Universidade Federal de Pernambuco
Centro de Informática

Graduação em Ciência da Computação

Um estudo comparativo entre SPEM e BPMN como padrões para modelagem de Processos de Software

Manoel Gilvan Calou de Araújo e Sá Filho

TRABALHO DE GRADUAÇÃO

Recife, agosto de 2007

Universidade Federal de Pernambuco
Centro de Informática

Manoel Gilvan Calou de Araújo e Sá Filho

Um estudo comparativo entre SPEM e BPMN como padrões para modelagem de Processos de Software

Monografia apresentada ao Centro de Informática da Universidade Federal de Pernambuco, como requisito parcial para obtenção do Grau de Bacharel em Ciência da Computação.

Orientador: Alexandre Marcos Lins de Vasconcelos
Co-orientador: Sandro Ronaldo Bezerra Oliveira

Recife, agosto de 2007

À minha família e amigos por serem os principais responsáveis por tudo que me tornei e conquistei.

Agradecimentos

Em primeiro lugar, agradeço a Deus por me proporcionar a oportunidade de estar cumprindo mais esta etapa tão importante de minha vida.

À minha família por ter me ajudado na minha formação educacional e moral, sendo fonte dos meus princípios e valores, e também, de minha felicidade. Tendo sempre me mostrado que sou capaz de realizar meus sonhos.

Aos meus amigos, pelas palavras de apoio, pelos momentos de descontração e união, que me fazem sentir querido.

À minha namorada, Karen, por sua compreensão e carinho, e por seu incentivo que me impulsionou e deu forças para realizar este trabalho.

Ao meu orientador professor Alexandre Vasconcelos por suas orientações e ajuda no desenvolvimento do trabalho.

Ao meu co-orientador Sandro Oliveira, pelo suporte constante ao longo deste período.

Aos colegas de trabalho, aos professores e funcionários do Centro de Informática da UFPE e a todas as outras pessoas que de alguma forma contribuíram, ao longo de minha vida, para que alcançasse mais este objetivo.

Muito obrigado.

Resumo

Com o objetivo de controlar e gerenciar o desenvolvimento de software, as empresas têm se preocupado com a definição de processos de desenvolvimento de software. Para tanto, a modelagem de processos auxilia na sua definição, além de permitir um maior entendimento do processo por todos os seus participantes. Nesse contexto, encontra-se o SPEM (*Software Process Engineering Metamodel*), como um dos padrões para modelagem de processos de software mais difundidos e aceitos.

Da mesma forma, o Gerenciamento de Processos de Negócio está relacionado com a definição, evolução e integração dos processos de negócio de uma organização, alinhando-os com as estratégias e objetivos da corporação. Visando ao suporte à definição e modelagem desses processos, foi desenvolvido o BPMN (*Business Process Modeling Notation*), uma notação para modelagem de processos de negócio de grande reconhecimento.

Este trabalho tem por objetivo principal a comparação entre SPEM e BPMN na modelagem de processos de software, defendendo, para isso, a idéia que o processo de software é também um dos processos de negócio de uma organização.

Palavras-chave: Processos de Software, Modelagem, Processos de Negócio, SPEM, BPMN.

Abstract

With the objective to control and to manage the software development, the companies have been worried about the definition of software processes development. In such a way, the process modeling assists in its definition, beyond allowing a greater understanding of the process by all its participants. In this context, it meets SPEM (Software Process Engineering Metamodel), as one of the most spread out and accepted standards for software processes modeling.

In the same way, the Business Processes Management is related with the definition, evolution and integration of the business processes of an organization, lining up them with the strategies and objectives of the corporation. Aiming to the support for the definition and modeling of these processes, BPMN was developed (Business Process Modeling Notation), a great recognition notation for business processes modeling.

This work has for main objective the comparison between SPEM and BPMN in the modeling of software processes, defending, for this, the idea that the software process is also one of the business processes of an organization.

Keywords: Software Processes, Modeling, Business Processes, SPEM, BPMN.

Sumário

Lista de figuras

Lista de Tabelas

1. Introdução

As crescentes globalização e competição no mercado de desenvolvimento de software têm gerado nas empresas de TI (Tecnologia da Informação) uma incessante busca pela qualidade no desenvolvimento de software. O caminho para alcançar essa qualidade passa por um bom gerenciamento do projeto, promovido com a ajuda de um Processo de Desenvolvimento e Manutenção de Software bem definido, compreendido e aceito por todos os envolvidos em sua implantação. [Araujo, 2004]

Dada a necessidade do bom entendimento do processo de desenvolvimento por aqueles que dele fazem parte, devem ser criados mecanismos que auxiliem nessa compreensão. Contribuindo para esse objetivo, foram desenvolvidos meta-modelos para a especificação de processos de software [Martins, 2004], dentre os quais o mais difundido e aceito é o *Software Process Engineering Metamodel* (SPEM) [SPEM, 2005].

Uma alternativa ao uso de meta-modelos para modelagem de processos de software é o uso de uma abordagem de Gerenciamento de Processos de Negócio como forma de alinhar os processos de desenvolvimento aos objetivos da organização. [Araujo, 2004] Nesse sentido, o *Business Process Management Initiative* (BPMI) desenvolveu um padrão para Notação de Modelagem de Processos de Negócio (BPMN - *Business Process Modeling Notation*). [BPMN, 2004]

Dessa forma, é importante analisar quais as os pontos fortes e fracos de cada abordagem, dependendo do contexto em que a organização de desenvolvimento de software está inserida, como ferramenta para decidir que metodologia para definição e modelagem dos processos de desenvolvimento de software será adotada.

1.1. Motivação

Os padrões para a modelagem de processos, seja de negócio ou de software, em particular SPEM e BPMN no caso deste trabalho, estão cada vez mais em evidência, devido à sua contribuição na definição, acompanhamento e evolução dos processos utilizados por uma organização.

O uso de padrões para Modelagem de Processos de Negócio como meio para as empresas de TI (Tecnologia da Informação) especificarem seus processos de desenvolvimento proporcionam a elas o conhecimento, gerência e evolução dos seus modelos de negócio [Araujo, 2004]. O confronto dessa abordagem com padrões já bem difundidos para a definição de processos, como o SPEM, pode levar à conclusão de qual metodologia conferirá maior produtividade e qualidade aos seus processos de desenvolvimento.

1.2. Objetivos

Este Trabalho de Graduação se propõe a apresentar um estudo comparativo entre dois padrões que despertam um crescente interesse na área de Engenharia de Software para modelagem de processos, o BPMN para processos de negócio e o SPEM para processos de software. O estudo tem o objetivo de ser um guia para que empresas que desenvolvam software possam fundamentar sua escolha pela definição do seu Processo de Software utilizando uma abordagem de Gestão de Processo de Negócio ou a tradicional, direcionada especificamente à Engenharia de Processo de Software.

1.3. Metodologia

O trabalho proposto por esse documento foi realizado através da pesquisa e estudo de literatura relacionada ao uso de SPEM e BPMN, que são mais bem detalhados no Capítulo 3, na modelagem e definição de processos, especialmente processos de desenvolvimento de software, a fim de reunir elementos suficientes à análise comparativa desses dois padrões na especificação de processos de software. Posteriormente, foi feito um mapeamento entre os elementos de modelagem dos dois padrões.

Esse mapeamento serviu de fonte de ferramenta para um estudo de caso, realizado modelando-se o processo de software de uma empresa através das duas notações. De posse do mapeamento e com os resultados do estudo de caso, foi possível promover um comparativo entre SPEM e BPMN, a partir da análise de algumas propriedades, no que diz respeito à modelagem de processos de software.

1.4. Estrutura do Trabalho

Além deste capítulo introdutório, este trabalho apresenta mais 4 (quatro) capítulos:

- Capítulo 2 - Processo de Software: Uma Visão Geral, esse capítulo apresenta conceitos e fundamentos essenciais para o bom entendimento do restante do trabalho;
- Capítulo 3 - Abordagens para a Modelagem de Processo de Software, no qual são discutidos conceitos que norteiam as duas abordagens para modelagem de processos de software confrontadas neste trabalho;

- Capítulo 4 - SPEM versus BPMN na Modelagem de Processo de Software, que tem como objetivo principal, apresentar uma análise comparativa entre os padrões, apoiada no mapeamento proposto entre eles e no estudo de caso realizado utilizando-se as duas notações;
- Capítulo 5 - Conclusões, capítulo que encerra o trabalho, trazendo conclusões acerca das contribuições do mesmo, apresentando também sugestões de continuidade através de trabalhos futuros.

Por fim, listamos as referências bibliográficas utilizadas neste trabalho.

2. Processo de Software: Uma Visão Geral

Na Engenharia de Software, uma das áreas que vêm recebendo maior atenção e despertando maior interesse em pesquisas é a tecnologia de processo de desenvolvimento de software, a qual será chamada, de agora em diante, apenas como processo de software. Esta tecnologia surgiu na década de 1980 e tem seu surgimento intimamente relacionado com os primeiros esforços de se estabelecer um ciclo de vida para o desenvolvimento de software, isto é, um conjunto bem definido de estados que possibilitem o acompanhamento do desenvolvimento [Oliveira, 06] [Reis, 2003] [Reis, 2004].

Podemos entender processo de software como um conjunto de atividade parcialmente ordenado empreendidas para transformar os requisitos dos usuários em soluções de software. Essas **atividades** são executadas por pessoas que ocupam determinado **papel** específico, podem consumir ou produzir **artefatos** (código fonte, documentos, diagramas, etc.) [Reis, 2004]. A figura 2.1 ilustra como se dá, a partir dos requisitos iniciais relacionados ao problema, a escolha de um modelo de processo de software que suportará o desenvolvimento, e posterior manutenção, do software que se destina a resolver o problema inicial.

O surgimento das ferramentas CASE (*Computer Aided Software Engineering*), que auxiliam no desenvolvimento de software, propiciou o advento dos primeiros ambientes de desenvolvimento de software – ADS, que se utilizam de ferramentas CASE para dar suporte a todo o processo de desenvolvimento de software. Mais recentemente, sentiu-se a necessidade de acrescentar a esses ADSs, ferramentas que apoiassem o controle e gerenciamento do processo de software, o que proporcionou o surgimento dos PSEEs (*Process-Centered Software Engineering Environments* ou ADSs Orientados ao Processo), permitindo a definição, modelagem e contínuo acompanhamento das atividades do processo [Oliveira, 06] [Reis, 2003].

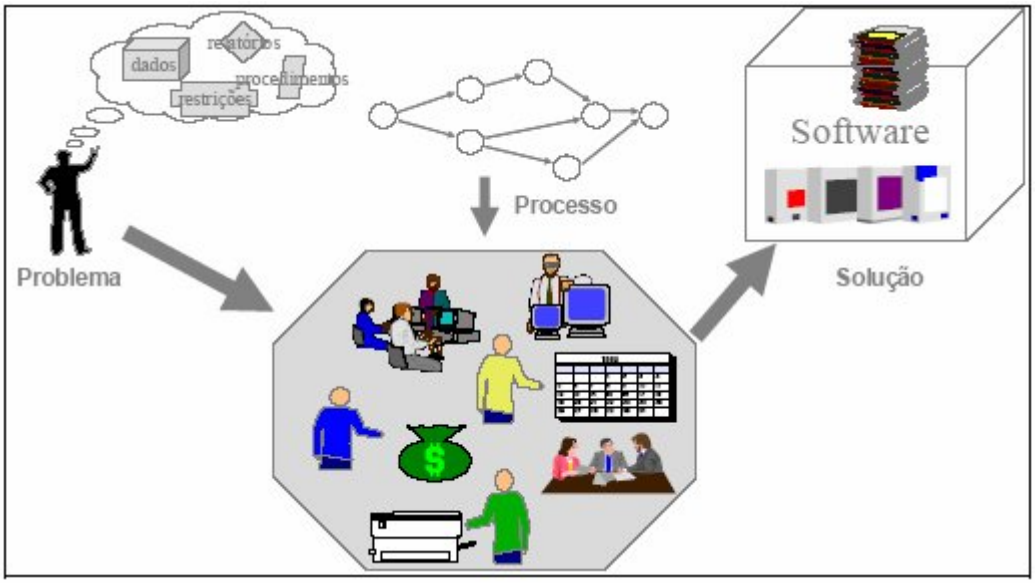


Figura 2.1 - Processo de Desenvolvimento de Software [Reis, 2004]

2.1. Fundamentos da Tecnologia de Processo de Software

A tecnologia de processo de software tem por objetivo possibilitar o desenvolvimento de softwares com a maior qualidade possível, para isso, é importante que ela esteja submetida ao rigor da Engenharia de Software. Dessa forma, o processo de software deve passar por etapas como especificação, análise, testes, medição e avaliação, entre outras, cuja finalidade é disciplinar o processo e melhorar o produto de software resultante. Para atender a essas etapas, faz-se necessária a utilização de linguagens de processo de software, permitindo descrever e automatizar o processo. Nesse contexto, a modelagem e execução de processos de software são fundamentais para alcançar maiores níveis de qualidade no desenvolvimento de software, o que tem levado a um aumento de pesquisas na área no desenvolvimento de ferramentas e ADSs que possibilitem esse objetivo [Reis, 2004].

Conforme dito anteriormente, pode-se entender processo de software como um conjunto de atividades necessárias para obter, a partir dos requisitos iniciais do usuário, uma solução em software. O processo de software é formado por um

conjunto de passos parcialmente ordenados relacionados a artefatos, pessoas, estruturas organizacionais e restrições, com a finalidade de produzir e manter a solução em software requerida [Reis, 2004].

Os passos do processo de software são atividades ou tarefas. Atividades são passos do processo gerenciados e tarefas são passos elementares (ações atômicas realizadas por uma organização), que conduzem à realização de uma atividade, e não são gerenciadas. Atividades incorporam e implementam procedimentos, regras e políticas, e têm como objetivo gerar ou modificar um dado conjunto de artefatos. Elas estão, freqüentemente, organizadas em redes bi-dimensional, como pode ser visto na figura 2.2 e estão associadas com papéis, ferramentas e artefatos [Oliveira, 06].

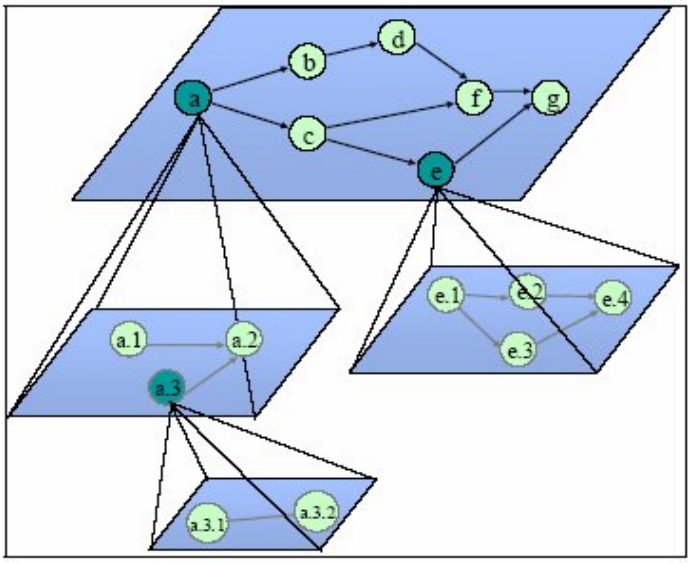


Figura 2.2 – Redes de Atividades

Uma **atividade** aloca recursos (por exemplo, computadores, impressoras e material de expediente), é escalonada, monitorada e atribuída a membros da equipe do projeto (agentes), que podem utilizar ferramentas para executá-la. Uma atividade também pode ser executada somente por ferramentas automatizadas, sem intervenção humana. A descrição da atividade pode especificar os artefatos necessários, as relações de dependência com outras atividades, a prioridade de

execução, o nível de complexidade, as datas de início e fim planejadas, os recursos a serem alocados e os agentes responsáveis pela mesma [Reis, 2004].

Um **agente** está relacionado com as atividades de um processo e pode ser uma pessoa ou uma ferramenta automatizada (quando a atividade é automática). Os agentes podem estar organizados em cargos, que possibilitarão diferentes percepções e responsabilidades sobre o que acontece durante o processo de software. Um agente gerente, por exemplo, perceberá os aspetos de controle e alocação de recursos e cronogramas para atividades, enquanto um desenvolvedor perceberá as suas atividade como atribuições que devem ser feitas para produzir um resultado (possivelmente através de agendas, onde as atividades em que ele está envolvido são relacionadas) [Oliveira, 06].

Um **artefato** é um produto criado ou modificado durante um processo. Tal produto é resultado de uma atividade e pode ser utilizado posteriormente como matéria-prima para a mesma ou para outra atividade a fim de gerar novos produtos, ou ainda pode ser parte do produto final de software, objetivo do processo de desenvolvimento. Desta forma, uma atividade pode consumir produtos (de entrada) e gerar novos produtos (de saída). Os produtos são freqüentemente persistentes e possuem versões [Reis, 2004].

A realização do processo é afetada pelas **restrições**, que podem atingir atividades, agentes, recursos, artefatos, papéis e seus relacionamentos. Uma restrição é uma condição definida que um passo de processo deve satisfazer antes ou depois de ser executado [Reis, 2004].

2.2.Meta-Processo de Software

Um processo de software e seu modelo têm uma natureza evolucionária, devido à necessidade de melhoria e correção contínua e devido à instabilidade do ambiente operacional. Em outras palavras, o modelo é primeiro estabelecido para representar o mundo real inicial, e, durante seu tempo de vida, é exposto a

mudanças causadas por eventos planejados e não-planejados internos e externos à organização, por isso, faz-se necessário um ciclo de vida para processo de software análogo ao ciclo de vida de produtos de software. As atividades do ciclo de vida de processo de software são chamadas de meta-atividades, e o processo de desenvolvimento e evolução de processo de software são denominados meta-processo [Oliveira, 06].

As fases do meta-processo são representadas em alto nível de abstração, de forma que cada fase pode ser decomposta em subfases de pouca granularidade. A seguir são descritas as meta-atividades básicas [Reis, 2003]. A interação destas pode ser vista na figura 2.3:

- **Provisão de Tecnologia:** a tecnologia de suporte a produção de software e de modelos de processo deve ser provida, incluindo as linguagens de modelagem de processo, modelos de processo prontos para reutilização e ferramentas para aquisição, modelagem, análise, projeto, simulação, evolução, execução e monitoração de modelos de processo;

- **Análise de Requisitos do Processo:** nesta fase são identificados requisitos a atividades de projeto de um novo processo, ou novos requisitos para um processo existente. Os requisitos resultantes especificam os recursos e propriedades que o processo deve oferecer;

- **Projeto do Processo:** provê a arquitetura geral e detalhada do processo. Nesta etapa as linguagens de modelagem do processo são utilizadas, havendo necessidade que satisfaçam aos requisitos;

- **Implementação ou Instanciação do Processo:** implementa a especificação do processo produzida pela atividade anterior. Nesta fase, é gerado um modelo de processo instanciado, contendo

informações detalhadas sobre prazos, agentes e recursos utilizados por cada atividade definida no processo;

- **Simulação do Processo:** a simulação ocupa um papel chave na verificação e validação dos processos definidos. A simulação é uma tarefa que, geralmente, é acompanhada tanto pelo projetista de processo quanto pelo gerente do desenvolvimento com o objetivo de antever o comportamento do projeto (execução do processo);

- **Execução do Modelo de Processo:** nesta etapa, o modelo de processo instanciado é executado através da invocação de ferramentas para guiar e assistir a realização do processo no mundo real. Informações sobre o andamento do processo (*feedback*) são coletadas e analisadas durante a execução;

- **Avaliação do Processo:** provê informação quantitativa e qualitativa, descrevendo o desempenho de todo o processo em execução. Esta fase ocorre simultaneamente à execução do modelo de processo e as informações adquiridas são utilizadas na atividade de análise de requisitos.

