PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
Programa de Pós–Graduação em Engenharia Elétrica

Renato Aurélio Sales Nascimento

UMA PROPOSTA BASEADA NA TÉCNICA DE DECISÃO MULTICRITÉRIO PARA SELECIONAR A CONFIGURAÇÃO DO PROCESSO DE DESENVOLVIMENTO DE SOFTWARE PARA UM PROJETO

Belo Horizonte
2012

Renato Aurélio Sales Nascimento

UMA PROPOSTA BASEADA NA TÉCNICA DE DECISÃO MULTICRITÉRIO PARA SELECIONAR A CONFIGURAÇÃO DO PROCESSO DE DESENVOLVIMENTO DE SOFTWARE PARA UM PROJETO

Dissertação apresentada ao programa de Pós-Graduação em Engenharia Elétrica da Pontifícia Universidade Católica de Minas Gerais, como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica.

Orientador: Carlos Alberto Marques Pietrobon
Co-Orientador: Petr Iakovlevitch Ekel

Belo Horizonte
2012

FICHA CATALOGRÁFICA
Elaborada pela Biblioteca da Pontifícia Universidade Católica de Minas Gerais

Nascimento, Renato Aurélio Sales
N244p Uma proposta baseada na técnica de decisão multicritério para selecionar a configuração do processo de desenvolvimento de software para um projeto / Renato Aurélio Sales Nascimento. Belo Horizonte, 2012.
110f.: il.

Orientador: Carlos Alberto Marques Pietrobon
Co-Orientador: Petr Iakovlevitch Ekel
Dissertação (Mestrado) - Pontifícia Universidade Católica de Minas Gerais. Programa de Pós-graduação em Engenharia Elétrica.

1. Software – Desenvolvimento. 2. Gestão do conhecimento. 3. Conjuntos difusos. 4. Processo decisório por critério múltiplo. I. Pietrobon, Carlos Alberto Marques. II. Ekel, Petr Iakovlevitch. III. Pontifícia Universidade Católica de Minas Gerais. Programa de Pós-Graduação em Engenharia Elétrica. IV. Título.

CDU: 681.3.03

Renato Aurélio Sales Nascimento

UMA PROPOSTA BASEADA NA TÉCNICA DE DECISÃO MULTICRITÉRIO PARA SELECIONAR A CONFIGURAÇÃO DO PROCESSO DE DESENVOLVIMENTO DE SOFTWARE PARA UM PROJETO

Dissertação apresentada ao programa de Pós-Graduação em Engenharia Elétrica da Pontifícia Universidade Católica de Minas Gerais, como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica.

_____

Prof. Dr. Carlos Alberto Marques Pietrobon (Orientador) – PUC Minas

_____

Prof. Dr. Petr Iakovlevitch Ekel (Co-Orientador) – PUC Minas

_____

Prof. Dr. Ana Liddy Cenni de Castro Magalhães - UFMG

Belo Horizonte, 13 de Dezembro de 2012

Ao meu pai e á minha mãe pelo incentivo e carinho.

AGRADECIMENTOS

Agradeço a todos que, de alguma forma, ajudaram na realização desta pesquisa. Em especial ao meu orientador Carlos Alberto Marques Pietrobon e ao meu co-orientador Petr Iakovlevitch Ekel pelo apoio, paciência, ajuda e orientação durante o período de realização desta pesquisa e principalmente pelas experiências profissionais e pessoais passadas. Agradeço aos meus pais, meus irmãos e minha família pela criação que tive e pelo apoio, amor e amizade que sempre recebi. Agradeço a meus amigos pela amizade e força e a todos por sempre acreditarem em mim e por entenderem os momentos em que não estive presente, os quais investi na realização desta pesquisa.

Aos amigos de mestrado e pesquisa, agradeço pela ajuda nos trabalhos e pelo companheirismo em diversos momentos.

Um agradecimento especial para a Pontifícia Universidade Católica de Minas Gerais e ao Programa de Pós-Graduação em Engenharia Elétrica pela formação recebida.

RESUMO

A instanciação do processo de desenvolvimento de software em um projeto é uma tarefa complexa que requer do grupo de gestão do projeto alto conhecimento dos requisitos, dos objetivos e das restrições do projeto como custo, qualidade e tempo. Este trabalho propõe um método para esta importante atividade através da técnica de decisão multicritério em grupo baseada na função de relação de preferências *Fuzzy* e gestão de conhecimento. O objetivo é aumentar a justificativa e eficiência real da escolha da configuração das atividades presentes no processo organizacional com base no processamento das preferências dos envolvidos na tomada de decisão e através da gestão de conhecimento das informações envolvidas. O método proposto foi aplicado em um projeto real de desenvolvimento de software com o objetivo de ser validado. Como resultado da aplicação, foi possível detectar melhorias significativas em fatores relacionados à gestão de qualidade do projeto tais como tempo e custo.

Palavras-chave: Processo de desenvolvimento de Software. Gestão do conhecimento. Conjuntos *Fuzzy*. Decisão Multicritério em Grupo.

ABSTRACT

The software developing project process instantiation is a complex task that requires from the project management group deep knowledge of requirements, objectives and project constraints such as cost, quality and time. This work proposes a method for this important activity using a multicriteria group decision technique based on fuzzy preference relation function and Knowledge Management. The goal is to increase the efficiency of the activities configuration choice from the organization process for a specific project based on the professional's preferences and through the knowledge management. The proposed method was applied to a real software development project in order to be validated. As result of method application, significant improvements in factors related to project quality management such as time and cost was found.

Keywords: Software Development Process. Knowledge Management. Fuzzy Sets. Multicriteria Group Decision.

LISTAS DE FIGURAS

LISTAS DE TABELAS

LISTAS DE QUADROS

SUMÁRIO

1. INTRODUÇÃO

Nos últimos anos foi presenciado uma crescente cobrança do mercado mundial com relação à qualidade com que os softwares são produzidos. A comprovação da capacidade de desenvolvimento de softwares com qualidade por parte da organização desenvolvedora se tornou um importante fator usado por clientes privados e governamentais para escolherem a empresa que irá produzir o software necessitado.

A capacidade de desenvolvimento está diretamente relacionada à gestão da qualidade. De acordo com Goetsch (2012), a gestão de qualidade de software tem como objetivo garantir que a qualidade do produto, o cronograma, o escopo e o orçamento estão de acordo com os requisitos do cliente e os procedimentos da organização. Sabe-se que o desenvolvimento de um software é um processo complicado, que requer a integração de diversas disciplinas, atividades técnicas, gerenciamento de projeto, etc.. (PRESSMAN, 2009)

Não importa o quanto elegante sejam os métodos utilizados para testar o produto final, o quanto é completo a documentação produzida, o quanto é estruturado a metodologia de desenvolvimento, a gestão de configuração e os planos do projeto, não importa o quanto avançado as ferramentas utilizadas são, o projeto do desenvolvimento de um software tem altas possibilidades de falhar se o sistema de gestão de qualidade não for efetivo (GOETSCH, 2012).

A forma como os projetos de desenvolvimento de software aplicam a gestão da qualidade é formalmente estabelecida através do processo de desenvolvimento de software definido internamente pela empresa (FAYAD, 1997). E, para garantir que estes processos são eficientes o suficientes para possibilitar a alta qualidade do software a ser produzido, muitas empresas buscam por avaliações de maturidade de processos de desenvolvimento de software como o CMMI (CHRISSIS et al, 1997) e o MPS.Br (MONTONI et al, 2009). Os resultados das avaliações são utilizados como comprovadores da capacidade e maturidade dos processos de desenvolvimento de software e, quando aplicados corretamente, trazem, comprovadamente, uma gama de benefícios ao projeto (NIAZI, 2006).

Um processo maduro define de forma detalhada um conjunto de atividades que precisam ser executadas durante o desenvolvimento de software. Juntamente com estas atividades, é definido um conjunto de produtos de trabalho que evidenciam a sua execução (ROUT, 2007).

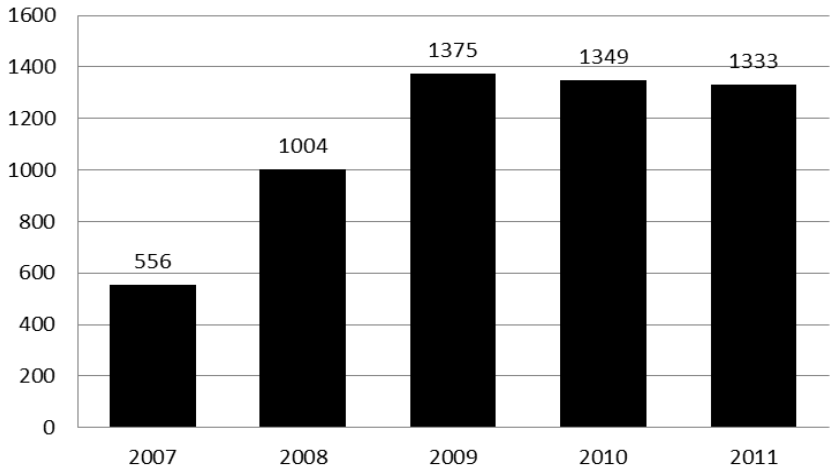
No inicio de um projeto, o grupo de gestão deve decidir quais atividades definidas no processo organizacional são aplicáveis para o projeto e quais os produtos de trabalho que

serão gerados como base para atender as características do projeto. Esta atividade é comumente conhecida como “instanciação do processo de software para o projeto” (ROUT, 2007).

A instanciação de processo é uma tarefa complexa que requer do grupo de gestão do projeto alto conhecimento do processo de desenvolvimento de software organizacional, dos requisitos envolvidos no projeto, dos objetivos do projeto e das suas restrições como custo, tempo e recursos humanos. A escolha indevida das atividades e produtos de trabalho irá certamente causar impactos na gestão de qualidade do projeto e no próprio projeto. A seleção de uma grande quantidade de atividades de forma não criteriosa leva a um aumento do custo do projeto e do tempo, uma vez que atividades que agregam pouco ou nenhum valor poderão ser ativadas. No caminho contrário, selecionar uma pequena quantidade de atividades ou deixar de selecionar uma que seja relevante, acarreta numa baixa visibilidade de gestão do projeto e impacta na qualidade do software produzido. Portanto, considerando que o processo de desenvolvimento de software possui maturidade e capacidade suficiente para possibilitar alta qualidade do software, o sucesso do projeto fica então fortemente dependendo de como este projeto irá aplicar o processo.

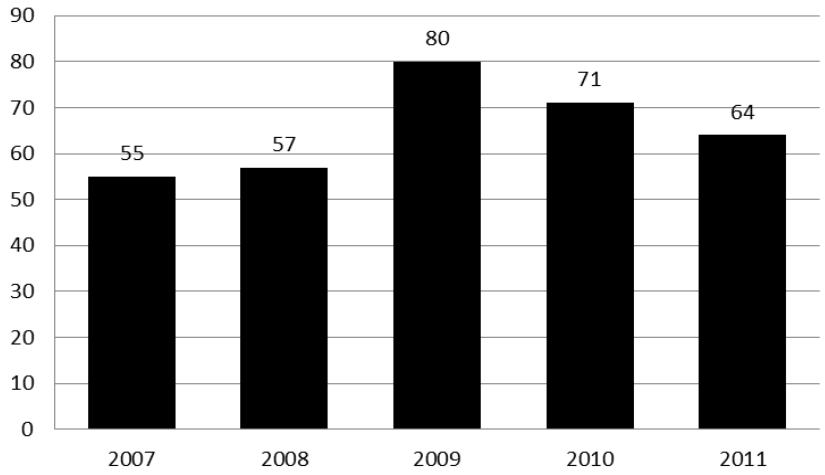
O interesse por melhorar a qualidade de software pode ser visto com o crescimento na obtenção de certificados de maturidade e capacidade de processos de desenvolvimento de software como mostrado nas Figuras 1 e 2. Porém, outra constatação importante é que, apesar dos esforços empenhados em CMMI, MPS-BR e outras propostas de melhorias de qualidade, muitas empresas ainda continuam possuindo um índice considerável de projetos com resultados negativos.

Figura 1 - Avaliações CMMI ao longo dos anos



Fonte: SEI (2012) (adaptado)

Figura 2 - Avaliações MPS.br ao longo dos anos



Fonte: Softex (2012) (adaptado)

De acordo com os resultados apresentados pelos relatórios *Chaos* desde 1994 até 2009 (Tabela 1) fornecido pela organização *The Standish Group* (Standish Group International, 2011), verifica-se que apenas cerca de 32% de todos os projetos de desenvolvimento de software são finalizados com sucesso (o que significa entregar o software dentro do prazo e custo previstos, com todas os requisitos atendidos) mesmo considerando os diversos ferramentais e conhecimentos de gestão de qualidade encontrado na literatura e nas próprias empresas.

Tabela 1 - Comparação dos resultados apresentados pelos relatórios Chaos

Apresentação Resumida dos Resultados do Relatório *Chaos* ate 2009								
	1994	**1996**	**1998**	**2000**	**2002**	**2004**	**2006**	**2009**
Successful	16%	27%	26%	28%	34%	29%	35%	32%
Challenged	53%	33%	46%	49%	51%	53%	46%	44%
Failed	31%	40%	28%	23%	15%	18%	19%	24%

Fonte: The Standish Group (1994 ate 2009)

O mau uso do processo de desenvolvimento de software é um dos principais fatores contribuintes para a falha de projetos (CERPA et al, 2009). E, certamente, a instanciação do processo de desenvolvimento de software é um fator de extrema importância dentro do contexto de uso do processo.

