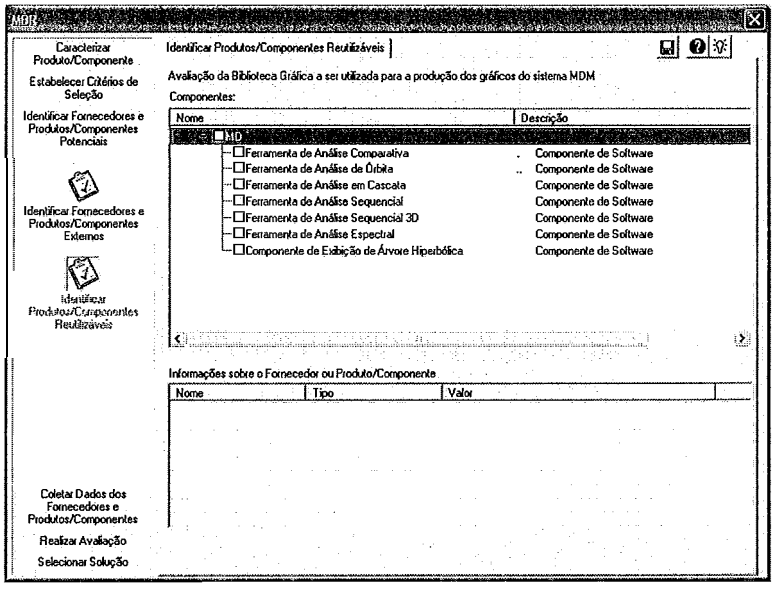


Figura 5.27 – Atividade "Identificar Produtos\Componentes Reutilizáveis"

A figura 5.28 corresponde à atividade de coleta dos dados dos fornecedores e dos produtos\componentes identificados nas atividades anteriores.

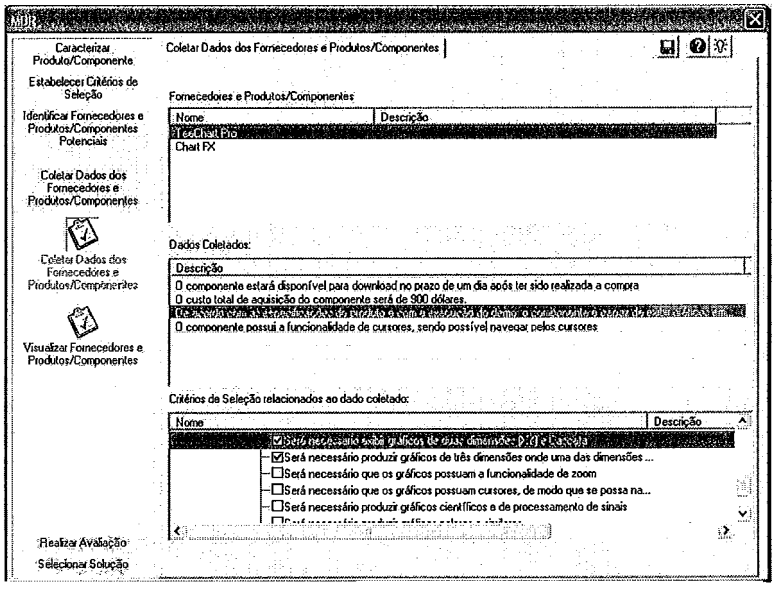


Figura 5.28 – Atividade "Coletar Dados dos Fornecedores e Produtos\Componentes"

Após selecionar um produto/componente ou fornecedor na lista existente na parte superior da tela, o projetista deve inserir na lista do meio os dados coletados em relação ao fornecedor ou produto/componente em questão e, depois disso, selecionar na lista localizada na parte inferior da tela os critérios de seleção que estão relacionados com o dado coletado. Ao finalizar esta atividade, pode-se visualizar as informações fornecidas através da atividade "Visualizar Fornecedores e Produtos\Componentes", representada pela figura 5.29.

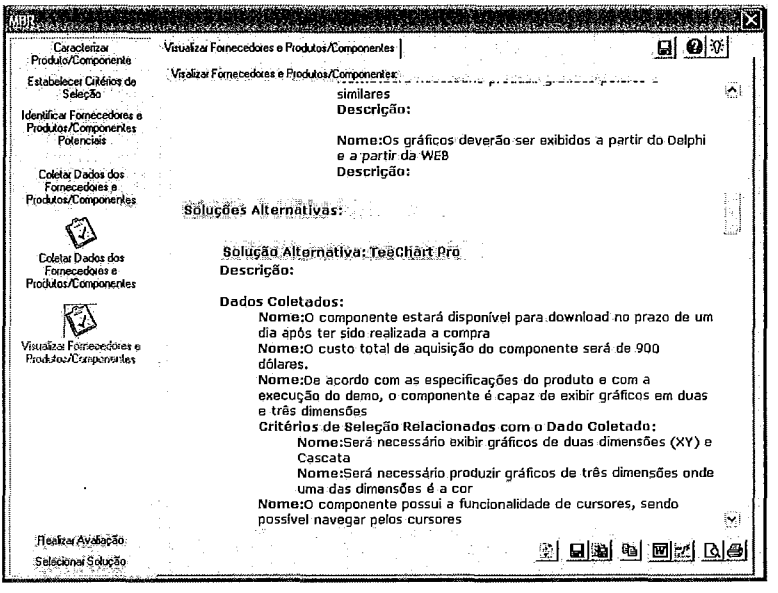


Figura 5.29 – Atividade "Visualizar Fornecedores e Produtos\Componentes"

A avaliação das opções em relação aos critérios estabelecidos ocorre da mesma forma que na ferramenta TechSolution. O projetista deve avaliar as soluções alternativas, neste caso os possíveis produtos\componentes e fornecedores, em relação aos critérios durante a atividade "Avaliar Soluções Alternativas em Relação aos Critérios", ilustrada pela figura 5.30. Após realizar a avaliação, deve-se visualizar o relatório da avaliação para que se possa selecionar a solução mais adequada. Assim como na ferramenta TechSolution, o relatório da avaliação sugere a seleção de uma das soluções. A atividade da ferramenta MBR que gera o relatório da avaliação é ilustrada pela figura 5.31.

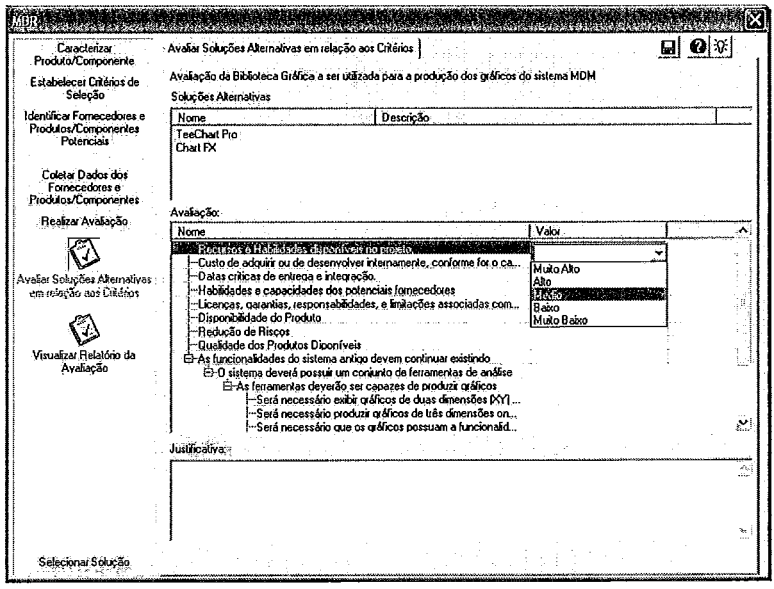


Figura 5.30 – Atividade "Avaliar Soluções Alternativas em relação aos Critérios"

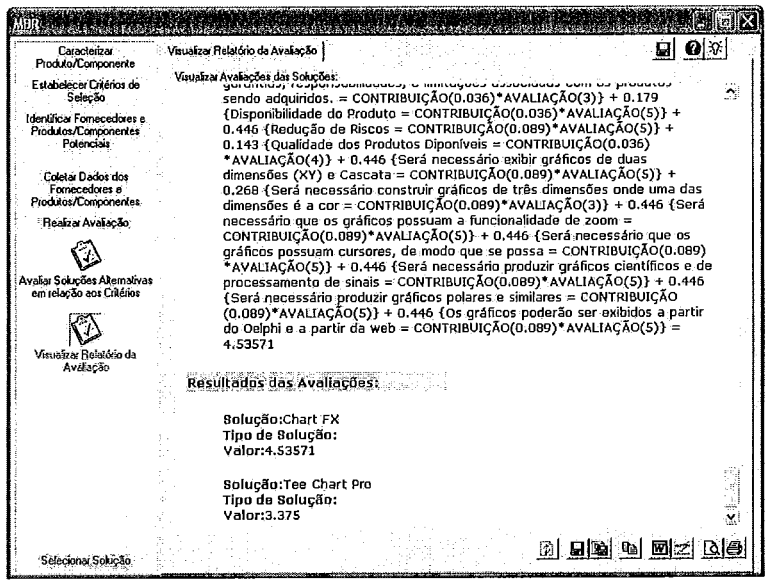


Figura 5.31 – Atividade "Visualizar Relatório da Avaliação"

Para selecionar a solução mais adequada, o projetista deve levar em consideração o resultado da avaliação, mas caso julgue necessário poderá re-executar alguma das atividades do processo. A atividade de seleção da solução é idêntica à atividade de seleção de solução da ferramenta TechSolution, sendo que no caso da ferramenta MBR, as opções

de solução são os produtos\componentes e os fornecedores avaliados. Após selecionar uma solução, deve-se justificar o porquê da escolha. Em seguida, pode-se gerar um relatório que descreve as atividades do processo decisório executado, assim como as informações fornecidas durante a execução do mesmo. A atividade na qual uma solução é selecionada é exibida pela figura 5.32 e a atividade na qual é gerado o relatório da avaliação é exibida na figura 5.33.