No meta-processo, é necessária a participação dos agentes humanos que operam na fase de execução do processo. Além disso, para que o processo seja modelado é requerida a participação de um projetista de processo, que é o responsável por descrever o processo a ser executado e um gerente do processo que deverá acompanhar a execução e a avaliação do processo, analisando seu desempenho. Em geral, os PSEEs estabelecem seu meta-processo através da provisão de tecnologia e paradigmas adotados para modelagem e execução de processos [Oliveira, 06].

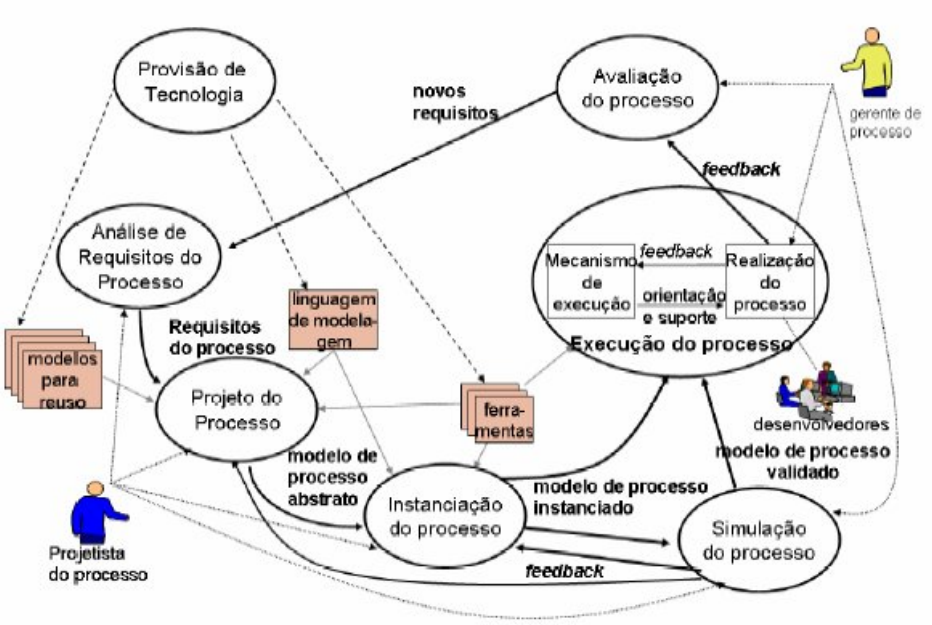


Figura 2.3 – Ciclo de vida de processo de software [Reis, 2003]

2.3. Definição de Processo de Software

Dentro de uma organização, cada projeto de desenvolvimento de software possui peculiaridades e características próprias, no entanto, sendo todos da mesma empresa, eles possuem muitas características em comum, tais como culturas organizacionais, recursos, etc. Tendo em vista esse cenário, é importante a definição de um processo de desenvolvimento padrão para a empresa, o que facilitará muito a definição de processos especializados para cada projeto, pois eles seguirão grande parte do que foi definido no processo padrão. A definição do processo padrão terá como base o meta-modelo de processo de software e levará em conta as características da organização.

Considerando que tipos de softwares diferentes apresentam características distintas e podem exigir abordagens de desenvolvimento diferentes, é necessário que o processo padrão da organização seja especializado, considerando os vários tipos de software que podem ser desenvolvidos pela organização (por exemplo, sistemas de informação), bem como os paradigmas de desenvolvimento (por

exemplo, orientação a objetos). Dessa forma, durante a etapa de especialização do processo de desenvolvimento, atividades podem ser adicionadas ou excluídas dependendo do contexto de desenvolvimento para o qual está sendo realizada a especialização.

Para o desenvolvimento de um projeto, é necessário considerar mais algumas variáveis para definir o processo de desenvolvimento, tais como, modelo de ciclo de vida, métodos e ferramentas utilizadas pelo projeto, os recursos humanos e materiais e suas responsabilidades dentro do processo, e os artefatos (produtos) consumidos e produzidos. A influência desses fatores é refletida na adaptação do processo especializado para o projeto, o que nada mais é que a sua instanciação [Mendes, 2005].

2.4. Modelagem de Processo de Software

O **modelo de processo de software** se refere à definição de processos como modelos, ou seja, é uma descrição abstrata do processo de software. A modelagem de processos pode receber suporte automatizado. Vários tipos de informação devem ser integrados em um modelo de processo de software para indicar quem, quando, onde, como e por que os passos são realizados. Para representar um modelo de processo de software é utilizada uma linguagem de modelagem do processo de software, a qual deve oferecer recursos para descrever e manipular os passos do processo [Acuña, 2001] [Oliveira, 06] [Reis, 2003].

Um **modelo do processo instanciado** ou **processo executável** é um modelo de processo pronto para execução. O modelo instanciado é considerado uma instância de um modelo abstrato, com objetivos e restrições específicos, envolvendo agentes, prazos, orçamentos, recursos e um processo de desenvolvimento. Este modelo pode ser interpretado por uma máquina de processo, a qual é um componente provido por um ambiente de desenvolvimento

de software (ADS) com capacidade de suportar a modelagem e a execução de processo de software (ADS Orientado a Processo) [Silva, 2006].

Por sua vez, um **projeto** é a instância de um processo, com objetivos e restrições específicos. Um projeto se destina a desenvolver um produto de software e envolve uma estrutura organizacional, prazos, orçamentos, recursos e um processo de desenvolvimento. Dessa forma, a gerência de projetos tem como responsabilidades o planejamento, controle e monitoração de um processo em execução, enquanto que a gerência de processos preocupa-se com a definição, análise e verificação dos modelos de processo, utilizando-se de informações coletadas durante sua execução, para que os modelos possam ser evoluídos [Reis, 2004].

A modelagem de um processo de software deve ser conduzida de modo a possibilitar o entendimento e a padronização do processo, para tanto se devem observar os principais objetivos para se modelar o processo de software [Acuña, 2001]:

1. **Facilitar o entendimento e a comunicação:** o modelo de processo contém informação suficiente para sua representação. Isso formaliza o processo e serve de base para treinamento.

2. **Suporte ao gerenciamento do processo e controle sobre ele:** possibilita estimativas e planejamento, através da simulação da tomada de algumas decisões, comparando-se o processo definido com o processo em andamento.

3. **Prover orientação automatizada do processo:** proporciona um ambiente de desenvolvimento de software efetivo, fornecendo orientação aos usuários, instruções e material de referência.

4. **Prover execução automatizada do processo:** um ambiente automatizado (PSEE) pode controlar o comportamento do processo

definido, coletar métricas e reforçar as regras para garantir a integridade do processo.

Modelos de processo diferentes podem enfatizar diferentes pontos de vista. Por exemplo, um modelo de processo pode focar nos agentes envolvidos com cada atividade, enquanto que outro modelo pode enfatizar o relacionamento entre as atividades. Existe um modelo que se refere à cultura organizacional e foca nas capacidades ambientais e deveres dos papéis envolvidos no processo de software.

Isso significa que cada modelo observa, foca em ou prioriza pontos particulares no mundo tão complexo do desenvolvimento de software. Um modelo é sempre uma abstração da realidade, e tal como toda abstração, nem todos os aspectos do processo são levados em consideração em um único modelo. Por isso, em geral, um modelo de processo pode ser dividido em vários sub-modelos que expressam diferentes perspectivas e pontos de vista [Acuña, 2001].

2.4.1. Perspectivas da Modelagem de Processos de Software

Para que um processo de software seja descrito adequadamente, muitas formas de informação devem estar integradas. Freqüentemente, as informações encontradas em um modelo de processo são: o que será feito (quais as atividades), quem o fará (quais os agentes), quando, onde, como e por que será feito, e quem é dependente de cada passo realizado. As linguagens de modelagem de processos diferem nas respostas dadas a estas questões e apresentam uma ou mais perspectivas diferentes relacionadas a estas respostas. As perspectivas representadas mais freqüentemente são [Reis, 2004]:

- **Perspectiva Funcional:** representa quais passos do processo estão sendo executados, e quais fluxos de informação são necessários para esses passos;

- **Perspectiva Comportamental:** representa quando os passos do processo são executados, assim como aspectos indicando como esses passos são realizados através de iterações, condições de tomada de decisão, critérios de entrada e saída, dentre outros;

- **Perspectiva Organizacional:** representa onde e por quem (agentes) da organização os passos do processo são executados, além de descrever os mecanismos físicos de comunicação usados para transferência de entidades, e a mídia física e localizações usadas para armazenar entidades;

- **Perspectiva Informacional:** representa as entidades de informação produzidas ou manipuladas por um processo. Estas entidades incluem dados, produtos (intermediários e finais) e objetos. Esta perspectiva inclui a estrutura das entidades de informação e os relacionamentos entre elas.

Estas perspectivas, quando combinadas, produzem um modelo do processo de software integrado, consistente e completo. Entretanto, poucas linguagens e padrões suportam todas as perspectivas.

3. Abordagens para a Modelagem de Processo de Software

Este trabalho tem o objetivo de estudar duas abordagens distintas para a modelagem de processos de software: a tradicional, utilizando tecnologias especificamente desenvolvidas com esse intuito; e a abordagem que vê o processo de software como um processo de negócio, como tantos encontrados em uma organização. Neste capítulo, serão tratados alguns aspectos para modelagem de processos de software. Para entender as duas abordagens escolhidas, algumas tecnologias envolvidas serão vistas, para que, posteriormente, a escolha dos padrões SPEM e BPMN possa ser justificada.

Finalmente, uma breve apresentação dos dois padrões escolhidos para modelagem de processos envolvidos com as abordagens propostas será realizada, a saber, SPEM, voltado para a modelagem de processos de software e BPMN, notação para a modelagem de processos de negócio. A escolha dessas duas notações se dá pela crescente aceitação que elas vêm obtendo na área de modelagem de processos, evidenciada pelo suporte que ambas recebem por parte do *Object Management Group* (OMG) [OMG, 2007], uma organização que possui grandes empresas como membros, sendo mundialmente reconhecida, e que tem como um dos objetivos a manutenção de padrões de modelagem.

3.1. Abordagem Tradicional para Modelagem de Processo de Software

Tradicionalmente, as pesquisas por tecnologias de processo de software foram conduzidas considerando-se apenas o contexto de uma equipe de desenvolvimento de software, ou seja, tendo uma visão vertical do desenvolvimento de software. Com esse pensamento, foram desenvolvidos tecnologias e padrões para definição e modelagem de processo, dentre os quais se destaca, neste trabalho, o meta-modelo SPEM.

Considerando-se as tecnologias existentes que foram desenvolvidas nesse contexto, a que mais se destaca é a de linguagens para modelagem de processos. Em vista disso, serão apresentadas algumas características das chamadas Linguagens de Modelagem de Processos (*Process Modeling Languages* – PMLs).

3.1.1. Linguagens de Modelagem de Processos de Software

Dada à complexidade envolvida nos processos de software, bem como a analogia que pode ser feita entre o ciclo de vida do processo e o ciclo de vida do produto de software, a comunidade de engenharia de software sentiu a necessidade de desenvolver linguagens para a modelagem de processos, com o objetivo de formalizar a sua definição, bem como possibilitar o suporte a todo o seu ciclo de vida. Essas linguagens possibilitam a representação precisa e compreensível dos vários elementos envolvidos no processo de software, vistos na seção 2.1, ou seja, as PMLs são constituídas de elementos centrais, com semânticas próprias, que representam os elementos comuns do processo de software, e um conjunto de associações necessárias à construção de um modelo de processo [Abdala, 2004].

Para atender aos objetivos apresentados acima, as linguagens de modelagem de processos necessitam de um conjunto de capacidades básico. No entanto, não há um consenso na literatura relacionada, acerca de qual seria um conjunto mínimo de capacidades. Em geral, toda linguagem de processos procura definir representações para atividades e recursos. Um conjunto básico de requisitos pode ser proposto a partir da análise de algumas linguagens [Reis, 2004]:

- **Descrição de atividades:** para definir as etapas que compõem um processo;
- **Descrição dos artefatos:** para definir o sistema de software que está sendo construído, incluindo o código-fonte de todos os artefatos produzidos;

- **Descrição dos recursos:** para especificar os recursos humanos e computacionais disponíveis para utilização nas atividades;

- **Relacionamentos entre entidades:** para indicar os vários tipos de interconexões semânticas entre atividades, artefatos e recursos;

- **Gerenciamento da Consistência:** para assegurar a satisfação das condições requeridas acerca das atividades interligadas, artefatos, e recursos e através das atividades.

3.1.2. Classificação das Linguagens de Processos de Software

Para classificar as linguagens de processo, é importante entender a distinção entre as diferentes fases do ciclo de vida do processo. Durante a especificação de requisitos (análise do processo), os objetos principais são entender o processo e explicitá-lo, para facilitar sua comunicação e entendimento. O modelo resultante descreve o comportamento esperado e desejado do processo, os papéis a serem realizados e outros procedimentos de alto nível. Por outro lado, a etapa de implementação se preocupa em produzir um suporte automatizado para o processo que facilite a obtenção dos objetivos e metas que foram definidos na fase de requisitos. O resultado desta etapa geralmente é um código complexo que inclui muitos detalhes relativos à integração com ferramentas e codificação de passos de processo. A maioria dos detalhes destas características é irrelevante durante a análise ou projeto.

Assim, uma classificação importante para as linguagens de processo, refere-se à fase do ciclo de vida de processo suportada e o nível de abstração fornecido [Reis, 2004]:

- **Linguagens de Especificação de Processo:** são utilizadas na etapa de especificação dos requisitos do processo;

- **Linguagens de Projeto de Processo:** oferecem recursos para serem utilizadas durante o projeto do processo;

- **Linguagens de Programação (implementação) de Processo:** são definidas para serem utilizadas durante a programação e execução de processos.

Muitas linguagens apresentadas na literatura podem ser classificadas simultaneamente como de Especificação, de Projeto e de Implementação de Processo, como por exemplo, Merlin, Marvel e SPELL.

3.2. Abordagem de Modelagem de Processos de Software sob o ponto de vista dos Processos de Negócio

Nesta seção, será apresentada uma abordagem diferente para a modelagem de processos de software, visando à defesa da idéia de que um processo de software é também um dos processos de negócio de uma empresa. Para isso serão vistos os conceitos de uma importante tecnologia desenvolvida para modelagem de processos de negócio, *Workflow*, com o objetivo de identificar intersecções entre processo de software e de negócio. No entanto, a fim de fundamentar essa abordagem, é necessário entender o que é um processo de negócio, o que será feito na seção 3.2.2..

3.2.1. A tecnologia de *Workflow*

Nos últimos anos, diversas tecnologias surgiram com o objetivo de auxiliar o envolvimento cooperativo de profissionais nos mais diversos tipos de atividades humanas. Neste contexto, as tecnologias conhecidas como *Workflow* e Processo de

Software, dentre outras, se destacaram como tecnologias inter-relacionadas que estabeleceram comunidades próprias dentro da Ciência da Computação.

A intersecção entre essas duas tecnologias ainda causa muitas divergências entre os autores. Alguns consideram Processo de Software um caso particular de *Workflow* para tratar o desenvolvimento de software, e esta será a idéia defendida neste trabalho. Apesar da discussão em torno da relação entre as duas tecnologias, é fato que, embora apresentem, muitas vezes, terminologia conflitante, elas possuem o mesmo objetivo: apoiar o envolvimento conjunto de agentes humanos com o apoio de sistemas de computação, justificado inclusive, pelo fato das duas tecnologias apresentarem elementos chave comum: comunicação, colaboração e coordenação [Oliveira, 2006].

O *Workflow* tem por principal objetivo a modelagem, automação e melhoria de processos de negócio (*business processes*), um processo de negócio é formado por um conjunto de atividades interligadas que buscam atingir um objetivo relacionado ao contexto organizacional. A automação de um processo de negócio através de um *Workflow* consiste na representação do processo compreensível por uma máquina, o que é alcançado pela utilização dos chamados Sistemas de Gerência de *Workflow* (*Workflow Management System* – WFMS), sistemas que fornecem um conjunto de ferramentas de apoio à definição e execução de *Workflows,* na figura 3.1 podem ser vistas algumas das tarefas realizadas por WFMSs [Silva, 2006]. Ao contrário da tecnologia de processo de software, os sistemas de *Workflow* foram desenvolvidos, primeiramente, na indústria e não no meio científico, o que trouxe dificuldades com relação à criação de padrões, isso levou ao surgimento de diversos tipos de WFMSs para os diversos tipos de processo de negócio. Em vista disto, em 1993 foi criada a *Workflow Management Coalition*, reunindo os principais fornecedores, universidades e usuários para definição de um modelo de referência e padrões de interoperabilidade entre os produtos [Reis, 2004].

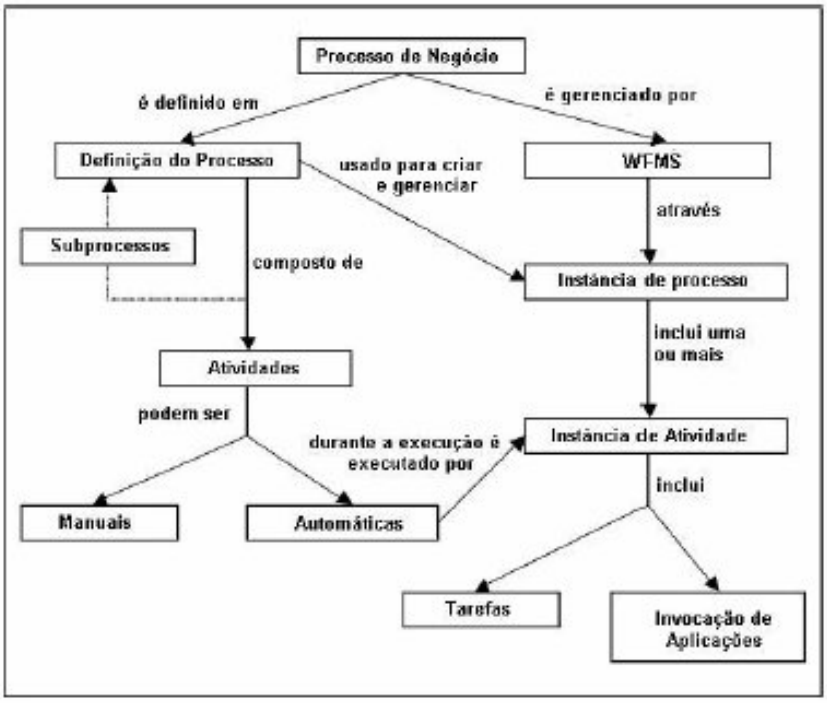


Figura 3.1 – Tarefas desempenhadas por um WFMS [Silva, 2006]

A seguir, será visto como *Workflow* pode apoiar uma abordagem diferente, baseada na Gestão de Processos de Negócio, para a definição e modelagem de processos de software.