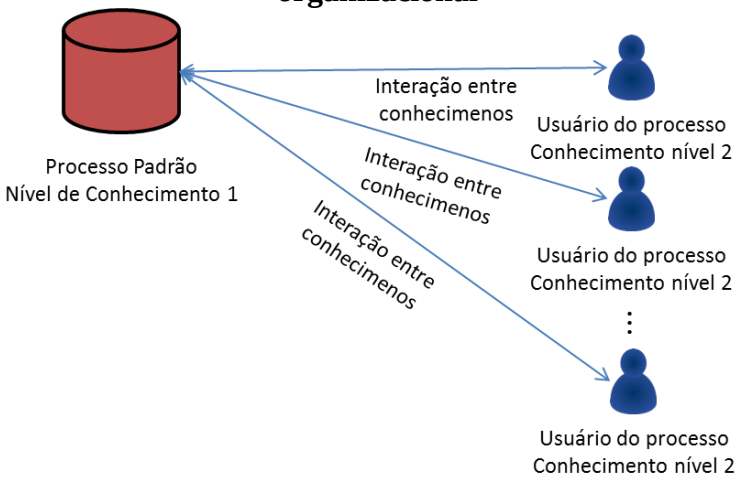
Melhorar as escolhas realizadas durante a instanciação do processo para o projeto é importante para garantir a expressão da maturidade do processo de desenvolvimento de software no projeto. Porém, as próprias características da atividade de instanciação dificultam atingir essa meta. Por exemplo, quando a instanciação do processo de software é executada

apenas por um único individuo (o gerente do projeto, por exemplo), se peca no quesito de confiabilidade. Por outro lado, quando é executada por um grupo de pessoas, é necessário lidar com diferentes pontos de vistas e diferentes níveis de conhecimento sobre o problema o que dificulta chegar a um resultado final coeso.

Além disso, as informações de entrada para as tomadas das decisões nessas atividades não são precisas e os julgamentos realizados muita vezes não podem ser expressos como “sim” ou “não” por envolver um grau de incerteza. Por exemplo, uma atividade do processo de desenvolvimento de software de uma empresa pode trazer algumas vantagens para um determinado projeto, porém, ao mesmo tempo, um custo relevante. A relação de custo e beneficio orienta a fazer uso dessa atividade? Decisões como esta fazem parte da rotina de instanciação do processo organizacional e normalmente não são simples de serem feitas devido a inúmeros fatores relacionados.

Outro fator relevante a ser considerado na problemática da instanciação do processo de desenvolvimento de software é que, à medida que uma organização de desenvolvimento de software evolui em nível de maturidade, é natural que o seu processo de desenvolvimento de software cresça e se torne, de certa forma, mais complexo. Um processo de alta maturidade possui um grande conjunto de informações proveniente de diferentes fontes de conhecimento e é capaz de suportar o desenvolvimento de um software nas mais diversas situações. Porém, é inevitável que um processo nesse nível seja mais complexo de ser usado e exija do usuário um maior conhecimento para determinar exatamente quais informações ali realmente agregam valor para ele ou não. Chega-se assim em uma situação em que um conhecimento de segundo nível (conhecimento do usuário) torna-se um fator limitador para a expressão do conhecimento agregado de primeiro nível (conhecimento presente no processo) (Figura 3). A interação entre conhecimentos desses níveis passa a ser então um fator crucial para atingir bons resultados.

Figura 3 - Interação entre conhecimentos durante o uso do processo organizacional



Fonte: Elaborado pelo Autor

Os fatos acima expõem o problema motivador deste trabalho: Como realizar a instanciação de processos de desenvolvimento de software de uma forma mais criteriosa e com maior objetividade para um projeto, considerando a escolha dos itens de projetos feitos a partir da opinião de diversos interessados? Qual conjunto de atividades é o melhor para um projeto que possui um conjunto específico de restrições e características? Como potencializar a qualidade de interação de conhecimento entre o usuário e o processo de desenvolvimento de software?

1.1. Objetivo

O grupo de gestão de projeto necessita decidir por um conjunto de atividades do processo de desenvolvimento de software organizacional que, segundo a sua opinião coletiva, se apresenta como a melhor configuração de atividades levando em consideração diversas restrições e características do projeto. Portanto, tem-se um contexto claro de tomada de decisão em grupo com múltiplos critérios envolvidos.

Técnicas de decisão multicritério em grupo (PEDRYCZ et al, 2011) propõem soluções para lidar com o julgamento de itens por um grupo quando se é necessário realizar a análise perante vários critérios. No entanto, o problema apresentado neste trabalho possui ainda uma característica importante: o julgamento dos especialistas se baseia em informações quantitativas incertas/imprecisas e qualitativas, ou seja, normalmente as informações não apresentam um alto grau de certeza. Devido a isso, a utilização da lógica *fuzzy* como apoio á tomada de decisão se torna um fator importante. Mais informações sobre as técnicas de

decisão multicritério em grupo e sobre a lógica *fuzzy* serão apresentadas nos próximos capítulos.

É importante considerar também que cada empresa e cada projeto de desenvolvimento de software possuem características únicas e que irão influenciar na tomada de decisão em questão. A solução para o problema exposto anteriormente deve, então, ser construída de forma que possa ser adaptada facilmente à diversidade presente no mercado de trabalho.

Ainda é relevante considerar os problemas relacionados à interação de conhecimento entre o processo padrão e seus usuários como um fator limitador da expressão da qualidade do processo no projeto. Técnicas de gestão de conhecimento podem ser adaptadas para tentar solucionar esse problema.

Este trabalho, portanto, tem como objetivo desenvolver uma abordagem formal baseada em técnicas de tomada de decisão em ambiente *fuzzy* para selecionar a configuração das atividades que irão compor o processo de desenvolvimento de software utilizado por um projeto. Com o uso desta abordagem, será possível justificar com maior eficiência a escolha da configuração das atividades relacionadas ao processo de desenvolvimento de software organizacional com base na formação e no processamento matemático das preferências dos profissionais envolvidos.

Outro objetivo deste trabalho é agregar a esse método técnicas de gestão de conhecimento para possibilitar que a interação de conhecimento entre o usuário de um processo de desenvolvimento de software e o próprio processo seja mais efetiva.

1.2. Metodologia

Inicialmente foi realizada uma revisão bibliográfica sobre os 5 pilares do método proposto por este trabalho: processo de desenvolvimento de software, maturidade de processos de software, o problema de decisão multicritério em grupo, gestão de conhecimento e, por fim, sobre os fundamentos dos conjuntos *fuzzy*. A revisão realizada teve como objetivo dar embasamento teórico para o desenvolvimento do método proposto neste trabalho.

O levantamento bibliográfico foi realizado por meio de uma revisão sistemática tendo como base fontes tais como o *Institute of Electrical and Electronic Enginners (IEEE)*, *SpringerLink* e *Association for Computing Machinery Digital Library (ACM).* Além disso, diversos livros de autores especialistas na área foram consultados. No IEEE, por exemplo, o termo “Gestão de Conhecimento” retornou 25.355 resultados e, quando associado ao termo “Engenharia de Software”, foi retornado

6.054 resultados. Desses resultados, foi realizado um filtro baseando-se nos objetivos deste trabalho e na relevância que os artigos possuem definida pela própria IEEE.

Após o levantamento bibliográfico, o próximo passo consistiu em estabelecer a proposta que permite avaliar as atividades do processo. Para isso, foi necessário definir um conjunto de critérios para a avaliação das atividades do processo de software, os termos linguísticos (EKEL et al, 2008) utilizados pelos avaliadores para expressarem suas opiniões de forma estruturada e a forma de consenso para se obter um valor único de avaliação para uma determinada atividade do processo.

Com o estabelecimento do método para avaliação das atividades de um processo, o próximo passo consistiu em definir a metodologia de tomada de decisão utilizando recursos da lógica *fuzzy*. Dessa forma, o julgamento dos especialistas envolvidos na atividade de instanciação do processo de software passa por um processamento matemático que possui como resultado uma configuração ótima de atividades para serem empregadas no projeto sob o ponto de vista das várias opiniões dos especialistas envolvidos.

Após a elaboração do método, foi necessário realizar a sua verificação e validação. Ambas realizadas através da aplicação do método em um caso real de projeto de desenvolvimento de software.

1.3. Organização do Trabalho

Além deste capitulo de introdução, este trabalho possui ainda mais 7 capítulos que então resumidamente descritos a seguir.

O Capítulo 2 – Processos de Desenvolvimento de Software – possui como finalidade analisar referências bibliográficas relacionadas aos processos de desenvolvimento de software, instanciação de processo, maturidade de processo e gestão de conhecimento.

O capítulo 3, denominado Tomada de Decisão Multcritério, apresenta a problemática de tomada de decisão e os fatores envolvidos neste processo.

O capítulo 4, sobre Lógica *Fuzzy*, apresenta os principais conceitos envolvidos com a lógica *fuzzy* e como a mesma pode ser utilizada para resolver problemas de tomada de decisão multicritério.

O capítulo 5 apresenta a método proposto por este trabalho combinando técnicas de decisão multicritério em grupo com a lógica *fuzzy* para aperfeiçoar a instanciação do processo de desenvolvimento de software.

No capitulo 6 é apresentado ao leitor os procedimentos de verificação e validação da proposta deste trabalho e seus resultados.

No capitulo 7, uma análise de requisitos levantados para o desenvolvimento de uma ferramenta de suporte para aplicação do método proposto é apresentada. Além disso, também é proposto uma arquitetura para sua implementação.

Por fim, no capitulo 8 é apresentado as conclusões finais do trabalho e dos resultados obtidos.

2. PROCESSOS DE DESENVOLVIMENTO SOFTWARE

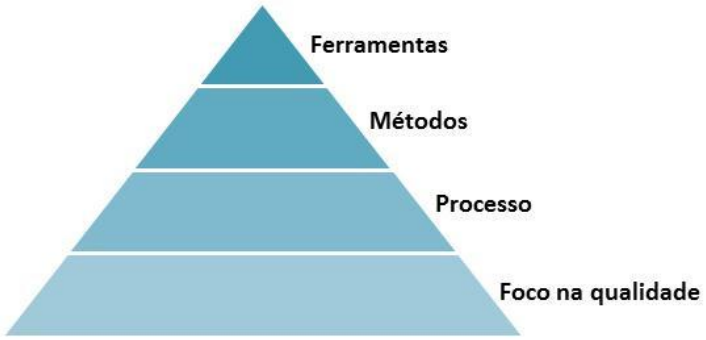
De maneira genérica, um processo pode ser definido como uma série de ações na qual uma ou mais entradas são utilizadas para produzir uma ou mais saídas (PRESSMAN, 2009).

Processos atuam sobre domínios bastante distintos, como projetos de engenharia, produção industrial, recrutamento de funcionários, entre outros. Cada um desses domínios apresenta necessidades e regras específicas. Por esta razão, os processos apresentam diferentes características em cada domínio em que são aplicados. Entretanto, é possível notar que existem similaridades entre processos de domínios distintos. Por exemplo, a maioria dos projetos de engenharia apresenta uma natureza flexível e dinâmica, não sendo possível encontrar um único modelo de processo capaz de atender as necessidades de todos os tipos de projeto de engenharia. Entretanto, reconhecidamente existem muitas similaridades entre diferentes tipos de projetos de engenharia. A maioria dos projetos de engenharia passa pelas seguintes grandes etapas: levantamento de requisitos, especificação e projeto.

Todas as características apresentadas acima estão presentes também no processo de engenharia de software. De acordo com Sommerville (2007), a engenharia de software é uma disciplina da engenharia que diz respeito a todos os aspectos da produção de software, desde os estágios iniciais de especificação do sistema até a sua manutenção após ter sido colocado em uso. Pressman (2009) define engenharia de software como o "estabelecimento e uso de princípios sólidos de engenharia para a construção e manutenção de software, com o objetivo de obter software confiável e que funcione corretamente em máquinas reais".

Pressman (2009) visualiza a engenharia de software como uma tecnologia em camadas conforme apresentado na Figura 4. Para ele, o foco na qualidade é condição imprescindível de suporte à engenharia de software. Se não houver um comprometimento da organização para com a qualidade, a engenharia de software não se sustenta. No modelo proposto por Pressman, a camada de processo é a camada chave que possibilita o desenvolvimento e a manutenção de software de forma racional e em tempo razoável. Essa camada faz a ligação entre os princípios de qualidade que regem a organização, as disciplinas necessárias para a construção e manutenção de software, os métodos e técnicas de cada disciplina e as ferramentas. A camada de método diz respeito ao conhecimento técnico sobre "como fazer". Os métodos abrangem um amplo conjunto de tarefas, tais como análise de requisitos, projeto, codificação, teste, entre outros. Finalmente, a última camada diz respeito às ferramentas que provêm suporte automatizado para os processos e métodos de engenharia de software.