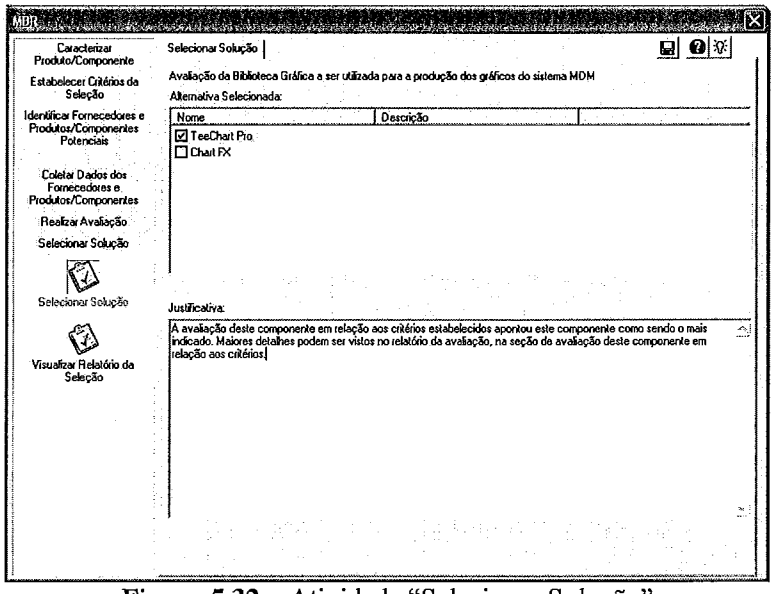


Figura 5.32 – Atividade "Selecionar Solução"

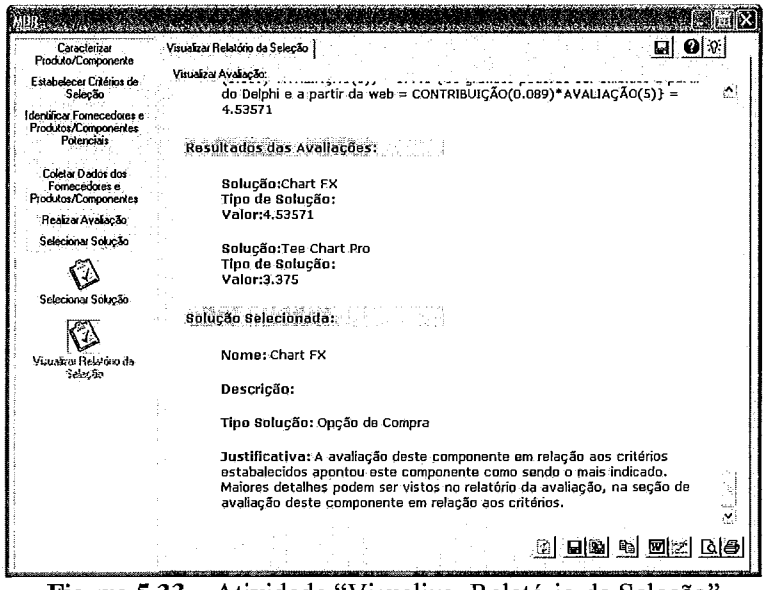


Figura 5.33 – Atividade "Visualizar Relatório da Seleção"

5.4 Interação das Ferramentas TechSolution e MBR com as Ferramentas da Estação TABA

As ferramentas TechSolution e MBR são executadas a partir dos ADSOrg. Para que possam ser utilizadas, é necessário selecioná-las para apoiar uma ou mais atividades do processo a ser executado em um determinado projeto. Esta seleção das ferramentas para um determinado processo ocorre durante a adaptação do processo padrão da organização para um projeto específico.

As ferramentas TechSolution e MBR possuem interfaces bem definidas para se comunicarem com certos serviços do ambiente instanciado e com determinadas ferramentas que também estão integradas ao mesmo. Desta forma, as ferramentas e serviços que se relacionam com as ferramentas TechSolution e MBR são:

- **AdaptPro**

O objetivo da ferramenta AdaptPro é apoiar o gerente do projeto na adaptação do processo padrão da organização para um projeto específico (BERGER, 2003). Durante a execução da ferramenta, é escolhido o modelo de ciclo de vida para o projeto e as atividades do processo padrão da organização são mapeadas em relação às fases do modelo de ciclo de vida selecionado. Caso necessário, o gerente de projeto pode incluir ou excluir atividades no processo que está sendo adaptado para o projeto. Além disso, também é necessário selecionar os artefatos que serão produzidos e consumidos pelas atividades do processo e indicar que ferramentas apoiarão a execução destas atividades. Neste sentido, o gerente deve indicar que atividades do processo do projeto serão apoiadas pelas ferramentas TechSolution e MBR. O resultado da execução do AdaptPro é a instanciação de um ADSOrg para um projeto específico, tal ADSOrg fornecerá suporte à execução das atividades do processo adaptado a partir do processo padrão da organização.

- **Tela de Controle dos Artefatos Produzidos e Consumidos**

A interface principal dos ADSOrg possui uma tela que possibilita a manipulação dos artefatos produzidos ou consumidos pelas atividades do processo. A figura 5.34 apresenta esta tela em destaque. Ao selecionar uma atividade na árvore existente no lado esquerdo da figura, os artefatos requeridos e os artefatos produzidos pela atividade selecionada são exibidos nas guias de artefatos da tela de controle dos artefatos.

Na parte inferior da tela que exibe os artefatos de acordo com a atividade selecionada existe uma barra de ferramentas com botões que possibilitam manipular o artefato selecionado. Os botões existentes antes da implementação da ferramenta TechSolution permitiam ao usuário: (i) criar um novo artefato a partir de um *template*, (ii) importar o artefato da área de trabalho do usuário para a Estação Taba, (iii) exportar o usuário da Estação Taba para a área de trabalho do usuário, (iv) abrir o artefato para edição, (v) excluir o artefato da Estação Taba, (vii) enviar o artefato por e-mail, (viii) bloquear o artefato para edição e (ix) desbloquear um artefato que houvesse sido bloqueado para edição.

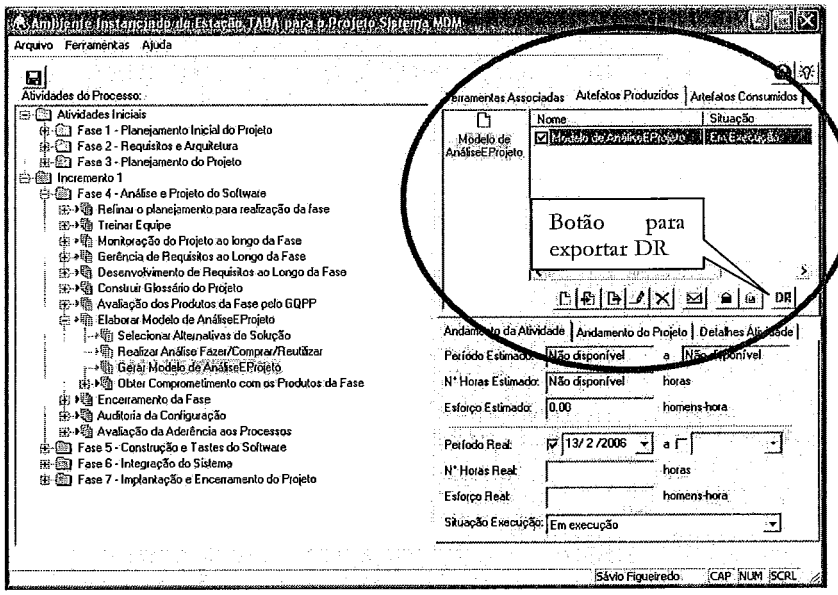


Figura 5.34 – Tela Principal do Ambiente Instanciado com destaque para a interface de manipulação de artefatos

Para permitir ao usuário da Estação Taba uma rápida consulta ao *Design Rationale* associado a um determinado artefato, foi inserido um novo botão na barra de ferramentas representado pelo símbolo  . Este botão permanece desabilitado enquanto não houver *Design Rationale* para o artefato selecionado. A partir do momento em que o usuário registrar um *Design Rationale* para o artefato em questão através da ferramenta *TechSolution*, o botão poderá ser utilizado para consultar o *Design Rationale* existente sobre aquele artefato. Assim, ao clicar no botão, será exibido um relatório descrevendo o *Design Rationale* relativo ao artefato selecionado.

- **ReqManager**

A ferramenta ReqManager, presente nos ADSOrg, tem como objetivo apoiar a gerência de requisitos. Para isto, a ferramenta ajuda o gerente na elaboração da matriz de rastreabilidade e na navegação bi-direcional entre os diversos níveis de requisitos e os artefatos do projeto, como por exemplo, o projeto dos componentes e os casos de teste.

Durante o processo de tomada de decisão apoiado pela ferramenta TechSolution, é necessário mapear as soluções alternativas definidas em relação aos requisitos. Assim, é possível visualizar os requisitos satisfeitos por uma determinada solução alternativa e os requisitos não estão sendo considerados por esta solução. Estes requisitos foram identificados através da ferramenta ReqManager.