3.2.2. Gestão de Processos de Negócio

A área de Gestão de Processos de Negócio tem surgido no mercado como uma necessidade das empresas atuais para conhecerem, gerirem e aprimorarem seus respectivos negócios. Os conceitos e atividades preconizadas por esta área de prática e também de pesquisa têm levado ao desenvolvimento de metodologias e ferramental para a gestão de processos em organizações.

Durante muito tempo, as empresas que desenvolviam software limitavam-se a desenvolver produtos para a maior quantidade de clientes possível, sem respeitar as necessidades individuais de cada um, o interessante era produzir mais. Com a evolução do mercado e o aumento da competição, impulsionado também

pela globalização, foi preciso olhar o cliente com outro ponto de vista, atualmente, é preciso conhecer o cliente, pois ele exige tratamento individualizado. Para atender às novas exigências impostas pelo mundo externo, as organizações procuraram conhecer melhor seu negócio, para isso é preciso formalizá-lo, facilitando seu entendimento e o controle sobre ele [Araujo, 2004].

Através da gestão dos processos de negócio é possível visualizar as atividades dos funcionários alinhadas dentro do contexto do negócio da organização. Isso ocasiona um ganho de eficiência da empresa na integração e evolução dos seus processos, pois estes estarão em concordância com as estratégias e metas da corporação [Filho, 2007].

A gestão de processos de negócio pode ser considerada um conjunto de métodos e práticas que auxiliam a organização na gestão do negócio através da identificação, entendimento e controle dos seus processos. Para isso, é necessário formalizar os processos de negócio buscando a obtenção do entendimento uniforme, e utilizando uma linguagem comum. Dessa forma, a modelagem de processos de negócio é fundamental nessa formalização, pois ajuda a responder às questões principais do processo de negócio: o que está sendo feito, por que está sendo feito, onde, por quem, quando e de que forma é feito [Araujo, 2004]. Pode-se dizer que o conjunto de métodos da gestão de processos de negócio possui as seguintes características [Filho, 2007]:

- Aperfeiçoamento contínuo ao invés da simples resolução de problemas;
- Orientação dos funcionários para uma visão horizontal (organizacional);
- Necessidades dos clientes direcionam os objetivos e tomadas de decisão;

- Gerentes de diversas funções recebem e produzem constantemente informações sobre processos intra e interfuncionais para os quais seus departamentos contribuem.

A figura 3.2 apresenta os principais elementos envolvidos na modelagem de processos de negócio de uma organização.

Nesse contexto, surge o termo BPM (*Business Process Management*), que pode ser definido como o desenvolvimento e manutenção de uma arquitetura de processos de negócio, que se utiliza de técnicas, metodologias, ferramentas e gerenciamento humano para garantir que os processos da organização sejam continuamente monitorados e melhorados. Dentre as tecnologias que dão suporte a BPM, encontram-se *Workflow* e o BPMN, que se propõe a realizar uma padronização para as notações de modelagem de processos de negócio, o que é um ponto falho da tecnologia de *Workflow*.

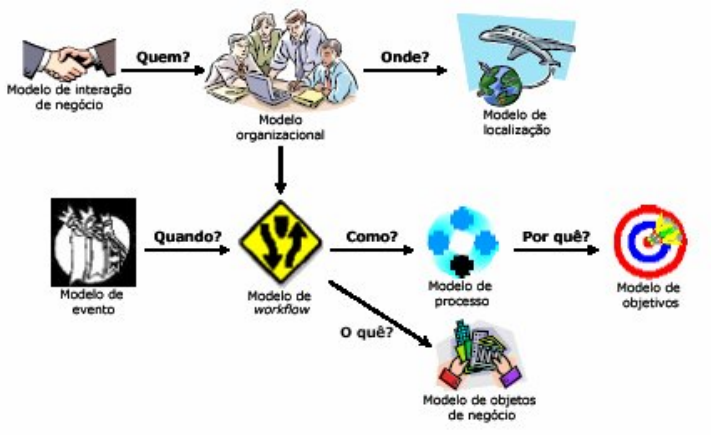


Figura 3.2 – Questões respondidas pela modelagem de processos de negócio [Araujo, 2004]

3.2.3. Modelagem de Processos de Software sob a ótica da Gestão de Processos de Negócio

Até boa parte do século XX, a realidade dentro das organizações era de forte divisão hierárquica, na qual cada unidade organizacional era responsável por determinadas atividades e se preocupava em aperfeiçoá-las independentemente dos outros setores da empresa. Isso quer dizer que as demais atividades destinadas a produzir o bem ou serviço de interesse do cliente não eram consideradas na estratégia de evolução de cada unidade. Essa abordagem ficou conhecida por divisão em "silos", o que sugere que cada departamento ou setor é isolado em seu próprio silo, não existindo relação entre as unidades, ou seja, sem gerenciamento entre os espaços "em branco" entre os silos [Filho, 2007].

A fim de superar essa visão que já não atende mais às necessidades do mercado, é necessário integrar e alinhar as diversas unidades organizacionais aos objetivos da empresa. Tendo em vista que a área de TI (Tecnologia da Informação) de uma empresa é também uma dessas unidades, com objetivos específicos, mas principalmente que têm por finalidade auxiliar a organização a atender as suas metas principais, é possível dizer que um processo de software pode ser considerado um caso específico de processo da organização. Dessa forma, pode-se concluir que a visão, conceitos, ferramentas e tecnologias, como *Workflow*, da área de Gestão de Processos de Negócio podem ser perfeitamente aplicadas à definição e, sobretudo, modelagem de processos de software, trazendo como um dos benefícios a possibilidade de alinhar o processo de software aos objetivos globais da organização [Araujo, 2004].

Para dar suporte a essa abordagem de definição e modelagem de processos de software, destacam-se algumas características dos WFMSs, que são importantes no apoio a esses processos [Silva, 2006]:

- **Modelagem de processos:** WFMSs oferecem linguagens de alto nível para a definição de processos, com suporte a edição gráfica, o que facilita a definição do processo;

- **Instanciação do modelo de processo:** os WFMSs permitem separar o modelo de processo de suas instâncias em execução, possibilitando maior flexibilidade na realização de alterações no processo de software, para que essas novas definições não afetem os processos em execução, ou então é possível que as instâncias em execução incorporem automaticamente as novas alterações no modelo;

- **Heterogeneidade:** Através de interfaces previamente definida, os WFMSs permitem a integração de sistemas de *Workflows* diferentes, podendo até ter um processo executado por outro sistema de *Workflow*. Isso é bastante útil em ambientes de desenvolvimento de softwares multidisciplinares, que podem ser distribuídos e envolver diversos setores de uma organização. Esse recurso está diretamente relacionado à idéia da visão horizontal de uma empresa, apresentada na seção 3.2.2;

- **Execução dos processos:** corresponde à instanciação do processo e suas atividades. No contexto de desenvolvimento de software, auxilia os participantes do processo a entenderem seu papel e visualizarem o andamento do processo. Além disso, as atividades de cada participante indicam suas responsabilidades, o que favorece o bom andamento do processo;

- **Supervisão:** a disponibilização de informações acerca do andamento do processo, fornece meios para a sua coordenação, supervisão e monitoramento, o que pode ser usado para encontrar futuros pontos de melhorias no processo;

Vale ressaltar, dentre essas características, a definição de processos de software multidisciplinares, ou seja, que envolvam não só uma equipe de desenvolvimento, mas também outros departamentos de uma organização, o que pode ser bastante útil no desenvolvimento de projetos mais complexos que necessitem agregar informações e regras relativas a diversas áreas ao processo.

Nas próximas seções, serão apresentados os padrões para modelagem de processos escolhidos para apoiar as duas abordagens de modelagem discutidas nesse trabalho, para que no próximo capítulo, um estudo comparativo entre os dois padrões possa ser realizado.

3.3.SPEM

O SPEM é um meta-modelo para definição e modelagem de processos de software, mantido pela OMG desde novembro de 2002. A execução de processos de software não está no escopo de sua especificação. Atualmente, o SPEM está na versão 1.1 de janeiro de 2005. [Oliveira, 2006] [Mendes, 2005] [SPEM, 2005] [Combemale, 2006].

No seu modelo conceitual, o SPEM está fundamentado na idéia principal de que um processo de desenvolvimento de software é uma colaboração entre entidades abstratas ativas, chamadas papéis do processo (*process roles*), que realizam operações, nomeadas atividades (*activities*), sob entidades concretas e tangíveis, chamadas produtos de trabalho (*work product*).

3.3.1. Arquitetura de Modelagem

Esse padrão utiliza uma abordagem orientada a objetos para modelagem de processos de software e utiliza UML como notação. A OMG define uma

arquitetura de modelagem de quatro camadas, que pode ser visualizada na figura 3.3 [Mendes, 2005] [SPEM, 2005].

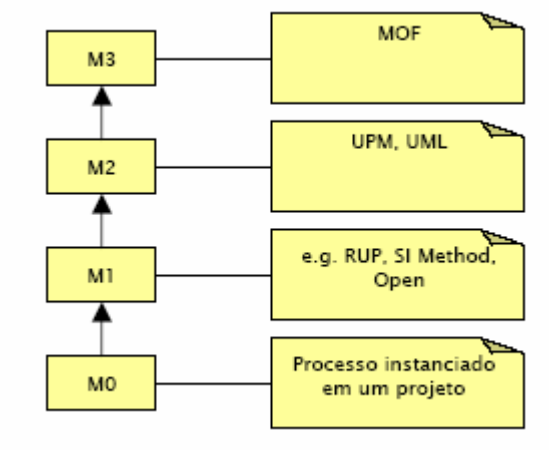


Figura 3.3 – Arquitetura de modelagem definida pela OMG [Mendes, 2005]

Os processos instanciados, ou seja, os processos que estão em execução em um projeto estão no nível M0. A correspondente definição desses processos, os processos abstratos, está situada no nível M1, podendo servir de exemplos, o RUP (*Rational Unified Process*), DMR *Macroscope*, IBM *Global Services Method*. O SPEM se encontra no nível M2, e serve de base para definição de processos do nível M1.

O SPEM está estruturado como um perfil de UML, e também como um meta-modelo baseado em MOF (*Meta-Object Facility*), uma tecnologia adotada pela OMG para definir meta-dados. Um meta-modelo MOF define a sintaxe abstrata de meta-dados na representação de um modelo MOF. Os meta-modelos MOF podem ser representados utilizando-se um subconjunto de UML (*Unified Modeling Language*).

3.3.2. Estrutura de Pacotes

A definição do SPEM (Software Process Engineering Metamodel) contém dois pacotes, ilustrada pela figura 3.4: o primeiro chamado *SPEM_Foundation*

representa uma extensão da especificação da UML 1.4 e contém os elementos básicos na qual a especificação SPEM foi construída; o segundo pacote chamado *SPEM_Extensions*, adiciona as abstrações necessárias e as semânticas requeridas para a engenharia de processo de software [SPEM, 2005] [Oliveira, 2006].

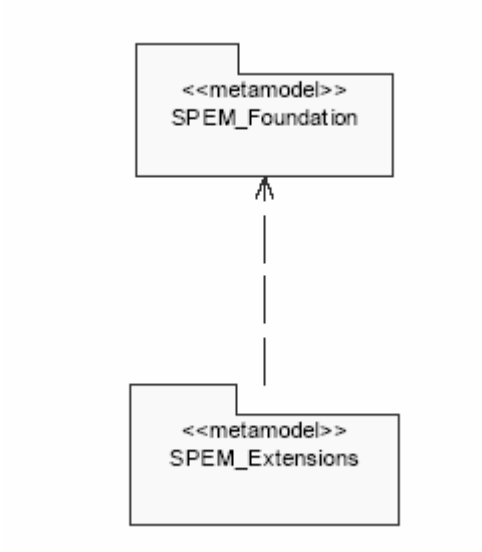


Figura 3.4 – Estrutura de pacotes do SPEM [SPEM, 2005]

3.3.3. *SPEM_Foundation*

Esse pacote é uma extensão de um subconjunto do meta-modelo UML 1.4 e está dividido em cinco sub-pacotes: *Data_Types, Core, Actions, State_Machines, Activity_Graphs* e *Model_Management.* Neles são encontrados os elementos básicos para a modelagem do processo, tais como elementos de tipos de dados (*Integer*, *UnlimitedInteger, String*, *AggregationKind, Boolean*, etc.), que representam ações (*Action, CallAction*, etc.), máquinas de estados (*StateMachine, State, Transiction*, etc.), diagramas de atividades (*ActivityGraph, ObjectFlowState, ActionState*, etc.) ou que servem para o gerenciamento dos modelos (*Package*).

O *SPEM_Foundation* contém ainda, elementos que são a base estrutural do modelo, são os que indicam relacionamento (*Association, AssociationEnd*), dependência (*Dependency*), ou que servem como base para outros elementos do

modelo (*ModelElement*, *PresentationElement*) [Mendes, 2005]. Para uma descrição completa dos sub-pacotes de *SPEM_Foundation*, consultar [SPEM, 2005].

3.3.4. *SPEM_Extensions*

O pacote *SPEM_Extensions* está estruturado em sub-pacotes. São eles: *Basic Elements, Dependencies, Process Structure, Process Components,* e *Process Lifecycle*. A figura 3.5 mostra as dependências entre eles e com relação ao *SPEM_Foundation* [Mendes, 2005]. A seguir, será abordado um pouco mais, cada sub-pacote [SPEM, 2005].

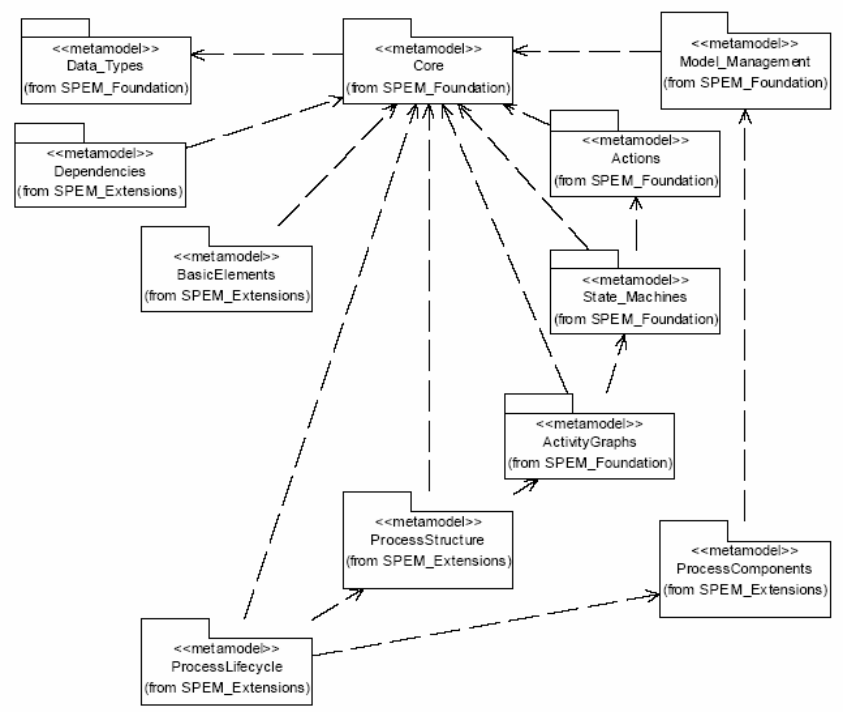


Figura 3.5 – Sub-pacotes do SPEM_Extensions e suas dependências [SPEM, 2005]

3.3.4.1. *Basic Elements* (Elementos Básicos)

Esse pacote contém os elementos básicos para a descrição de processos. Os elementos do pacote são:

- ***ExternalDescription:*** a todo *ModelElement* está relacionada uma ou mais *ExternalDescription*. Esse elemento contém uma descrição de um elemento do modelo (*ModelElement*) útil para quem está lendo a descrição do processo;

- ***Guidance*****:** são elementos responsáveis por prover informação mais detalhada (orientação) sobre os *ModelElements.* Podem ser de vários tipos, dependendo da família do processo. Como, por exemplo, *Technique*, *Checklist, Guideline* e *Template*:

3.3.4.2. *Dependencies* (Dependências)

Esse pacote é formado por subclasses das classes *Abstraction, Usage* e *Permission* do pacote *SPEM_Foundation*, e que têm semântica definida pela UML 1.4. Abaixo, uma breve descrição das dependências suportadas pelo SPEM [SPEM, 2005].

- ***Categorizes*****:** associa um pacote a um elemento de outro pacote, e serve para indicar que elementos de processo possuem múltiplas categorias. Essa funcionalidade é geralmente útil, pois atual em conjunto com as Disciplinas, que serão vistas mais adiante, para dar uma classificação alto nível para todos os elementos;

- ***Impacts*****:** Atua de um produto de trabalho (*work product*) para outro, indicando que alterações em um produto de trabalho podem invalidar outro;

- ***Import*:** uma dependência do tipo *Import* indica que todo o conteúdo do pacote alvo é adicionado ao *namespace* do pacote origem;

- ***Precedes*:** atua de uma atividade a outra ou de um *WorkDefinition* a outro, para indicar relação do tipo terminar para iniciar (*finish-start*) ou terminar para terminar (*finish-finish*), dependendo valor do atributo *kind* (tipo);

- ***RefersTo*:** Atua de um elemento de processo a outro, para garantir que eles pertencem ao mesmo *ProcessComponent*, que será visto no pacote *ProcessLifeCycle*. Geralmente é aplicado quando o texto de um elemento de processo se refere a outro, por nome ou pelo conteúdo, elemento, enfatizando de forma explícita na representação estrutural a relação entre os elementos.

- ***Trace*:** uma dependência desse tipo pode ser aplicada a quaisquer dois elementos do modelo de qualquer tipo. Ela serve para rastrear requisitos e mudanças através dos modelos.