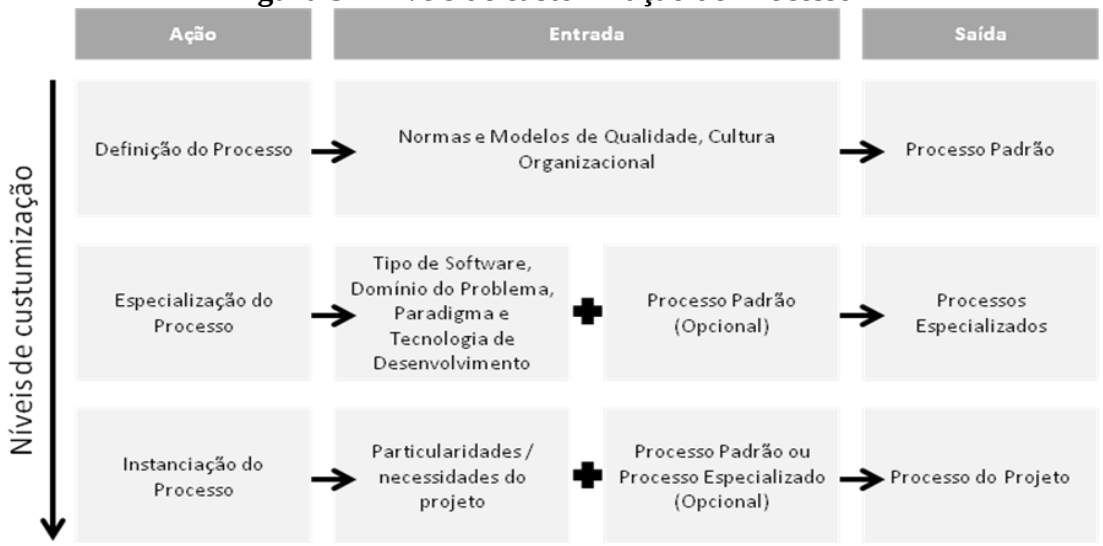
Figura 4 - Camadas da Engenharia de Software



Fonte: Elaborado pelo Autor

De forma geral, é comum que organizações estabeleçam um conjunto de ativos (sub-processos, atividades, tarefas, artefatos, papéis, etc.) como parâmetro para os projetos de desenvolvimento de software que possam vir a ocorrer. Esse conjunto de ativos corresponde aos elementos das camadas de processo, métodos e ferramentas apresentadas anteriormente e constituem o que é denominado de Processo Padrão de Desenvolvimento de Software de uma organização (SOMMERVILLE, 2007). Devido ao fato de que organizações podem atender a domínios diferentes de desenvolvimento de software, ainda é comum que se crie especializações do Processo Padrão com o objetivo de atender com maior eficiência às necessidades específicas daqueles domínios. Em nível de projeto, ainda pode-se identificar necessidades específicas do projeto que podem ser provenientes de restrições como tempo, orçamento, qualidade e etc.. Devido a isso, é comum que a gestão do projeto ainda faça uma customização do processo padrão (seja ele especializado ou não) para atendimento de suas necessidades. Essa atividade de customização é denominada instanciação do processo padrão de desenvolvimento de software. O modelo para definição de Processos em Níveis apresentado pelo ISO/IEC 12207 (ISO, 1998) descreve tais procedimentos de customização do processo de software (Figura 5).

Figura 5 - Níveis de customização de Processo



Fonte: Elaborado pelo Autor

Uma vez que o processo padrão seja estabelecido, o conhecimento agregado ali pode ser reutilizado pelo projeto através da instanciação do processo no projeto. Essa atividade tem como objetivo adaptar o processo às características específicas do projeto e é, portanto, de grande importância.

O fato de que processos de software devem ser criados ou adaptados para os ambientes específicos nos quais serão aplicados, de forma a serem aceitos e terem seu uso maximizado, é conhecido e pesquisado ha bastante tempo. Basili (1988) apresenta uma metodologia e a ferramenta que a suporta denominada TAME (*Tailoring A Measurement Environment*) que objetiva a melhoria do processo de software, por meio de sua adaptação para as características de um ambiente e projeto específicos.

Mais recentemente, métodos mais sofisticados para a adaptação de processos de software têm sido desenvolvidos. Por exemplo, os que usam conhecimento no processo de adaptação utilizando técnicas de Inteligência Artificial para representar, armazenar, recuperar e utilizar corretamente esse conhecimento. Benger (2003) define um método de instanciação de processos de software apoiado pela ferramenta AdaptPro, através da qual é disponibilizado ao gerente de projetos (pessoa responsável por gerenciar o andamento de determinado projeto de software) o conhecimento sobre instanciação de processos de software acumulado pela organização de software em projetos anteriores. Esta abordagem fundamenta-se nos conceitos de gerência de conhecimento e de ambientes de desenvolvimento de software orientados à organização.

Rupprecht et al. (2000), apresentam uma abordagem fundamentada em um kit de construção de processos de software, que consiste de um repositório para o gerenciamento de

blocos básicos para construção de processos de software e um vetor de operadores para adaptar tais blocos. Este kit foca no projeto conceitual de processos de software em termos de ontologias, enquanto ao mesmo tempo oferece suporte automatizado para adaptação de processos de software.

Henninger (2002) apresenta uma abordagem que utiliza um sistema baseado em regras, organizadas sob a forma de árvores de decisão, para adaptar processos de software para as necessidades específicas de projetos individuais e usa técnicas de aprendizagem organizacional para modificar processos de software, de forma a atender às necessidades da organização. Esta abordagem é suportada pela ferramenta GUIDE, uma aplicação Web que utiliza o repositório baseado em casos BORE para capturar experiências em projetos anteriores. Esta abordagem utiliza a combinação de inferência baseada em conhecimento com raciocínio baseado em casos para adaptar processos de software, em virtude dos diferentes tipos de conhecimento envolvidos na adaptação.

Seguindo a tendência atual da área de utilização de processos padrões na definição de processos especializados, Machado (2002), a ferramenta DefPro, construída para auxiliar na definição de um processo padrão, de acordo com a norma ISO/IEC 12207, com modelos de maturidade tais como CMMI e ISO/IEC 15504 e com as características da organização de software. Coelho, por sua vez, em (COELHO, 2002), apresenta o MASP (Modelo de Adaptação de Processos de Software), o qual tem por objetivo adaptar o processo padrão de uma organização para os projetos conduzidos na mesma, baseado nas características desses projetos e em adaptações anteriores, utilizando uma abordagem orientada a artefatos. Soluções baseadas em políticas de instanciação, tal como descrito no modelo APSEE (COSTA, 2006), adotam uma postura diferente por fornecer solução automatizada para a escolha de agentes e recursos a partir de critérios genéricos definidos a priori.

As inúmeras pesquisas relacionadas à atividade de instanciação de processo de software como as apresentadas nos parágrafos anteriores demostram o reconhecimento no meio cientifico da importância da atividade de instanciação de processo e suas problemáticas. Em especial, devido ao fato de que essa atividade tende a se tornar cada vez mais complexa à medida que os processos de desenvolvimento de software cresçam e se tornem mais maduros. A questão de maturidade de software (e seu impacto nessa atividade) passa a ser um fator importante também a ser analisado. De acordo com o *Software Engineering Institute* (SEI) (SEI, 2012), a melhoria de processo de software pode ser entendida como um programa de atividades definidas com o objetivo de melhorar o desempenho e a maturidade do processo

organizacional, bem como os resultados de tal programa. Na próxima subseção, a questão de melhoria e maturidade de processo de desenvolvimento de software será melhor detalhada.

2.1. Maturidade do Processo de Desenvolvimento de Software

Apesar das possíveis diferenças que processos de desenvolvimento de software possam ter, eles se baseiam em uma estrutura similar de boas práticas que foram ao longo do tempo refinadas e comprovadamente consideradas eficientes (FALBO et al, 2004). Com base nesse fato, especialistas na área da engenharia de software foram capazes de estabelecer parâmetros para medir a maturidade de um processo de desenvolvimento de software específico. Portanto a maturidade de um processo de software pode ser definido como a aderência que o processo em questão possui com o conjunto de boas práticas previamente estabelecido.

Para servirem como base para a avaliação de maturidade em questão, organizações especializadas na área acabaram por estabelecer modelos que apresentassem um conjunto de boas práticas, alinhadas com produtos de trabalho típicos que comprovem a sua execução, assim como também metodologias de avaliações. Internacionalmente, destacasse como um destes modelos o CMMI (CHRISSIS et al, 1997) e nacionalmente temos como destaque o MR.MPS-SW (MONTONI et al, 2009).

Ambos os modelos citados acima estabelecem níveis de maturidade equivalentes, Tabela 2, que são relacionados ao atendimento de certo subconjunto de práticas ou resultados apresentados. O atendimento às exigências de cada nível de maturidade é cumulativo, de forma que, para que uma organização seja considerada nível A de maturidade na visão do MR.MPS-SW, por exemplo, é necessário que ela atenda de alguma forma todos os resultados esperados dos processos associados a cada nível anterior mais os resultados esperados associados ao nível A.

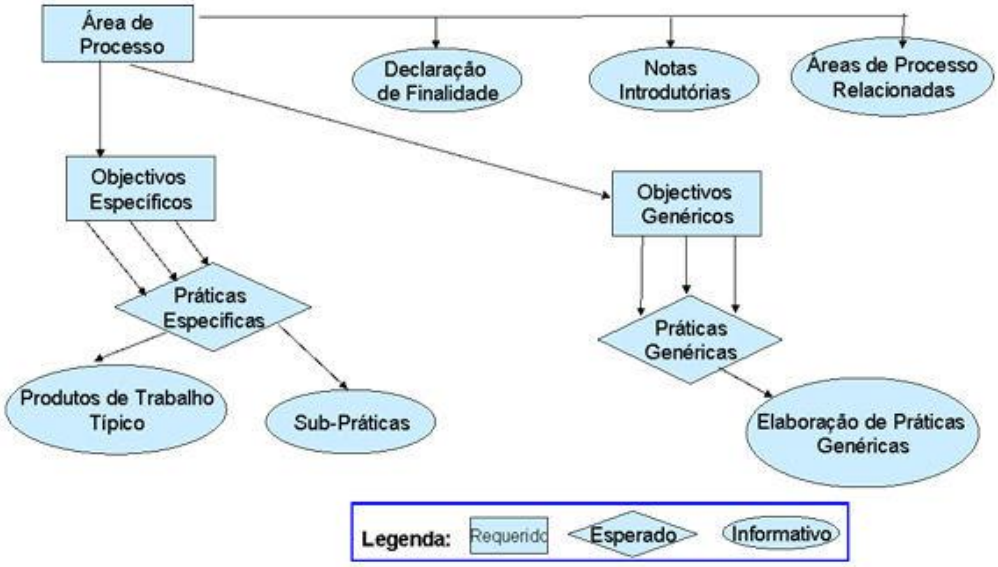
Quadro 1 - Níveis de Maturidade de Processo

CMMI	MPS.Br	Descrição
1	Não é definido	Executado: Processo de software caracterizado como "ad hoc". Poucos processos de desenvolvimento definido e o sucesso dependem de esforço individual.
2	G	Gerenciado: As políticas de gerencia de desenvolvimento de software são definidas e seguidas. A execução do projeto é planejada e a execução é confrontada com o plano.
	F	
3	E	Definido: O processo básico de software para as atividades de gestão e engenharia é documentado, padronizado e integrado em um processo de software padrão para a organização.
	D	
	C	
4	B	Gerenciado quantitativamente: Medidas detalhadas do processo de software e da qualidade do produto são realizadas. O processo e os produtos de software e da qualidade do produto são quantitativamente compreendidos e controlados.
5	A	Em Otimização: A melhoria contínua do processo é proporcionada pelo *feedback* quantitativo do processo e pelas ideias e tecnologias inovadoras.

Fonte: Elaborado pelo Autor

No modelo CMMI as disciplinas de processo são chamadas de áreas de processo. constituindo os seus principais componentes do CMMI. Como definido em (CHRISSIS et al, 1997), os outros elementos do modelo CMMI são: definição de propósito, notas introdutórias, áreas de processo relacionadas, objetivos específicos, objetivos genéricos, práticas específicas, produtos de trabalho típicos, sub-práticas, práticas genéricas e detalhamento de práticas genéricas. Estes componentes, para efeito de avaliação de maturidade, são classificados entre requeridos, esperados e informativos. Os componentes requeridos são o foco das avaliações de maturidade. A figura 6 provê uma forma de visualização da estrutura do modelo, bem como da classificação dos seus componentes.

Figura 6 - Estrutura do Modelo CMMI



Fonte: Elaborado pelo Autor

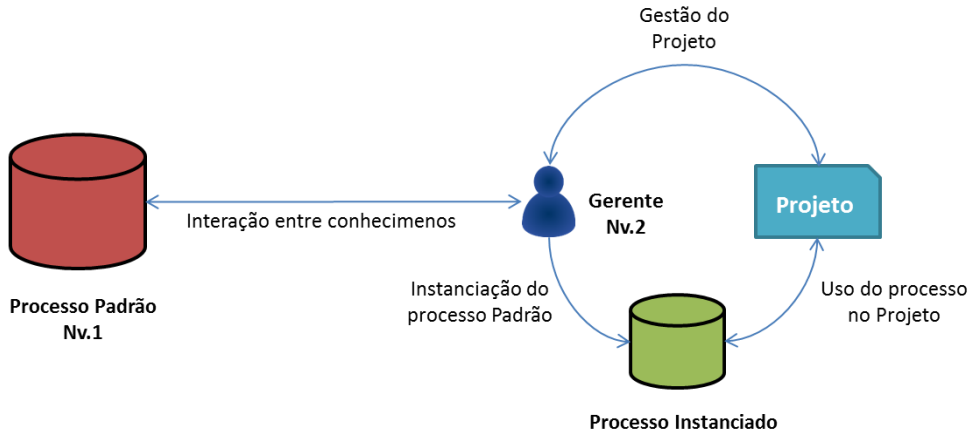
Uma prática específica é a descrição de uma atividade que é considerada importante para o atendimento de um objetivo específico associado e, por isso, são elementos dinâmicos. Uma vez refletidos em um processo, as práticas são de suma importância para que o projeto atinja os objetivos de qualidade/maturidade configurados no processo. É basicamente nas práticas específicas e em seus elementos filhos (produtos de trabalho típico e sub-práticas) que a gestão de conhecimento possui mais força, uma vez que são esses elementos que, uma vez refletidos no processo, vão estabelecer a interface direta com o usuário.

