ELICITAÇÃO DOS REQUISITOS DE META-AMBIENTES DE ENGENHARIA DE SOFTWARE ATRAVÉS DA ANÁLISE DE NEGÓCIOS

M.A.B. Serrano[1]
mamelia@furnas.com.br
A.v. Staa[2]
arndt@inf.puc-rio.br

Resumo

Utilizando técnicas de análise de negócios, são identificadas e justificadas as propriedades essenciais de meta-ambientes de engenharia de software, dentro de uma abordagem mais ampla para o processo de desenvolvimento de software, visando a instanciação de ambientes de negócio dirigidos à engenharia de software. O contexto do desenvolvimento de software é organizado em três eixos ortogonais, atuando em dois ambientes de negócio diferentes: geração das necessidades (domínio de indivíduos), solução do problema (domínio do negócio) e implementação da solução do problema (domínio da tecnologia), aplicados ao negócio engenharia de software e ao negócio aplicação. O modelo baseia-se em ciclos de realimentação e é construído para evoluir e ser capaz de aprender a partir das necessidades de uso.

Palavras Chaves: *ambiente de engenharia de software, análise de negócios, processos dirigidos por realimentação, meta ambientes de engenharia de software, requisitos de software*

Abstract

The essential properties of Software Engineering Meta-Environment are identified. The identification process is based on mission analysis techniques, and examines the software process from a fairly wide scope, envisaging software environment instantiation, adequately geared towards the software engineering mission. The software development process is organized in three orthogonal domain axes: user domain, mission domain, and technology domain –, and operate in two mission oriented environments: mission of the application, and mission of the software engineering environment. The model is based on feedback loops, and has been designed in order to allow easy evolution and knowledge acquisition.

Keywords: *feedback driven processes, mission analysis, business planing, meta-environments, software engineering environments, software requirements.*

---

[1] Trabalho suportado por Furnas Centrais Elétricas S.A., Coordenação de Organização e Informática (OI.G)

[2] Trabalho apoiado por: CNPq, Bolsa de Pesquisador 300029/92-6

1. Introdução

O objetivo do presente trabalho é identificar e justificar requisitos de meta-ambientes de engenharia de software assistidos por computador, dentro de uma abordagem mais ampla para o desenvolvimento de software, visando a integração entre a organização para quem é desenvolvido o software, a organização que desenvolve o software, as ferramentas disponíveis e os papéis desempenhados pelas pessoas que atuam no desenvolvimento e uso. Para identificar os requisitos de meta-ambientes e de ambientes de engenharia de software serão utilizados princípios de análise de negócio [KAST92]. Esta abordagem visa desenvolver e evoluir ambientes de engenharia de software que sejam ao mesmo tempo efetivos, eficientes e eficazes.

Tem sido observado que ambientes de desenvolvimento devem ser dirigidos para o domínio do problema [BROWN92, NUSEIBEH93, FUGETTA94]. Tem sido observado, ainda, que ferramentas CASE são freqüentemente abandonadas, constituindo o que vem sendo depreciativamente chamado de "*shelf ware*". Tendo em vista a multitude de domínios, levando a um número grande de pequenos nichos de mercado, o custo de desenvolver ambientes especializados para cada um, pode facilmente exceder os benefícios auferidos ao utilizá-los. Meta-ambientes visam justamente agilizar e reduzir os custos de desenvolvimento e de evolução de instâncias de ambientes específicas e fortemente dirigidas para domínios de problemas. No entanto, para que meta-ambientes possam vir a ser instrumentos eficazes, é necessário entender muito bem as necessidades do negócio engenharia de software, as necessidades do negócio aplicação e a interação existente entre estes dois.

Uma característica essencial de meta-ambientes, dos ambientes com eles instanciados e dos sistemas desenvolvidos com o apoio destas instâncias é a necessidade de contínuo aprimoramento ou evolução [LEHMAN96]. Portanto, não basta apenas *construir* corretamente um software, ou o ambiente utilizado para desenvolvê-lo, é preciso poder *evolui-los* correta e continuamente. A evolução se dá tanto no negócio da aplicação, como no negócio engenharia de software e nos meta-ambientes instanciadores, e visa manter a proximidade do serviço do software e do ambiente utilizado para o seu desenvolvimento com a realidade externa na qual estão inseridos e que se encontra em contínua evolução.

A engenharia de software presta serviços para o negócio aplicação, através de uma relação que se acha em contínua adaptação e modificação. A engenharia de software ainda depende fortemente dos indivíduos e de suas características pessoais. Para reduzir a dependência de pessoal de alta proficiência, está surgindo um novo paradigma de desenvolvimento no qual engenheiros de software de elevada proficiência – fabricantes – criam ferramentas e "*frameworks*" especializados, que são utilizados por desenvolvedores – construtores, usuários – para criar aplicações dentro de domínios estabelecidos [NIERSTRASZ95]. Este paradigma depende fortemente de ferramentas poderosas e dirigidas para o domínio da aplicação. Desta forma o especialista da aplicação pode desenvolver aplicações adequadas[3] sem necessitar um nível de treinamento elevado em informática.

Muitos dos modelos de processos descritos na literatura partem de uma especificação de requisitos [ROMBACH90, GHEZZI91, BOEHM94, DARIMONT95], que uma vez estabeleci-

[3] O termo *adequado* é usado no sentido de conformidade exata com a realidade [AURÉLIO86].

da, é congelada. Outros prevêem uma explícita gestão da evolução de requisitos [HUMPHREY89]. Sem o apoio de ferramentas poderosas e adequadas, a qualidade do produto dependerá das qualidades individuais dos desenvolvedores, da sua criatividade e conhecimento tecnológico. Dependerão, ainda, dos usuários, das suas capacidades de comunicação e de seus conhecimentos do problema. Os modelos de processo precisam evoluir, ampliar sua abrangência, para que se tenha um melhor entendimento do problema a resolver, para que se possa agilizar o desenvolvimento e a evolução, e para que se possa produzir software de melhor qualidade [LEHMAN96].