3.3.4.3. *Process Structure* (Estrutura do Processo)

Esse pacote define os principais elementos estruturais a partir dos quais uma definição de processo é construída [Mendes, 2005]. Os principais elementos desse pacote são [SPEM, 2005]:

- ***WorkProduct and WorkProductKind*:** um produto de trabalho é qualquer coisa produzida, consumida ou modificada durante o processo. Pode ser um documento, um código, um modelo, etc. Um *WorkProduct* identifica uma classe de produto de trabalho afetado pelo processo. Já um *WorkProductKind*, identifica o tipo de

produto, tais como, documento de texto, diagrama UML, código fonte, etc. Um *WorkProduct* pode estar associado a um *ProcessRole*, que representa o papel responsável por sua produção, e também, a uma máquina de estados, que estabelece em quais estados o produto de trabalho pode aparecer;

- ***WorkDefinition and ActivityParameter*:** é um tipo de *Operation* (operação) que descreve o tipo de trabalho desenvolvido no processo. A sua principal subclasse é *Activity* (atividade), mas *Phase*, *Iteration* e *LifeCycle* (do pacote *Process Lifecycle*) também são subclasses. Um *WorkDefinition* possui um conjunto de entradas e saídas que se referem a ele através de um *ActivityParameter*. Algumas associações e observações do *WorkDefinition*:

    - Um *WorkDefinition* pode ser decomposto utilizando-se a associação *subWork*, trazendo a idéia de sub-tarefas;

    - Possui um *ProcessPerformer*, representando o papel primário que executa o *WorkDefinition;*

    - Os conceitos de WBS (*Work-Breakdown Structure*) podem ser descritos com algumas construções do SPEM: a decomposição em *subWorks* traz a idéia da hierarquia; as decomposições de *WorkDefinitions* podem ser representadas em detalhes no diagrama de atividades, limitados a um nível de aninhamento; a dependência de precedência traz a possibilidade de seqüenciar elementos de um mesmo nível da WBS.

- ***Activity* e *Step*:** é a subclasse principal de *WorkDefinition*. Ela descreve um trabalho realizado por um *ProcessRole*. Uma atividade pode ser constituída de elementos atômicos, os *Steps*. Algumas associações de *Activity*:

- Uma atividade pertence a um *ProcessRole*, que é o responsável por ela, e pode ser referenciada por outros papéis, que são auxiliares na atividade;

- Embora não seja proibido, *Activity*, geralmente, não utiliza a estrutura *subWork*, para ser decomposta, já que isso é feito utilizando-se *Steps*;

- *Steps* herdam de *ActionState*, logo, o fluxo de uma atividade pode ser representado por diagramas de atividades.

- ***ProcessPerformer* e *ProcessRole*:** um *ProcessPerformer* representa o executor de um conjunto de definições de trabalho (*WorkDefinitions*) no processo. Sua subclasse *ProcessRole* representa responsabilidades sobre produtos de trabalho (*WorkProducts*) e papéis que executam e auxiliam em atividades (*Activitiy*) específicas.

3.3.4.4. *Process Components* (Componentes do Processo)

As classes desse pacote têm por objetivo dividir uma ou mais definições de projeto em partes próprias que podem ser colocadas sob gerência de configuração e controle de versão [Mendes, 2005]. Abaixo, os principais elementos do pacote [SPEM, 2005]:

- ***Package*:** assim como em UML, é um *container* (recipiente) que pode possuir e importar elementos de definição de processo. Pacotes e *Categorizes* (Categorização) podem ser usados para categorizar elementos de descrição de processo. Um pacote é criado para representar cada categoria, e todos os elementos dentro do

pacote são relacionados através de uma dependência do tipo *Categorizes*, para representar os membros da categoria;

- ***ProcessComponent*:** é um bloco da descrição do processo que é consistente internamente e pode ser reusado com outros componentes de processo para montar um processo completo. Um componente de processo importa um conjunto de elementos de definição de processo, que deve ser auto-contido, ou seja, nenhum dos seus elementos pode ter dependências do tipo *ReferTo* para um elemento fora do *ProcessComponent*;

- ***Process*:** é um *ProcessComponent* (componente de processo), que tem por finalidade ser único como um processo completo. Ele se distingue de outros componentes de processo pelo fato de que ele não deve ser composto por outros componentes. A classe *Process* também pode representar uma família de processos, sob a qual, múltiplos processos sobrepostos podem ser definidos;

- ***Discipline*:** é uma especialização particular de *Package* (pacote), que divide as atividades de um processo de acordo com um tema. Isso implica que as *Guidances* (guias) e *WorkProducts* associados à atividade devem estar contidos na disciplina também. A inclusão de uma atividade em uma disciplina é realizada através de uma dependência do tipo *Categorize*, com a restrição de que uma atividade só pode estar em uma disciplina.

3.3.4.5. *Process Lifecycle* (Ciclo de Vida do Processo)

Os elementos desse pacote ajudam a definir como o processo será executado. Eles descrevem o comportamento geral do processo e são usados para auxiliar no planejamento, execução e monitoramento do processo. Esses elementos

são ainda, fundamentais para definir a "forma" do processo com relação ao tempo e a estrutura do seu ciclo de vida, em termos de iterações e fases [Mendes, 2005]. A seguir, uma breve descrição dos seus elementos [SPEM, 2005]:

- ***Phase*:** uma fase é uma especialização de *WorkDefinition*, na qual suas pré-condições definem os critérios de entrada nela e seus objetivos (*milestones*) definem seus critérios de saída. Geralmente, está associada a algumas restrições seqüenciais, ou seja, existem datas para os *milestones* ao longo do tempo, e é assumido que há mínima ou nenhuma sobreposição de atividades;

- ***Lifecycle*:** um ciclo de vida de processo é definido como uma seqüência de fases para atingir um objetivo específico. Ele define o comportamento de um processo completo para ser executado em um projeto ou programa;

- ***Iteration*:** é a composição de um *WorkDefinition* com um objetivo (*milestone*) menor no processo;

- ***Precondition* e *Goal*:** a cada *WorkDefinition* pode ser associado uma pré-condição e um objetivo. *Peconditions* e *Goals* são restrições, onde restrições são representadas em termos de *BooleanExpression* (expressão booleana). As condições são expressas em termos dos estados dos *WorkProducts* que são parâmetros da *WorkDefinition* ou de *WorkDefinitions* internas.

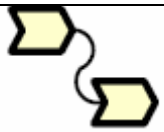
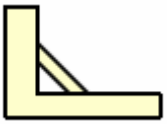
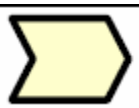
3.3.4.6. Estereótipos do SPEM *Profile*

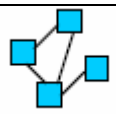
Nas seções anteriores, foram discutidos os elementos do SPEM como meta-modelo, mas o SPEM também é definido como um UML *Profile*, o que adiciona a ele a expressividade da UML. UML *Profiles* são extensões da

semântica padrão da UML, com o objetivo de customizar a UML para um domínio ou propósito específico.

Além disso, o uso do SPEM como um UML *Profile* faz com que uma larga quantidade de desenvolvedores, já familiarizados com UML, possa utilizar esses conhecimentos no domínio da modelagem de processos de software. Na tabela 3.1, são apresentados os estereótipos do SPEM *Profile*, que apresentam notações próprias do SPEM. Para visualizar a lista completa de estereótipos, consultar [SPEM, 2005].

Tabela 3.1 Estereótipos do SPEM Profile

Estereótipo	**Classes Base**	**Estereótipo Pai**	**Notação**
WorkProduct	*Core::Class* *ActivityGraphs::ObjectFlow* *State*		
WorkDefinition	*Core::Operation* *ActivityGraphs::ActionState* *UseCases::UseCase*		
Guidance	*Core::Comment*		
Activity	*Core::Operation* *ActivityGraphs::ActionState* *UseCases::UseCase*	*WorkDefinition*	
ProcessPerformer	*UseCases::Actor*		

Estereótipo	Classes Base	Estereótipo Pai	Notação
ProcessRole	*UseCases::Actor*	*ProcessPerformer*	
ProcessPackage	*ModelManagement::Package*		
Phase	*Core::Operation* *ActivityGraphs::ActionState* *UseCases::UseCase*	*WorkDefinition*	
Process	*ModelManagement::Package*	*ProcessPackage*	
Document	*Core::Class* *ActivityGraphs::ObjectFlow* *State*	*WorkProduct*	
UMLModel	*Core::Class* *ActivityGraphs::ObjectFlow* *State*	*WorkProduct*	

3.4. BPMN

O BPMN (*Business Process Modeling Notation*) é um padrão para a modelagem de processos de negócio. Inicialmente desenvolvido pela BPMI (*Business Process Management Initiative*), esse padrão é mantido, hoje, em conjunto pela BPMI e OMG. Essa união foi bastante importante na tentativa de transformar o BPMN num padrão para a indústria, visto que a OMG é uma organização de grande reputação no desenvolvimento de padrões para a indústria de desenvolvimento de software [BPMN, 2004] [Filho, 2007] [Owen, 2003].

O objetivo principal do BPMN é prover uma notação rapidamente compreensível por todos os participantes do negócio, desde os analistas de negócio, que farão os primeiros rascunhos dos processos, até os desenvolvedores técnicos responsáveis por implementar a tecnologia, que irão realizar esses processos, como também, as pessoas envolvidas no negócio, que gerenciando e monitorando o processo. Dessa forma, o BPMN procura criar uma ponte padrão para o desnível semântico entre modelagem de processos de negócio e implementação de processos.

Outro objetivo do BPMN, não menos importante, é garantir que linguagens desenvolvidas para a execução de processos de negócio, tais como, e principalmente, BPEL4WS (*Business Process Execution Language for Web Services*), podem ser visualizadas com uma notação orientada a negócios. Em sua especificação [BPMN, 2004], encontra-se um mapeamento entre BPMN e BPEL4WS.

BPMN provê a criação dos BPD (*Business Process Diagram*), que são diagramas voltados para o uso de pessoas que modelam e gerenciam processos de negócio, suportando conceitos como o B2B (*Business to Business*), ou seja, processos inter-organizacionais. Outro recurso do BPMN é a possibilidade da sua extensão por modeladores e ferramentas, através de novos marcadores nos elementos estendidos, mas sem que sua forma básica seja alterada. Isso faz com

que necessidades de modelagem não suportadas pela notação padrão possam ser atendidas.

O BPMN cobre a criação de vários tipos de processos de negócio, e permite a criação de processos de negócio fim a fim. Ele define três principais tipos de sub-modelos dentro de um modelo BPMN completo [Filho, 2007]:

- **Processos privados (internos):** representações de processos internos à organização, correspondem às atividades do processo e como elas interagem entre si. Na figura 3.6, um exemplo de processo privado.

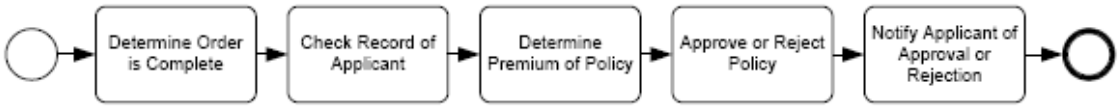


Figura 3.6 – Exemplo de processo privado [BPMN, 2004]

- **Processos abstratos (públicos):** correspondem às interações entre processos privados e outras entidades de negócio (outro processo ou participante) que são externas a esse processo. Apenas as atividades e mecanismos de controles de fluxo que são usadas para a comunicação externa ao processo privado são mostrados nesse tipo de modelo, qualquer outra atividade interna não é exibida no modelo de processo abstrato. A figura 3.7 ilustra um exemplo desse tipo de modelo.

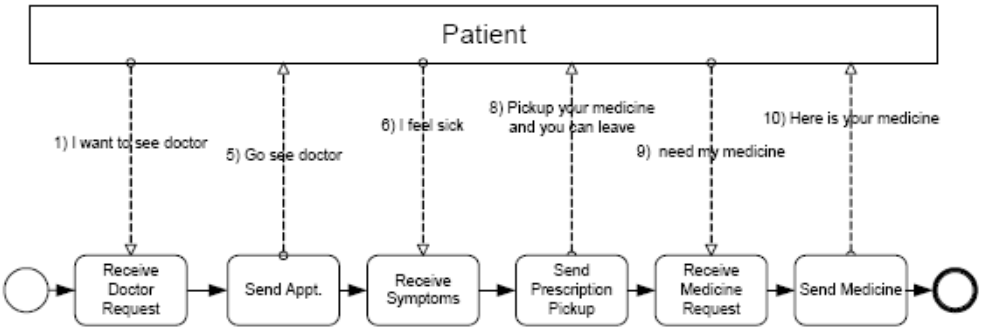


Figura 3.7 – Exemplo de processo abstrato [BPMN, 2004]

- **Processos de colaboração (global):** representam as interações entre duas ou mais entidades de negócio. Essas interações são definidas como um conjunto de atividades que representam um padrão para a troca de mensagens entre as entidades envolvidas. Na figura, 3.8, um exemplo de processo de colaboração.

Como pode ser visto na figura 3.8, o BPMN possibilita estabelecer diferentes pontos de vista sobre o diagrama do processo. Cada ponto de vista ressalta as atividades sobre as quais um participante possui responsabilidades e controle. No exemplo abaixo, há dois pontos de vista: o do paciente e o do recepcionista ou médico.

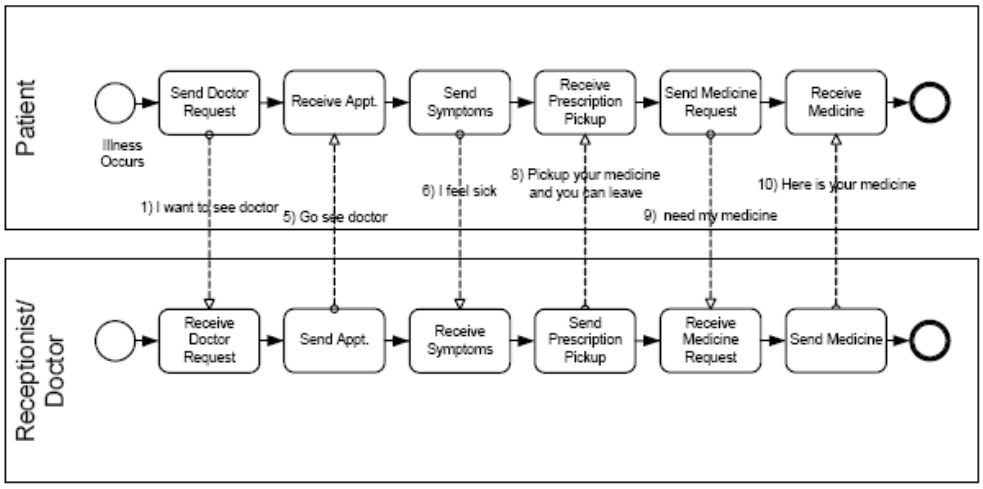


Figura 3.8 – Exemplo de processo de colaboração [BPMN, 2004]

3.4.1. Elementos Gráficos

Os diagramas de processos de negócio modelados com BPMN permitem a compreensão de processos desde os mais simples até os mais complexos processos de negócio. Além disso, o BPMN permite o mapeamento dos seus modelos para linguagens de execução de processos de negócio, graças ao suporte dado por atributos dos elementos gráficos. Esses elementos são organizados em quatro categorias [BPMN, 2004]: *Flow Objects* (Objetos de Fluxo), *Conecting Objects* (Objetos de Conexão), *Swimlanes* (Raias) e *Artifacts* (Artefatos). A seguir, serão

vistos os principais elementos de cada categoria, e que são a base para outros elementos derivados, em mais detalhes, considerando-se o propósito de comparação com SPEM como notação para modelagem de processos de software, objetivo deste trabalho. Para a lista completa dos elementos gráficos do BPMN, consulte [BPMN, 2004].

3.4.1.1. *Flow Objects*

Os elementos de fluxo são os elementos responsáveis por definir o comportamento de um processo de negócio. Há três elementos de fluxo principais: *Events* (Eventos), *Activities* (Atividades) e *Gateways* (Passagens). Na tabela 3.2 podem ser vistos seus principais elementos [BPMN, 2004].

Tabela 3.2 Flow Objects

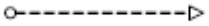
Elemento	**Descrição**	**Notação**
Event	Um evento é algo que acontece durante a execução do processo. Ele afeta a execução do processo e, geralmente, possui uma causa (*trigger*) e um impacto (*result*). Eles podem ser de três tipo, baseado no momento em que afetam o processo: *Start* (Início), *Intermediate* (Intermediário) e *End* (Fim). Eventos do tipo início e intermediário também podem ser divididos quanto ao seu *trigger* (gatilho): *Message, Timer, Error, Cancel, Compensation, Rule, Link, Multiple, Terminate.*	

Elemento	Descrição	Notação
Activity	Atividade é um termo genérico para um trabalho desempenhado por uma organização. Uma atividade pode ser atômica ou não-atômica (composta). Os tipos de atividades que são parte de um modelo de processo são: *Process* (Processo), *Sub-Process* (Sub-Processo) e *Task* (Tarefa). *Task* e *Sub-Process* são representados por retângulos arredondados. *Process* não possui notação gráfica associada (é o modelo em si) ou é o conteúdo de um *Pool* (será visto mais adiante).	
Gateway	Um *Gateway* é usado para controlar a divergência ou convergência de fluxos de seqüência. Dessa forma, ele irá determinar ramificação, bifurcação, ligação e junção de caminhos. Símbolos internos à notação gráfica irão indicar o tipo de controle.	

3.4.1.2. *Connecting Objects*

Os elementos de conexão são os elementos responsáveis por conectar os objetos de fluxo. Há três tipos principais de objetos de conexão, que podem ser vistos na tabela 3.3 [BPMN, 2004].

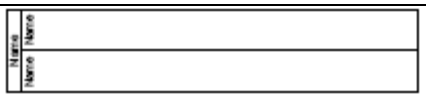
Tabela 3.3 Connecting Objects

Elemento	Descrição	Notação
Sequence Flow	Um fluxo de seqüência é usado para indicar a ordem em que atividades serão executadas em um processo. Os fluxos de seqüência podem ser de quatro tipos: *Uncontrolled Flow* (Fluxo sem Controle), *Conditional Flow* (Fluxo condicional), *Default Flow* (Fluxo Padrão) e *Exception Flow* (Fluxo de Exceção).	
Message Flow	Um fluxo de mensagem é usado para mostrar a troca de mensagens entre dois participantes do processo. Em BPMN, dois *Pools* separados no diagrama representam dois participantes (uma entidade de negócio ou um papel do processo, por exemplo).	
Association	Usada para associar informação a objetos de fluxo. Objetos gráficos ou textuais, que não são de fluxos podem ser associados a objetos de fluxo.	

3.4.1.3. *Swimlanes*

São elementos que possibilitam o agrupamento de elementos primários, como atividades eventos e associações, em “raias”, o que pode indicar o conjunto de atividades de uma entidade do negócio, por exemplo. Na tabela 3.4, são vistos esses elementos [BPMN, 2004].

Tabela 3.4 Swimlanes

Elemento	Descrição	Notação
Pool	Uma "piscina" representa um participante do processo. Ele representa um recipiente que separa um conjunto de atividades de outro *Pool*, geralmente no contexto de B2B (*Business To Business).*	
Lane	Uma "raia" é uma sub-partição de um *Pool*. *Lanes* são usados para organizar e categorizar as atividades.	