Em relação à ferramenta MBR, que tem como objetivo apoiar a tomada de decisão entre desenvolver, comprar ou reutilizar um determinado componente, os requisitos são utilizados para definir o componente sobre o qual a análise será realizada. A ferramenta MBR exibirá para o projetista os requisitos identificados na ferramenta ReqManager e o projetista deve selecionar os requisitos que o componente a ser produzido, comprado ou reutilizado deverá satisfazer.

- **Acknowledge**

A ferramenta Acknowledge permite o registro e a consulta do conhecimento associado com as atividades do processo de desenvolvimento/manutenção a fim de possibilitar a reutilização do conhecimento organizacional (MONTONI, M., 2003). A figura 5.35 exibe a interface principal da ferramenta TechSolution. Os ícones existentes na parte superior direita da figura 5.35 representam a interface das ferramentas TechSolution com a ferramenta Acknowledge.

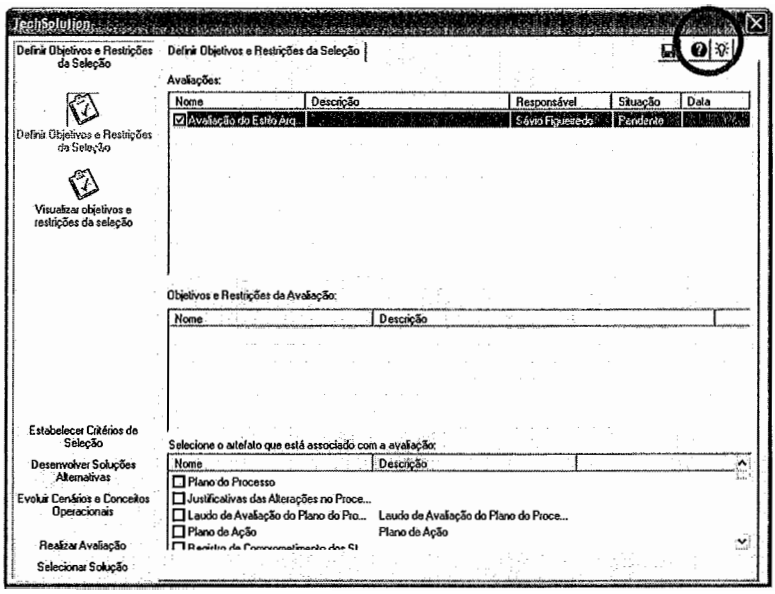


Figura 5.35 – Interface Principal da Ferramenta TechSolution destacando os botões que executam a ferramenta Acknowledge

O botão com o ícone  permite que o usuário da ferramenta registre um conhecimento em relação à atividade do processo da ferramenta que está sendo executada no momento. O usuário poderá registrar por exemplo uma idéia, uma lição aprendida ou uma dúvida. Após passar por um processo de aquisição do conhecimento, no qual este conhecimento será filtrado e empacotado pelos gerentes de conhecimento da organização, o conhecimento poderá se tornar disponível para consulta por outros usuários da ferramenta que estiverem executando a atividade. Desta forma, a pessoa que desejar ter acesso ao conhecimento disponível na organização sobre aquela atividade da ferramenta poderá consultá-lo através do botão com o ícone  .

- **Sapiens**

O objetivo da ferramenta Sapiens é permitir a descrição e visualização das estruturas organizacionais, englobando os profissionais alocados e as competências requeridas e possuídas ao longo dessas estruturas (SANTOS *et al.*, 2004). Esta ferramenta pode ser executada a partir do Meta-Ambiente, do Ambiente Configurado ou então do ADSOrg. A figura 5.36 apresenta a interface principal da ferramenta Sapiens.

Desta forma, a ferramenta Sapiens possibilita o cadastro das organizações clientes e fornecedoras de uma determinada organização e dos produtos que estas organizações produzem. Além disso, também é possível identificar os produtos que a organização que está desenvolvendo o projeto possui. Durante a execução da atividade de identificação dos fornecedores e produtos\componentes externos na ferramenta MBR, a ferramenta apresenta para o usuário uma lista de fornecedores e de produtos\componentes que estes fornecedores possuem. De forma análoga, durante a execução da atividade de identificação dos produtos\componentes que poderiam ser reutilizados, a ferramenta apresenta para o usuário a lista de produtos que a organização que está desenvolvendo o projeto possui. Estas informações representam os dados registrados através da ferramenta Sapiens.

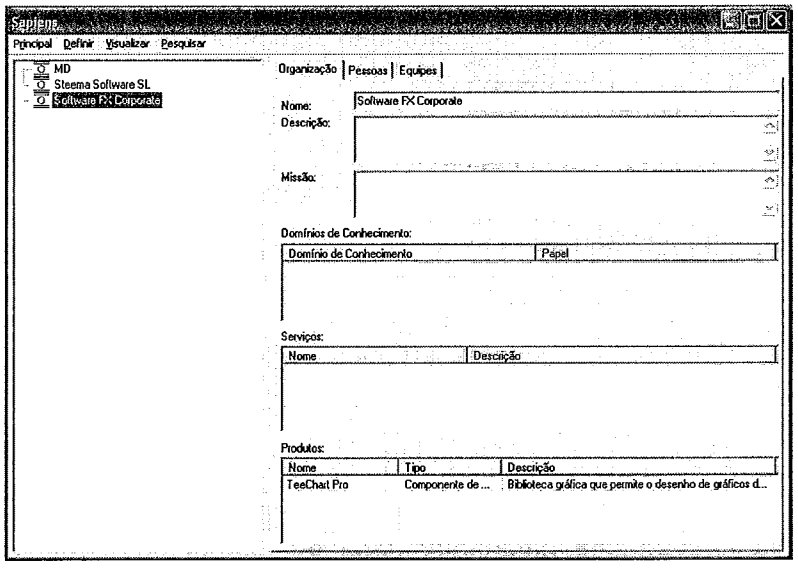


Figura 5.36 – Interface Principal da Ferramenta Sapiens exibindo o cadastro de um componente produzido por um fornecedor.

- **VerificationManager**

Alguns produtos da Solução Técnica, como o projeto, devem ser verificados em relação aos requisitos pré-estabelecidos. Esta verificação é apoiada nos ADSOrgs pela ferramenta VerificationManager que apóia o planejamento e a execução das atividades de verificação ao longo do processo de desenvolvimento.

- **ValidationManager**

A ferramenta ValidationManager foi definida e implementada como um apoio ferramental para a execução da atividade de validação na Estação Taba. Esta ferramenta apóia o planejamento da validação, desde a identificação de quais artefatos devem ser validados até a definição de um plano completo para a realização da validação no projeto.

5.5 Considerações Finais

Este capítulo apresentou a abordagem definida para apoiar a Solução Técnica em projetos de Software. Foram descritas as atividades do processo definido com o objetivo de fornecer suporte à seleção de alternativas de projeto e as atividades do processo definido com o objetivo de fornecer suporte à decisão de fazer, comprar ou reutilizar um determinado componente do produto. As ferramentas TechSolution e MBR, implementadas para apoiar, respectivamente, os processos descritos também foram descritas. Em seguida, foram especificados os serviços e as ferramentas da Estação TABA com os quais as ferramentas TechSolution e MBR interagem.

O próximo capítulo apresentará a conclusão desta dissertação, além de discutir as contribuições deste trabalho e destacar aspectos que poderiam ser evoluídos.

CAPÍTULO 6 – CONCLUSÃO

6.1 Introdução

Esta dissertação apresentou uma abordagem para apoio ao processo de Solução Técnica. Para atingir este objetivo foi realizada uma revisão da literatura técnica sobre os conceitos relacionados a processos de software, destacando a interseção do processo de Solução Técnica com o processo de desenvolvimento. A revisão da literatura também englobou abordou Ambientes de Desenvolvimento de Software, a Estação TABA e suas ferramentas e *Design Rationale*.

Em seguida, foram descritos os processos definidos para apoiar a Solução Técnica em projetos de software. O primeiro processo tem como objetivo a seleção de alternativas de projeto e o segundo processo apóia a escolha entre fazer, comprar ou reutilizar um determinado componente do produto a ser desenvolvido. Finalmente foram descritas as duas ferramentas implementadas para apoiar a execução destes processos: TechSolution e MBR.

Este capítulo apresenta as conclusões desta dissertação, destacando as contribuições do trabalho e as perspectivas futuras para continuidade do mesmo, indicando aspectos que podem ser aprofundados.

6.2 Conclusão e Contribuições

O processo de Solução Técnica é executado durante o desenvolvimento de um produto e tem início quando os requisitos do software estiverem definidos e aprovados. O objetivo do processo de Solução Técnica é projetar e implementar uma solução para os requisitos em questão.

Nesta dissertação foi descrita uma abordagem para apoiar a Solução Técnica em projetos de software. Esta abordagem apóia os processos de seleção de alternativas de projeto e de escolha entre fazer, comprar ou reutilizar um determinado componente do produto. A abordagem está baseada, principalmente, nas áreas de processo Solução Técnica e Análise de Decisão e Resolução do CMMI, nos processos de Solução Técnica e de Análise de Decisão e Resolução do MPS.BR e na norma ISO/IEC 14102 e em outras abordagens existentes na literatura (BURGE, 2005a) (FRANCISCO, 2004) (XAVIER, 2001).