As exigências feitas hoje à engenharia de software impõem uma visualização, análise e modelagem de uma interação de negócio para negócio. Além das necessidades individuais dos usuários, é preciso que sejam atendidas as necessidades dos papéis desempenhados por estes usuários. Mais do que isto, é preciso que o próprio desenvolvimento seja visto como negócio e visualize as necessidades do desenvolvedor e do seu papel no negócio engenharia de software. É preciso que a geração das necessidades, quer sejam dos indivíduos, de seus negócios, ou de construção do software, estejam inseridas no contexto de desenvolvimento.

Um ambiente de negócio de engenharia de software baseado no processo, constituindo uma instância de um meta ambiente automatizado, cujo modelo é especificado usando-se técnicas de análise de negócios, sinaliza um caminho para atender de forma satisfatória às exigências da engenharia de software. O ambiente é baseado no processo, é auto regulado por "*feedback*", é voltado às necessidades de uso e é evolutivo, adaptando-se continuamente às mudanças do meio em que está inserido [SERRANO97].

O trabalho inicia com a apresentação do "*Contexto de Desenvolvimento*" dentro de um enfoque mais amplo, identificando as necessidades do ambiente de produção de software. A seguir são discutidos os "*Requisitos do Ambiente Automatizado*", identificando e justificando as propriedades do ambiente automatizado. Na seção "*Trabalhos Relacionados*", apresentamos as linhas de pesquisa atuais dentro da área do problema. Finalmente, nas "*Conclusões*" são apresentadas as contribuições trazidas pelo presente trabalho.

2. Contexto de Desenvolvimento

É necessária uma mudança na abrangência do contexto de desenvolvimento, que parte da interface negócio engenharia de software / negócio aplicação, priorizando ciclos de avaliação e realimentação. Com a mudança de paradigma de desenvolvimento, o alinhamento estratégico e operacional entre os dois negócios é sistematizado, a especificação de requisitos, tanto do ambiente de desenvolvimento como da aplicação, ao invés de ser rígida, evolui junto com os ambientes envolvidos e com o relacionamento entre ambos, mantendo-o em equilíbrio dinâmico com o meio onde se acha inserido.

Tal como já vem sendo preconizado por inúmeros autores, o requisito básico mais abstrato de um software é de atender as necessidades de seus usuários e dos negócios onde estes se acham inseridos. Este requisito muda a preocupação com as interfaces do sistema, para a preocupação com os serviços a serem apoiados pelo software, que terão como *conseqüência* a definição das interfaces [NUSEIBEH93, WEAVER95]. Técnicas de elicitação de requisitos, conforme discutidas na literatura [GOLDIN94, DARIMONT95], procuram contribuir para a produção

de especificações mais adequadas. Mesmo assim, é esperado que os requisitos evoluam durante e após o desenvolvimento [LEHMAN96, HUMPHREY89].

Várias são as interfaces de interação identificadas (figura 1), e que devem ser atendidas, analisadas e modeladas. Cada uma delas apresenta características próprias (interfaces homem-homem, interfaces homem-máquina; interfaces máquina-máquina), que contribuem para a especificação das necessidades a serem atendidas pelo ambiente automatizado.

Um ciclo de avaliação dos serviços prestados é gerado em cada uma das interfaces de comunicação, provocando mudanças das necessidades. De cada ciclo de realimentação resulta um aprendizado, quer seja interno ao ambiente de negócio, quer seja externo. Este aprendizado se traduz em uma mudança de requisitos, que provoca a evolução do ambiente automatizado, quer seja para modificação das ferramentas em uso em cada ambiente de negócio, quer seja na modificação da forma de execução do trabalho e conseqüentemente de sua organização.

Lehman [LEHMAN96] define o processo de desenvolvimento de software como sendo um processo de múltiplos agentes, múltiplos níveis de abstração, múltiplos ciclos de um sistema de realimentação. Para que se tenha melhorias sensíveis no processo de desenvolvimento de software é necessário uma maior abrangência do contexto de desenvolvimento, é preciso que sejam identificados os ciclos de realimentação que influenciam o sistema de software e que são por ele influenciados.

Dentro do enfoque proposto, o ambiente de desenvolvimento e produção de software fica então identificado por suas necessidades, e que podem ser divididas em dois grandes grupos, ver figura 1. No primeiro grupo se acham as necessidades do ambiente de desenvolvimento de software decorrentes da identificação e captura dos requisitos externos do ambiente automatizado (necessidades do negócio), ou seja:

- entender / registrar / modelar o negócio aplicação apoiado pelo sistema objetivo (interfaces de nº 1, nº 2 e nº 7 da figura 1);
- entender / registrar / modelar o negócio de engenharia de software apoiado pelo ambiente automatizado (interfaces de nº4, nº 5 e nº 7);
- entender / registrar / modelar a interface entre o negócio engenharia de software e negócio aplicação (interfaces de nº 7);
- entender / registrar / modelar os serviços da engenharia de software (interfaces de nº 4 e nº 7);
- entender / registrar / modelar os serviços da aplicação (interfaces de nº 1 e nº 7);
- entender / registrar os serviços do desenvolvedor do sistema objetivo e do ambiente automatizado (interface de nº 5);
- entender / registrar os serviços dos usuários do sistema objetivo (interface de nº 2);

No segundo grupo se acham as necessidades do ambiente de desenvolvimento de software decorrentes da identificação e captura dos requisitos externos (necessidades do usuário) e requisitos internos do ambiente automatizado, ou seja:

- entender / registrar / modelar o ambiente automatizado de forma adequada ao negócio engenharia de software e ao negócio aplicação;