3.4.1.4. *Artifacts*

Artefatos são usados para prover informação adicional sobre o Processo. Há três artefatos padrões, no entanto, ferramentas e modeladores podem adicionar quantos tipos de artefatos forem necessários. Na tabela 3.5, são descritos os três principais tipos de artefatos usados no diagrama de processo modelado por BPMN [BPMN, 2004].

Tabela 3.5 Artifacts

Elemento	Descrição	Notação
Data Object	Objetos de dados são considerados artefatos porque eles não têm efeito direto no fluxo de seqüência ou fluxo de mensagens do processo, mas eles provêem informação acerca do que as atividades precisam para serem executadas e o que elas produzem.	
Group (uma caixa que engloba um grupo de objetos para propósitos de documentação)	Um grupo de atividades que não afeta o fluxo de seqüência. O agrupamento pode ser realizado para fins de análise ou documentação. Grupos também podem ser usados para identificar atividades de uma transação distribuída que é mostrada entre *Pools*.	
Text Annotation (ligada a uma Associação)	Anotações de texto são mecanismos utilizados pelos modeladores para prover informação adicional aos leitores do diagrama de processo de negócio.	

4. SPEM versus BPMN na Modelagem de Processo de Software

Neste capítulo, serão comparadas as duas abordagens para modelagem de processo que foram vistas no Capítulo 3. A abordagem tradicional, através de um meta-modelo para modelagem de processos de software, e outra alternativa, na qual é utilizada uma tecnologia para modelagem de processos de negócio, tratando, dessa forma, o processo de software como um dos processos de negócio possível de ser definido em uma organização. Para isso, foi necessária a escolha de duas tecnologias representantes de cada metodologia. A decisão tomada foi pelo uso de SPEM e BPMN, por razões já apresentadas no capítulo anterior.

O objetivo desse comparativo é trazer à comunidade da engenharia de software informações úteis à escolha de uma ou outra abordagem, através da análise do contexto organizacional sob o qual se definirá o processo, incluindo culturas, características dos recursos humanos, relações entre as diversas áreas da empresa. Essas informações, aliadas aos resultados obtidos nesse trabalho, fornecerão meios e argumentos para uma empresa decidir que tecnologia seria mais adequada na definição e modelagem do seu processo de desenvolvimento.

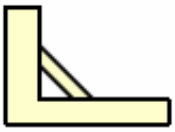
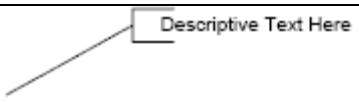
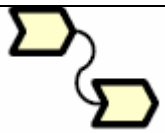
Primeiramente, para estabelecer o comparativo entre as duas tecnologias, é necessário que façamos o mapeamento entre os seus elementos, o que será feito na seção 4.1. Feito isso, analisaremos as possibilidades suportadas por cada padrão, indicando vantagens e desvantagens de cada abordagem.

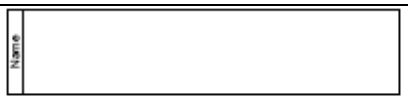
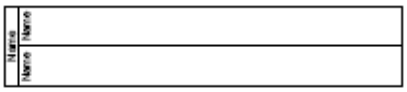
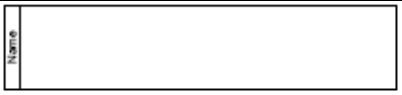
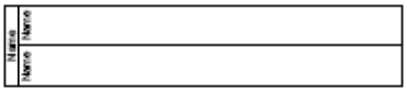
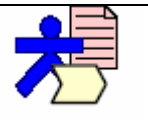
Vale ressaltar que o objetivo do comparativo limita-se ao escopo de definição e modelagem do processo de software, não sendo interesse, deste trabalho, analisar suporte à execução dos processos através de uma linguagem de execução que possa ser mapeada pelas notações do SPEM ou do BPMN.

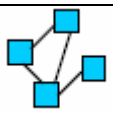
4.1. Mapeamento entre SPEM e BPMN

Ao estabelecer o comparativo entre SPEM e BPMN, devemos considerar que estamos realizando um confronto entre as duas tecnologias para um objetivo específico: modelagem de processos de software. Sendo assim, tomaremos como ponto de partida para a comparação, o meta-modelo SPEM, visto que ele é um padrão especificamente desenvolvido para a modelagem de processos de software. Na tabela 4.1, apresentamos a proposta de mapeamento dos principais elementos do SPEM para os elementos que podemos estabelecer como correspondentes a eles no BPMN [SPEM, 2005] [BPMN, 2004].

Tabela 4.1 Mapeando os estereótipos do SPEM para o PBMN

SPEM	BPMN	Justificativa / Restrições
 *WorkProduct*	 *Data Object*	Representam qualquer tipo de produto consumido ou gerado durante o processo, e que provêem informação adicional às atividades.
 *Guidance*	 *Text Annotation*	Existem vários tipos de *Guidance* no SPEM, o que lhe confere maior especificidade comparando-se com o uso do *Text Annotation*.
 *WorkDefinition*	 E*mbedded Sub-Process*	*WorkDefinition* representa qualquer tipo de trabalho realizado em um processo que é constituído por outras atividades ou *WorkDefinitions*, por isso será mapeado para um *Embedded Sub-Process*, que pode ser formado por várias atividades.

SPEM	BPMN	Justificativa / Restrições
 *Activity*	 *Task*	Um *Activity* representa uma atividade que não pode ser dividida em outras, assim como *Task*, no BPMN, é uma tarefa atômica.
 *ProcessPerformer*	 *Pool*  *Lane*	*ProcessPerformer* representa o executor de um conjunto de *WorkDefinitions*, enquanto que um *Pool* engloba uma série de *Tasks*, representando um participante do processo. *Lanes* são subdivisões de um *Pool*.
 *ProcessRole*	 *Pool*  *Lane*	*ProcessRole* representa um papel do processo que executa e auxilia em várias *Activities*, enquanto que um *Pool* engloba uma série de *Tasks*, representando um participante do processo. *Lanes* são subdivisões de um *Pool*.
 *ProcessPackage*	 *Independent Sub-Process*	*ProcessPackage* contém elementos de definição do processo (atividades, papéis, produtos) através de dependências do tipo *Categorizes.* Um *Independent Sub-Process* também pode englobar papéis, atividades e produtos em sua definição.
 *Phase*	 *Embedded Sub-Process*	*Phase* é uma especialização de *WorkDefinition*, portanto também o mapearemos para *Sub-Process.*

SPEM	BPMN	Justificativa / Restrições
 *Process*		*Process*, no meta-modelo SPEM, é um processo auto contido, independente e completo. No BPMN, não existe uma notação específica para *Process*, ele é representado pelo diagrama de processo como um todo.
 *Document*	 *Data Object*	*Document* é um tipo de *WorkProduct* que possui um estereótipo próprio, logo, foi mapeado para um *Data Object*.
 *UMLModel*	 *Data Object*	*UMLModel* é um tipo de *WorkProduct* que possui um estereótipo próprio, logo, foi mapeado para um *Data Object*.

4.1.1. Considerações acerca do mapeamento dos elementos

Alguns dos mapeamentos propostos necessitam que levantemos algumas considerações e restrições quanto ao seu uso:

4.1.1.1. *Guidance* para *Annotation*

Um *Guidance*, no meta-modelo SPEM, é um tipo específico para apresentar guias e orientações na realização de uma *Activity* ou *WorkDefinition*, por exemplo. Através da classe *GuidanceKind*, pode representar vários tipos, tais como: *Guidelines*, *Techniques*, *Metrics*, *Examples*, *UML Profiles*, *Tool mentors*. Eles conferem uma maior expressividade semântica ao modelo, quanto ao tipo de orientação que fornecem, por isso, ao mapearmos para o elemento *Text Annotation*

do BPMN, estaremos perdendo essa expressividade, já que *Text Annotation* é apenas um elemento textual, o que, contudo, não invalida o mapeamento.

4.1.1.2. *WorkDefinition* para *Embedded Sub-Process*

Um *WorkDefinition* é usado para definir um trabalho que deve ser decomposto em várias atividades ou outros trabalhos (*WorkDefinition*). Dessa forma, o seu correspondente no modelo BPMN deverá ser um *Embedded Sub-Process*, pelas razões expostas abaixo:

- Um *WorkDefinition* será constituído apenas por outros *WorkDefinitions* ou *Activities* e poderá estar associado a *WorkProducts*, através da classe *ActivityParameter*, indicando se eles são entradas ou saídas. Um *Sub-Process* pode ser de três tipos: *embedded*, *independent* ou *reference*. *Embedded Sub-Process* é um sub-processo que é dependente do seu processo pai, e possui visibilidade dos dados globais desse processo. Ele não pode possuir *Pools* (papéis em nosso mapeamento), podendo apenas, ser associado a eles, assim como um *WorkDefinition*, sendo formado por outras atividades, objetos de conexão e artefatos.

- Os *WorkDefinitions* e atividades constituintes do principal podem ser entendidos como as atividades e sub-processos do *Embedded Sub-Process*, bem como os *WorkProducts* que indicam entradas e saídas, podem ser associados aos artefatos dos *InputSets* e *OutputSets*, atributos de qualquer atividade de um modelo BPMN, e, portanto, presentes no *Embedded Sub-Process*.

4.1.1.3. *ProcessPerformer/ProcessRole* para *Pools* e *Lanes*

ProcessPerformers e *ProcessRoles*, de maneira geral, representam executores do processo, com a diferença que *ProcessRole* é usado para indicar um papel específico do processo, enquanto *ProcessPerformer* é associado a um participante mais geral, que não pode ser encaixado em nenhum dos papéis do processo. O BPMN dá muita liberdade ao projetista do processo quanto ao uso de *Pools* e *Lanes*, estabelecendo apenas que eles representam participantes do processo, sendo as *Lanes* subdivisões de um *Pool*, ficando a cargo do projetista definir que tipo de divisão será realizada, por exemplo, de cargos em especializações deles, departamentos divididos em setores, etc.

4.1.1.4. *ProcessPackage* para *Independent Sub-Process*

ProcessPackage é um *container* (repositório) que pode possuir ou importar elementos de definição do processo, tais como *WorkDefinitions*, *Activities*, *WorkProducts*, *ProcessRole*, etc. Já os *Sub-Process*, do tipo *independent*, são sub-processos independentes do processo pai no qual estão sendo representados, ou seja, ao ser chamado no seu fluxo, apenas instancia o sub-processo, com suas atividades, *pools*, artefatos, etc. Dessa forma, podemos mapear todos os componentes de um pacote para objetos dentro de um *Independent Sub-Process*, visto que ele pode conter qualquer tipo de elemento de um diagrama de processo do BPMN.

4.1.1.5. *Phase* para *Embedded Sub-Process*

A primeira justificativa para este mapeamento está relacionada ao fato de *Phase* ser uma especialização de *WorkDefinition*, portanto foi mapeada para o

mesmo elemento: *Embedded Sub-Process*. No entanto, há algumas características de *Phase* que devem ser consideradas:

- *Phase* é um elemento que deve ser usado para representar o ciclo de vida de um processo, isso, na verdade, não é uma restrição ao mapeamento, mas, ao mapear uma *Phase* para o BPMN, essa característica deve ficar clara no modelo BPMN.

- As pré-condições de uma fase definem seus critérios de entrada e seus objetivos (geralmente chamados *milestones*), os critérios de saída. Pré-condições podem ser modeladas através de eventos (do tipo *Rule*, por exemplo, que estabelece regras para a seqüência de um fluxo) em BPMN, assim como os *milestones*, especialmente, usando-se eventos de tempo (*Time*), que possibilita a criação de datas para o fim ou começo de atividades.

- Internamente, as fases precisam satisfazer restrições de seqüencialidade, através do cumprimento de datas de *milestones*, mas também necessitam que produtos de trabalho sejam gerados, através, inclusive, do consumo de outros tantos. Para satisfazer esse comportamento, podemos utilizar o atributo *IORules* da atividade do BPMN, que permite a criação de regras de produção de artefatos de saída, a partir de outros de entrada, bem como utilizar eventos, notadamente de tempo, no fluxo de um *Embedded Sub-Process* para implementar essas restrições de seqüenciamento.

4.1.1.6. Uso de *Data Object*

O SPEM possui estereótipos diferentes para representar alguns tipos de *WorkProducts*, a saber: *UMLModel* e *Document*. Isso não impede que ambos sejam mapeados para o elemento *Data Object* de BPMN, pois os dois derivam de

WorkProduct. Além disso, um *Data Object* possui um atributo chamado *Properties*, que representa um conjunto de *Property*, um tipo suportado por BPMN. Um *Property* possui um nome e um tipo (*String*, *Boolean*, etc.), com isso, é possível criar um *Data Object* com propriedades específicas de um *UMLModel* ou um *Document*.

4.1.2. Perspectivas do modelo de processo

O SPEM, por ser também um *UML Profile*, provê vários tipos de diagramas, que permitem representar diferentes perspectivas de um modelo de processo de software. Dentre esses diagramas, destacamos [SPEM, 2005]:

- **Diagrama de classes:** usado, principalmente, para representar herança, dependências, associações, relacionamento entre *ProcessPerformer* ou *ProcessRole* e *WorkProduct*. A figura 4.1 ilustra um exemplo desse tipo de diagrama.

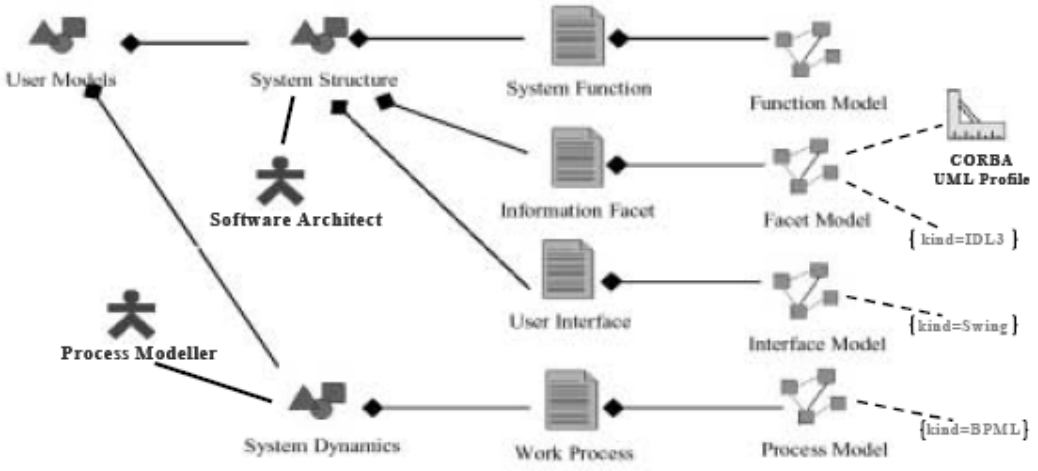


Figura 4.1 – Exemplo de diagrama de classes [SPEM, 2005]

- **Diagrama de pacotes:** permite a representação de *Process*, *ProcessComponents*, *ProcessPackages* e *Disciplines*. A figura 4.2 apresenta um diagrama de pacotes.

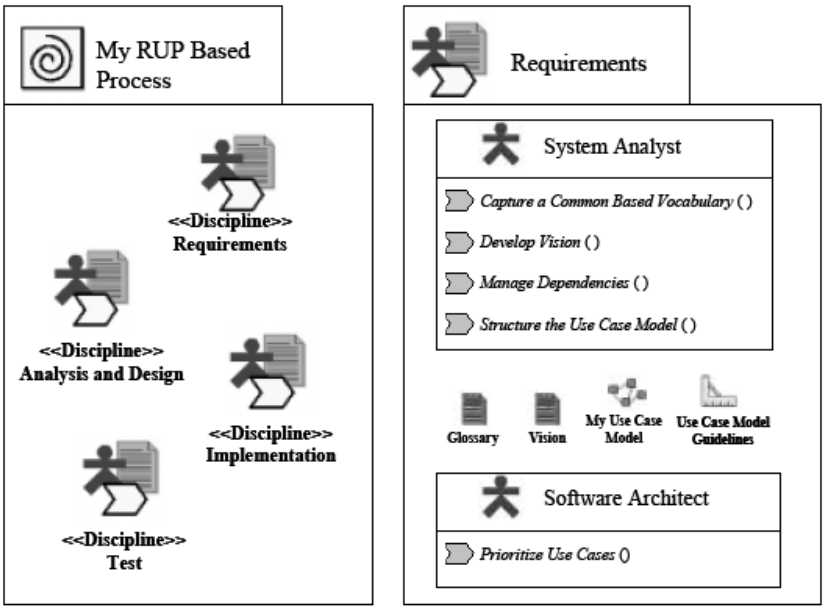


Figura 4.2 – Exemplo de diagrama de pacotes [SPEM, 2005]

- **Diagrama de caso de uso:** diagramas de caso de uso mostram o relacionamento entre papéis de processo e as principais definições de trabalho (*WorkDefinition*). A figura 4.3 mostra um exemplo desse diagrama.

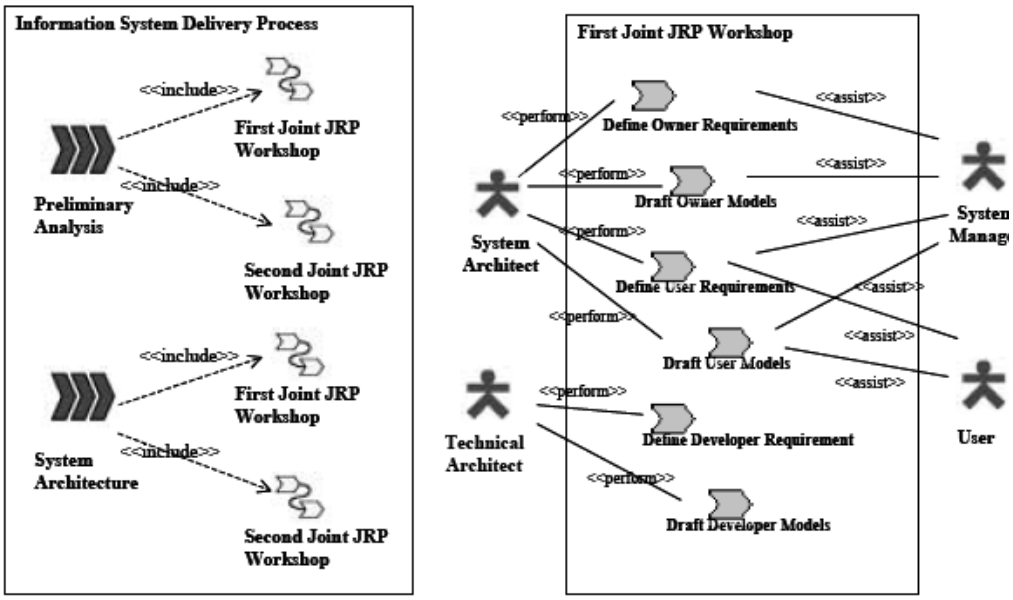


Figura 4.3 – Exemplo de diagrama de caso de uso [SPEM, 2005]

- **Diagramas de seqüência:** podem ser usados para exibir padrões de iterações de instâncias de elementos de modelo do SPEM.