Neste momento três empresas já estão utilizando a abordagem e as ferramentas propostas nesta dissertação com o intuito de apoiar a implementação do processo de Solução Técnica em seus projetos. Estas empresas visam a obtenção do Nível de Maturidade C do MPS.BR e do Nível 3 do CMMI. Este contato com a indústria de software traz benefícios para ambos os lados, pois, pelo lado do meio acadêmico, avaliações da utilização das abordagens propostas permitem calibrar as abordagens para fornecer um melhor apoio às organizações e, pelo lado da indústria, facilita o crescimento da maturidade das organizações no uso dos processos, através do apoio ferramental.

Como fruto deste trabalho houve a publicação de um artigo no Workshop de Manutenção de Software Moderna realizado em novembro de 2005 na cidade de Manaus-AM. O artigo destacava a importância da utilização do DR, gerado e mantido pela ferramenta TechSolution nos projetos de manutenção de software (FIGUEIREDO *et al.*, 2005).

Dentre as contribuições deste trabalho, podemos destacar:

- Definição de um processo de seleção de alternativas de projeto;
- Definição de um processo de análise entre fazer, comprar ou reutilizar um componente do produto.
- Definição e implementação da Ferramenta TechSolution para apoiar o processo de seleção de alternativas de projeto;

- Definição e implementação da ferramenta MBR para apoiar o processo de análise entre fazer, comprar ou reutilizar um componente do produto.
- Integração das ferramentas implementadas às ferramentas da Estação TABA e aos ambientes instanciados.
- A identificação e organização do conhecimento na literatura para apoiar a seleção do estilo arquitetural mais adequado para um projeto de software e a sua inserção na Estação TABA.
- A extensão do modelo de dados da Estação TABA para armazenar o DR dos projetos.

6.3 Perspectivas Futuras

A utilização da ferramenta por empresas que estão objetivando o alcance do nível de maturidade C do MPS.BR e do Nível 3 do CMMI torna possível realizar uma avaliação das ferramentas e dos processos definidos para obter um *feedback* sobre estes e permitir a sua melhoria através das questões levantadas pelas empresas. Esta avaliação estará inserida em um contexto mais geral, onde serão avaliadas todas as ferramentas TABA e será realizada como parte de uma tese de doutorado.

Neste momento, já podem ser percebidos como oportunidades de melhoria nas abordagens:

- Permitir a execução das ferramentas MBR a partir do Ambiente Configurado, apoiandoa decisão de compra, desenvolvimento ou reutilização de um determinado componente pela organização, e não apenas no contexto de um projeto específico.
- Evoluir o processo de escolha de alternativas de projeto e a ferramenta TechSolution para que esta possa apoiar o processo de Análise de Decisão e Resolução.

- Implementar na ferramenta TechSolution mecanismos de pesquisa em relação ao DR existente para um determinado artefato ou projeto.
- Alterar a avaliação das soluções alternativas na ferramenta TechSolution e a avaliação entre as opções de fazer, comprar ou reutilizar um determinado componente na ferramenta MBR para possibilitar a avaliação por mais de uma pessoa.
- Evoluir as listas de critérios de seleção sugeridas pelas ferramentas TechSolution e MBR.
- Inserir na ferramenta TechSolution conhecimento sobre outros tipos de decisões que geralmente são tomadas durante a Solução Técnica, além da escolha do estilo arquitetural, tais como a escolha de táticas de projeto[4].

[4]Uma tática é uma decisão do *design* que influencia no controle da resposta de um atributo de qualidade

REFERÊNCIAS BIBLIOGRÁFICAS

ANDRADE, J.M.S., 2005, *Avaliação de Processos de Software em ADSOrg*, Dissertação de M. Sc., COPPE/UFRJ, Rio de Janeiro, Brasil.

ANTONIOL, G., GRADARA, S., VENTURI, G., 2004, Methodological issues in a CMM Level 4 implementation. Software Process: Improvement and Practice 9(1), p. 33-50.

ARTEMIS 7, 2006, http://www.aisc.com/Product/1, acessado em jan/2006.

BANDINELLI, S., DI NITTO, E., FUGGETTA, A., 1996, Supporting Cooperation in the SPADE-1 Environment. IEEE Trans. on Software Engineering, Vol. 22, No. 12, pp. 841-865.

BARCELLOS, M.P., 2003, Planejamento de Custos em Ambientes de Desenvolvimento de Software Orientados à Organização, Dissertação de M. Sc., COPPE/UFRJ, Rio de Janeiro, Brasil.

BARCELLOS, M. P., FIGUEIREDO, S. M., ROCHA, A.R.C., TRAVASSOS, G.H., 2003, "Utilização de Métodos Paramétricos, Analogias, Julgamento de Especialistas e Conhecimento Organizacional no Planejamento de Tempo e Custos de Projetos de Software", *In: Anais do II Simpósio Brasileiro de Qualidade de Software*, pp. 17-31, Fortaleza, Brasil.

BASS, L., CLEMENTS, P., KAZMAN, R., 2003, *Software Architecture in Practice*. Addison-Wesley, 2 edition.

BEN-SHAUL, I.Z., SKOPP, P.D., HEINEMAN, G.T., TONG, A.Z., POPOVICH, S.S., VALETTO, G., 1994, Integrating groupware and process technologies in the Oz environment. In: *Proc. of the 9th Int. Software Process Workshop*, pp.: 114–116, 5-7 Oct.

BERGER, P., 2003, Instanciação de Processos de Software em Ambientes Configurados na Estação TABA, Dissertação de M. Sc., COPPE/UFRJ, Rio de Janeiro, Brasil.

BOEHM, B., BOSE, P., 1994, A Collaborative Spiral Software Process Model Based on Theory W, Third International Conference on the Software Process, Reston, VA, pp. 59-68.

BOSE, P.: 1995, A Model for Decision Maintenance in the WinWin Collaboration Framework, Knowledge Based Software Engineering (KBSE '95), pp. 105-113.

BOWEN, S., MAURER, F., 2002, Process support and knowledge management for virtual teams doing agile software development. In.: *Proc. of the 26th Annual Int. Computer Software and Applications Conference (COMPSAC)* pp:1118–1120, 26-29 Aug.

BOUCHRIHA, H., D´AMOURS, S., LADET, P., 2002, A "make or buy" decision model with economies of scale, Article de conférence avec actes, 2002.

BUCKINGHAM-SHUM, S. J., HAMMOND, N., 1994, Argumentation based design rationale: What use at what cost? Human-Computer Studies, 40(4):603–652, April 1994.

BROWN, D. C., BANSAL, R.: 1991, Using Design History Systems for Technology Transfer, in Computer Aided Cooperative Product Development, D. Sriram, R. Logcher and S. Fukuda (Eds.), Lecture Notes Series, No. 492, Springer-Verlag, New York, pp. 544-559.

BURGE, J. E., 2005a, "Software Engineering Using design RATionale", PhD Dissertation, CS Dept., WPI, May.

BURGE, J. E., 2005b, Design Rationale Types and Tools, Disponível em http://web.cs.wpi.edu/Research/aidg/DR-Rpt98.html, acessado em 12/2005.

BURGE, J. E., BROWN, D.C., 2000, "Reasoning with Design Rationale", *Artificial Intelligence in Design '00*, J. Gero (ed.), Kluwer Academic Publishers, Netherlands, 2000, pp. 611-629.

BURGE, J. E., BROWN, D. C., 2001, Design Rationale for Software Maintenance, Proceedings of the 16th IEEE International Conference on Automated Software Engineering, p.433, November 26-29.

BURGE, J. E., BROWN, D. C., 2002, "Discovering a Research Agenda for Using Design Rationale in Software Maintenance", Computer Science Technical Report, Worcester Polytechnic University, WPI-CS-TR-02-03.

CÁNEZ, L., PROBERT, D., PLATTS, K., 2001, Testing a Make-or-Buy Process, *Proceedings of the Twelfth Annual Conference of the Production and Operations Management Society*, POM-2001, March 30 – April 2, Orlando.

CHEN, A., MCGINNIS, B., ULLMAN, D., DIETTERICH, T., 1990, *Design History Knowledge Representation and Its Basic Computer Implementation,* The 2nd International Conference on Design Theory and Methodology, ASME, Chicago, IL, pp. 175-185.

CHRISSIS, M. B., KONRAD, M., SHRUM, S., 2003, *CMMI: Guidelines for Process Integration and Product Improvement*, Addison-Wesley, 2003.

CONKLIN, J., 1989, "Design Rationale and Maintainability". Vol. II: Software Track, *Proceedings of the Twenty-Second Annual Hawaii International Conference* on Volume 2, 3-6 Jan. Page(s): 533-539. Vol. 2 Digital Object Identifier 10.11 09/HICSS. 1989.48049.

CONKLIN, J., BEGEMAN, M. L., 1988, gIBIS: A hypertext tool for exploratory policy discussion. ACM Transactions on Office Information Systems, v. 6, n. 4, pp. 303-331.

CONKLIN, J., BURGESS-YAKEMOVIC, K., 1995, A Process-Oriented Approach to Design Rationale, In Design Rationale Concepts, Techniques, and Use, T. Moran and J. Carrol, (eds), Lawrence Erlbaum Associates, Mahwah, NJ, pp. 293-428.

DEBOU, C., KUNTZMANN-COMBELLES, A., 2000, Linking software process improvement to business strategies: experiences from industry, Software Process: Improvement and Practice 5(1), p. 55-64.

DELLEN, B., KOHLER, K., MAURER, F.: 1996, Integrating Software Process Models and Design Rationales, In: *Proceedings Knowledge-based Software Engineering*, Syracuse, NY, IEEE Computer Society Press, pp. 84-93.