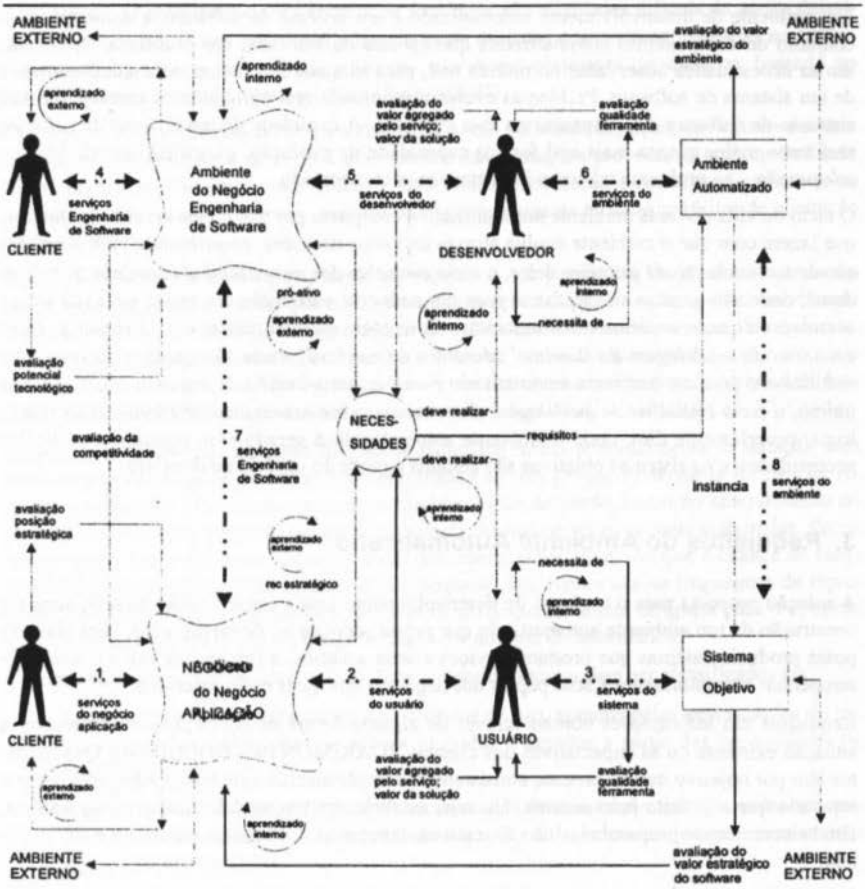

Figura 1 - Contexto do Desenvolvimento de Software

- entender / registrar / modelar os serviços do ambiente automatizado de forma adequada ao negócio de engenharia de software (interface de nº 6);
- entender / registrar / modelar os serviços do sistema objetivo de forma adequada ao negócio aplicação (interface de nº 3);
- entender / registrar / modelar a interface entre o ambiente automatizado e o sistema objetivo (interface de nº 8);
- construir / evoluir o ambiente automatizado e sistema objetivo mantendo a consistência dos serviços prestados (interface de nº 1, nº 2, nº 3, nº 4, nº 5, nº 6, nº 7 e nº 8).

Um ambiente de desenvolvimento automatizado é um sistema de software, e como tal é um conjunto de componentes automatizados que apoiam ou resolvem um problema. Problemas são as necessidades observadas no mundo real, para as quais se procura uma solução através de um sistema de software. Problemas evoluem no mundo real obrigando os correspondentes sistemas de software a acompanharem esta evolução. A qualidade de um sistema de software será tanto maior quanto mais ágil for sua capacidade de evolução, garantindo proximidade – *adequação* – ao problema tal como instantaneamente percebido.

O ciclo de vida do meta ambiente automatizado é composto por três ciclos de vida evolutivos, que fazem com que o ambiente evolua através de várias iterações, contribuindo para a agilização de sua evolução. O primeiro deles, o *ciclo evolutivo das necessidades*[4] (domínio de indivíduos), onde são gerados os requisitos para um ambiente automatizado, tendo em vista as necessidades de seus usuários e dos ambientes de negócio ao qual pertencem. O segundo, *ciclo evolutivo da modelagem do domínio*[5] (domínio do negócio), onde são gerados os requisitos reutilizáveis para um ambiente automatizado e/ou para uma família de sistemas objetivos. Por último, o *ciclo evolutivo da modelagem do meta ambiente automatizado*[6] (domínio da tecnologia) propriamente dito, onde o ambiente automatizado é gerado e/ou mantido a partir das necessidades, e os sistemas objetivos são gerados a partir do modelo do domínio.

3. Requisitos do Ambiente Automatizado

A solução proposta para o contexto de desenvolvimento visa a especificação de requisitos e a construção de um ambiente automatizado que presta serviços ao desenvolvedor, para que este possa produzir sistemas que prestam serviços a seus usuários, a fim de que estes possam desempenhar satisfatoriamente seus papéis nos negócios nos quais estão inseridos.

Requisitos são informações obtidas através de alguma forma de elicitação e documentam a situação existente ou as expectativas dos clientes[7] [DARIMONT95, GOLDIN94]. Os requisitos têm por objetivo definir precisa, consistente e completamente e de forma adequada o que é esperado que seja feito pelo sistema. Ou seja, estabelecem um modelo funcional do sistema. Estabelecem ainda propriedades não funcionais, tais como critérios de aceitação e desempenho.

A identificação dos requisitos de um sistema sempre apresenta dificuldades. Uma das razões é o erro de se assumir que todas as necessidades do sistema possam ser identificadas e definidas precisamente em um único ponto do processo de desenvolvimento de software. Cada pessoa envolvida tem uma percepção diferente das suas necessidades, e estas percepções variam com o tempo e com o nível de abstração, motivadas em parte pelo entendimento do problema e seu contexto, e em parte porque o contexto por si só se modifica [BUSTARD96].

---

[4] Apresentado na figura 1 pelas interfaces de no. 2, 3, 5 e 6, e pelos ciclos de avaliação da qualidade da ferramenta, do valor agregado pelo serviço, da competitividade.

[5] Apresentado na figura pelas interfaces de no. 1, 2, 4, 5 e 7, e pelos ciclos de avaliação do potencial tecnológico, da posição estratégica e da competitividade.

[6] Apresentado na figura pelas interfaces de no. 3, 6 e 8, e pelos ciclos de avaliação da qualidade das ferramentas nos dois ambientes de negócio, do valor estratégico do software, do valor estratégico do ambiente.