- **Diagramas de estados:** são úteis na representação do comportamento dos elementos do modelo.

- **Diagrama de atividades:** diagramas de atividades permitem representar a seqüência de atividades com seus produtos de trabalho de entrada e saída bem como os estados dos objetos de fluxo. Raias (*swimlanes*) podem ser usadas para separar as responsabilidades dos papéis de processo. A figura 4.4 exibe um diagrama de atividades.

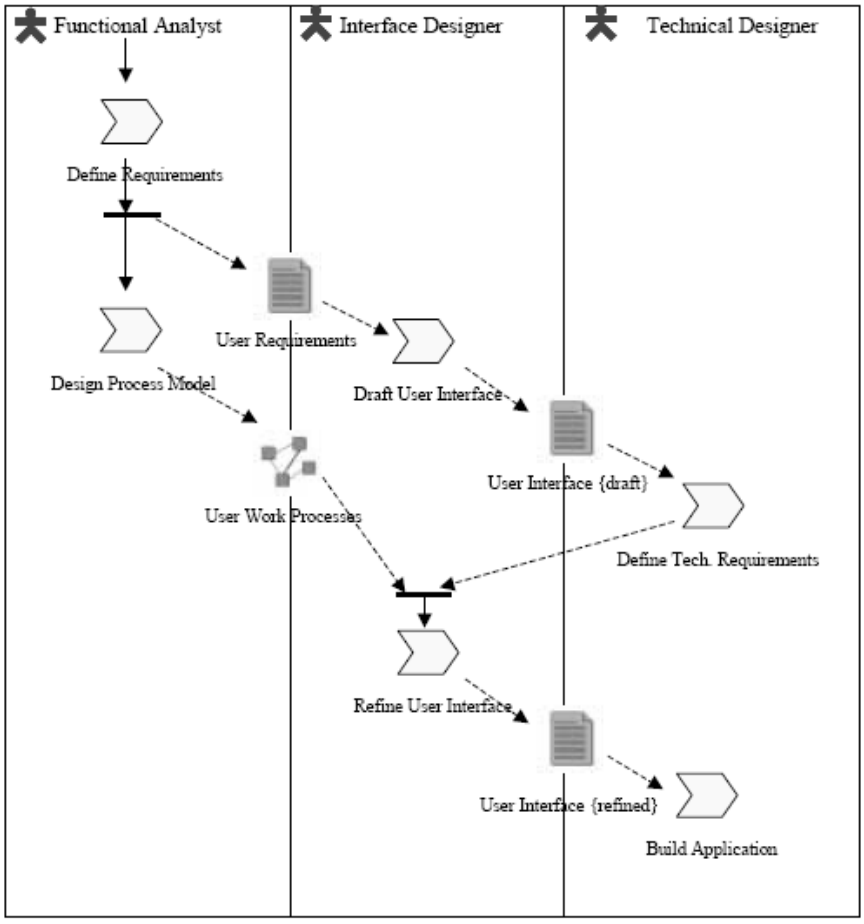


Figura 4.4 – Exemplo de diagrama de atividades [SPEM, 2005]

Do outro lado, o BPMN não possui diferentes tipos de diagramas, quanto à sua forma, permitindo apenas a representação dos chamados diagramas de processo de negócio. Os diagramas do BPMN indicam o fluxo de atividades na sua seqüência e devem ser lidos a partir dos eventos de início (*start events*). Fazendo uma analogia com os tipos de diagramas suportados por SPEM, podemos dizer que a perspectiva fornecida pelo diagrama do BPMN se assemelha à apresentada pelo diagrama de atividades do SPEM, que também representa a seqüência de atividades do processo. Na figura 4.5, ilustramos um exemplo de diagrama de processo do BPMN.

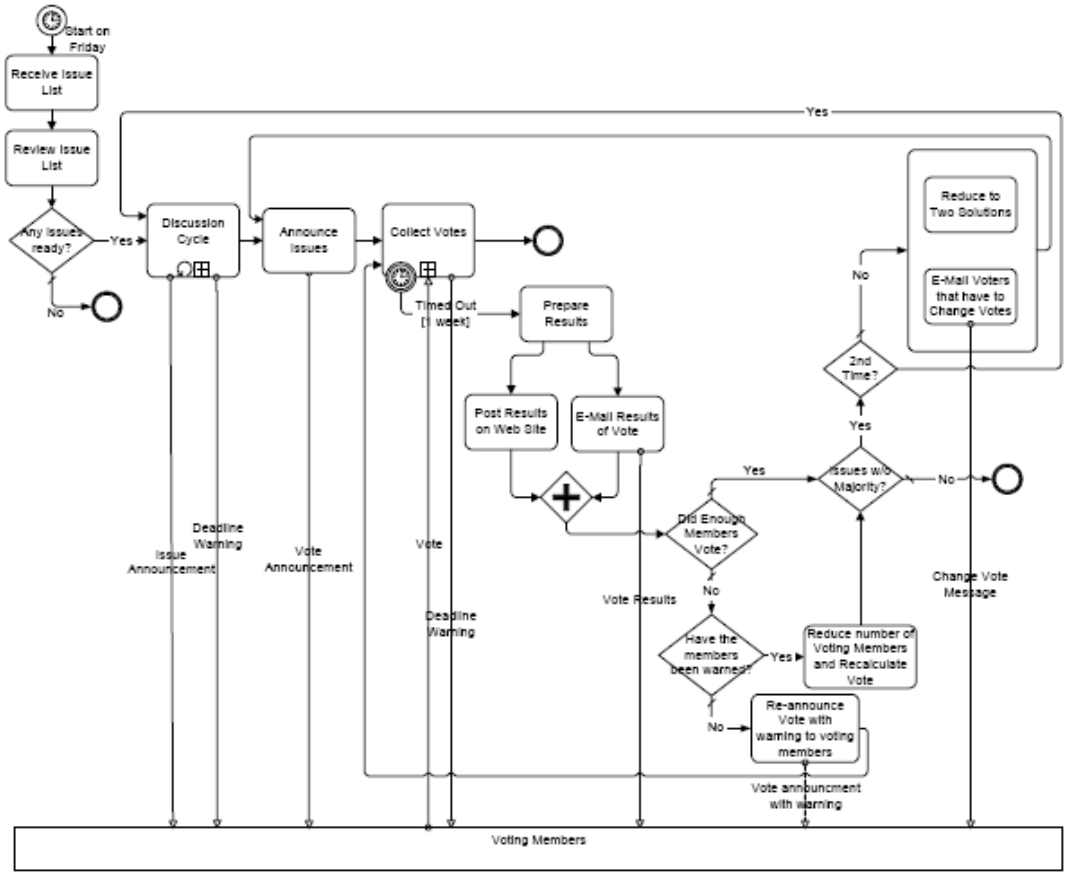


Figura 4.5 – Exemplo de diagrama de processo do BPMN [BPMN, 2004]

4.2. Estudo de Caso – Processo de desenvolvimento de uma empresa de TI (Tecnologia da Informação)

Nesta seção, será realizado um estudo de caso da modelagem de processo de software através do SPEM e do BPMN, tendo como base o mapeamento proposto na seção 4.1 e objetivando sua validação através da aplicação em um caso real.

Será analisado o processo de desenvolvimento utilizado em um dos projetos de uma empresa multinacional, que atua há mais de vinte anos no desenvolvimento de soluções para a gestão de negócios. A organização é uma empresa de grande porte, conta com clientes em mais de 30 países e cerca de 1100 colaboradores em 12 países. No entanto, é importante destacar que o projeto em questão foi iniciado há pouco menos de 1 ano, e que o processo de desenvolvimento ainda está em fase de definição.

4.2.1. O Processo de Desenvolvimento

Antes de realizarmos a modelagem do processo através de SPEM e BPMN, é preciso apresentar o modelo atualmente proposto pela empresa. Neste momento, faz-se necessário destacar que o processo de desenvolvimento em análise neste trabalho é ainda incipiente e não está completamente definido, nem tampouco é utilizado na sua totalidade. Também por isso, não apresenta clareza quanto a regras e restrições sobre o seu fluxo, o que dificulta o seu entendimento e, por conseqüência, sua modelagem. Entretanto, isso não faz com que a modelagem que proporemos seja insatisfatória, pois o processo abrange, ao menos em parte, todos os níveis de detalhamento propostos por ele, como, por exemplo, o de fases e descrição de atividades.

A figura 4.6 ilustra a visão mais macro do processo. É importante destacar que a modelagem realizada pela empresa não fez uso de nenhuma representação diagramática padronizada pela comunidade científica e/ou comercial, mas sim, usou uma notação informal.

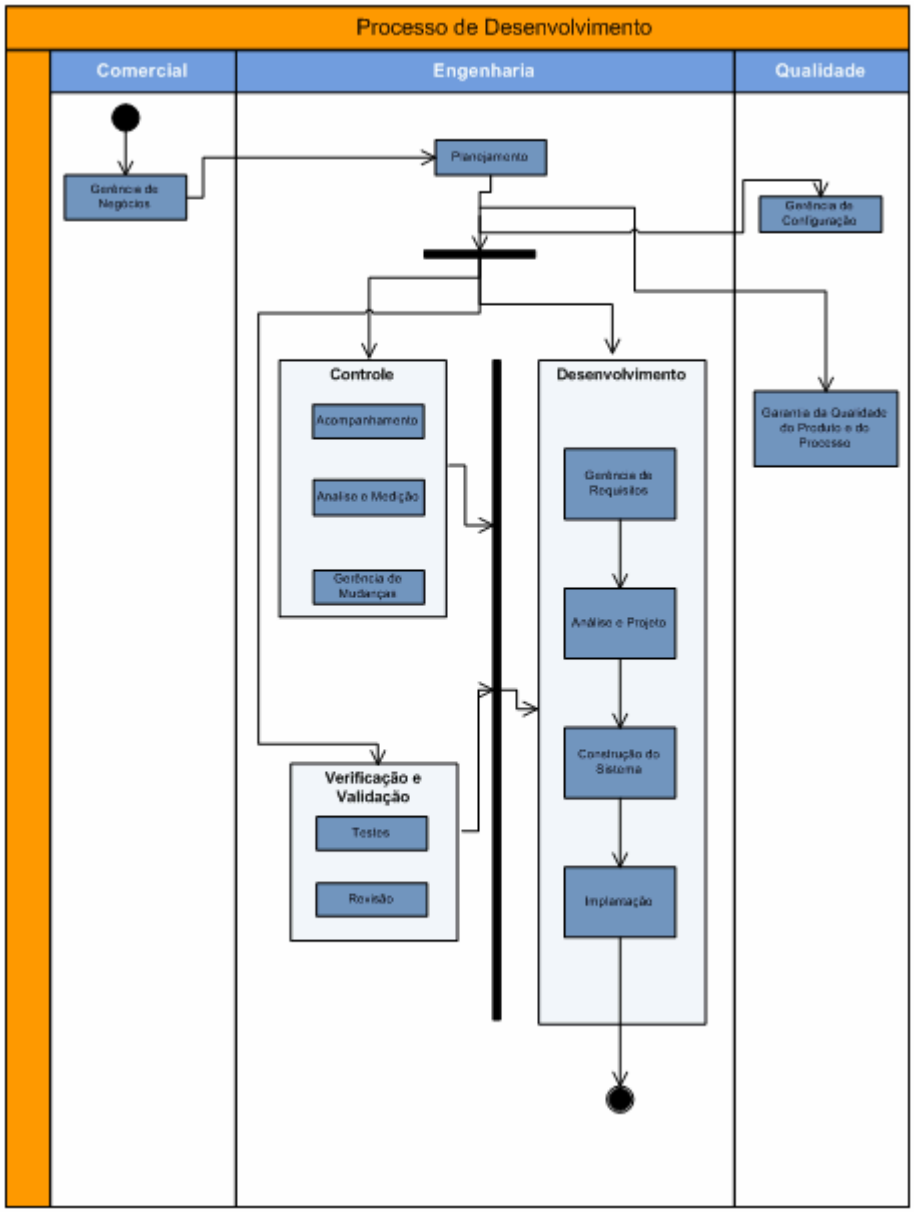


Figura 4.6 – Modelo do processo de desenvolvimento analisado definido pelo SEPG da empresa

O estudo de caso será conduzido de forma semelhante ao mapeamento realizado anteriormente, ou seja, iniciaremos modelando o processo através do SPEM, para, a partir do mapeamento, obter o modelo na notação BPMN.

4.2.2. Identificando critérios para a modelagem SPEM

O primeiro passo para modelar o processo é descobrir as correspondências entre os elementos identificados no modelo original e os do meta-modelo SPEM. Dessa forma, visualizando a figura 4.6, identificamos as seguintes correspondências:

- Os retângulos maiores que estão dispostos em forma de raias (comercial, engenharia e qualidade) serão representados por *Phases*, pois é possível perceber que eles representam o ciclo de vida do processo, caracterizando o fluxo de trabalho da empresa;

- Os retângulos na cor azul clara (controle, verificação e validação e desenvolvimento), serão definidos por sub-processos, visto que são auto contidos e independentes. Nesse ponto, é preciso fazer uma ressalva ao modelo definido pela empresa. Visto que planejamento e controle são práticas realizadas durante todo o ciclo de vida de um processo e não apenas na fase de engenharia, para uma melhor organização do ciclo de vida, faremos uma adaptação no processo. Desvincularemos as "caixas" de planejamento e controle da fase de engenharia e criaremos mais uma fase no modelo de ciclo de vida, chamada "Planejamento e Controle", o qual dará suporte contínuo às demais fases: comercial, engenharia e qualidade;

- Finalmente, os retângulos em cor azul escuro (teste, gerência de requisitos, gerência de configuração, etc.) correspondem a *ProcessPackages*, visto que criam categorias para as atividades e outros elementos que os compõem, como pode ser visualizado na figura 4.7, que apresenta o detalhamento para Gerência de Requisitos.

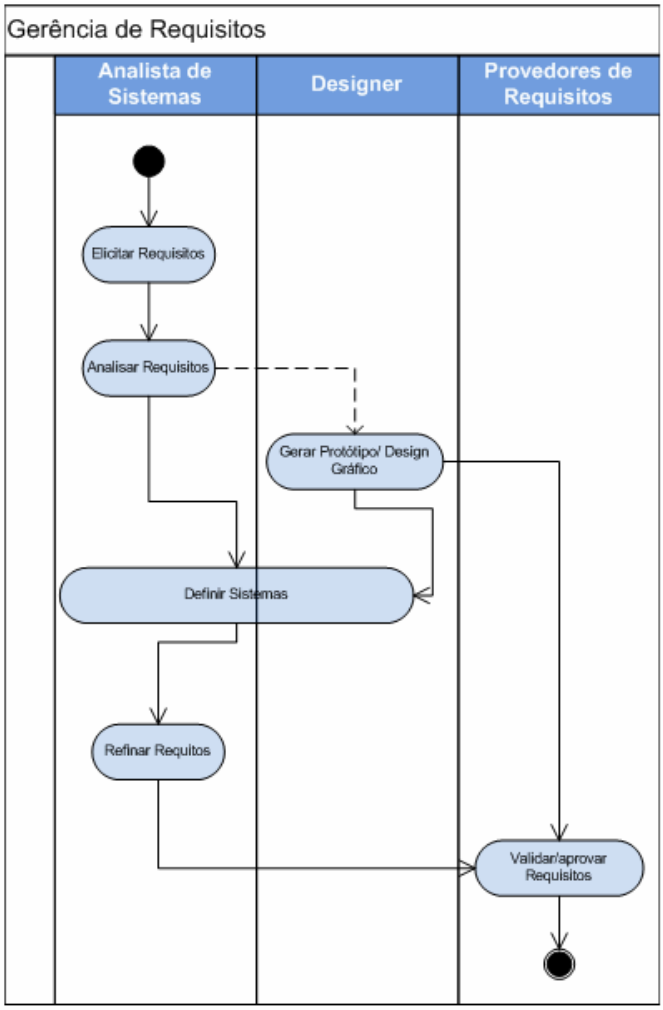


Figura 4.7 – Detalhamento do elemento Gerência de Requisitos do modelo de processo definido pela empresa

Analisando o detalhamento proposto para os retângulos em azul escuro, tomando como exemplo o de Gerência de Requisitos (figura 4.7), encontramos novas correspondências entre o modelo atual do processo e os elementos de SPEM:

- Os retângulos com bordas arredondadas em cor azul claro representam atividades, que não possuem mais nenhuma subdivisão (atividade elementar), por conseguinte, fica claro que serão visualizados através de *Activities* no modelo feito através do SPEM;

- Os retângulos maiores, em forma de raias (analista de sistemas, *designer*, provedores de requisitos), são os executores das atividades, por isso são equivalentes a *ProcessPerformers* e *ProcessRoles*.