Um processo aderente ao nível 5 do CMMI, por exemplo, deve na realidade estar aderente a centenas de práticas específicas apresentadas pelo modelo, que estão divididas entre as 22 áreas de processo. É coerente concluir que um processo com tal maturidade apresente, portanto, um conjunto grande de práticas específicas que permitam a cobertura completa destas áreas. A instanciação de um processo com tal maturidade passa a ser uma tarefa complexa devido à grande gama de decisões que precisam ser tomadas e, ao mesmo tempo, vital para que a maturidade daquele processo se reflita nos resultados do projeto. O grupo de gestão do projeto deve conduzir essa instanciação de forma criteriosa, levando em consideração múltiplos critérios envolvidos (e muitas vezes conflitantes) para o sucesso do projeto, tais como qualidade, tempo e orçamento.

O aumento de informação presente no processo acaba por dificultar sua interação com seus usuários. A interação de conhecimento agregado do processo e dos membros envolvidos na instanciação é outro fator importante que deve ser levado em consideração para o sucesso

da atividade, embora claramente seja algo desafiador devido à complexidade do processo. Essa interação está representada na Figura 7.

Figura 7 - Interação entre conhecimento durante a instanciação do processo padrão



Fonte: Elaborado pelo Autor

Dada a importância que a gestão de conhecimento pode ter dentro do contexto de instanciação de conhecimento, na próxima subseção a questão da gestão de conhecimento será melhor detalhada.

2.2. Gestão do Conhecimento Organizacional

De acordo com Gottschalk (2005) a gestão do conhecimento está relacionada com a descrição, organização, compartilhamento e desenvolvimento de conhecimento em uma organização. Para o autor, gerenciar o conhecimento é gerenciar atividades de uso intensivo do conhecimento. Trata-se, basicamente, da identificação e uniformização do conhecimento coletivo, de forma a auxiliar a organização a competir em seu mercado. Para tanto, o autor considera que a gestão de conhecimento é um método para atendimento aos objetivos corporativos a partir da coleta, criação, síntese e compartilhamento de informações, pensamentos e experiências. Montana (2001) define a gestão do conhecimento como uma disciplina que tem por foco a geração, aquisição, compartilhamento, proteção, distribuição e utilização do conhecimento, capital intelectual e ativos intangíveis.

Segundo Alavi (2001), o conhecimento organizacional diz respeito a: conhecimentos sobre clientes e consumidores, produtos, processos e competidores, que podem incluir melhores práticas, *know how* e regras heurísticas, padrões, códigos de software, processos de

negócio e modelos; arquiteturas, tecnologias e estruturas de negócio; experiências de projetos (propostas, planos de trabalho e relatórios); e ferramentas utilizadas para implementar um processo, tais como listas de verificação e levantamentos.

De modo geral, os ativos de conhecimento de uma organização constituem elementos produzidos na execução dos processos organizacionais. Esses elementos podem ser tangíveis, como por exemplo, qualquer artefato que seja produto de trabalho intelectual (relatórios, cronogramas, planilhas, código de software, etc.) ou podem ser intangíveis, como o conhecimento tácito, habilidades e experiências adquiridas pelo indivíduo ou grupo de indivíduos envolvidos no trabalho intelectual de produção e criação de tais artefatos.

Segundo Nonaka (1996), o conhecimento organizacional é criado por meio de interações humanas entre indivíduos e entre diferentes tipos de conhecimento (tácito ou explícito). Tal processo social e epistêmico de criação do conhecimento organizacional compreende o que os autores denominam de modos de conversão do conhecimento: socialização (conversão do conhecimento individual tácito para conhecimento tácito coletivo), externalização (conversão do conhecimento tácito para conhecimento explícito), combinação (conversão do conhecimento explícitos separados em conhecimento explícito sistêmico) e internalização (conversão de conhecimento explícito para conhecimento tácito) (Figura 8). Ainda é afirmado que a criação do conhecimento ocorre por meio da espiral de conhecimento (Figura 9).

Figura 8 - Modos de conversão de conhecimento

Fonte: Elaborado pelo Autor

Figura 9 - Espiral de Conhecimento

Fonte: Nonaka (1996)

Outra forma de entender a natureza da gestão do conhecimento organizacional e como ela ocorre em organizações é a partir do exame do que se denomina na literatura de "facilitadores do conhecimento" (*knowledge enablers*). Do ponto de vista de Lee (2003) os facilitadores do conhecimento podem ser entendidos como mecanismos organizacionais que facilitam a ocorrência do conhecimento e estimulam a sua criação, compartilhamento e utilização em organizações. Além disso, para melhor entender a gestão do conhecimento também devem ser examinados os principais motivos de falhas e causas de insucesso de iniciativas de gestão do conhecimento organizacional.

Ao medir os fatores críticos de sucesso para a gestão do conhecimento em pequenas e médias empresas, Wong (2005) verificou como os principais facilitadores ou inibidores da gestão do conhecimento organizacional os seguintes: liderança e suporte gerencial, cultura, tecnologia da informação, estratégia e propósito da organização, infraestrutura organizacional, medição do processo de implantação da gestão do conhecimento, processos e atividades de gestão de conhecimento, auxílio motivacionais (sistema de incentivos), recursos (principalmente financeiros), educação e treinamento e gestão de recursos humanos.

Em geral, a pesquisa sobre a gestão do conhecimento na área de engenharia de software se concentra na aplicação dessa gestão para potencializar a melhoria contínua do processo de engenharia de software, como se pode verificar, por exemplo, em (FALBO et al. 2004) (CORBIN et al. 2007) e (KUKKO et al. 2008), através do desenvolvimento/melhoria dos *knowledge enablers*. Este trabalho, porém, não objetiva exatamente a melhoria contínua

do processo de desenvolvimento de software e sim a melhoria de sua utilização. A melhoria do processo de desenvolvimento com a utilização do método aqui proposto pode acontecer como uma consequência de seu melhor entendimento e aplicação.

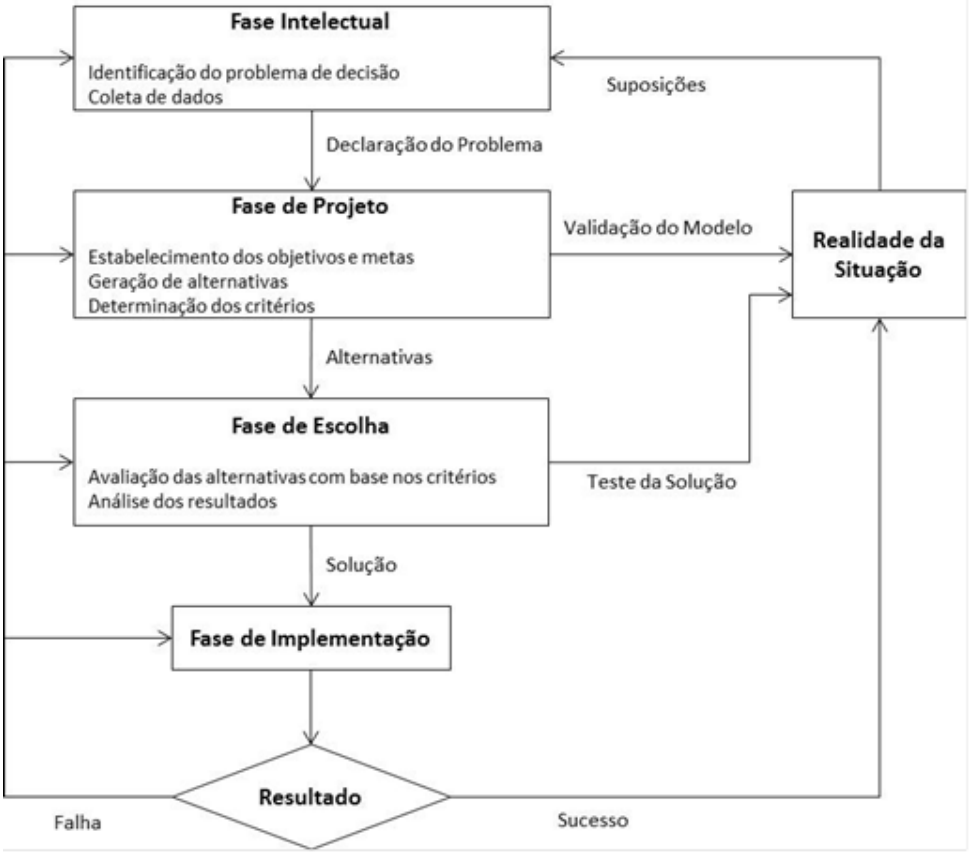
A gestão de conhecimento pode ser uma ferramenta extremamente útil quando se lida com situações de tomadas de decisão. O conhecimento de decisões tomadas no passado e suas consequências é uma informação valiosa para novas tomadas de decisão. Na próxima seção será apresentado com maior detalhe como é o processo de tomada de decisão e os fatores envolvidos no processo.

3. TOMADA DE DECISÃO MULTICRITÉRIO

A tomada de decisão é um processo cognitivo associado à avaliação, comparação, seleção, ordenação e/ou priorização de um curso de ações sobre um conjunto de alternativas. Todo processo dessa natureza produz uma escolha final, comumente denominada solução. Em geral, o processo começa quando é necessário encontrar uma solução, mas não existe a certeza de que ela será aceita por todas as pessoas envolvidas no processo decisório. O processo pode também ser visto como uma atividade que envolve raciocínio, que pode ser formal ou intuitivo e pode ser baseado em suposições explícitas ou tácitas.

A Figura 10 apresenta um esquema de processo de tomada de decisão proposto por (LU et. al, 2007):

Figura 10 - Esquema de processo de tomada de decisão



Fonte: LU et. al (2007)

O processo de tomada de decisão começa com a Fase Intelectual, na qual a realidade é analisada, o problema é identificado e declarado. Na Fase de Projeto, um modelo que

representa o sistema é construído, a partir de considerações que simplificam a realidade e descrevem as relações entre todas as variáveis. O modelo é então validado e os critérios são definidos para a avaliação das alternativas de solução do problema. A fase de Escolha inclui a seleção de uma das possíveis soluções propostas para o modelo. Esta solução é testada a fim de determinas a sua viabilidade. Se a solução escolhida for razoável, segue-se à ultima fase, de Implementação. Uma implementação bem sucedida resulta na solução do problema real. Uma falha provoca o retorno a uma das fases anteriores.

O problema de tomada de decisão é extremamente comum dentro do contexto da engenharia de software, embora raramente seja reconhecido formalmente. Decisões importantes são tomadas desde a realização de uma proposta de um projeto a um cliente até á conclusão do projeto. Avaliar o custo de um projeto, definir a equipe, escolher a metodologia de desenvolvimento, aceitar ou não um extra escopo e alterar ou não um cronograma são exemplos dessas decisões.

3.1. Modelos de Tomada de Decisão

Tomada de Decisão Multicritério (*Multiple-Criteria Decision Making* (MCDM)) refere-se à tomada de decisão na presença de múltiplos e conflitantes critérios. Situações dessa natureza acontecem diariamente e compartilham as seguintes características (CHEN, 1992):

a) Possuir múltiplos critérios, que podem ser objetivos ou atributos que qualificam os itens julgados;
b) Apresentar conflitos entre os critérios;
c) Apresentar diferentes unidades de medida para os critérios.

Com base na primeira característica, é possível classificar os problemas MCDM em duas categorias (LU et al, 2007):

a) Tomada de Decisão Multiobjetivo, *Multi-Objective Decision Making* (MODM) – são problemas que trabalham com espaços contínuos de decisão e estão associados, principalmente, à programação matemática com múltiplas funções objetivo (funções que permitem diferenciar alternativas).
b) Tomada de Decisão Multiatributo, *Multi-Atribute Decision Making* (MADM) – são problemas que trabalham com espaços discretos de decisão.

Os problemas MADM possuem um conjunto de alternativas bem definido e conhecido. Como exemplo desse tipo de problema, pode-se citar a escolha de membros que irão compor uma equipe de projeto de desenvolvimento de software dado seus atributos.

Diferentemente dos problemas MADM, os MODM possuem um conjunto de alternativas infinitas e normalmente definidas por uma função matemática. Uma vez que uma possível alternativa seja identificada, é estabelecido o julgamento de quão perto essa alternativa satisfaz os objetivos envolvidos na decisão. Como exemplo desse tipo de problema, pode-se imaginar uma situação onde se deseja aumentar a qualidade do desenvolvimento de um software que está diretamente ligado a variáveis como tempo, custo entre outros. Dependendo dos valores dessas variáveis, chega-se a uma alternativa diferente, e então essa alternativa pode ser julgada.

Para ampliar o conhecimento sobre MODM e MADM, alguns conceitos e terminologias definidos em são apresentados abaixo (CHEN et al, 1992) (PEDRYCZ et al, 2011) :

a) **Decisor** – também conhecido como Sujeito de Decisão, Agente de Decisão ou Tomador de Decisão, é a pessoa ou grupo de pessoas que, direta ou indiretamente, proporciona o juízo de valor que poderá ser usado no momento de avaliar as alternativas disponíveis, com o objetivo de identificar a melhor escolha.
b) **Critérios** – são padrões ou conjuntos de regras que permitem avaliar as alternativas mediante objetivos ou atributos. Ex.:qualidade, tempo, custo.
c) **Objetivos** – são reflexos de desejo dos decisores e indicam a direção na qual eles querem trabalhar. Ex.: Aumentar a lucratividade.
d) **Moderador** – também denominado Analista, é a pessoa ou conjunto de pessoas (grupo moderador) encarregado de modelar o problema e, eventualmente, fazer as recomendações e orientações relativas à seleção final.
e) **Conjunto de Alternativas** – também chamado de Conjunto de Escolha, é um conjunto finito, do ponto de vista prático, constituído por um numero relativamente pequeno de elementos que permite alcançar os objetivos de uma operação. As alternativas devem ser diferentes, exaustivas (a inclusão de novas alternativas implica na reformulação do modelo) e excludentes (não são permitidas soluções mistas). Ex.: Atividades de processo, ferramentas de desenvolvimento, etc..
f) **Coeficientes de Importância** – são pesos ou importâncias atribuídas aos especialistas e a critérios, a fim de diferenciá-los.