[7] *Cliente* é usado no sentido abrangente. Podem ser pessoas, processos ou outros sistemas de software.

Além disso, as atividades da engenharia de requisitos são atividades difíceis de serem definidas e organizadas, já que envolvem a ligação entre decisões corporativas, baseadas no conhecimento e experiência no negócio aplicação, e o desenvolvimento de sistemas, baseado no conhecimento e experiência tecnológica [REAIMS 95].

Conclui-se, então, que requisitos de software estão em constante mutação, em virtude das mudanças de necessidades e de tecnologias usadas, das alterações provocadas pelo aprendizado do problema, seu contexto e sua solução. Esta contínua evolução é considerada inerente, ou seja, impossível de ser eliminada, mesmo se dispuséssemos de técnicas perfeitas de elicitação de requisitos.

As conseqüências de uma *especificação de requisitos não produzida, ou produzida de forma inadequada* são, entre outras, que o *volume de alterações* durante e após o desenvolvimento tende a ser grande; o *custo* e o *tempo* requeridos são significativamente maiores do que o realmente necessário, conseqüência direta do volume de retrabalho (tentativa e erro); e o *risco técnico* tende a ser grande. São exemplos de tais riscos: sistema incompleto, sistema com funções irrelevantes, sistema não evolutivo, sistema não integrável [HUMPHREY89, STAA95].

Enquanto os problemas da especificação de requisitos forem principalmente de notação, eles podem ser minimizados ou até mesmo eliminados através do uso de técnicas formais de registro e validação. No entanto, quando os problemas da definição forem do entendimento do problema a resolver, pouco pode ser feito através do uso de técnicas mais elaboradas. Precisamos então aumentar a participação *eficaz* dos interessados, de modo que a chance de ocorrência de tais problemas seja tornada muito pequena. Isto requer que as linguagens de representação utilizadas para a comunicação entre os participantes sejam dirigidas para o domínio do problema, ao invés de serem dirigidas para a tecnologia de informática.

Os requisitos a serem atendidos pelo ambiente automatizado podem ser classificados em:

- *requisitos externos*, voltados para os cenários de uso, gerados pelas necessidades do negócio onde as ferramentas se acham inseridas e gerados a partir das necessidades de seus "usuários";
- *requisitos internos*, voltados para as necessidades de construção, gerados pelas necessidades de implementação, funcionalidade requerida e desempenho desejado para atender aos requisitos externos;
- *requisitos sistêmicos*, gerados a partir da forma de percepção e modelagem da realidade [SERRANO97].

3.1 Requisitos Externos

O que é relevante, segundo a ótica do presente trabalho, é que o contexto do ambiente de desenvolvimento vê a engenharia de software como um ambiente de negócio que atua em parceria com o ambiente de negócio aplicação, e o fato de que um sistema de software precisa apoiar os usuários de forma que estes possam servir ao negócio aplicação e engenharia de software em que atuam, em níveis de qualidade satisfatórios para este negócio. O serviço do software precisa então atender às necessidades de cada usuário e às necessidades do ambiente de negócio onde se encontra inserido. Os requisitos, identificados sob a ótica de cada *usuário* (interfaces de nº 3, 6 e 8 da figura 1) e os identificados sob a ótica do *negócio* (interfaces de nº 1, 2, 4, 5 e 7), precisam ser consistentes, não conflitantes e completos.

3.1.1 Serviços do Negócio - Interação do Negócio com o Meio Externo

As trocas do negócio engenharia de software com o meio externo (interface nº 4), permitem que este faça uma avaliação de seu potencial tecnológico, decorrente da posição que deseja ocupar no mercado competitivo. Desta avaliação pode resultar a geração de novas necessidades em função do público alvo a atingir. De maneira análoga, o relacionamento do negócio aplicação com o ambiente externo (interface nº 1) gera uma avaliação de sua posição estratégica, que pode modificar os serviços prestados pelo negócio, sua visão de mercado e políticas de gerenciamento.

A mudança da missão do negócio, de seus objetivos estratégicos, quer seja para o negócio engenharia de software, quer seja para o negócio aplicação, evolução da plataforma tecnológica (hardware, software, comunicação), organização do trabalho, provoca a mudança dos serviços prestados por seu pessoal, que, conseqüentemente, terão suas necessidades alteradas. Por outro lado, com a alteração da visão externa do negócio, o valor estratégico do software (ambiente automatizado e sistema objetivo) pode sofrer modificações.

As mudanças provocadas pelo redirecionamento da visão do negócio (negócio engenharia de software, negócio aplicação), quer seja por mudanças nos serviços prestados por seu pessoal (desenvolvedor, interface nº 5 da figura 1 e usuário, interface nº 2), quer seja por mudanças no valor estratégico do software, provocam mudanças de requisitos para o ambiente automatizado. Estas mudanças de requisitos, quando absorvidas pelo sistema de software, geram mudanças nos serviços prestados (interfaces nº 3 e nº 6), fazendo com que os sistemas de software estejam coadunados com os serviços do negócio engenharia de software e o negócio aplicação (interfaces nº 5 e nº 2).

É preciso conhecer o impacto causado pela introdução do ambiente automatizado no ambiente de negócio (ciclo de avaliação do valor estratégico do ambiente automatizado), o que significa ser necessário conhecer quais são os objetivos estratégicos do negócio, quais são suas premissas, quais são seus critérios de sucesso, como se relaciona com o ambiente externo.

3.1.2 Serviços do Usuário - Interação do Usuário com o Negócio Aplicação

Os serviços que o usuário presta ao negócio aplicação (interface nº 2) são dependentes do papel que desempenha, do processo que define este papel, de suas atribuições, do grupo ao qual pertence, do projeto onde está alocado. É preciso conhecer o impacto causado pelo sistema objetivo no serviço do usuário (ciclo de avaliação do valor da solução, do valor agregado pelo serviço), e o seu reflexo sobre o negócio aplicação.