Subindo ao nível de maior granularidade do modelo do processo em estudo, encontramos o detalhamento das atividades. Como exemplo, utilizaremos a atividade **Elicitar Requisitos** (figura 4.8), por ser a que possui a maior quantidade de elementos que podem ser modelados dentre as atividades do processo. Analisando a figura 4.8, é possível identificar algumas informações que podem ser modeladas através do SPEM:

- Os artefatos de entrada e saída (geração e consumo) podem ser representados por *WorkProducts* que são *Inputs* e *Outputs*, respectivamente, para uma atividade;

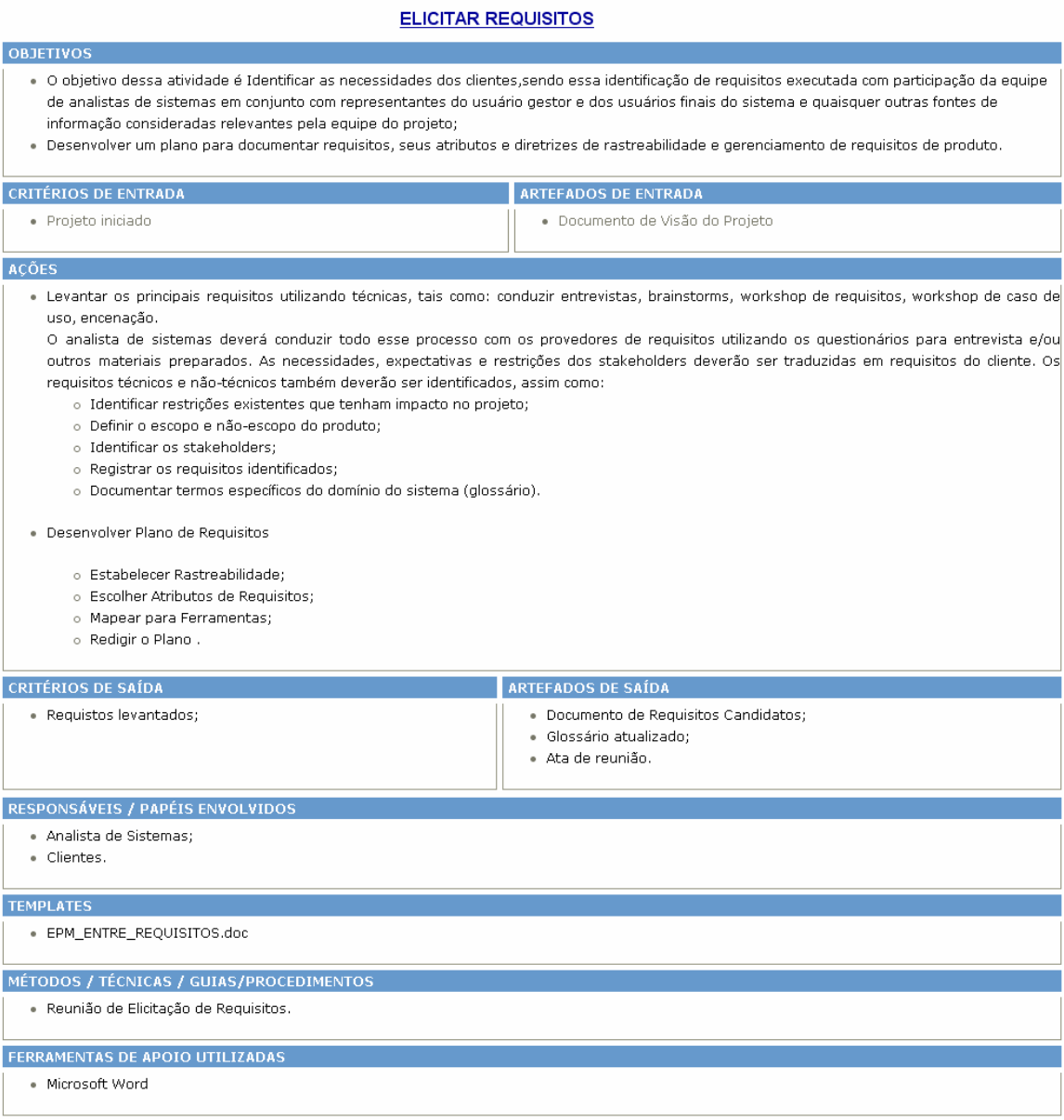
- As ações são equivalentes aos *Steps* no modelo SPEM. Assim, para modelar as ações, seria necessário criar mais um diagrama para cada atividade, no qual definiríamos o fluxo através de *Steps*, no entanto, consideramos que esse diagrama não é mais representativo nem esclarecedor que uma descrição textual como a apresentada na figura 4.8. Dessa forma, não chegaremos ao nível de detalhe da modelagem do fluxo de uma atividade;

- Responsáveis/Papéis envolvidos, como já apresentamos anteriormente, são equivalentes a *ProcessPerformers* e *ProcessRoles*;

- *Templates* serão modelados através de *Guidances* do tipo *Template;*

- Métodos/Técnicas/Guias/Procedimentos serão identificados por *Guidances* dos tipos *Guideline* e *Technique*;

- Para ferramentas de apoio não temos uma equivalência direta, mas utilizaremos *Guidance* do tipo *ToolMentor*, que orientam como utilizar uma ferramenta relacionada à determinada atividade.



ELICITAR REQUISITOS

OBJETIVOS

- O objetivo dessa atividade é Identificar as necessidades dos clientes,sendo essa identificação de requisitos executada com participação da equipe de analistas de sistemas em conjunto com representantes do usuário gestor e dos usuários finais do sistema e quaisquer outras fontes de informação consideradas relevantes pela equipe do projeto;
- Desenvolver um plano para documentar requisitos, seus atributos e diretrizes de rastreabilidade e gerenciamento de requisitos de produto.

CRITÉRIOS DE ENTRADA

- Projeto iniciado

ARTEFADOS DE ENTRADA

- Documento de Visão do Projeto

AÇÕES

- Levantar os principais requisitos utilizando técnicas, tais como: conduzir entrevistas, brainstorms, workshop de requisitos, workshop de caso de uso, encenação.
  O analista de sistemas deverá conduzir todo esse processo com os provedores de requisitos utilizando os questionários para entrevista e/ou outros materiais preparados. As necessidades, expectativas e restrições dos stakeholders deverão ser traduzidas em requisitos do cliente. Os requisitos técnicos e não-técnicos também deverão ser identificados, assim como:
  - Identificar restrições existentes que tenham impacto no projeto;
  - Definir o escopo e não-escopo do produto;
  - Identificar os stakeholders;
  - Registrar os requisitos identificados;
  - Documentar termos específicos do domínio do sistema (glossário).
- Desenvolver Plano de Requisitos
  - Estabelecer Rastreabilidade;
  - Escolher Atributos de Requisitos;
  - Mapear para Ferramentas;
  - Redigir o Plano .

CRITÉRIOS DE SAÍDA

- Requistos levantados;

ARTEFADOS DE SAÍDA

- Documento de Requisitos Candidatos;
- Glossário atualizado;
- Ata de reunião.

RESPONSÁVEIS / PAPÉIS ENVOLVIDOS

- Analista de Sistemas;
- Clientes.

TEMPLATES

- EPM_ENTRE_REQUISITOS.doc

MÉTODOS / TÉCNICAS / GUIAS/PROCEDIMENTOS

- Reunião de Elicitação de Requisitos.

FERRAMENTAS DE APOIO UTILIZADAS

- Microsoft Word

Figura 4.8 – Descrição da atividade Elicitar Requisitos do modelo de processo definido pela empresa

4.2.3. Modelo do processo por SPEM

Após definirmos as correspondências e os critérios para a modelagem do processo através de SPEM, podemos agora apresentar o modelo resultante dessa análise. De forma a considerar todos os aspectos discutidos na seção 4.2.2, optamos por apresentar quatro tipos de diagramas que consideramos necessários para uma representação satisfatória do modelo:

- Diagrama para o ciclo de vida, no qual são encontradas as fases e seu fluxo;
- Diagramas representando cada fase, com seus pacotes e sub-processos;
- Diagramas para cada sub-processo, no qual podemos identificar seu fluxo;
- Diagramas de atividades para cada pacote de processo, contendo o fluxo das atividades, seus produtos consumidos e gerados, os papéis responsáveis, etc.

Para fins de estudo de caso, consideramos desnecessário modelar todo o processo, sendo suficiente a modelagem de um exemplo para cada diagrama descrito acima, o que facilmente poderia ser replicado para os demais ativos de processo da empresa. Assim, o primeiro diagrama modelado foi o representando o ciclo de vida do processo (figura 4.9).

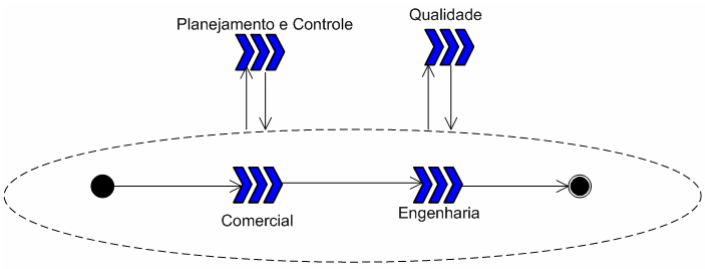


Figura 4.9 – Diagrama de ciclo de vida do processo

No diagrama da figura 4.9, é possível observar que as fases **Planejamento e Controle** e **Qualidade** são fases de apoio ao processo, que ocorrem durante todo o seu ciclo de vida e, por isso, não fazem parte do fluxo principal.

Para exemplificar um diagrama representando uma fase, escolhemos a fase de **Engenharia**, por possibilitar um modelo mais rico em variedade de elementos de processo. A figura 4.10 ilustra o modelo para a fase de **Engenharia**, que é constituída por dois sub-processos: **Desenvolvimento** e **Verificação e Validação**.

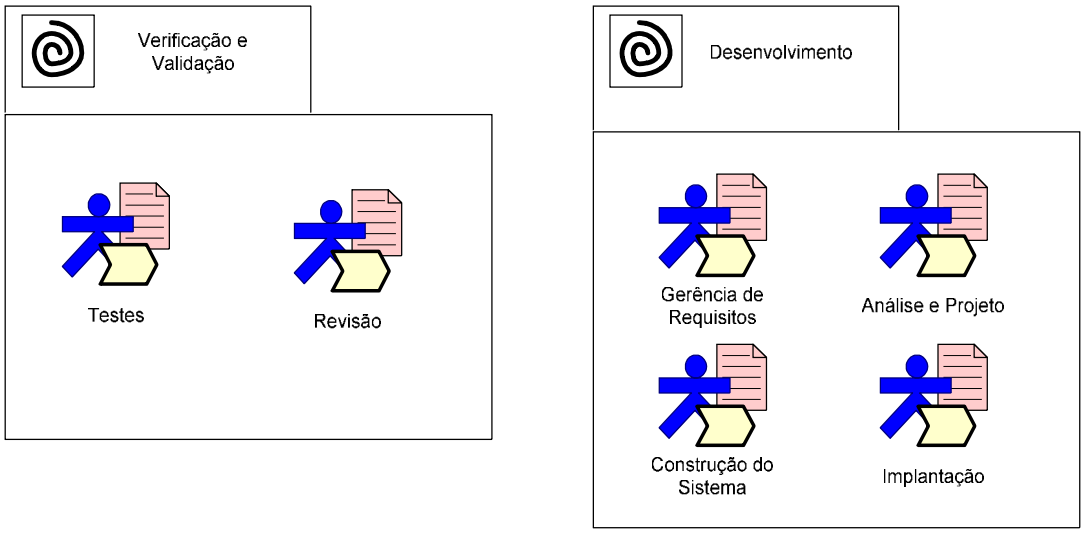


Figura 4.10 – Diagrama de pacotes da fase Engenharia

Como exemplo de modelagem de um sub-processo, escolhemos o sub-processo de **Desenvolvimento**, o seu diagrama é visto na figura 4.11. Nesse diagrama, é possível identificar as etapas do sub-processo, através dos pacotes que o compõem, bem como seu fluxo.

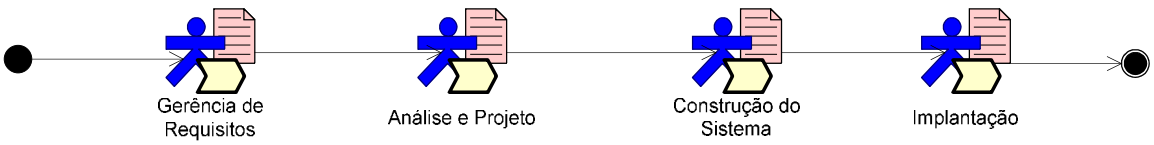


Figura 4.11 – Diagrama do sub-processo de Desenvolvimento

O último tipo de diagrama que iremos exemplificar é o diagrama de atividades que representa o trabalho desprendido num *ProcessPackage* do

processo, através do fluxo de atividades realizadas no pacote. Para o nosso exemplo, tomaremos o diagrama de **Gerência de Requisitos** do processo original, que pode ser visualizado na figura 4.7.

A figura 4.12 mostra a modelagem proposta para o pacote **Gerência de Requisitos** através do SPEM, baseado no modelo de processo original da empresa. Além do fluxo de atividades por seus responsáveis, identificados por *ProcessRoles*, podemos observar na figura a presença de artefatos que são produzidos e consumidos pelas atividades (Documento de Requisitos Formais, Casos de Uso, Protótipo, por exemplo), *UMLModel* (Casos de Uso), e *Guidances* (Template ASE_REQ_DRC.DOC, Template EPM_ATA.DOC, por exemplo). Esses elementos foram obtidos a partir das descrições das atividades do processo da empresa, que foi exemplificada através da figura 4.8, a qual exibe a descrição da atividade **Elicitar Requisitos**. Podemos observar que o diagrama não apresenta *Guidances* do tipo *Technique*, *Guideline* e *ToolMentor*, como podemos encontrar na descrição de algumas atividade do modelo original. A representação desses elementos foi suprimida apenas por considerarmos que elas não acrescentariam valor ao modelo, visto que são apenas tipos diferentes de *Guidances*, já representadas pelos *templates*.

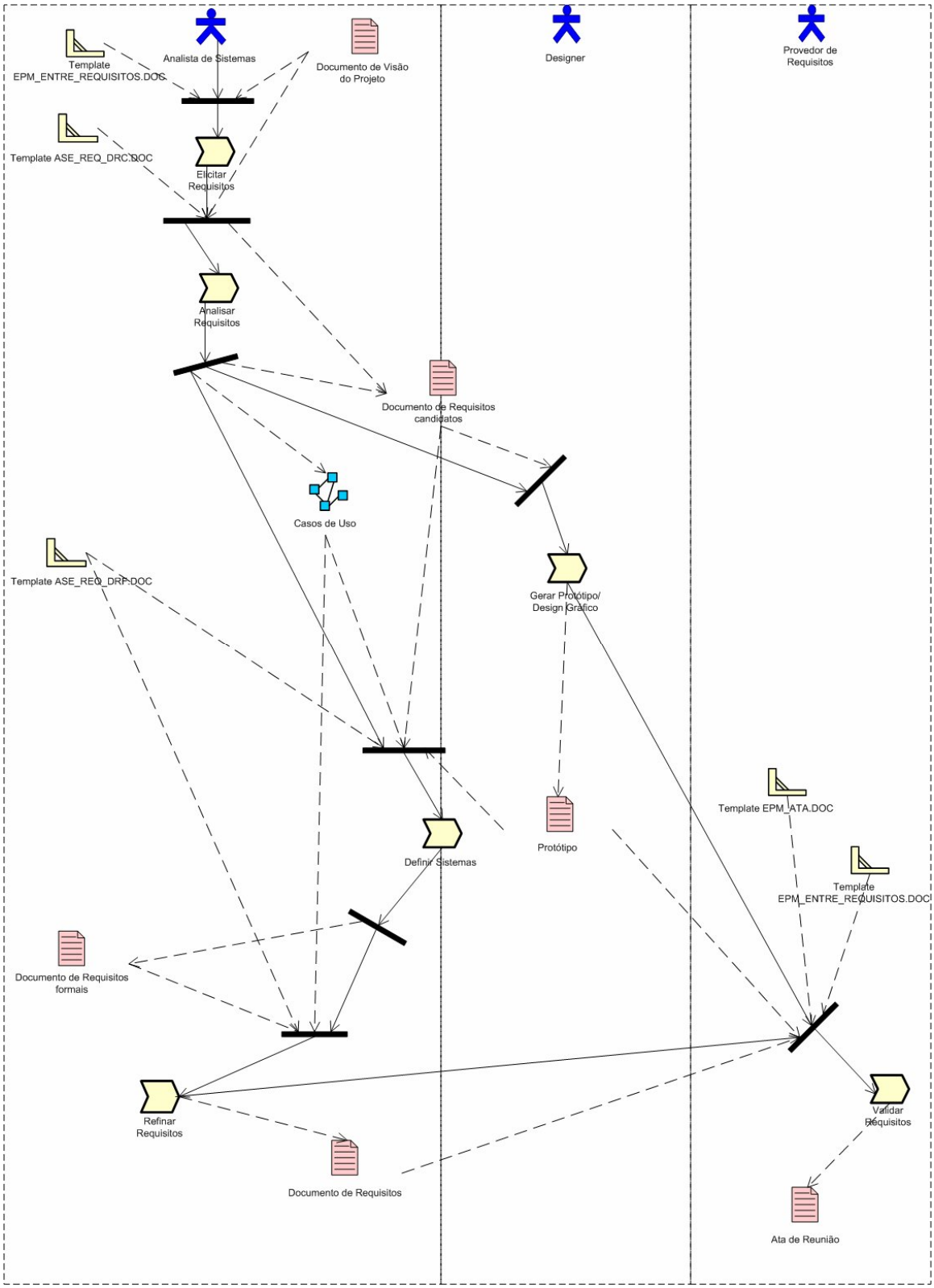


Figura 4.12 – Diagrama de atividades do pacote Gerência de Requisitos do modelo de processo da empresa

4.2.4. Modelo do processo por BPMN

Realizaremos, agora, a modelagem do processo da empresa utilizando a notação BPMN. Para isso, utilizaremos como base o modelo de processo da empresa, por ser a principal fonte de informação para estabelecer o comportamento do processo. Além disso, faremos uso do modelo obtido através do SPEM e do mapeamento entre SPEM e BPMN proposto na seção 4.1 deste trabalho, a fim de garantir a consistência entre os modelos nas duas tecnologias, além de validar o mapeamento proposto anteriormente. Adicionalmente, essa modelagem tem por objetivo avaliar as capacidades fornecidas por BPMN para a modelagem de processos de software. A avaliação com relação à presença de algumas características para modelos de processo tanto em BPMN quanto em SPEM serão discutidas na seção 4.3.

Para modelar o processo utilizando BPMN, procederemos de forma semelhante à que conduzimos a modelagem por SPEM. Sendo assim, exibiremos quatro tipos de diagramas:

- Um diagrama que representa a visão mais macro do processo, ou seja, o fluxo por suas fases;
- Um diagrama que exibe o comportamento interno a uma fase, representada por um *Embedded Sub-Process*, conforme o mapeamento constante na tabela 4.1;
- Um diagrama que apresenta o fluxo de um dos sub-processos de uma fase;
- Um diagrama que retrata a estrutura interna de um pacote, a seqüência de atividades que o compõem, os papéis, etc. No modelo

BPMN, o pacote será modelado utilizando-se um *Independent Sub-Process*, também de acordo com o mapeamento proposto;

Devido a um dos objetivos da modelagem ser a validação do mapeamento entre os padrões abordados no trabalho, os exemplos de cada tipo de diagrama serão os mesmos que os utilizados na modelagem SPEM. Logo, o primeiro diagrama modelado ilustra as fases do processo e seu fluxo, como mostra a figura 4.13.