3.2. Tomada de Decisão Multicritério em Grupo

O problema de decisão multicritério em grupo possui os mesmos objetivos do problema de tomada de decisão multicritério padrão, porém implica na necessidade de lidar com opiniões coletivas. Normalmente o procedimento para sua resolução implica em ordenar, por ordem de importância, as alternativas de um conjunto X, de acordo com critérios de um conjunto C, e considerando a opinião coletiva de uma equipe E de especialistas (PEDRYCZ et al, 2011). Trata-se de um problema comum em diversas fases do processo de desenvolvimento de software, em especial, durante a fase de planejamento do projeto, na qual várias decisões relacionadas à gestão e arquitetura são tomadas.

O processo de decisão multicritério em grupo exige que, de certa forma, a opinião de cada especialista seja agregada criando-se uma opinião coletiva. Encontra-se na literatura várias metodologias para resolver este problema que vão desde simples reuniões até a modelagem matemática. A decisão de qual técnica utilizar está diretamente ligada à complexidade do problema (número de elementos envolvidos) e à sua importância, sendo que será mais indicado o uso de técnicas que se baseiam em modelagens matemáticas quanto maior forem essas duas variáveis. (LU et al, 2007)

O problema de instanciação do processo organizacional para um projeto deve ser enxergado como um problema complexo de decisão multicritério em grupo, devido ao grande número de variáveis e fatores envolvidos e de grande importância para resultados do projeto, como já discutido durante a introdução deste trabalho. Portanto, a metodologia apresentada aqui para a realização desta atividade terá como base a modelagem e o desenvolvimento matemático do problema.

O problema de decisão multicritério em grupo pode ser inicialmente modelado matematicamente por meio dos seguintes conjuntos:

a) $X = \{X_1, X_2, ..., X_n\}$ é um conjunto finito e discreto de itens candidatos que se deseja avaliar.
b) $C = \{C_1, C_2, \ldots, C_m\}$ é um conjunto finito e discreto de critérios envolvidos na tomada de decisão. Estes critérios podem ser tanto de natureza quantitativa quanto qualitativa (Por exemplo, qualidade de software, custo, tempo e etc.).
c) $P = \{P_1, P_2, \ldots, P_m\}$ é um conjunto finito e discreto de pesos que são associados aos critérios, uma vez que critérios podem ter relevâncias diferentes dependendo do contexto.

d) $E = \{E_1, E_2, \ldots, E_q\}$ é um conjunto finito e discreto de especialistas envolvidos no processo de decisão.
e) $W = \{W_1, W_2, \ldots, W_q\}$ é um conjunto finito e discreto de pesos que são associados à opinião de cada especialista envolvido no processo, uma vez que a opinião de um especialista com maior experiência no contexto pode ser mais valorizado do que um com menos experiência.

Dessa forma, todos os especialistas pertencentes ao conjunto E podem ser associados aos pesos pertencentes ao conjunto W e devem analisar cada item candidato do conjunto X frente a cada critério do conjunto C. Os critérios do conjunto C podem ainda ser associados a pesos do conjunto P.

3.3. Considerações Finais

É claramente possível considerar a atividade de instanciação do processo de desenvolvimento de software como um problema de decisão com múltiplos critérios em grupo. O conjunto de especialistas pode ser representado pelo grupo que compõe a gestão do projeto. Os itens para serem julgados são as atividades, que compõem o processo organizacional da empresa. E, como exemplo de critérios envolvidos neste contexto, pode-se citar a qualidade do produto a ser desenvolvido, os riscos envolvidos com o projeto ou mesmo as características envolvidas com o orçamento ligado a aquele projeto.

Contextualizando o esquema apresentado por (LU et al, 2007) ilustrado na Figura 10 no problema de tomada de decisão durante a instanciação de processo de software trata-se de um caso de MADM, uma vez que o conjunto de alternativas é discreto, não finito e bem definido, no qual percebe-se que as ações de Validação do Modelo Construído e do Teste da Solução antes de sua implementação são difíceis de serem realizadas dado às características da atividade. Como são exatamente essas as ações que possuem como objetivo garantir a qualidade da tomada de decisão, é possível inferir aqui a fragilidade que este processo pode possuir com relação a erros. Isso demonstra a importância de um maior formalismo na execução da atividade e evidencia a necessidade de uso de outras técnicas de suporte à tomada de decisão. Com essa finalidade, o próximo capítulo irá apresentar elementos da teoria dos conjuntos *fuzzy* e como essa teoria pode ser utilizada para suportar a tomada de decisão de modo a potencializar sua eficiência.

4. LÓGICA *FUZZY*

Seres humanos são capazes de lidar com processos bastante complexos, baseados em informações imprecisas ou aproximadas. A estratégia adotada pelos operadores humanos é também de natureza imprecisa e geralmente possível de ser expressa em termos linguísticos. A teoria de conjuntos *fuzzy* e os conceitos de lógica *fuzzy* podem ser utilizados para traduzir, em termos matemáticos, a informação imprecisa expressa por um conjunto de regras linguísticas.

A teoria de conjuntos *fuzzy* foi concebida por ZADEH (ZADEH, 1996) com o objetivo de fornecer um ferramental matemático para tratamento de caráter impreciso ou vago. A Lógica f*uzzy*, baseada nessa teoria, foi inicialmente construída a partir dos conceitos já estabelecidos de lógica clássica; operadores foram definidos à semelhança dos tradicionalmente utilizados e outros foram introduzidos ao longo do tempo, muitas vezes por necessidades de caráter eminentemente prático.

Nas subseções seguintes serão apresentados os conceitos fundamentais de conjuntos *fuzzy* e de lógica *fuzzy*, assim como algumas definições e operadores que permitem abordar os mecanismos de inferência.

4.1. Conjuntos *Fuzzy*

Na teoria clássica dos conjuntos, o conceito de pertinência de um elemento a um conjunto fica bem definido. Dado um conjunto A em um universo X, os elementos deste universo simplesmente pertencem ou não pertencem àquele conjunto (PEDRYCZ et al, 2011). Isso pode ser expresso pela função característica $f_A(x)$ = { 1 se e somente se $x \in A$ / 0 se e somente se $x \notin A$}.

O fundador da lógica *fuzzy*, ZADEH (1996), propôs uma característica mais ampla, generalizando a função característica, de modo que ela pudesse assumir um número infinito de valores no intervalo [0,1]. Um conjunto *fuzzy* A em um universo X é definido por uma função de pertinência $\mu_A(x):X \rightarrow [0,1]$ e representado por um conjunto de pares ordenados $A = \{\mu_A(x)/x\}$, $x \in X$, onde $\mu_A(x)$ indica o quanto x é pertinente ao conjunto A. Um determinado elemento de X pode pertencer a mais de um conjunto *fuzzy*, com diferentes graus de pertinência.

4.2. Variáveis Linguísticas

Durante o processo de decisão em grupo, diversos tipos de incerteza podem surgir, principalmente durante a construção da opinião coletiva, uma vez que pode-se encontrar opiniões completamente opostas sobre um mesmo item. A forma como essa incerteza é expressa durante o processo é um importante fator para o sucesso da aplicação da lógica *fuzzy*. Devido à importância que a expressão de incertezas assume durante o processo de decisão em grupo, o uso de termos linguísticos tem sido considerado uma forma realística e eficiente de tratar este problema, possibilitando a expressão dessas incertezas de forma prática, simples e realística (EKEL et al, 2008).

Um termo linguístico (ou estimativa linguística), neste contexto, nada mais é do que um conjunto de representações literárias pré-determinadas utilizadas para expressar o julgamento de um determinado item pelo especialista. Podem ser citados, a título de exemplificação, os termos “grande”, “pequeno”. “médio”, “curto”, “longo”, “importante”, etc..

Formalmente, um termo linguístico é caracterizado por uma quíntupla (N, T(N), X, G, M) (EKEL et al, 2008), onde:

a) N: nome da variável;
b) T(N): conjunto de termos de N, ou seja, o conjunto de nomes dos valores linguísticos de N;
c) X: universo de discurso;
d) G: regra sintática para gerar os valores de N como uma composição de termos de T(N), conectivos lógicos, modificadores e delimitadores;
e) M: regra semântica, para associar a cada valor gerado por g um conjunto *fuzzy* em X.

Uma exemplificação do uso de termos linguísticos será apresentada na seção seguinte deste trabalho, onde estes serão empregados.

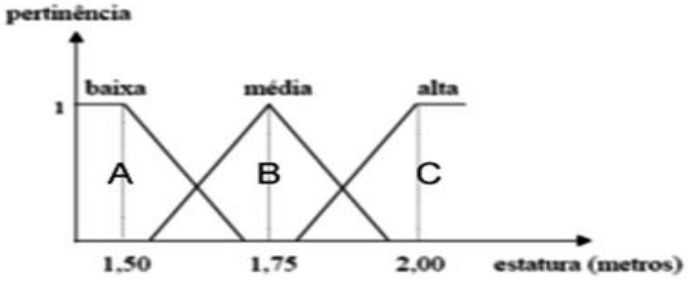
4.3. Funções de Pertinência

Como já mencionado, na teoria clássica, um dado elemento do universo em discurso (domínio) pertence ou não pertence ao referido conjunto. Porém, na teoria dos conjuntos *fuzzy* existe um grau de pertinência de cada elemento a um determinado conjunto.

Por exemplo, considerando os conjuntos das pessoas com alta renda e conjunto das pessoas altas, podemos verificar que não existe uma fronteira bem definida para decidirmos quando um elemento pertence ou não ao respectivo conjunto. É possível dizer que uma pessoa é alta ou que possui renda alta, porém existe certo grau de veracidade ou pertinência nisso dependendo da pessoa. Por exemplo, uma pessoa de 2 metros pode ser classificada como alta, porém uma pessoa de 2,3 metros possuí uma pertinência maior ao conjunto de pessoas altas do que a anterior. Com os conjuntos *fuzzy* podemos definir critérios e graus de pertinência para tais situações através do que é chamado de função de pertinência.

As funções de pertinência podem ter diferentes formas, dependendo do conceito que se deseja representar e do contexto em que serão utilizadas (GOMIDE, 1995). Para exemplificar o quanto o contexto é relevante nas funções de pertinência e de sua distribuição ao longo de um dado universo, considere-se a variável linguística estatura (de pessoas), constituída dos seguintes termos: T(estatura) = {baixo, média, alta}. A esses faz-se corresponder conjuntos *fuzzy* A, B e C, respectivamente, definidos por suas funções de pertinência. Uma escolha possível de funções de pertinência é apresentada na Figura 11.

Figura 11 - Exemplo de Função de Pertinência



Fonte: Elaborado pelo Autor

Na definição acima, estaturas de até 1,5 metros apresentam grau de pertinência igual a 1 no conjunto A, até a altura de 1,5 metros; a partir daí, o grau de pertinência decresce até zero, à medida que a estatura aumenta. Considera-se que a estatura de 1,75 metros é

“totalmente compatível” com o conjunto B, ao passo que a estatura 1,9 (aproximadamente) é mais compatível com estatura alta, porém não com uma pertinência de 100%.

Funções de pertinência podem ser definidas a partir da experiência e da perspectiva do usuário, mas é comum fazer-se uso de funções de pertinência padrão, como, por exemplo, as de forma triangular, trapezoidal e Gaussiana. Em aplicações práticas as formas escolhidas inicialmente podem sofrer ajustes em função dos resultados observados (GOMIDE, 1995).

4.4. Relações *Fuzzy*

Uma relação exprime a presença ou a ausência de uma associação (ou interação) entre elementos de dois ou mais conjuntos (PEDRYCZ et al, 2011). Formalmente, dado dois universos X e Y, a relação R definida em X x Y é um subconjunto do produto cartesiano do dois universos, de tal forma que R: X x Y $\rightarrow \{0,1\}$. Ou seja, se algum x $\in$ X e y $\in$ Y estiverem relacionados, R(x, y) = 1; caso contrário, R(x, y) = 0. Isto pode ser expresso pela seguinte função característica (ou função de pertinência bivalente):

$$f(x,y) = \begin{cases} 1 \text{ se e somente se } (x,y) \in R \\ 0 \text{ em caso contrário} \end{cases} \quad (1)$$

As relações podem ser expressas de forma analítica (para universos infinitos, por exemplo), ou de forma tabular, muito utilizada no caso de universos finitos (discretos). Esta última forma recebe o nome de matriz de relações, cujos elementos são ou zero ou um.

Relações *fuzzy* generalizam o conceito de relações e representam o grau da associação entre elementos de dois ou mais conjuntos *fuzzy*. Exemplos de caráter linguístico seriam: x é muito maior do que y, x está próximo de y, etc. Formalmente, dados dois universos X e Y, a relação *fuzzy* R é um conjunto *fuzzy* em X x Y, caracterizada por uma função de pertinência $\mu_R(x, y) \in [0,1]$, onde x $\in$ X e y $\in$ Y (PEDRYCZ et al, 2011).

Em um contexto de relação de preferência dentro do modelo apresentado em (PEDRYCZ et al, 2011), por exemplo, a pertinência $\mu_R(X_i,X_j) = 1$ indica indiferença entre a preferência do item X_i sobre o item X_j. De forma semelhante, $\mu R (Xi,Xj) = 0$ indica a máxima preferência entre o item Xi sobre o item Xj.