3.1.3 Serviços do Desenvolvedor - Interação do Desenvolvedor com o Negócio Engenharia de Software

Os serviços prestados pelo desenvolvedor (interface nº 5) são relativos à produção e/ou manutenção de sistemas objetivo de qualidade assegurada e com produtividade para o negócio aplicação, e à produção e/ou manutenção de ambientes automatizados de qualidade assegurada com produtividade para o negócio engenharia de software e para apoiá-lo em suas tarefas. É preciso conhecer o impacto causado pelo ambiente automatizado no serviço do desenvolvedor (ciclo de avaliação do valor de solução, do valor agregado pelo serviço) e o seu reflexo no negócio engenharia de software.

3.1.4 Serviços da Engenharia de Software - Interação do Negócio Engenharia de Software com o Negócio Aplicação

A relação entre o negócio engenharia de software e o negócio aplicação (interface nº 7) precisa ser uma relação onde o primeiro possa fornecer sistemas objetivos adequados às necessidades de uso, para que o segundo possa executar sua missão de forma competitiva. A relação de parceria estabelece um ciclo de avaliação de competitividade que gera requisitos para o ambiente automatizado.

A obtenção de um relacionamento eficiente e eficaz entre o negócio engenharia de software e o negócio aplicação é muito difícil de ser obtido. Vários são os fatores que contribuem para isto: falibilidade humana; incapacidade de se adquirir uma compreensão completa e correta desta interface de comunicação devido a sua complexidade, tamanho e abrangência; o fato de que a evolução do conhecimento básico, experimental e tecnológico estabelece uma base contínua de mudanças. Por isto esta relação envolve muitas variáveis e diferentes pontos de vista, e como tal, o alinhamento entre estes dois ambientes não é um evento estanque, isolado, mas um processo de contínua adaptação e modificação [LUFTMAN93]. A visualização e modelagem da interface entre estes dois domínios ainda não foi adequadamente considerada pelas áreas de engenharia de software e melhoria da qualidade [REAIMS95].

Cada um dos dois ambientes precisa manter uma integração entre a sua posição externa no mercado competitivo (missão, políticas, metas), consistente com sua estrutura interna (processos, plataforma tecnológica, organização). Por outro lado, os dois ambientes precisam apresentar uma interdependência entre seus domínios de atuação, o que significa uma integração a nível estratégico, ou seja, o negócio engenharia de software deve ser capaz de suportar as estratégias do negócio aplicação, e uma integração a nível operacional, onde o negócio engenharia de software seja capaz de oferecer uma arquitetura de informação, uma arquitetura de sistemas de software, e uma plataforma tecnológica compatível com as expectativas e requisitos internos do negócio aplicação [HENDERSON93].

A interdependência entre os dois ambientes de negócio é uma relação dinâmica de naturezas diversas, dependendo das necessidades e expectativas do negócio aplicação em relação ao apoio a ser recebido pelo negócio engenharia de software. Neste caso, o alinhamento entre os ambientes pode ser de dois tipos [LUFTMAN93]. Em um dos casos, a engenharia de software atua como recurso estratégico do negócio aplicação, ao fornecer uma infra-estrutura de sistemas, plataforma tecnológica (hardware, software, comunicação) e uma infra-estrutura de informação compatíveis com as decisões estratégicas do negócio aplicação. No outro tipo de relação, a engenharia de software atua como um parceiro pró-ativo para sistemas do negócio aplicação, criando novas oportunidades de mercado pela oferta de novas ferramentas e novas soluções para o negócio aplicação.

3.1.5 Serviços do Ambiente Automatizado - Interação com o Desenvolvedor

Os serviços prestados pelo ambiente automatizado ao desenvolvedor (interface nº 6) são procedimentos realizados, resultados obtidos (sistemas de software, informações, ferramentas), produtos gerados pelo ambiente para apoiar o desenvolvedor em suas tarefas, tarefas estas, determinadas pelo papel que desempenha no negócio engenharia de software.

3.1.6 Serviços do Sistema Objetivo - Interação do Usuário e Sistema Objetivo

Os serviços prestados pelo sistema de software ao usuário (interface nº 3) são procedimentos realizados, resultados obtidos, produtos gerados pelo sistema e que apoiam as tarefas a serem executados pelo usuário, de acordo com o seu papel no negócio aplicação. A interação do usuário com o sistema objetivo permite a avaliação da utilidade, da utilizabilidade e do desempenho da ferramenta (ciclo de avaliação de qualidade do sistema objetivo).

3.1.7 Serviços do Ambiente Automatizado - Interação com o Sistema Objetivo

A parceria que se estabelece no domínio da tecnologia, entre ambiente automatizado / sistema objetivo (interface nº 8), gera requisitos para o ambiente automatizado. Os serviços prestados pelo ambiente automatizado são relativos a geração, manutenção e gerenciamento do sistema objetivo.

3.2 Requisitos Internos

O conhecimento do software a desenvolver é traduzido por uma coletânea de fatos. Estes fatos são registrados em documentos muito diferentes e ao mesmo tempo interdependentes. Cada um deles tem características próprias e informações peculiares a serem transmitidas, especificadas pelos ambientes parceiros negócio aplicação / negócio engenharia de software e por seus usuários, que devem ser armazenadas no computador, para que sejam geradas ferramentas de qualidade que apoiem efetivamente os processos organizacionais.

O modelo de um ambiente automatizado leva, portanto, à modelagem de um fluxo de documentos, que usam representações de complexidade e nível de abstração diversa para traduzir o conhecimento adquirido. As representações se utilizam de linguagens de representação que conseqüentemente são bastante diversas: especificação de requisitos, especificação de objetos, projeto do sistema de software, código fonte, código executável, manuais, ajuda ao usuário (help) [STAA95].