DERMIANE, J. C., KABA B. A., WASTELL, D., 1999, "Software Process: Principles, Methodology and Technology". Lecture Notes in Computer Science 1500. Springer.

ECLIPSE, 2006, http://www.eclipse.org/, acessado em jan/2006.

FARIAS, L., 2002, Planejamento de Riscos em Ambientes de Desenvolvimento de Software Orientados à Organização, Dissertação de M. Sc., COPPE/UFRJ, Rio de Janeiro, Brasil.

FARIAS, L. *et al.*, 2003, *RiscManager: uma Ferramenta par Planejamento de Riscos com apoio de Gerência do Conhecimento*, I Workshop Tecnologias da Informação e Gerência do Conhecimento, Fortaleza, Brasil.

FERREIRA, A, I, F., CERQUEIRA, R., SANTOS, G., 2005, Implementando MPS BR Nível F como preparação para certificação CMMI Nível 3, In: I Encontro de Implementadores de MPS.BR, pp. -, Brasília, Brasil, Junho.

FIGUEIREDO, S., 2004, Gerência de Configuração em Ambientes de Desenvolvimento de Software Orientados à Organização, Monografia de Final de Curso de B.Sc., UFRJ, Rio de Janeiro, Brasil.

FIGUEIREDO, S., ROCHA, A.R., SANTOS, G., MONTONI, M., NATALI, A.C., 2005, "Apoio à Manutenção de Software através de Design Rationale em Ambientes de Manutenção de Software Taba" , *II Workshop de Manutenção de Software Moderna (WMSWM-05)*, Manaus, Brasil.

FISCHER, G., LEMKE, A., MCCALL, R., MORCH, A., 1995, Making Argumentation Serve Design, in Design Rationale Concepts, Techniques, and Use, T. Moran and J. Carroll, (eds), Lawrence Erlbaum Associates, pp. 267-294.

FRANCISCO, S. D., 2004, *DocRationale* - uma ferramenta para suporte a Design Rationale de artefatos de Software. São Carlos-SP, março de 2004. 123p. Dissertação de Mestrado. Instituto de Ciências Matemáticas e de Computação de São Carlos, Universidade de São Paulo.

FUGGETA, A., 2000, Software Process: A Roadmap. In: *22nd International Conference on the Future of Software Engineering*, Limerick, Ireland: ACM-Association for Computing Machinery, p. 25-34.

GALOTTA, C., 2000, Netuno: Um Ambiente de Desenvolvimento de Software Orientado ao Domínio de Acústica Submarina, Dissertação de M. Sc., COPPE/UFRJ, Rio de Janeiro, Brasil.

GARCIA, A. C. B., SOUZA, C. S., 1997, Add+: Including rhetorical structures in active documents. Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 11(2):109–124, April 1997.

GOLUBIÉ, S., 2005, Influence of Software Development Process Capability on Product Quality, In: *8th International Conference on Telecommunications*, Zagreb, Croatia, June 5-17.

GOMES, A., 2001, Avaliação de processos de software baseada em medições, Tese de MSc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, Jul.

GRUBER, T. R., RUSSELL, D. M., 1990, Design Knowledge and Design Rationale: A Framework for Representation, Capture, and Use. Technical Report, Knowledge Systems Laboratory, Stanford University, July 27, 1990.

GRUBER, T., 1990, Model-based Explanation of Design Rationale, In: *Proceedings of the AAAI-90 Explanation Workshop*, Boston, July 30, 1990.

GRUDIN, J., 1995, Evaluating Opportunities for Design Capture, In: *T Moran and J Carroll (Eds), Design Rationale Concepts, Techniques, and Use,* Lawrence Erlbaum Associates, NJ, pp. 453-470.

HU, X., PANG, J., PANG, Y., ANTWOOD, M., SUN, W., REGLI, C. W., 2000, A survey on design rationale: Representation, capture and retrieval. Engineering with Computers: An Int'l Jour. for Simulation-Based Engineering, v. 16, p. 209-235.

IBRAHIM, L., PYSTER, A., 2004, A Single Model for Process Improvement, Lessons Learned at the US FAA, IEEE IT-Pro, Vol. 6, No. 3, May/June.

ISO/IEC 12207, 2000, Information Technology – Software Lyfe-Cycle Processes.

ISO/IEC PDAM 12207, 2002, *ISO/IEC 12207 Information Technology - Amendment to ISO/IEC 12207*, INTERNATIONAL STANDARD ORGANIZATION, Montreal: ISO/IEC JTC1 SC7.

ISO/IEC PDAM 12207, 2004, *ISO/IEC 12207 Information Technology - Amendment to ISO/IEC 12207*, INTERNATIONAL STANDARD ORGANIZATION, Montreal: ISO/IEC JTC1 SC7.

ISO/IEC 14102, 1995, Information technology - Guideline for the evaluation and selection of CASE tools.

ISO/IEC 15504, 2004, *Information Technology – Software Process Assessment,* Parts 1-5, International Organization for Standardization and the International Electrotechnical Commission, Geneva, Switzerland.

ISO/IEC 15504 –1, 2004, Information Technology – Process Assessment, - Part 1: Concepts and Vocabulary

ISO/IEC 15504 –2, 2004, Software Engineering – Process Assessment – Part 2: Performing an Assessment

ISO/IEC 15504 –3, 2004, Software Engineering – Process Assessment – Part 3: Guidance on Performing an Assessment.

ISO/IEC 15504 –4, 2004, Software Engineering – Process Assessment – Part 4: Guidance on use for process improvement and process capability determination.

ISO/IEC 15504 –5, 2004, Software Engineering – Process Assessment – Part 5: An Exemplar Process Assessment Model.

ISO/IEC 9126, 2001, *Software engineering - Product quality - Part 1: Quality model.*

KAWALEK, P., WASTELL, D. G., 1996 "Organisational design for software development: a cybernetic perspective" In: *Lecture Notes in Computer Science. Proc. of the 5th European Workshop on Software Process Technology* (EWSPT'96). Springer-Verlag. pp. 258-270.

KLEIN, M.: 1992, DRCS: An Integrated System for Capture of Designs and Their Rationale, In: *Artificial Intelligence in Design '92, Gero*, J. (ed.), Kluwer Academic Publishers, pp. 393-412.

KOMIYAMA, T., SUNAZUKA, T., KOYAMA, S., 2000, Software process assessment and improvement in NEC - current status and future direction, Software Process: Improvement and Practice 5(1), p. 31-44.

LEE, J., 1990, SIBYL: a tool for managing design rationale. In: *Proceedings o fthe Conference on Computer Supported Cooperative Work*, Los Angeles, EUA, p. 79-92.

LEE, J., 1997, Design Rationale Systems: Understanding the issues. *IEEE expert/Intelligent Systems and Their Applications*, v. 12, n. 3, p. 78-85.

LEE, J., LAI, K., 1991, What's in design rationale. Human-Computer Interaction, 6(3-4): 251-280.

MARTINS, F., 2004, Ambientes de Desenvolvimento de Software Orientado à Organização Baseado em Instrumentação Virtual, Dissertação de M.Sc., COPPE/UFRJ, Rio de Janeiro, Brasil.

MCCALL, R., 1991, PHI: A Conceptual Foundation for Design Hypermedia. Design Studies 12, pp.30-41.

MCLEAN, A., YOUNG, R., BELLOTI, V., MORAN, T., 1991, Question, options, and criteria: Elements of design space analysis. Human-Computer Interaction, v.6, n. 3-4, p. 210-250.

MCLEAN, A., YOUNG, R., BELLOTI, V., MORAN, T., 1995, Questions, Options and Criteria: Elements of Design Space Analysis, in *Design Rationale Concepts, Techniques, and Use*, T. Moran and J. Carroll (Eds.), Lawrence Erlbaum Associates, NJ, 1995, pp. 201-251.

MINH, N. N., WANG, A.I., CONRADI, R., 1997, Total Software Process Model Evolution in EPOS Experience Report. In: *Proc. of the 19th Int. Conf. on Software Engineering*, pp: 390–399, May 17-23.

MONTONI, M., 2003, *Aquisição de Conhecimento no Desenvolvimento de Software*, Tese de M. Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

MONTONI, M., MIRANDA, R., ROCHA, A.R., TRAVASSOS, G.H., 2004, "Knowledge Acquisition and Communities of Practice: an Approach to Convert Individual Knowledge into Multi-organizational Knowledge", *In: Proceedings of the LSO 2004*, pp. 110-121, Banff, Canadá.

MONTONI, M., SANTOS, G., VILLELA, K., ROCHA, A.R., TRAVASSOS, G., FIGUEIREDO, S., MAFRA, S., ALBUQUERQUE, A., MIAN, P., 2005, "Enterprise-Oriented Software Development Environments to Support Software Products and Processes Quality Improvement", *In: Proceedings of the PROFES 2005*, pp.370-384, Oulu, Finlândia.

MOREAU, B., LASSUDRIE, C., NICOLAS, B., HOMME, O., ANTERROCHES, C., GALL, G., 2003, Software Process: Improvement and Practice 8(3), p. 135-144.

MOURA, L, M, V., ROCHA, A. R . C., 1992, *Ambientes de Desenvolvimento de Software*, Publicações Técnicas COPPE/UFRJ, ES 271/92, Rio de Janeiro, Brasil.

MPS.BR, 2005a, MPS.BR - Melhoria de Processo do Software Brasileiro, Guia Geral (v. 1.0).

MPS.BR, 2005b, MPS.BR - Melhoria de Processo do Software Brasileiro, Guia de Aquisição (v. 1.0).