4.5. Função de Pertinência de Relações de Preferência *Fuzzy*

As estimativas *fuzzy* das alternativas para o estabelecimento das relações *fuzzy* podem ser utilizando as funções correspondentes $\mu[f_{C_k}(X_i)]$ k = 1,...,m, $X_i \in X$. A disponibilidade das estimativas $\mu[f_{C_k}(X_i)]$ k = 1,...,m e $X_i \in X$ possibilita a construção de relações de grau de preferência que podem ser descritas da seguinte forma como demonstrado em (PEDRYCZ et al, 2011):

$$\eta\{\mu[f_{c_k}(X_i)], \mu[f_{c_k}(X_j)]\} = \mu_{R_{c_k}}(X_i, X_j) = \sup_{X_i, X_j \in X f_{C_k}(X_i) \geq f_{C_k}(X_j)} min\{\mu[f_{c_k(X_i)}], \mu[f_{c_k(X_j)}]\}; \quad (2)$$

$$\eta\{\mu[f_{c_k}(X_j)], \mu[f_{c_k}(X_i)]\} = \mu_{R_{c_k}}(X_j, X_i) = \sup_{X_j, X_i \in X f_{C_k}(X_j) \geq f_{C_k}(X_i)} min\{\mu[f_{c_k(X_j)}], \mu[f_{c_k(X_i)}]\}. \quad (3)$$

Nas expressões acima, $\eta\{\mu[f_{c_k}(X_i)], \mu[f_{c_k}(X_j)]\}$ e $\eta\{\mu[f_{c_k}(X_j)], \mu[f_{c_k}(X_i)]\}$ correspondem aos graus de preferências $\mu[f_{C_k}(X_i)] \succeq \mu[f_{C_k}(X_j)]$ e $\mu[f_{C_k}(X_j)] \succeq \mu[f_{C_k}(X_i)]$ onde $\succeq$ significa que uma alternativa não é pior que a outra.

Através de (2) e (3), é possível avaliar os níveis das preferências das alternativas comparadas e construir as matrizes das relações de preferência *fuzzy*.

É importante ressaltar aqui que as expressões apresentadas acima não representam o único meio pelo qual podemos avaliar os níveis de preferência das alternativas. É possível construir expressões com operadores diferentes que possuem o mesmo objetivo, porém apresentam perspectivas diferentes de lidar com a informação. Um conjunto com maiores detalhes sobre essa questão é apresentado em (PEDRYCZ et al., 2011).

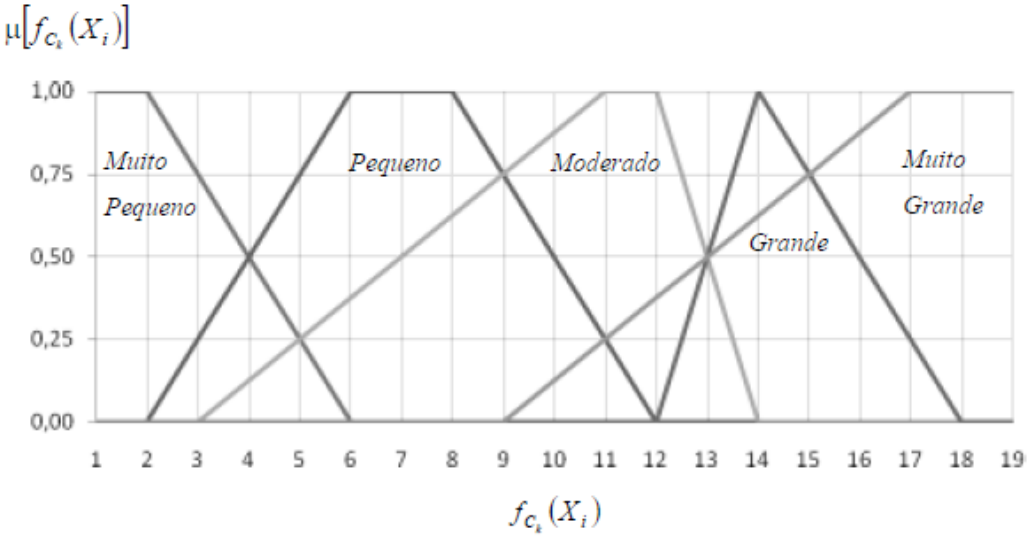
4.6. Aplicação de Relações de Preferência *Fuzzy* na Tomada de Decisão Multicritério

Esta seção tem como objetivo apresentar a forma como as relações de preferência *fuzzy* podem ser utilizadas para analisar os problemas de decisão multicritério. Isso será feito utilizando o problema apresentado abaixo.

Considerando um conjunto de alternativas X = $\{X_1,...,X_3\}$ avaliadas por pelo especialista E_1, do ponto de vista de maximização de dois critérios C = $\{C_1, C_2\}$. Considere ainda que o especialista irá expressar sua preferência utilizando o gráfico de função de

pertinência, apresentado na Figura 12, mapeado com as variáveis linguísticas “Muito Pequeno”, “Pequeno”, “Moderado”, “Grande” e “Muito Grande”.

Figura 12 - Função de Pertinência utilizada no exemplo



Fonte: Elaborado pelo Autor

Considerando ainda os resultados da avaliação do especialista E_1 sobre os itens perante os critérios C_1 e C_2 apresentados abaixo.

a) C_1: X_1 possui importância moderada, X_2 possui importância pequena e X3 possui importância muito pequena.
b) C_2: X_1 possui importância pequena, X_2 possui importância moderada e X_3 possui importância grande.

Com base nessas informações, é possível aplicar a técnica de modelagem de relações de preferência *fuzzy* para auxiliar o especialista E_1 na sua tomada de decisão. Na próxima subseção, tal aplicação será explicitada.

4.6.1. Aplicando a Técnica de Relações de Preferência Fuzzy

As opiniões do especialista E1 perante os itens julgados e contextualizados dentro dos critérios C_1 e C_2 podem ser escritas da seguinte forma: $f_{C1}(X_1)$ = Moderado; $f_{C1}(X_2)$ = Pequeno; $f_{C1}(X_3)$ = Muito Pequeno; $f_{C2}(X_1)$ = Pequeno; $f_{C2}(X_2)$ = Moderado; $f_{C2}(X_3)$ = Grande.

Levando em consideração a Figura 12 e o julgamento do Especialista E_1 apresentado acima, pode-se aplicar as expressões (2) e (3) para se obter a matriz de relações f*uzzy*. Para a aplicação de (2) e (3) é necessário construir o produto cartesiano utilizando o operador *min* entre cada conjunto de funções de pertinências *fuzzy* (eixo y do gráfico apresentado na figura 6) baseados nos termos linguísticos utilizados pelo especialista. Os produtos cartesianos obtidos para o critério C_1 são exemplificados nas figuras 13, 14 e 15 a seguir.

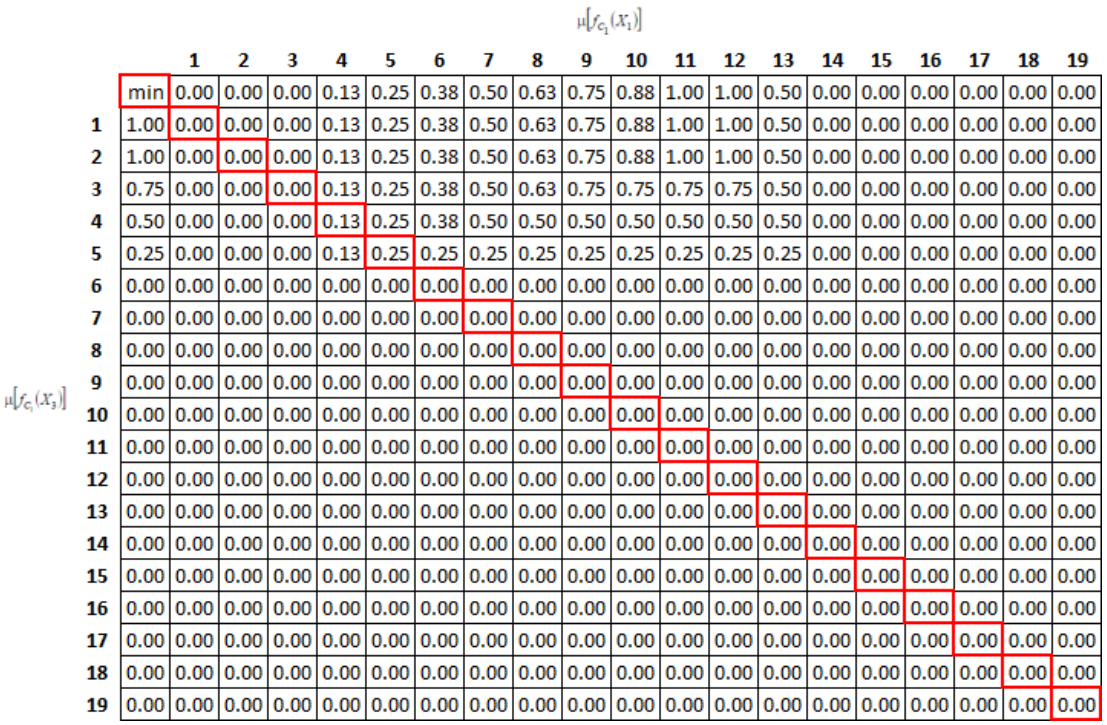
Figura 13 - Produto Cartesiano entre X1 e X2

$\mu[r_{C_1}(X_1)]$

$\mu[r_{C_1}(X_2)]$	min	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
3	0.25	0.00	0.00	0.00	0.13	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.00	0.00	0.00	0.00	0.00	0.00
4	0.50	0.00	0.00	0.00	0.13	0.25	0.38	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.00	0.00	0.00	0.00	0.00	0.00
5	0.75	0.00	0.00	0.00	0.13	0.25	0.38	0.50	0.63	0.75	0.75	0.75	0.75	0.50	0.00	0.00	0.00	0.00	0.00	0.00
6	1.00	0.00	0.00	0.00	0.13	0.25	0.38	0.50	0.63	0.75	0.88	1.00	1.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00
7	1.00	0.00	0.00	0.00	0.13	0.25	0.38	0.50	0.63	0.75	0.88	1.00	1.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00
8	1.00	0.00	0.00	0.00	0.13	0.25	0.38	0.50	0.63	0.75	0.88	1.00	1.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00
9	0.75	0.00	0.00	0.00	0.13	0.25	0.38	0.50	0.63	0.75	0.75	0.75	0.75	0.50	0.00	0.00	0.00	0.00	0.00	0.00
10	0.50	0.00	0.00	0.00	0.13	0.25	0.38	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.00	0.00	0.00	0.00	0.00	0.00
11	0.25	0.00	0.00	0.00	0.13	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.00	0.00	0.00	0.00	0.00	0.00
12	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
13	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
14	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
16	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
17	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
18	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
19	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Fonte: Elaborado pelo Autor

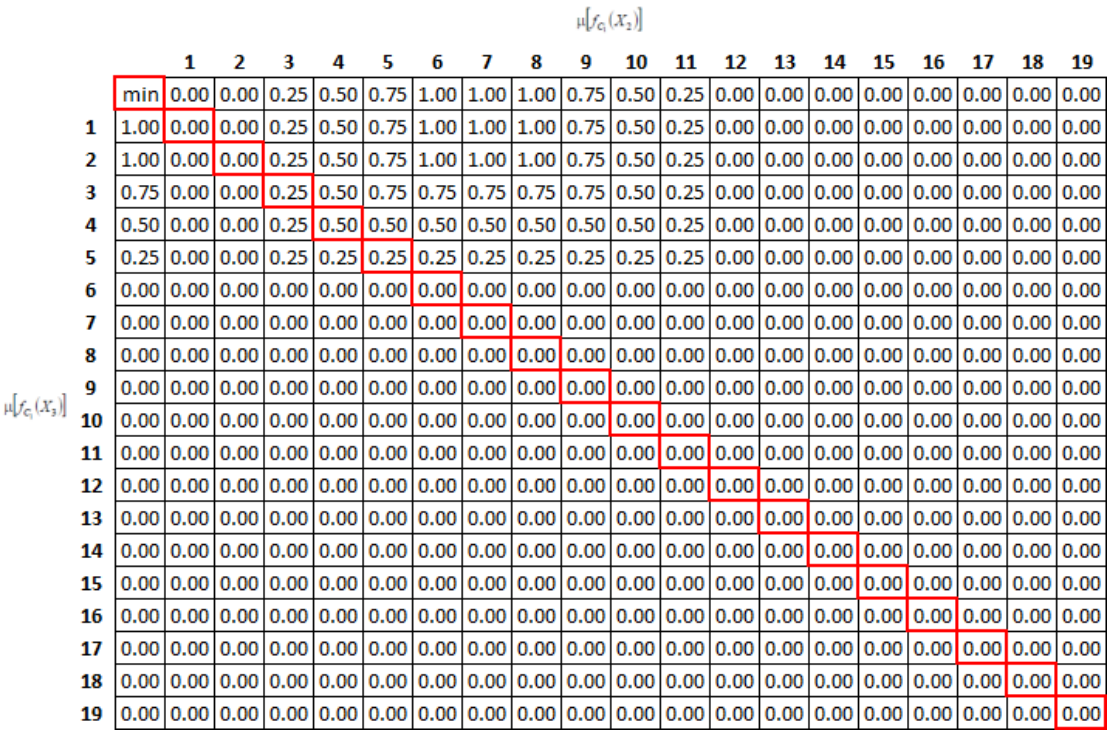
Figura 14 - Produto Cartesiano entre X1 e X3