O conhecimento a ser modelado pode mudar muito durante o processo de modelagem e especificação. O conhecimento é muitas vezes incompleto devido à complexidade. A medida que o ambiente automatizado vai sendo modelado, vai crescendo o conhecimento adquirido, a informação vai sendo detalhada. A interação dos usuários com o ambiente automatizado gera um fluxo contínuo de mudanças, da mesma forma que a operação do ambiente automatizado inserido no ambiente de negócio. Este fluxo contínuo de mudanças existe em cada um dos ciclos de realimentação (ciclos de avaliação), apresentados na figura 1. As mudanças são provocadas muitas das vezes, pelo fato de que as pessoas não conhecem a priori as suas necessidades futuras. Esta descoberta vai se dando na medida em que as pessoas interagem com o ambiente automatizado e vivenciam sua funcionalidade e serviços oferecidos. O ambiente externo se modifica, modificando as necessidades do ambiente de negócio e, conseqüentemente, as necessidades de sua infra-estrutura interna. O ambiente automatizado deve:

- garantir que cada um de seus "usuários" (o desenvolvedor, o usuário do sistema objetivo, o próprio sistema objetivo) e os ambientes de negócio no qual estão inseridos, utilizem seus próprios métodos e técnicas específicas, e que métodos, técnicas, "usuários" e ambiente de negócio possam evoluir em conjunto e em harmonia. O ambiente automatizado precisa de representações e linguagens de representação, com capacidade de especificar qualquer outra linguagem de representação, especificando métodos, ferramentas,

processos e requisitos. Isto se faz necessário para que a ferramenta gerada seja útil e utilizável.

- permitir capturar a especificação do ambiente e, então, gerar o ambiente a partir desta especificação, capacitando-se a gerar uma variedade de instâncias de ambientes customizados para as diferentes necessidades de uso. O ambiente automatizado precisa armazenar a sua especificação na base de conhecimento do sistema de software, e apresentar mecanismos para instanciar a especificação genérica de métodos, ferramentas, linguagens de representação para ambientes automatizados específicos. Isto garante ao ambiente automatizado flexibilidade, capacidade de evolução adequada aos problemas cuja solução é por ele apoiada, capacitando-o a apresentar maior velocidade de desenvolvimento e evolução.
- ter a capacidade de se modificar e evoluir com as modificações, evolução do processo de desenvolvimento, mudanças de requisitos e necessidades externas, habilitando-se a desenvolver de forma sistemática novos métodos de desenvolvimento de sistemas, novas linguagens de representação e novas técnicas. O ambiente deve ter a capacidade de definir sua estrutura a partir do processo, e deve apresentar maleabilidade e flexibilidade para modificar seu processo e estrutura em decorrência de modificações na sua relação com o meio externo (mudança de requisitos). Isto garante ao ambiente automatizado equilíbrio dinâmico, mantendo-o adequado ao meio externo, garantindo-lhe as propriedades de utilidade e utilizabilidade.
- apresentar características de uma abordagem sistêmica de integração de esforços, onde nenhuma das partes é mais fundamental do que as outras, e todas têm que ser compatíveis, capacitando-o a se modificar e incorporar novas ferramentas e tecnologias que vão surgindo no mercado.
- apresentar portatilidade e independência de plataforma tecnológica (hardware, software, comunicação). Isto se faz necessário para que o ambiente seja capaz de atender a novas necessidades pela incorporação de novas soluções tecnológicas de forma fácil e econômica.
- apresentar uma estrutura maleável que favoreça um regime de parcerias e alianças, na busca da colaboração mútua, permitindo que a comunicação entre as partes seja feita através de diversos meios físicos, capacitando-se a centralizar funções naturalmente compartilhadas e distribuir funções naturalmente customizáveis. O ambiente deve apresentar duas instâncias: uma com características integrativas, cujo papel é o de integração e coordenação das demais, executando funções naturalmente centralizadas; e outra com características auto afirmativas, cujo papel é o de produção, possivelmente em várias instâncias, executando funções diferenciadas, naturalmente distribuídas. Isto facilita a integração do ambiente automatizado com a organização do trabalho, qualquer que seja a estrutura usada pelo ambiente de negócio, além de garantir ao ambiente automatizado uma estrutura flexível, de evolução e manutenção fácil e econômica.
- modelar o conhecimento como percebido pelos "usuários" formando uma rede de objetos interconectados, onde são priorizadas as relações e interdependências entre objetos e malhas de realimentação, capacitando-o ao reúso de especificações e à evolução dinâmica. O ambiente automatizado deve ser capaz de registrar, modelar e armazenar os ambientes de negócio aplicação e engenharia de software, bem como as necessidades dos usuários do sistema objetivo e desenvolvedor. É preciso que este conhecimento modela-

do e armazenado por objetos fortemente interligados e interdependentes, que formam uma rede com capacidade de crescer ou diminuir dependendo das necessidades e do conhecimento adquirido. Isto facilita ao ambiente automatizado uma evolução em conjunto com o problema cuja solução está apoiando, fazendo com que se mantenha útil, utilizável e econômico.

- apresentar características distribuídas quanto ao: hardware, operando em plataformas diferentes; quanto ao software, construído para processos distribuídos; quanto aos dados, operando em uma rede de comunicação de dados, capacitando-o a apoiar o trabalho individual, em grupo, em múltiplos grupos, localizados em vários locais de um mesmo ambiente de negócio ou de múltiplos ambientes de negócio [PERRY88].
- deve apresentar uma organização multinível de unidades automatizadas, onde conceitos, valores, critérios válidos para uma unidade são válidos para todas. Cada unidade é um ambiente autônomo, com seu método, repositório, ferramentas, representações e linguagens de representação próprias, envolvendo uma pessoa ou um grupo de pessoas, dependendo da abrangência da unidade automatizada. Isto facilita ao ambiente automatizado lidar com a complexidade e abrangência dos problemas cuja solução é por ele apoiada.

Os requisitos internos do ambiente automatizado, decorrentes dos requisitos externos e enfoque sistêmico usado na modelagem, implicam no uso da tecnologia de meta ambientes no seu processo de construção.

4. Trabalhos Relacionados

As pesquisas de modelagem de ambientes, buscam nos requisitos internos do ambiente automatizado subsídios para a especificação de sua funcionalidade, características, propriedades e serviços a oferecer [FORTE92, BROWN92]. A forma de modelar muitas vezes não permite a evolução do ambiente automatizado de maneira fácil e rápida. O trabalho apresentado busca nos requisitos externos subsídios para a determinação dos requisitos internos do ambiente automatizado, especificando e construindo uma ferramenta a partir das necessidades de uso.