OLIVEIRA, K., 1999, Modelo para Constrição de Ambientes de Desenvolvimento de Software Orientados a Domínio, Tese de D. Sc., COPPE/UFRJ, Rio de Janeiro, Brasil.

OLIVEIRA, K.M., SANTOS, G., ZLOT, F., ROCHA, A.R.C., 2000, *A Estação TABA e Ambientes de Desenvolvimento de Software Orientados a Domínio*, Caderno de Ferramentas - XIV Simposio Brasileiro de Engenharia de Software, pp. 343-346, João Pessoa, Brasil.

PAULK, M., CURTIS, B., CHRISSIS, M., WEBER, C., 1993, Capability maturity model for software, version 1.1. Technical Report CMU/SEI-93-TR-24, Software Engineering Institute, Carnegie Mellon University: Pittsburgh, PA.

PENA-MORA, F., SRIRAM, D., LOGCHER, R., 1995, Design Rationale for Computer-Supported Conflict Mitigation, ASCE Journal of Computing in Civil Engineering, pp. 57-72.

PFLEEGER, S, L., 2004, "*Engenharia de Software: Teoria e Prática*", Prentice Hall, 2ª Edição.

POLTROCK, S. E., GRUDIN, J., 1999, Cscw, groupware and workflow: Experiences, state of the art, and future trends. ACM SIGCHI Tutorial, Pittsburgh, PA, April.

PRESSMAN, R. S., 2001, *"Software Engineering: A Practitioner's Approach"*, McGraw-Hill International Editions, 5th ed.

RAMESH, B., SENGUPTA, K., 1995, Multimedia in a design rationale decision support system. Decision Support Systems, 15(3):181–196, November 1995.

ROCHA, A, R., MONTONI, M., SANTOS, G., OLIVEIRA, K., NATALI, A, C., MIAN, P., CONTE, T., MAFRA, S., BARRETO, A., ALBUQUERQUE, A., FIGUEIREDO, S., SOARES, A., BIANCHI, F., CABRAL, R., DIAS, A., 2005, Dificuldades e Fatores de Sucesso na Implementação de Processos de Software Utilizando o MR-MPS e o CMMI, In: *I Encontro de Implementadores de MPS.BR*, pp. 40-47, Brasília, Brasil, Junho.

SANTOS, G., 2003, Representação da Distribuição do Conhecimento, Habilidades e Experiências através da Estrutura Organizacional, Dissertação de M. Sc., COPPE/UFRJ, Rio de Janeiro, Brasil.

SANTOS, G. *et al.*, 2003, *SAPIENS: uma Ferramenta para Descrição e Recuperação de Competências em uma Organização*, I Workshop Tecnologias da Informação e Gerência do Conhecimento, Fortaleza, Brasil.

SANTOS, G., MONTONI, M., ROCHA, A.R., FIGUEIREDO, S., MAFRA, S., ALBUQUERQUE, A., PARET, B.D., AMARAL, M., 2005, "Using a Software

Development Environment with Knowledge Management to Support Deploying Software Processes in Small and Medium Size Companies", *In: Proceedings of the LSO 2005*, pp. 72-76, Kaiserslautern, Alemanha.

SANTOS, G., VILLELA, K., SCHNAIDER, L., ROCHA, A.R.C., TRAVASSOS, G.H., 2004, "Building Ontology Based Tools for a Software Development Environment", *In : Proceedings of the LSO 2004*, pp. 19-30, Banff, Canadá.

SCHNAIDER, L., 2003, Planejamento da Alocação de Recursos Humanos em Ambientes de Desenvolvimento, de Software Orientados à Organização, Dissertação de M. Sc., COPPE/UFRJ, Rio de Janeiro, Brasil.

SCHNAIDER, L., SANTOS, G., MONTONI, M., ROCHA, A.R.C., 2004, "MedPlan: uma Abordagem para Medição e Análise em Projetos de Desenvolvimento de Software", in Anais do III Simpósio Brasileiro de Qualidade de Software, Brasília, Brasil.

SHAW, M., GARLAN, D., 1996, "Software Architectures--Perspectives on an Emerging" Discipline, Prentice Hall.

SHEARD, S, A., 1997, The Frameworks Quagmire, A Brief Look, Technical Report, Software Productivity Consortium.

SHIPMAN, F., MCCALL, R., 1997, Integrating Different Perspectives on Design Rationale: Supporting the Emergence of Design Rationale from Design Communication. Artificial Intelligence in Engineering Design, Analysis, and Manufacturing, 11(2):141–154.

SOUZA, A, S., OLIVEIRA, J, L., 2005, Experiências de Implantação de Processo de Software em Goiás, In: *I Encontro de Implementadores de MPS.BR*, pp. 16-24, Brasília, Brasil, Junho

SOUZA, C, R. B., WAINER, J., RUBIRA, C. M., R., 1997, An annotation model for the cooperative software development. In *III Workshop on Hypermedia and Multimedia Applications*, pages 143-154, São Carlos, Brazil, 23-25 May 1997.

SOUZA, C, R. B., WAINER, J., SANTOS, D. B., DIAS, K. L., 1998, A model and tool for semi-automatic recording of design rationale in software diagrams. In: *Proceedings of the 6th String Processing and Information Retrieval Symposium & 5th International Workshop on Groupware*, Cancun, Mexico, 1998, p. 306-313.

SWEBOK, 2004, The Software Engineering Process, In: *Guide to the Software Engineering Body of Knowledge*, Chapter 9, IEEE, Computer Society Press.

TUFFLEY, A., GROVE, B., MCNAIR, G., 2004, SPICE for Small Organisations. Software Process: Improvement and Practice 9(1), p. 23-31.

VIANA, P, W, P., VASCONCELLOS, J, F, A., 2005, Implantação e Certificação no Modelo de Referência MPS.BR no nível G em uma empresa do Pólo de Software AmazonSoft, In: *I Encontro de Implementadores de MPS.BR*, pp. 9-15, Brasília, Brasil, Junho.

VILLELA, K. V. C., 2004, Definição e Construção de Ambientes de Desenvolvimento de Software Orientados à Organização, Tese de D. Sc., COPPE/UFRJ, Rio de Janeiro, Brasil.

VILLELA, K., SANTOS, G., MONTONI, M., BERGER, P., FIGUEIREDO, S. M., MAFRA, S., ROCHA, A. R., TRAVASSOS, G. H., 2004, *Definição de Processos em Ambientes de Desenvolvimento de Software Orientados a Organização*, III Simpósio Brasileiro de Qualidade de Software, Brasília-DF.

VILLELA, K. V. C., ROCHA, A. R., TRAVASSOS, G. H., 2000, Ambientes de Desenvolvimento de Software Orientados a Organização. Publicação Técnica COPPE/UFRJ – ES530/00 Rio de Janeiro, RJ. Abril.

XAVIER, J, R., 2001, Criação e Instanciação de Arquiteturas de Software Específicas de Domínio no Contexto de uma Infra-Estrutura de Reutilização, Dissertação de Mestrado, Universidade Federal do Rio de Janeiro, Brasil.

ZLOT, F., 2002, Conhecimento de Tarefa em Ambientes de Desenvolvimento de Software Orientados a Domínio, Dissertação de M. Sc., COPPE/UFRJ, Rio de Janeiro, Brasil.

ZLOT, F. et al., 2002, Modeling Task Knowledge to Support Software Development, In: Proceedings of the 14th international conference on Software engineering and knowledge engineering - SEKE'02, pp. 35-42, Ischia, Itália.

ANEXO I

Formulário utilizado no *Survey* descrito em ROCHA *et al.* (2005)

INSTRUÇÕES

O objetivo desta pesquisa é identificar dificuldades e fatores de sucesso relacionados à implementação de processos de software.

O público-alvo desta pesquisa são engenheiros de software com experiência em implementação de processos de software.

Este formulário é composto de duas seções. A primeira seção deve ser preenchida com informações sobre o implementador de processo de software. A segunda seção deve ser preenchida com informações sobre dificuldades encontradas ou percebidas durante a realização de atividades de implementação de processos de software em pequenas, médias ou grandes empresas desenvolvedoras de software.

1. Identificação do implementador de processo

Nome:	
E-mail:	
Instituição Implementadora:	

2. Formulário de identificação de dificuldades e fatores de sucesso relacionados à implementação de processo de software

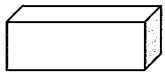
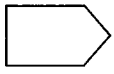
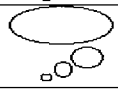
	Modelo de Negócio Cooperado em grupo de empresa (pacote)	Modelo de Negócio Específico para cada empresa (personalizado)
Dificuldades		
Não tem água encanada	X	
Inexistência de computadores		X
Fatores de sucesso		
Ter cafeteira no laboratório	X	X

ANEXO II

Linguagem Utilizada para Modelagem de Processos

A linguagem proposta para modelagem de processos organizacionais é composta de elementos gráficos que podem ser do tipo área, objeto ou associação, onde uma associação estabelece uma relação entre dois objetos e uma área agrupa objetos, definindo um contexto para os mesmos. Objetos ainda permitem adornos, utilizados para representar explicitamente características dos objetos. A seguir, cada elemento da linguagem é brevemente apresentado.