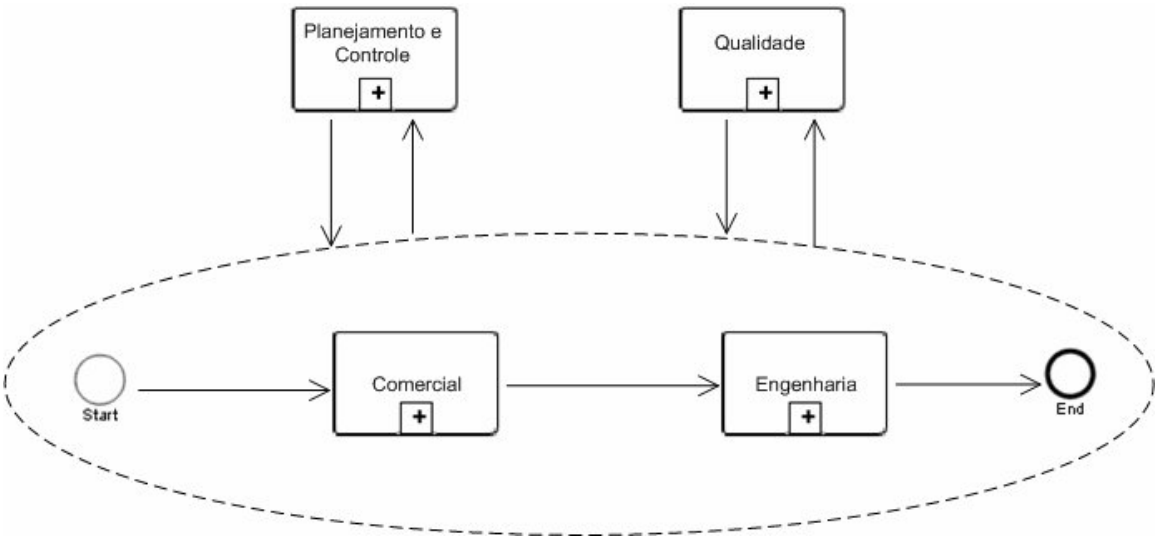


Figura 4.13 – Diagrama de fases do modelo de processo da empresa

O próximo diagrama (figura 4.14) representa a fase de **Engenharia**, com os seus sub-processos de **Desenvolvimento** e **Verificação e Validação**. Lembramos que as fases são representadas por *Embedded Sub-Process*, e por isso não podem apresentar *Pools* e *Lanes*.

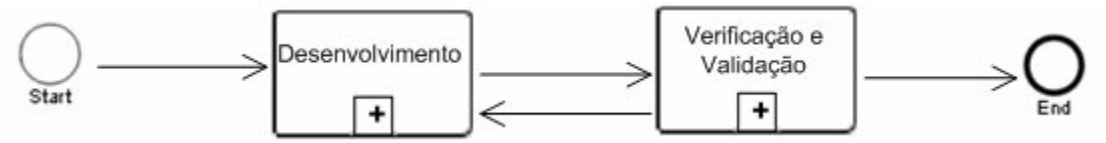


Figura 4.14 – Diagrama da fase de Engenharia do modelo de processo da empresa

O sub-processo escolhido na modelagem SPEM para apresentar o seu diagrama, como foi visto na seção 4.2.3, foi o de **Desenvolvimento**. Como na modelagem SPEM, ele apareceu como um fluxo entre pacotes de processo, de

acordo com o mapeamento proposto na tabela 4.1, no modelo BPMN ele apresentará um fluxo entre *Independent Sub-Process*, o que pode ser conferido na figura 4.15.

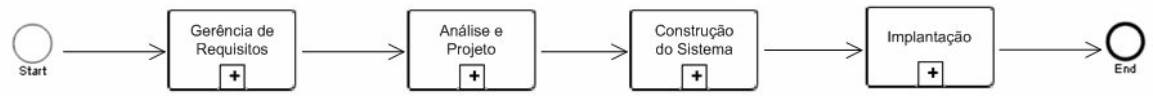


Figura 4.15 – Diagrama do sub-processo Desenvolvimento do modelo de processo da empresa

Por fim, modelamos o fluxo de atividades constituintes de um *Embedded Sub-Process* que representa um pacote de processo. Seguindo a escolha feita para o modelo SPEM, a figura 4.16 ilustra o diagrama para o *Embedded Sub-Process* **Gerência de Requisitos**.

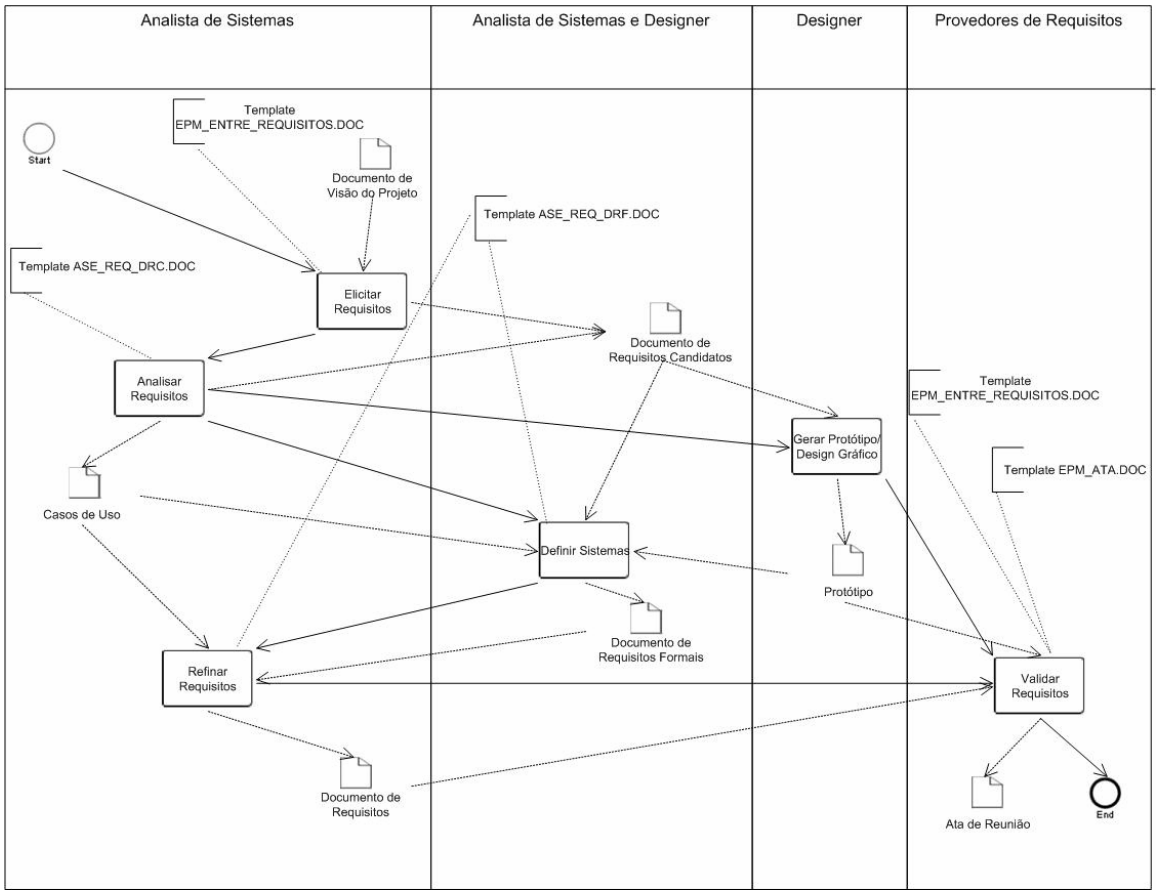


Figura 4.16 – Diagrama de atividades de Gerência de Requisitos do modelo de processo da organização

Conforme foi dito anteriormente, o processo da empresa que está sendo modelado ainda está em processo de elaboração, o que faz com que ele não nos possibilite extrair todas as regras sobre o seu fluxo. Por essa razão e também pelo fato de fugir ao escopo do estudo de caso, a definição de um processo de software, não foi possível realizar a modelagem com outros elementos que adicionam controle sobre o fluxo de atividades e artefatos do processo, tais como *Events* e *Gateways*.

4.3. Comparativo entre SPEM e BPMN

Realizado o mapeamento e o estudo de caso, faremos agora uma análise dos dois padrões com relação à presença de algumas características desejáveis para um modelo de processo de software. Assim, verificaremos a presença dos seguintes atributos na modelagem de processos de software através das duas notações, usando como base as recomendações presentes em [Pérez, 2007]:

- **Expressividade:** capacidade de representar a complexidade e todos os ativos dos processos de software;
- **Reuso:** capacidade de promover o reuso dos ativos constantes no modelo de processo;
- **Gestão:** suporte à gestão das instâncias do processo (planejamento, monitoramento e controle);
- **Evolução:** facilidade de identificar pontos ineficientes do processo, visando à melhoria e evolução dos mesmos;
- **Multinível:** capacidade de proporcionar desde visões mais alto nível do processo até outras com grande quantidade de detalhes;

- **Entendimento:** capacidade de compreensão do modelo por parte de todos os envolvidos no processo, sejam pessoas da organização ao qual o processo se destina ou os seus clientes, sobretudo aqueles que não são especialistas em modelagem de processo;

- **Integração organizacional:** possibilidade de se estabelecer integração e interação com os processos de outras áreas da organização, facilitando o alinhamento da definição dos processos com os objetivos globais da organização.

Para realizarmos a avaliação dos padrões quanto aos critérios definidos, definiremos os níveis de classificação quanto a sua presença:

- **Não Apresenta:** o padrão não apresenta a característica;

- **Apresenta Parcialmente:** a característica é encontrada, mas não de forma completamente satisfatória;

- **Apresenta Integralmente:** a notação dá total suporte à característica de modelagem avaliada.

A tabela 4.2 mostra a avaliação realizada sobre o SPEM e BPMN quanto aos critérios de modelagem de processos selecionados, utilizado os níveis definidos acima.

Tabela 4.2 Comparativo entre SPEM e BPMN

Notações / Características	SPEM	BPMN
Expressividade	Apresenta Integralmente	Apresenta Parcialmente
Reuso	Apresenta Integralmente	Apresenta Integralmente
Gestão	Apresenta Integralmente	Apresenta Parcialmente
Evolução	Apresenta Parcialmente	Apresenta Parcialmente
Multinível	Apresenta Integralmente	Apresenta Integralmente

Notações / Características	SPEM	BPMN
Entendimento	Apresenta Parcialmente	Apresenta Integralmente
Integração organizacional	Não Apresenta	Apresenta Integralmente

Acerca do comparativo realizado, é necessário realizarmos algumas considerações em cima da avaliação de alguns critérios:

- **Expressividade:** por não ser uma notação voltada especificamente para a modelagem de processos de software, o BPMN perde um pouco de expressividade, como, por exemplo, na representação de orientações adicionais mais específicas para a realização de atividades, que no SPEM podem ser modeladas através do elemento *Guidance*;

- **Evolução:** a modelagem realizada através dos dois padrões, por si só, não é suficiente para permitir a evolução do processo. A execução automatizada através do uso de linguagens de modelagem de processo pode auxiliar bastante na identificação de pontos falhos do processo, através de métricas fornecidas sobre o seu andamento. Nesse sentido, em [BPMN, 2004] encontra-se um mapeamento entre BPMN e a linguagem de execução **BPEL4WS** (*Business Process Execution Language for Web Services*);

- **Entendimento:** o ganho de expressividade obtido por SPEM na modelagem de processos de software tem como efeito colateral uma diminuição na capacidade de prover entendimento do modelo para pessoas que não são da área de engenharia de software. Isso não acontece com os modelos BPMN, visto que é um padrão para modelagem de processos de negócio em geral e tem como um dos

principais objetivos fornecer uma notação compreensível por todos os envolvidos no processo, segundo [BPMN, 2004];

- **Integração organizacional:** a especificidade dos elementos de processo fornecidos por SPEM, dificulta ou até mesmo impede que se consiga essa qualidade de modelos de processo. Já processos de software modelados através do BPMN, podem ser integrados com os demais modelos de processo de negócio de uma organização.

5. Conclusões

Neste trabalho, foi possível confrontar dois importantes padrões para modelagem de processos, SPEM e BPMN, visando à modelagem de processos de software. Para isso, foi preciso fundamentar uma abordagem diferente para modelagem de processos de software, na qual o processo é visto dentro de um contexto organizacional, ou seja, ele pode ser entendido como um dos processos de negócio da empresa. Isso faz com que a definição e modelagem de um processo de software possam ser realizadas com a finalidade de permitir a integração com demais processos organizacionais e alinhamento dos objetivos com as metas globais da instituição.

A sustentação dessa abordagem não tradicional para modelagem de processos de software foi obtida, primeiramente, através do mapeamento dos principais elementos do SPEM, voltado para modelagem de processo de software, para os elementos da notação BPMN, destinada a processos de negócio.

Posteriormente à realização do mapeamento, apresentamos um estudo de caso da modelagem do processo de software utilizado em um dos projetos de uma empresa de grande porte, através das duas notações. Lembramos que o processo em questão ainda está em fase de definição, o que prejudicou a qualidade do estudo de caso, no entanto, a modelagem realizada nos permitiu validar o mapeamento entre os padrões proposto, além de dar subsídios para o comparativo realizado entre as notações.

No comparativo, procuramos identificar algumas das principais qualidades desejáveis para um padrão de modelagem de processos de software para, a partir daí, e com base no mapeamento realizado entre SPEM e BPMN e, principalmente, com a experiência obtida através do estudo de caso, avaliarmos os padrões quanto ao nível de atendimento a cada uma das características escolhidas. Apresentando

pontos fortes e falhos de cada notação, espera-se que esse comparativo sirva como uma fonte de informação para quem necessita realizar a escolha por um padrão para modelagem de processos, com isso, o trabalho terá alcançado o seu principal objetivo.

5.1. Trabalhos Futuros

Após a conclusão deste trabalho, é possível identificar algumas possibilidades de continuidade em oportunidades futuras. Dentre elas, talvez a mais importante seja realizar um estudo de caso mais completo, o qual contemplaria a definição e modelagem de um processo de software completo para uma empresa de relativo porte utilizando SPEM e BPMN. Sendo interessante que a organização possua características que ofereçam oportunidades de integração do processo de software com outros processos de negócio porventura definidos pela instituição, a fim de que se possa avaliar o comportamento do modelo realizado com o uso do BPMN.

Após a definição e modelagem do processo através das duas notações, seria interessante a aplicação de ambos em projetos reais, possibilitando a obtenção de *feedback* e parâmetros que indiquem a eficácia de cada modelo. Esses trabalhos trariam como benefício adicional a possibilidade de extensão do mapeamento proposto entre os dois padrões, tanto contemplando elementos ainda não mapeados, como também identificando pontos de melhoria ou correção desse mapeamento.

5.2. Considerações Finais

O objetivo deste trabalho foi fornecer à comunidade de engenharia de software informações sobre dois importantes padrões para a modelagem de processos, que vêm obtendo crescente aceitação internacional e suporte de grandes empresas, através da OMG. Com o estudo apresentado, espera-se que seja possível obter uma visão geral da modelagem de processos de software realizada de forma mais tradicional, através do SPEM, ou fazendo uso de uma abordagem vista sob a ótica do Gerenciamento de Processos de Negócio, através do BPMN. A partir daí, deseja-se que o trabalho tenha provido meios para auxiliar na escolha da abordagem que melhor se adequa à realidade de uma organização.

Referências

[Abdala, 2004] Abdala, Martha Adriana Dias (2004) "Uma Abordagem para a Gerência das Modificações e da Configuração em um Ambiente Integrado para o Desenvolvimento e Gestão de Projetos de Software", Orientador Prof. Dr. Nilson Sant'Anna. Dissertação de Mestrado, Instituto Nacional de Pesquisas Espaciais, São José dos Campos, Brasil.

[Acuña, 2001] Acuña, Silvia T.; Ferré, Xavier (2001) "Software Process Modelling". World Multiconference on Systemics, Cybernetics and Informatics (SCI 2001). Orlando, FL (USA).

[Araujo, 2004] Araujo, R.; Capelli, C.; Gomes, JR, A. G.; Pereira, M.; Iendrike, H. S.; Ielpo, D.; Tovar, J. A. (2004) "A Definição de Processos de Software sob o ponto de vista da Gestão de Processos de Negócio". Artigo publicado no VI Simpósio Internacional de Melhoria de Processos de Software (SIMPROS). Departamento de Informática Aplicada/UNIRIO, Rio de Janeiro, Brasil.

[BPMN, 2004] BPMI - Business Process Modeling Notation (BPMN) versão 1.0, Maio de 2004.

[Combemale, 2006] Combemale, Benoit; Crégut, Xavier; Caplain, Alain; Coulette, Bernard (2006) "Towards a Rigorous Process Modeling with SPEM". ICEIS 2006 - Proceedings of the Eighth International Conference on Enterprise Information Systems.

[Filho, 2007] Filho, João Bosco A. Pinto (2007) "Gestão por Processos de Negócio: Uma Adaptação da Metodologia de Rummler-Brache Baseada numa Aplicação Real", Orientador Prof. Hermano Perrelli de Moura. Dissertação de Mestrado, CIn/UFPE, Recife, Brasil.

[Martins, 2004] Martins, Paula Ventura; Silva, Alberto Rodrigues (2004) "Comparação de Metamodelos de Processos de Desenvolvimento de Software". Artigo publicado no 5º Encontro para a Qualidade nas Tecnologias da Informação e Comunicações (QUATIC), Porto (Universidade Portucalense), Portugal.

[Mendes, 2005] Mendes, Rodrigo Cavalcante (2005) "Modelagem e Avaliação do CMMI no SPEM para Definição de um Meta-Processo de Software", Orientador Prof. Alexandre Marcos Lins de Vasconcelos. Trabalho de Graduação, CIn/UFPE, Recife, Brasil.

[Oliveira, 06] Oliveira, S. R. B. (2006) "Processo de Software: Princípios, Ambientes e Mecanismos de Execução", Orientador Prof. Alexandre Marcos Lins de Vasconcelos. Exame de Qualificação do Doutorado, CIn/UFPE, Recife, Brasil.

[OMG, 2007] Object Management Group (OMG) - http://www.omg.org/. Último acesso em 05 de agosto de 2007.

[Owen, 2003] Owen, Martin; Raj, Jog (2003) "BPMN and business process management; Introduction to the new business process modeling standard," White Paper. Popkin Software. September, 2003.

[Pérez, 2007] Pérez, Juan Diego (2007) "Notaciones y lenguajes de procesos. Una visión global", Supervised by Prof. Dr. Amador Durán Toro. Research Report in in partial fulfilment of the requirements for the degree of Ph.D. in Computer Engineering, University of Sevilla, Sevilla, Spain.

[Reis, 2003] Reis, Carla Alessandra L. (2003) "Uma Abordagem Flexível para Execução de Processos de Software Evolutivos", Orientador Prof. Dr. Daltro José Nunes. Tese de doutorado, Universidade Federal do Rio Grande do Sul, Instituto de Informática, Porto Alegre, Brasil.

[Reis, 2004] Reis, Carla Alessandra L. (2004) "Introdução à Modelagem de Processos de Software". XI SEMANA DE INFORMÁTICA DA UFPA, Belém, Brasil.

[Silva, 2006] Silva, Renato A. C.; Soares, Liziane S.; Braga, José L. (2006) "Workflow Aplicado a Engenharia de Software Baseada em Processos: Uma Visão Geral". INFOCOMP (Journal of Computer Science), vol. 5, n°3, p.76-84.

[SPEM, 2005] OMG - Software Process Engineering Metamodel Specification versão 1.1. Janeiro de 2005.

Orientador

Prof. Ph.D. Alexandre Marcos Lins de Vasconcelos

Aluno

Manoel Gilvan Calou De Araújo e Sá Filho