Os modelos de desenvolvimento encontrados na literatura, são na sua maioria, modelos rígidos, fixos, totalmente automatizados [FORTE92, JACCHERI93, FUGGETTA94]. Utilizam-se de processos de desenvolvimento que se desenrolam em uma só direção, não contemplam realimentações, não permitem a interação com o usuário, e por conseqüência não apresentam a capacidade de lidar com modificações freqüentes. É preciso um processo de aprendizado, de negociação e de resolução de conflitos, onde modificações constantes são necessárias para se convergir para uma solução de consenso. O presente trabalho procura identificar os pontos que precisam assegurar realimentação, onde é necessário garantir o aprendizado, para que através de um processo de modificações sucessivas o ambiente automatizado evolua corretamente com os problemas.

Os ambientes de engenharia de software centrados no processo [JACCHERI93, CONRADI94, EPOS95, WEAVER95] e meta ambientes [FINDEISEN94, STAA95, METAEDIT96] de maneira geral não modelam explicitamente a relação de parceria negócio aplicação / negócio engenharia de software, e não contemplam a geração das necessidades, como parte integrante do contexto de desenvolvimento. O presente trabalho focaliza tais pontos, melhorando

o entendimento do *que fazer* e *por que fazer*, mantendo o ambiente automatizado em equilíbrio dinâmico com o meio externo.

Lehman [LEHMAN96] apresenta um projeto de pesquisa (projeto FEAST - *Feedback, Evolution And Software Technology*, iniciado em outubro de 1996), dentro de uma abordagem mais ampla do desenvolvimento de software, que busca identificar, modelar e quantificar os ciclos de realimentação que influenciam o sistema de software e que são por ele influenciados. Ele defende a idéia de que é preciso uma abordagem multidisciplinar envolvendo questões técnicas, cognitivas e comportamentais, tanto da teoria da organização e gerenciamento, como da ciência da computação, para que se tenha resultados significativos de melhorias na produção de software. O presente trabalho apresenta o início de uma solução que caminha dentro de uma abordagem bem mais abrangente para a interação entre o domínio aplicação e o domínio de desenvolvimento.

5. Conclusões

A direção proposta neste trabalho, embora complexa, é uma forma viável de atacar o problema, na medida que favorece o seu entendimento e facilita a evolução da solução e o reúso de conhecimento, pela visualização, análise e modelagem de uma interação bem mais abrangente entre o domínio aplicação e o domínio de desenvolvimento, uma interação de negócio para negócio, modelando explicitamente a relação de parceria entre os dois ambientes de negócio. O uso de técnicas de análise de negócios favorece uma visão do todo e não apenas de parte do problema ou de sua solução. É a proposta de uma nova abordagem de solução.

Com a visão do contexto do desenvolvimento do software, que parte da interface negócio engenharia de software / negócio aplicação, a geração das necessidades, quer sejam dos indivíduos, quer sejam de seus negócios, quer sejam de construção do software, estão inseridas no contexto de desenvolvimento. A especificação de requisitos ao invés de ser rígida, evolui junto com os ambientes envolvidos e com o relacionamento entre ambos, mantendo o ambiente automatizado em equilíbrio dinâmico com o meio onde se acha inserido. O trabalho apresenta por exemplo, os pontos onde ocorrem realimentações, provocando sucessivas alterações. Embora o modelo, neste momento, não esteja preocupado com técnicas de implementação, técnicas de gerência de configuração e de controle de versão permitirão manter sob controle a evolução, que passará a se dar em passos discretos.

O modelo contempla ciclos de realimentação em cada uma das interfaces geradoras de necessidades quer seja interno ao ambiente de negócio, quer seja externo. A realimentação conduz a um aprendizado, que se traduz em um processo sucessivo de alterações, resultado de uma mudança de requisitos, que provoca a evolução do ambiente automatizado, quer seja para modificação das ferramentas em uso em cada ambiente de negócio, quer seja na modificação da forma de execução do trabalho e conseqüentemente de sua organização.

Um ambiente de requisitos evolutivo e que não utilize ferramentas é um ambiente de elevado custo, com uma freqüência grande de falhas, sendo difícil garantir a integridade e consistência das informações. Paralelamente a isto, boa parte da sistematização das atividades do processo de geração pode ser perdida. Junte-se a isto o fato de que esforços são gastos na tentativa de organizar e manter atualizadas as informações coletadas. O uso de ferramentas também vem a oferecer a validação automática dos requisitos e a geração da apresentação da especificação

em sua forma final. No entanto, a multitude de domínios torna economicamente inviável o desenvolvimento de ferramentas sob medida, a menos que se disponha de um meta-ambiente que facilite e agilize a instanciação de ferramentas adequadas simultaneamente ao negócio engenharia de software e ao negócio aplicação. Este trabalho visa portanto contribuir com a determinação de requisitos para a construção de tais ferramentas que facilitem a elicitação, registro e controle de requisitos.

Bibliografia

[AMANA93] "Em Busca de uma Administração mais Integrativa", *Amana Desenvolvimento e Educação*, Amana Key, Leadership Review Janeiro/Fevereiro 1993.

[AURÉLIO86] Buarque de Holanda Ferreira, Aurélio "Novo Dicionário da Língua Portuguesa", *Editora Nova Fronteira*, 1986.

[BOEHM94] Boehm, B.; Bose, P.; Horowitz, E.; Lee, M.J.; "Software Requirements as negotiated with conditions", In *IEEE International Conference on Requirements Engineering*, pg. 74-83, IEEE Computer Society Press, 1994.

[BROWN92] Brown, A. W.; Earl, A.N.; McDermid, J.A.; "Software Engineering Environments - Automated Support for Software Engineering", *McGraw Hill Book Company*, 1992.