Tabela A2.1 – Definição e Notação dos Objetos (VILLELA, 2004)

Objeto	**Notação**	**Definição**
Processo		Objeto referente ao conceito de mesmo nome definido na ontologia de organização. *Atributos Especiais:* Origem (Interno, Externo)
Evento		Objeto que representa um acontecimento no ambiente que provoca o início ou fim de um processo. A notação é proveniente do produto comercial de workflow ARIS *ToolSet*.
Ator		Objeto que representa um pessoa, agente ou unidade organizacional. Estes conceitos encontram-se definidos na ontologia de organização.
Atividade		Objeto referente ao conceito de mesmo nome definido na ontologia de organização. *Atributos Especiais:* Origem (Interna, Externa) Granularidade (Elementar ou Composta)
Estado Inicial		Objeto puramente notacional, proveniente dos diagramas de estado e que indica onde é iniciado o fluxo de atividades que definem um processo ou uma atividade composta
Estado Final		Objeto puramente notacional, proveniente dos diagramas de estado e que indica onde é encerrado o fluxo de atividades que definem um processo ou uma atividade composta
Conhecimento Explícito		Objeto que representa um conhecimento que pode ser expresso em palavras e números e ser facilmente transmitido e compartilhado.
Conhecimento Implícito		Objeto que representa um conhecimento que é altamente pessoal e difícil de formalizar, o que o torna também difícil de ser compartilhado.

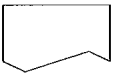
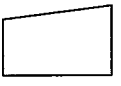
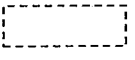
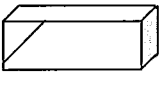
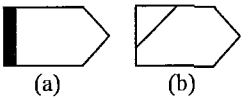
Comunicação		Objeto que representa a comunicação de dados ou informações a partir da, ou para a, execução de uma atividade. A comunicação pode ser verbal ou escrita e exemplos são e-mail e fax.
Repositório (Meio Magnético)		Objeto que representa um meio magnético para o armazenamento de dados e informações. A notação é proveniente do produto comercial de workflow ARIS *ToolSet*.
Arquivo (Local Físico)		Objeto que representa um local físico para armazenamento de documentos e comunicações escritas.
Documento		Objeto referente ao conceito de mesmo nome definido na ontologia de organização. A notação é proveniente do produto comercial de workflow ARIS *ToolSet*.
Componente de Hardware		Objeto referente ao conceito de mesmo nome definido na ontologia de organização. A notação é baseada na notação de componente da UML.
Componente de Software		Objeto referente ao conceito de mesmo nome definido na ontologia de organização. A notação é baseada na notação de componente da UML.
Peça		Objeto referente ao conceito de mesmo nome definido na ontologia de organização.
Matéria-Prima		Objeto referente ao conceito de mesmo nome definido na ontologia de organização.
Bem		Objeto referente ao conceito de mesmo nome definido na ontologia de organização. A notação fornecida pode ser substituída por uma mais significativa para o objeto específico do modelo como, por exemplo, o logotipo do software. *Atributos Especiais:*Tipo (Usufruto, Software, Hardware e Equipamento de Produção)
Nota Explicativa		Objeto que permite que notas explicativas sejam adicionadas ao modelo. *Atributos Especiais:*Texto

Tabela A2.2 – Definição e Notação dos Adornos (VILLELA, 2004)

Objeto	**Notação com Adornos**	**Definição dos Adornos**
Processo		Adorno que indica que o processo é externo, ou seja, que é executado por outra organização.
Atividade	 (a) (b)	(a) Adorno que indica que a atividade é composta, o que significa que ela pode ser decomposta em sub-atividades; (b) Adorno que indica que a atividade é externa, ou seja, que é executada por outra organização.

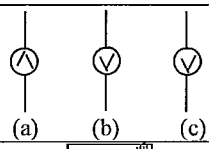
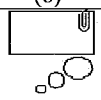
Operação Lógica		(a) Adorno que indica a operação lógica E; (b) Adorno que indica a operação lógica OU; (c) Adorno que indica a operação lógica OU Exclusivo.
Conhecimento Explícito		Adorno que indica que foi especificado um caminho para acesso ao conhecimento disponível em meio magnético. Este adorno só deve ser utilizado se a visualização do modelo for apoiada por uma ferramenta de software que permita o acesso ao conhecimento. *Atributos Especiais:* Localização do Arquivo

Tabela A2.3 – Definição e Notação das Áreas (VILLELA, 2004)

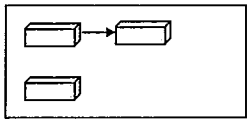
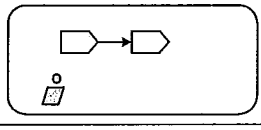
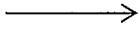
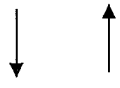
Objeto	Notação	Definição
Grupo de Processos		Área que agrupa processos relacionados.
Área de Ator		Área que agrupa atividades executadas por um ator ou grupo de atores. O ator ou o grupo de atores também precisa estar contido na área.

Tabela A2.4 – Definição e Notação das Associações (VILLELA, 2004)

Objeto	Notação	Definição
Fluxo de Controle		Associação que indica a passagem de controle do objeto origem para o objeto destino. O c1 e o c2 indicados na notação são os rótulos das condições estabelecidas para que a passagem de controle ocorra. *Atributo Especial:* Condição, formada por rótulo e descrição
Fluxo de Entrada/Saída		Associação que estabelece um insumo (se o fluxo é de entrada) ou um produto de uma atividade (se o fluxo é de saída). Quando o objeto de origem ou destino é um armazenador (repositório ou arquivo), a notação pode incluir os rótulos das informações trafegadas, existindo, então, um atributo especial. *Atributo Especial:* Informação, formada por rótulo e descrição
Associação Não Direcionada		Associação que não indica passagem de controle nem estabelece insumos e produtos para uma atividade, sendo utilizada para conectar bens de produção (software, hardware e equipamentos) utilizados como recursos para execução das atividades e para conectar eventos que atuam sobre processos, provocando o seu início ou fim. No segundo caso, um atributo especial é definido. *Atributo Especial:* Papel do Evento (Iniciador, Terminador)
Associação de Nota Explicativa		Associação que estabelece que uma nota explicativa é referente a um elemento do modelo.

ANEXO III

Norma ISO/IEC 14102 – Guia para a Seleção e Avaliação de Ferramentas CASE

o A norma ISO/IEC 14102 (1995) estabelece os processos e atividades a serem aplicados durante a avaliação de ferramentas CASE e seleção da ferramenta CASE mais apropriada dentre as diversas candidatas.

A avaliação e seleção de ferramentas CASE é composta de quatro processos: (i) Processo de Iniciação, (ii) Processo de Estruturação, (iii) Processo de Avaliação e (iv) Processo de Seleção.

Processo de Iniciação

O processo de iniciação tem a finalidade de definir os objetivos e requisitos gerais da avaliação e seleção a serem realizadas para estabelecer os requisitos de alto nível e definir aspectos gerenciais do esforço (cronograma, recursos, custo). O processo de iniciação é composto de três atividades:

o Ajuste do objetivo: Durante esta atividade, deve-se estabelecer o raciocínio e a política geral para a avaliação e seleção. As restrições da avaliação e seleção devem ser definidas e as expectativas devem ser quantificadas e qualificadas.

o Estabelecimento dos critérios de seleção: Nesta atividade, os critérios de seleção a serem utilizados nas atividades do processo de seleção são estabelecidos, sendo necessário definir a importância relativa dos critérios identificados.

o Planejamento do Projeto: Nesta atividade é elaborado um plano que inclui informação de planejamento genérica e também informação que define a estrutura do esforço de avaliação e seleção. Assim, é definido um cronograma para as atividades de avaliação e seleção, os recursos necessários para a realização destas atividades e a forma como a execução do plano será monitorada são planejados.

Processo de Estruturação

O processo de estruturação tem por objetivos: (i) elaborar um conjunto de requisitos estruturados, baseados nas características das ferramentas CASE, contra os quais as ferramentas CASE devem ser avaliadas (ii) e obter a informação necessária das ferramentas CASE para permitir a avaliação. Um conjunto de guias e informações gerais da organização deve estar disponível para ser consultado durante a execução deste processo, que é composto de três atividades:

Análise dos Requisitos: O objetivo desta atividade é transformar as necessidades organizacionais em estruturas mensuráveis. Durante a definição dos requisitos, os requisitos para a ferramenta CASE são coletados e organizados de acordo com as características da ferramenta CASE. Um conjunto abrangente de requisitos é necessário para selecionar a ferramenta CASE mais adequada e o processo de estruturação provê maior facilidade para a repetição do processo de avaliação.

o Coleta de Informação sobre a Ferramenta CASE: Durante esta atividade, deve-se coletar informações que indiquem o estado da arte atual das ferramentas CASE. As atividades de coleta da informação e de identificação das ferramentas CASE candidatas podem requerer diversas iterações para identificar as ferramentas mais promissoras para avaliações futuras de modo rápido e eficiente.

o Identificação das ferramentas CASE que são candidatas finais: Nesta atividade, as ferramentas CASE candidatas são identificadas para avaliação usando os resultados das atividades "Análise dos Requisitos" e "Coleta de Informação sobre a Ferramenta CASE".

Avaliação do Processo

O propósito da avaliação do processo é produzir relatórios técnicos de avaliação que serão utilizados pelo processo de seleção. Cada processo de avaliação resulta em um perfil da qualidade e de outras características da ferramenta avaliada, sendo que não são feitas comparações entre as ferramentas durante este processo. O processo de avaliação é composto de três atividades:

- Preparação para avaliação: Nesta atividade ocorre a finalização dos vários detalhes da avaliação em um plano de avaliação.

- Avaliação das ferramentas CASE: O objetivo desta atividade é medir, pontuar (*rating*) e avaliar (*assessment*) as ferramentas CASE. Neste momento, as ferramentas CASE são avaliadas em relação a cada um dos critérios pré-estabelecidos.