$\mu[f_{C_1}(x_3)]$	$\mu[f_{C_1}(x_1)]$																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
	min	0.00	0.00	0.00	0.13	0.25	0.38	0.50	0.63	0.75	0.88	1.00	1.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00
1	1.00	0.00	0.00	0.00	0.13	0.25	0.38	0.50	0.63	0.75	0.88	1.00	1.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00
2	1.00	0.00	0.00	0.00	0.13	0.25	0.38	0.50	0.63	0.75	0.88	1.00	1.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00
3	0.75	0.00	0.00	0.00	0.13	0.25	0.38	0.50	0.63	0.75	0.75	0.75	0.75	0.50	0.00	0.00	0.00	0.00	0.00	0.00
4	0.50	0.00	0.00	0.00	0.13	0.25	0.38	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.00	0.00	0.00	0.00	0.00	0.00
5	0.25	0.00	0.00	0.00	0.13	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.00	0.00	0.00	0.00	0.00	0.00
6	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
7	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
9	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
12	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
13	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
14	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
16	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
17	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
18	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
19	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Fonte: Elaborado pelo Autor

Figura 15 - Produto Cartesiano entre X2 e X3



$\mu[f_{C_1}(x_3)]$	$\mu[f_{C_1}(x_2)]$																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
	min	0.00	0.00	0.25	0.50	0.75	1.00	1.00	1.00	0.75	0.50	0.25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
1	1.00	0.00	0.00	0.25	0.50	0.75	1.00	1.00	1.00	0.75	0.50	0.25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	1.00	0.00	0.00	0.25	0.50	0.75	1.00	1.00	1.00	0.75	0.50	0.25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
3	0.75	0.00	0.00	0.25	0.50	0.75	0.75	0.75	0.75	0.75	0.50	0.25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
4	0.50	0.00	0.00	0.25	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
5	0.25	0.00	0.00	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
6	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
7	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
8	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
9	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
12	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
13	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
14	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
16	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
17	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
18	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
19	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Fonte: Elaborado pelo Autor

Na matriz apresentada pela Figura 13, por exemplo, o operador *min* é executado entre as funções de pertinência das alternativas X_2 e X_3. Observa-se, por exemplo, na escala 5 do universo de discurso, que o item X_2 (classificado como Pequeno) possui uma função de pertinência a esse conjunto equivalente a 0.75. Já o item X_2 (classificado como Muito Pequeno) possui uma pertinência de 0.25 ao conjunto Muito Pequeno na mesma escala do universo de discurso. Portanto, o operador *min* entre os dois resulta no valor de 0.25, como mostrado na matriz.

Dado os produtos cartesianos mostrados acima, é possível concluir a aplicação de (2) e (3) obtendo-se, respectivamente, o supremo (função *sup*) da porção superior e inferior de cada produto cartesiano.

$$\mu_{R_{C_1}}(X_1,X_2)=\sup_{\substack{X_1,X_2\in X\\ f_{C_1}(X_1)\geq f_{C_1}(X_2)}}\min\left\{\mu\left[f_{C_1}(X_1)\right],\mu\left[f_{C_1}(X_2)\right]\right\}=1\ ; \tag{4}$$

$$\mu_{R_{C_1}}(X_2,X_1)=\sup_{\substack{X_1,X_2\in X\\ f_{C_1}(X_2)\geq f_{C_1}(X_1)}}\min\left\{\mu\left[f_{C_1}(X_2)\right],\mu\left[f_{C_1}(X_1)\right]\right\}=0{,}75\ ; \tag{5}$$

$$\mu_{R_{C_1}}(X_3,X_1)=\sup_{\substack{X_1,X_3\in X\\ f_{C_1}(X_3)\geq f_{C_1}(X_1)}}\min\left\{\mu\left[f_{C_1}(X_3)\right],\mu\left[f_{C_1}(X_1)\right]\right\}=0{,}25\ ; \tag{6}$$

$$\mu_{R_{C_1}}(X_2,X_3)=\sup_{\substack{X_2,X_3\in X\\ f_{C_1}(X_2)\geq f_{C_1}(X_3)}}\min\left\{\mu\left[f_{C_1}(X_2)\right],\mu\left[f_{C_1}(X_3)\right]\right\}=1\ ; \tag{7}$$

$$\mu_{R_{C_1}}(X_1,X_3)=\sup_{\substack{X_1,X_3\in X\\ f_{C_1}(X_1)\geq f_{C_1}(X_3)}}\min\left\{\mu\left[f_{C_1}(X_1)\right],\mu\left[f_{C_1}(X_3)\right]\right\}=1\ ; \tag{8}$$

$$\mu_{R_{C_1}}(X_3,X_2)=\sup_{\substack{X_2,X_3\in X\\ f_{C_1}(X_3)\geq f_{C_1}(X_2)}}\min\left\{\mu\left[f_{C_1}(X_3)\right],\mu\left[f_{C_1}(X_2)\right]\right\}=0{,}5\ . \tag{9}$$

Utilizando-se os resultados das aplicações de (2) e (3) apresentados acima e considerando $\mu_R(X_i,X_i) = 1$ (dado que não há preferência entre dois itens iguais), pode-se construir a seguinte matriz de relação *fuzzy* para o critério C1:

$$R_{C_1}=\begin{bmatrix}1 & 1 & 1\\ 0{,}75 & 1 & 1\\ 0{,}25 & 0{,}5 & 1\end{bmatrix}.$$

De acordo com a definição apresentada em (ORLOVSKI, 1978), as relações de preferência *fuzzy* obtidas até o momento são classificadas como não rigorosas e representam o grau com que uma alternativa "não é pior" que outra.

Uma relação não rigorosa R pode ser representada por uma relação rigorosa R^r e uma relação indiferente R_i . É possível dizer que uma alternativa X_i é rigorosamente melhor que X_j se $(X_i, X_j) \in R$ e $(X_j, X_i) \notin R$. O subconjunto de todos estes pares é a relação de preferência rigorosa R^r . Como $(X_i, X_j) \in R^{-1}$ é equivalente a $(X_j, X_i) \in R$ (PEDRYCZ et al, 2011), é possível utilizar a relação inversa para construir $R^r = R \setminus R^{-1}$. Se $(X_i, X_j) \in R^r$, então X_i domina X_j, ou seja, X_i é melhor que X_j.

Considerando que $\mu_R^{-1}(X_i, X_j) = \mu_R(X_j, X_i)$, a função de pertinência que corresponde a R^r pode ser definida como:

$$\mu_R^r(X_j, X_i) = \max\{\mu_R(X_i, X_j) - \mu_R(X_j, X_i); 0\}. \quad (10)$$

Dessa forma, podemos obter a seguinte matriz de relação *fuzzy* rigorosa aplicando (10) sobre R_{C_1} :

$$R_{C_1}^r = \begin{bmatrix} 0 & 0{,}25 & 0{,}75 \\ 0 & 0 & 0{,}5 \\ 0 & 0 & 0 \end{bmatrix}.$$

Em particular, $\mu_{Rr}(X_j, X_i)$ é a função de pertinência do conjunto *fuzzy* de todas as alternativas X_i, que são estritamente dominadas por X_j. O complemento $1- \mu_{Rr}(X_j, X_i)$ fornece o conjunto *fuzzy* de alternativas que são não dominadas por outras alternativas de X. Para escolher o conjunto de todas as alternativas, que são "não dominadas" por outras alternativas de X, é necessário obter a interseção de todos os $1- \mu_{Rr}(X_j, X_i)$, $X_i \in X$ para todo $X_j \in X$. Esta interseção pode ser obtida aplicando a seguinte correlação:

$$\eta\{\mu_{R_r}^{ND}(X_i) = 1 - \sup \mu_R(X_i, X_j)\} \quad (11)$$

.

A aplicação dessa correlação sobre a matriz obtida anteriormente nos permite obter um vetor que apresenta o grau de pertinência das alternativas não dominadas:

$$\mu_{R_{C1}^r}^{ND}(X_k) = [1 \quad 0.75 \quad 0.25] \quad (12)$$

.

Com esse valor é possível ordenar as alternativas por ordem de importância. Logo, como pode ser visto, a alternativa X_1 é a melhor dentre as analisadas, seguido da alternativa X_2 e X_3 respectivamente.

Realizando o mesmo desenvolvimento aplicado para o critério C_1, pode-se obter a matriz $\mu_{R^r_{C_2}}^{ND}(X_k)$ que corresponder à matriz que apresenta o grau de pertinência das alternativas não dominadas no contexto do critério C_2:

$$R_{C_2} = \begin{bmatrix} 1 & 0{,}75 & 0 \\ 1 & 1 & 0{,}5 \\ 1 & 1 & 1 \end{bmatrix};$$

$$R^r_{C_2} = \begin{bmatrix} 0 & 0 & 0 \\ 0{,}25 & 0 & 0 \\ 1 & 0{,}5 & 0 \end{bmatrix};$$

$$\mu_{R^r_{C_2}}^{ND}(X_k) = \begin{bmatrix} 0 & 0{,}5 & 1 \end{bmatrix}.$$

Assim, para o Critério C_2, a alternativa X_1 é pouco pior que a alternativa X_2 e indiscutivelmente pior que a alternativa X_3, e a alternativa X_2 é pior que a alternativa X_3.

As modelagens acima foram feitas sem levar em consideração que o que se deseja na verdade é uma solução única, ou seja, ordenar X_1, X_2 e X_3 levando em consideração os dois critérios ao mesmo tempo. Para se obter uma única resposta, a ideia é que se estabeleça uma agregação entre as matrizes de relação *fuzzy* não rigorosas com o uso de operadores específicos.

Dentre os métodos de agregação existentes, é possível apontar as escalas de conversão de Chen e Hwang (CHEN et al, 1992), o algoritmo de Baas e Kwakernaak (BAAS et al, 1977), o método *fuzzy* Delphi (CHENG, 1999) e o método de agregação de Yager (YAGER, 1993). Cada um dos métodos apresenta operadores de agregação específicos para se obter a interseção entre as matrizes. A seguir, a agregação entre as matrizes será exemplificada utilizando o operador *min*, como pode ser visto em (PEDRYCZ et al, 2011):

$$R_{C_1} = \begin{bmatrix} 1 & 1 & 1 \\ 0{,}75 & 1 & 1 \\ 0{,}25 & 0{,}5 & 1 \end{bmatrix};$$

$$R_{C_2} = \begin{bmatrix} 1 & 0{,}75 & 0 \\ 1 & 1 & 0{,}5 \\ 1 & 1 & 1 \end{bmatrix};$$

$$R_{C_{1,2}} = R_{C_1} \cap R_{C_2} = \begin{bmatrix} 1 & 0{,}75 & 0 \\ 0{,}75 & 1 & 0{,}5 \\ 0{,}25 & 0{,}5 & 1 \end{bmatrix}.$$

Avançando para a obtenção da matriz de pertinências não dominadas, tem-se:

$$R'_{C_{1,2}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0{,}25 & 0 & 0 \end{bmatrix};$$

$$\mu^{ND}_{R'_{C_{1,2}}}(X_k) = [0{,}75 \quad 1 \quad 1] \quad .$$

O vetor obtido acima indica que os itens X_2 e X_3 são mais importantes que o item X_1 e igualmente importantes entre si, levando em consideração a opinião do especialista no contexto dos dois critérios (C_1 e C_2) ao mesmo tempo.

4.6.2. Tomada de Decisão Baseada na Modelagem de Relações Fuzzy

Os resultados obtidos em 4.6.1 permitem ordenar as alternativas de acordo com suas importâncias dadas as opiniões do especialista E1. Porém, em um processo de tomada de decisão, é importante ainda que seja considerado que cada critério pode ter importância diferente e que a opinião de um determinado especialista possa ter maior peso.

Para adicionar a importância de cada critério durante o desenvolvimento da construção da matriz de relação *fuzzy*, pode-se utilizar a seguinte expressão:

$$\mu_{R_{C_{1,\ldots,m}}}(X_i, X_j) = \min_{k=1,\ldots,m} \{ \mu_{R_{C_k}}(X_i, X_j)\, P_k \}, \qquad (13)$$

onde $\mu_{R_{C_{1,\ldots,m}}}(X_i, X_j)$ representa a preferência da alternativa X_i sobre a alternativa X_j, levando em conta m critérios; $\mu_{R_{C_k}}(X_i, X_j)$ representa a preferência da alternativa X_i sobre a alternativa X_j, com base no critério C_k; P_k é o peso que representa a importância associada a cada critério tal que $P_k \in [0,1]$ e $\sum P_k = 1_{\ k=1,\ldots,m}$.

De forma semelhante, para adicionar o resultado da importância de cada especialista, pode-se aplicar a seguinte expressão:

$$\mu_R^{E_{1,\dots,q}}(X_i, X_j) = \min_{k=1,\dots,q} \{ \mu_{R_{C_k}}(X_i, X_j) w_k \} . \quad (14)$$

Caso seja desejado aplicar tanto a importância dos especialistas quanto dos critérios, pode-se aplicar as duas expressões apresentadas acima, uma no resultado da outra, ou seja, na matriz resultado obtida da aplicação de (13), aplica-se (14) ou vice-versa.