[BUSTARD96] Bustard, D.W. "RUNES - Requirements Engineering Process & Tools", *URL: http://www.infc.ulst.ac.uk/informatics/ise/se/re/runes.html*.

[CONRADI93] Conradi, R.; Fernström, C.; Fuggetta, A. "A Conceptual Framework for Evolving Software Processes", PROMOTER book : "Software Process Modelling and Technology, October 1993, EPOS TR 187 URL: http://www.idt.unit.no/~epos/ bibliografia.html/.

[DARIMONT95] Darimont, R. "Process Support for Requirements Elaboration", Thèse présentée en vue de l'obtention du grade de Docteur en Sciences Appliquées, *UCL, Université Catholique de Louvain*, 23 juin 1995.

[DAVENPORT94] Davenport, T.H.; "Reengenharia de Processos", *Editora Campus*, 1994.

[DAVIS91] Davis, S.; Davidson, B.; "Visão 2020 - Administrando sua Empresa Hoje para Vencer Amanhã", *Editora Campus*, 1991.

[EPOS95] "EPOS Home Page", URL: http://www.idt.unit.no/~epos/OVERVIEW/epos/ epos.html, February 1995.

[FINDEISEN94] Findeisen, P. "The Metaview System", Department of Computing Science, University of Alberta, URL: *http://web.cs.ualberta.ca/~softeng/Metaview/ system/documentation.html*, June 1994.

[FINKELSTEIN92] Finkelstein, A.; Kramer, J.; Nuseibeh, B.; Goedicke, M. "ViewPoints: A FrameWork for Integrating Multiple Perspectives in System Development" *International Journal of Software Engineering and Knowledge Engineering*, 2(1), pp 31-58, Mar 1992.

[FORTE92] Forte, G.;.Norman, R. J.; "A Self-Assessment by the Software Engineering Community", *Communications of the ACM*, pp28-32, April 1992.

[FUGETTA94] Fuggetta, A.; Ghezzi C. "State of the Art and Open Issues in Process-Centered Software Engineering Environments", *J.Systems and Software*, pp 53-60, 1994.

[GHEZZI91] Ghezzi, C.; Jazayeri, M.; Mandrioli, D.; "Fundamentals of Software Engineering", *Prentice-Hall International Inc*, 1991.

[GOLDIN94] Goldin, L.; "A Method for Aiding Requirements Analysts in Requirements Elicitation for Large Software Systems"; PhD Thesis, *Israel Institute of Technology*, July 1994.

[HENDERSON93] Henderson, J.C.; Venkatraman, N., "Strategic Alignment: Leveraging information technology for transforming organizations", *IBM Systems Journal*, vol 32,nº1, pag 04 - 16, 1993.

[HUMPHREY89] Humphrey, W.S.; *Managing the Software Process*; Addison-Wesley; 1989

[INFOESTRATÉGIA93] "Conquistando Competitividade por meio da Tecnologia de Informação", *Amana Desenvolvimento e Educação*, Dezembro de 1993.

[JACCHERI93] Jaccheri, M.L.; Conradi, R. "Techniques for Process Model Evolution in EPOS", IEEE TSE on Software Process Model Evolution, December 1993, EPOS TR 188, URL: *http://www.idt.unit.no/~epos/bibliografia.html.*

[KAST92] Kast, F.E.; Rosenweig, J.E., "Organização e Administração - Um Enfoque Sistêmico", vol 1e 2, *Enio Matheus Guazzelli & Cia Ltda*, 1992.

[KEEN93] Keen, P.G.W., "Information Tecnology and the management difference: A fusion map", *IBM Systems Journal*, vol 32, nº1, pag 17 - 39, 1993.

[LEHMAN96] Lehman, M.M.; "Process Improvement - The Way Forward", *Brazilian Symposium on Software Engineering*, SBES'96, São Carlos, 18 oct 1996.

[LUFTMAN93] Luftman, J. N.; Lewis, P. R.; Oldach, S. H. "Transforming the enterprise: The alignment of business and information technology strategies", *IBM Systems Journal*, vol.32, nº1, pag 198-221, 1993.

[METAEDIT96] "MetaEdit+: Linking Software Development to Business Modeling", Parallel Performance Group, URL: *http://www.jsp.fi/metacase/start.html*, May 1996.

[NIERSTRASZ95] Nierstrasz, O.; Tsichritzis, D.; *Object-Oriented Software Composition*; Prentice Hall; 1995.

[NUSEIBEH93] Nuseibeh, B.; Kramer, J.; Finkelstein, A., "ViewPoints: A Vehicle for Method and Tool Integration", IEEE Proceedings of International Workshop on CASE (CASE92), July 1992, URL: *http://www-dse.doc.ic.ac.uk/dse-papers.*

[PINE 94] Pine II, B. J., "Personalizando Produtos e Serviços - Customização Maciça, a Nova Fronteira da Competição dos Negócios", *MAKRON Books do Brasil Editora Ltda* 1994.

[REAIMS95] "REAIMS - Requirements Engineering Adaption and Improvement for Safety and Dependability", ESPRIT Project 8649, *URL http://www.comp.lancs. ac.uk/ computing/research/cseg/projects/reaims.*

[ROMBACH90] Rombach, H.D. "Software Specifications: A Framework", SEI Curriculum Module SEI-CM-11-2.1, SEI/CMU, January 1990. URL:*ftp://ftp.sei.cmu. edu/pub/education.*

[SERRANO97] Serrano, M.A.B. "Análise de Negócio Aplicada à Modelagem de Meta Ambientes Automatizados", *Tese de Doutorado*, Departamento de Informática, PUC-Rio, 1997

[STAA95] Staa, A. V.; Cowan, D.D. "An Overview of the TOTEM Software Engineering Meta-Environment", *Monografias em Ciência da Computação*, 35/95, Departamento de Informática, PUC-Rio, 1995.

[WEAVER95] "PROCESS WEAVER - General Information Manual - Version PW2.1", Cap Gemini Sogeti, URL:*http://mark.cginn.cgs. fr:4747/PW1.html/*, March 1995.