- Relato da avaliação: O resultado final das atividades de avaliação será um relatório de avaliação contendo informações sobre todas as ferramentas avaliadas. Este relatório será utilizado pelas atividades do processo de seleção.

Processo de Seleção

O objetivo do processo de seleção é identificar a ferramenta CASE mais apropriada dentre as ferramentas candidatas e assegurar que a ferramenta CASE recomendada satisfaz os objetivos originais. O processo de seleção compara os resultados das avaliações das ferramentas candidatas para determinar qual é a mais apropriada para seleção, sendo composto de quatro atividades:

- Preparação para a seleção: Durante a execução desta atividade, deve-se terminar com a avaliação dos critérios de seleção e definir o algoritmo de seleção a ser aplicado. O algoritmo de seleção determina como dados gerados durante várias avaliações são combinados e comparados para resultar em classificações para cada candidato.

- Avaliação dos Resultados da Avaliação: Esta atividade tem a finalidade de aplicar o algoritmo de seleção em relação aos resultados das avaliações.

o Recomendação da Decisão de Seleção: O resultado desta atividade é a recomendação da ferramenta CASE mais adequada.

o Validação da Decisão de Seleção: A atividade final do processo deve ser a validação da recomendação de seleção, na qual os objetivos e guias de seleção originais devem ser revistos e comparados com os resultados das avaliações e outros dados relacionados com a seleção recomendada. Uma checagem deve ser feita para garantir que se a recomendação é aceita e os objetivos de alto nível serão alcançados. Uma ferramenta adequada pode não ser encontrada ou a decisão pode ser feita entre o desenvolvimento de uma nova ferramenta ou a modificação de uma ferramenta existente ou até mesmo abandonar todo o processo de avaliação e seleção.

ANEXO IV

Resumo do estudo apresentado em (XAVIER, 2001) que descreve uma abordagem de seleção de padrões arquiteturais

Tendo estudado as diferentes influências que uma solução arquitetural traz para o projeto da arquitetura, os autores perceberam que as que mais tem relação com a fase de especificação do software e exercem grande impacto sobre a aceitação do produto são as características de qualidade da aplicação.

Para que pudessem explorar o conhecimento sobre os benefícios e dificuldades comuns à utilização de estilos e padrões arquiteturais, era preciso optar por um modelo de qualidade que orientasse o processo de formalização deste conhecimento. No contexto do trabalho realizado foi escolhido como ponto de partida o modelo definido pela norma ISO/IEC 9126-1 (ISO/IEC 9126, 1992). Essa escolha recebeu a influência de trabalhos que relacionam algumas características de qualidade ao projeto arquitetural de software, como BASS *et al.* (2003).

Em relação aos estilos arquiteturais a serem avaliados, optou-se pelos estilos arquiteturais orientados a objetos descritos a seguir:

Camadas: Estilo arquitetural que propõe a decomposição de aplicação em grupos de subtarefas, onde cada grupo se apresenta em um nível particular de abstração.

Pipes & Filters: Este estilo arquitetural suporta a divisão de uma função ou processo em várias etapas de processamento seqüenciais. Cada etapa recebe, processa e disponibiliza uma cadeia de dados e o fluxo de saída de uma etapa corresponde ao fluxo de entrada da etapa seguinte.

BlackBoard: Neste estilo, vários subsistemas organizam o conhecimento para a construção de uma possível solução aproximada ou parcial através do uso de uma memória compartilhada. Cada subsistema é especializado para solucionar uma fase particular da tarefa completa, e todos os subsistemas trabalham juntos para a aquisição da solução. Tais subsistemas não interagem diretamente entre si e nem há uma seqüência determinada para as suas ativações. Em vez disto, a direção que o sistema toma é, principalmente, determinada pelo estado corrente da aplicação. A principal variação deste estilo é o Repositório, uma generalização onde há a especificação de um componente responsável pelo controle interno da aplicação. Subsistemas que utilizam bancos de dados tradicionais podem ser considerados como exemplos de repositórios.

Broker: Pelo uso deste estilo, uma aplicação pode processar serviços de outras aplicações simplesmente pelo envio de mensagens a objetos mediadores sem se preocupar com questões específicas relacionadas à comunicação entre processos (acoplamento, localização, transmissão de dados, etc.).

Modelo Vista Controle (MVC): Propõe a divisão da aplicação em três categorias principais de componentes. O componente modelo encapsula dados e funcionalidades principais da aplicação. O componente visão mostra informações ao usuário obtidas de um modelo. Cada visão apresenta um componente controlador que recebe as entradas do usuário (na forma de eventos disparados) e os traduz em solicitações de serviços ao modelo e/ou visão. O modelo é independente de suas possíveis representações, bem como do comportamento de entrada, e pode apresentar múltiplas visões.

Presentation-Abstraction-Control (PAC): Define uma estrutura para subsistemas na forma de uma hierarquia de agentes cooperativos. Cada agente é responsável por um aspecto específico da funcionalidade da aplicação e é composto por três componentes: apresentação, abstração e controle.

MicroKernel: Propõe a separação de um conjunto de funcionalidade mínimo das funcionalidades estendidas e partes específicas de clientes. O encapsulamento dos serviços fundamentais da aplicação deve ser realizado no componente *microkernel.* As funcionalidades estendidas e específicas devem ser distribuídas entre os componentes restantes da hierarquia.

Reflection: Uma arquitetura que é dividida em duas partes principais: o nível meta, que fornece uma representação do software para o auto-conhecimento da estrutura e comportamento do sistema através de componentes chamados meta-objetos, o nível base, que define a lógica da aplicação, onde a implementação do sistema utiliza os meta-objetos através de um protocolo de meta-objetos.

Para avaliar os estilos arquiteturais em relação às características de qualidade definidas, utilizou-se a escala "muito alto, alto, médio, baixo, muito baixo e desconhecido". Esta classificação permite avaliar o esforço necessário para a satisfação de uma determinada característica de qualidade caso um estilo arquitetural seja adotado. O relacionamento denominado "desconhecido" representa os casos em que para determinada característica arquitetural de qualidade não há informação suficiente que permita uma avaliação segura de sua relação. Os significados dos itens desta escala podem ser vistos a seguir:

Muito Bom (MB): Característica principal do padrão utilizado, sendo sua aplicação completamente favorável;

Bom (B): A utilização do padrão é favorável, mas exige algum esforço de projeto para alcançar o benefício almejado;

Médio (M): A utilização do padrão aplica-se a situações em que o esforço desprendido se justifica perante o benefício encontrado;

Ruim (R): A utilização do padrão é injustificável na maioria das situações;

Muito Ruim (MR): A utilização do padrão é altamente desfavorável;

Desconhecido (D): Quando não há menção sobre esta relação de esforço;

Para a identificação da relação entre os estilos arquiteturais e as características de qualidade identificadas, houve inicialmente o preenchimento de um questionário por parte de um grupo de dez pessoas, sendo este grupo formado em sua maioria por alunos do curso de graduação e pós graduação em informática da UFRJ que prestavam ou já haviam prestado uma disciplina relacionada ao desenvolvimento de software com ênfase em projeto. Após a interpretação dos resultados, percebeu-se que para a maioria dos participantes a relação existente entre as características arquiteturais de qualidade e o projeto arquitetural de um software não é evidente. O resultado obtido poderia indicar que as pessoas desconhecem a aplicação dos padrões arquiteturais visando especificamente a obtenção de características arquiteturais específicas. Os autores acreditaram que a orientação dada ao planejamento do estudo dificultou o entendimento dos objetivos e, então, procuraram realizar uma reformulação sobre a forma de avaliação dos padrões arquiteturais.

O segundo estudo foi baseado em cenários de projeto, mantendo a idéia de que desenvolvedores de software com experiência em projeto arquitetural e detalhado podem contribuir com conhecimento prático sobre a utilização de padrões arquiteturais. Porém, a avaliação deste conhecimento foi realizada através de um exemplo que procurou estar mais próximo de uma situação real de desenvolvimento.

Para a execução do estudo, foi solicitado aos especialistas que, para cada cenário de projeto, indicasse e justificasse, os estilos arquiteturais mais e menos aplicáveis ao contexto descrito, em um número de pelo menos dois padrões para cada situação. O especialista deveria ainda descrever algumas observações em relação aos motivos que o levaram a excluir os restantes dos estilos das situações anteriores.

A tabela A4.1 exibe os resultados da aplicação deste novo estudo:

Tabela A4.1 – Resultado da reavaliação dos padrões arquiteturais utilizados (XAVIER, 2001)

Padrões Arquiteturais / Características Arquiteturais	Camadas	Pipes and Filters	Blackboard	Broker	MVC	PAC	Microkernel	Reflection
Interoperabilidade	B	D	D	B	D	M	M	D
Segurança de Acesso	MB	D	M	B	B	M	B	MR
Maturidade	M	R	MR	M	B	M	B	D
Tolerância a Falhas	M	R	R	R	M	M	R	MR
Recuperabilidade	M	MR	R	MR	D	R	R	D
Operacionalidade	B	D	R	D	MB	MB	D	D
Comportamento em relação ao tempo	M	B	MR	MR	M	R	M	R
Comportamento em relação aos recursos	R	R	MR	MR	M	R	M	R
Modificabilidade	B	M	MB	B	B	R	MB	MB
Testabilidade	MB	R	MR	R	B	MR	M	D
Adaptabilidade	MB	MR	D	B	B	D	B	B