Dessa forma, para enriquecer o exemplo aqui apresentado, considere ainda que o critério C_1 e C_2 tenham respectivamente os pesos 0,4 e 0,6. Aplicando as expressões (13) e (14) nas duas matrizes de relacionamento *fuzzy* não rigorosas, definidas anteriormente para o critério 1 e critério 2 e realizando a agregação através do operador *min*, temos:

$$R_{C_{1,2}} = R_{C_1} \cap R_{C_2} = \begin{bmatrix} 0,4 & 0,4 & 0 \\ 0,3 & 0,4 & 0,3 \\ 0,1 & 0,2 & 0,4 \end{bmatrix};$$

$$R'_{C_{1,2}} = \begin{bmatrix} 0 & 0,1 & 0 \\ 0 & 0 & 0,1 \\ 0,1 & 0 & 0 \end{bmatrix};$$

$$\mu_{R'_{C_{1,2}}}^{ND}(X_k) = [0,9 \quad 0,9 \quad 0,9] .$$

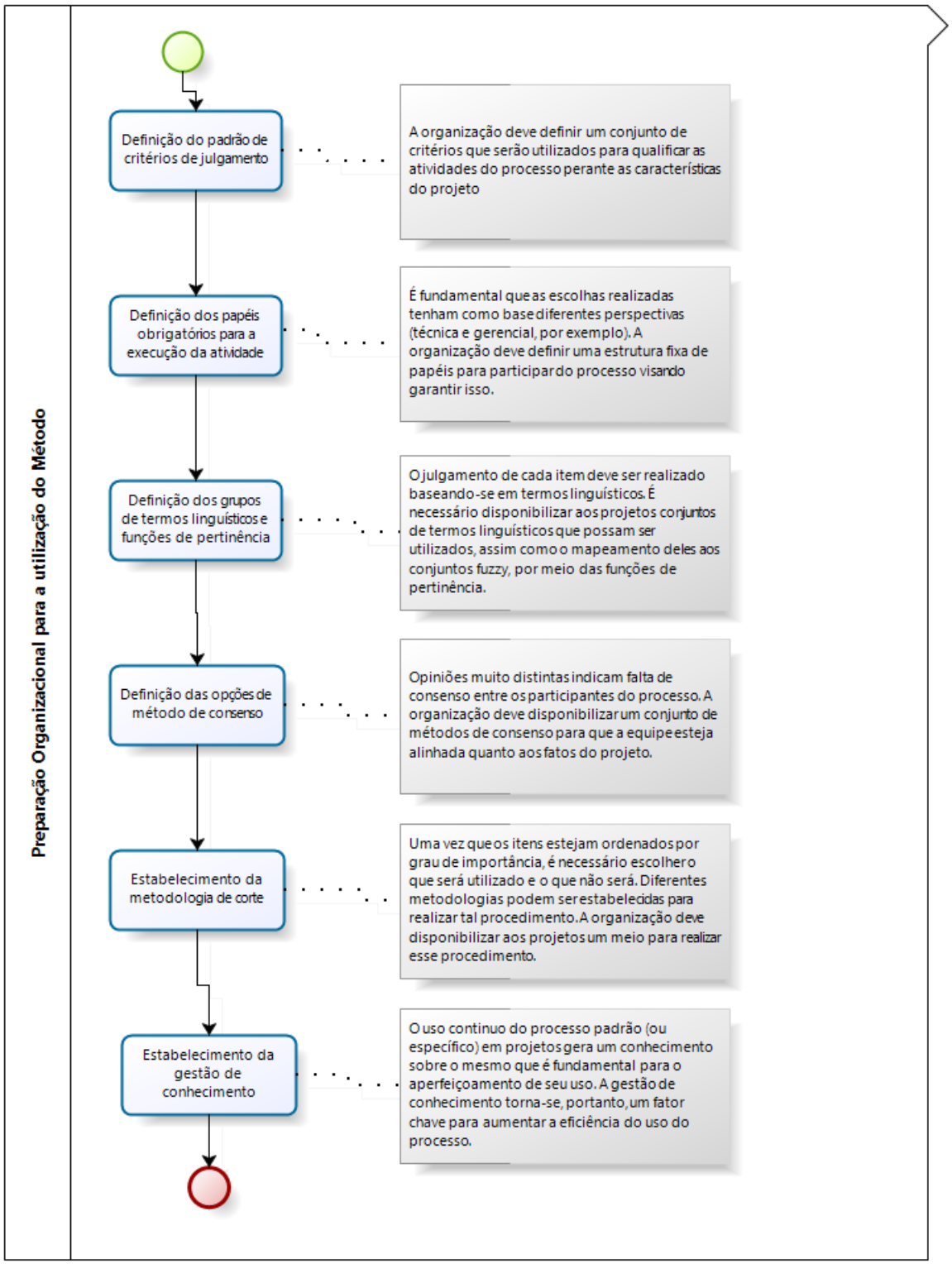
Com o surgimento dos pesos, verifica-se para esse exemplo que todas as alternativas são de igual importância. Essa é uma das piores situações que se poderia chegar com a utilização do método, embora extremamente rara. O resultado acima indica que todos os itens julgados possuem igual importância, portanto não fica claro quais deles deveriam ser escolhidos. Em uma situação como essa, o aconselhavel é que os especialistas discutam entre si novamente as alternativas julgadas e refinem suas opiniões sobre eles. Após isso, a aplicação do método pode ser realizada novamente.

5. O MÉTODO PROPOSTO PARA INSTANCIAÇÃO DE PROCESSOS

Como já mencionado nos capítulos anteriores, a atividade de instanciação do processo padrão é de extrema importância para que o potencial de qualidade do processo efetivamente se expresse durante a execução do projeto. Neste capítulo é apresentado um método para auxilio na execução dessa atividade. O objetivo do uso desse método é potencializar a justificativa e eficiência real da escolha da configuração das atividades relacionadas ao processo de desenvolvimento de software da organização com base na formação e no processamento matemático das preferências dos profissionais envolvidos e do uso de técnicas de gestão do conhecimento.

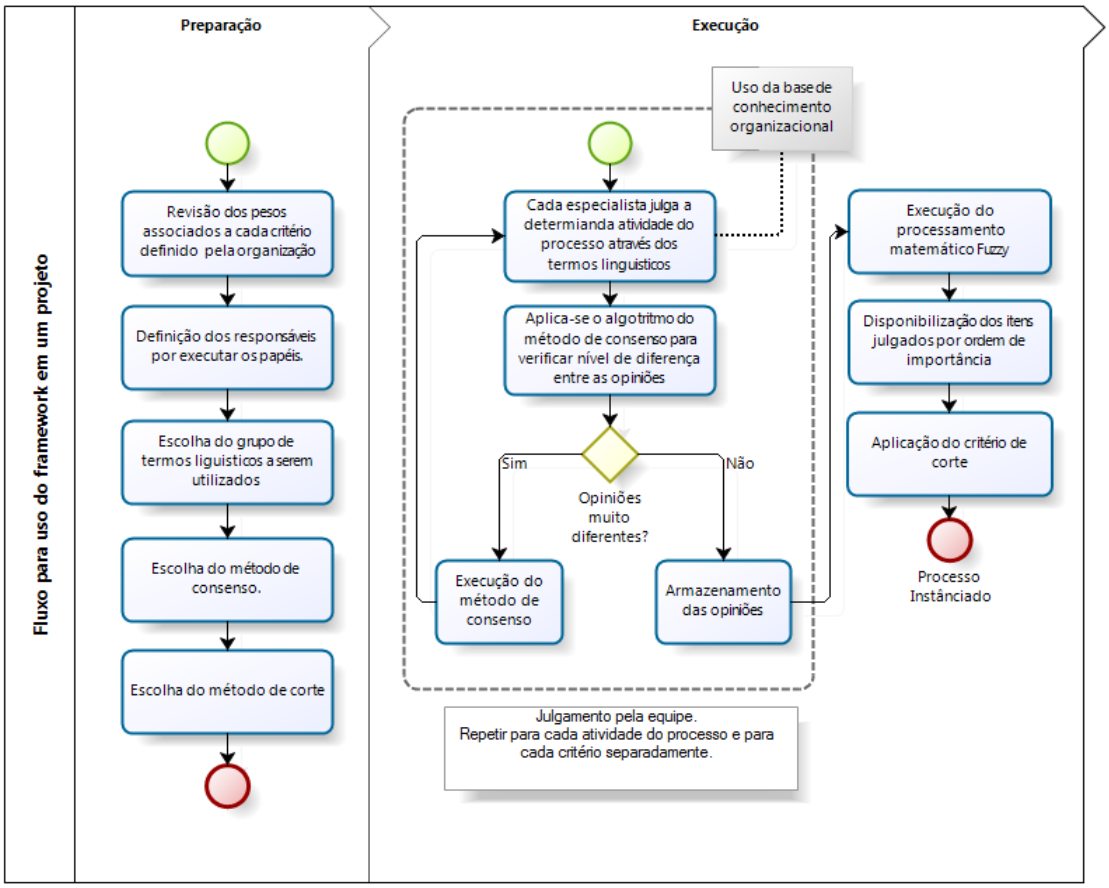
Os dois diagramas apresentados nas Figuras 16 e 17 ilustram em alto nível, o processo de implementação do método proposto. O primeiro deles, Figura 16, apresenta, em alto nível, os procedimentos necessários para a implementação do método proposto no nível organizacional (da empresa). O segundo, Figura 17, apresenta, em alto nível, os procedimentos necessários para a implementação do método proposto no nível de projeto. Nas próximas seções, cada item apresentado no diagrama será melhor detalhado. Esses diagramas apresentam a visão um passo a passo do método empregado, visando seu entendimento e aplicação do método.

Figura 16 – Preparação organizacional para a utilização do método



Fonte: Elaborado pelo Autor

Figura 17 - Fluxo para a aplicação do método em um projeto



Fonte: Elaborado pelo Autor

5.1. Definição dos critérios de julgamento padrão utilizados na tomada de decisão

Existem basicamente existem 4 variáveis que são os pilares na condução de qualquer projeto: tempo, custo, escopo e qualidade (Project Management Institute, 2008). O objetivo do uso de um processo de desenvolvimento de software é potencializar o controle sobre essas 4 variáveis, de modo que o projeto atinja seu objetivo tanto do ponto de vista do cliente (qualidade, tempo e escopo) quanto do ponto de vista da empresa (custo). Tendo como base estes fatores, optou-se por utilizar como critérios base de julgamento das atividades estes quatro importantes fatores de gestão.

Algumas organizações podem preferir por utilizar uma lista diferente de critérios da definida aqui como padrão. Não há problema que se faça essa customização uma vez que é importante que o método se adapte às características da empresa. Porém, é de extrema importância que exista uma única lista de critérios para o processo padrão da empresa (ou processo específico). Isso é necessário para que haja uma linguagem comum entre os

membros da organização, o que contribui para uma gestão de conhecimento mais fluente e eficaz. Empresas que desejem critérios mais detalhados, por exemplo, podem optar por utilizar as 9 áreas de conhecimento de projeto definidas no PMBOK: Escopo, Tempo, Custo, Qualidade, Integração, Recursos Humanos, Comunicação, Riscos e Aquisição (Project Management Institute, 2008).

Projetos de alta qualidade entregam o produto, serviço ou resultado solicitado dentro do escopo, no prazo, dentro do orçamento e com a qualidade de produto elevada. A relação entre esses fatores ocorre de tal forma que, se algum dos quatro fatores mudar, provavelmente, pelo menos um dos outros fatores será afetado. Os gerentes de projetos também gerenciam projetos em resposta a incertezas. Um risco do projeto é um evento ou condição incerta que, se ocorrer, terá um efeito positivo ou negativo em pelo menos um objetivo do projeto.

Dessa forma, frente a uma atividade do processo padrão, o grupo de especialistas deve refletir sobre sua importância frente a cada um destes quatro critérios. Para auxiliar nessa reflexão, é apresentado na Tabela 3 um conjunto de perguntas que podem ser feitas durante a análise. As perguntas apresentadas são apenas sugestões, ou seja, a empresa possui liberdade para estabelecer o seu próprio conjunto de perguntas. Os usuários devem estar cientes do que se trata cada um dos critérios e como a atividade julgada se relaciona com eles. As perguntas devem auxiliar nessa reflexão.

Quadro 2 - Lista base de perguntas para reflexão sobre os critérios

Critério	**Perguntas**
Custo	A atividade em análise é importante para a gestão de custos do projeto? A remoção desta atividade trará ganho significativo para o orçamento do projeto, sem prejudicar significativamente a qualidade do desenvolvimento do software?
Tempo	A atividade em análise é importante para a gestão do cronograma do projeto? A remoção desta atividade trará ganho significativo para o cronograma do projeto sem prejudicar significativamente a qualidade do desenvolvimento do software?
Qualidade	A atividade em questão é importante para garantir a qualidade do software produzido? A remoção desta atividade traz ganhos expressivos com relação a outros fatores com pouco impacto na perspectiva da qualidade do software?

Escopo	A atividade em análise é importante para a gestão do escopo do projeto? A remoção desta atividade traz pouco impacto na perspectiva da gestão de escopo?

Fonte: Elaborado pelo Autor

Conforme o nível de experiência e maturidade da organização aumenta, espera-se que a quantidade de questões aumente, fornecendo assim um questionamento mais rico para a análise dos critérios.

Um ponto importante com relação aos critérios apresentados é que, dependendo das características do projeto, elas podem ter diferentes níveis de importância. Para um projeto com cliente estratégico para a organização, por exemplo, a qualidade e tempo podem ser fatores muito importantes, porém o custo pode ser um fator de baixa relevância. Dessa forma, é proposto aqui também a associação de pesos a cada critério, de forma que esses pesos expressem a importância que cada critério possui para o projeto. Na escala proposta neste trabalho, os pesos variam entre 0 a 5, sendo que o valor 5 expressa a máxima importância do atendimento a aquele critério. Como padrão, todos os valores recebem peso 3 (mediano) (tal procedimento é realizado no momento da construção das matrizes de preferencia *fuzzy)*. A revisão da importância de cada critério deve ser realizada internamente em cada projeto pelos especialistas e representa um passo importante na customização do método no nível de projeto, como mostrado no fluxograma no início do capítulo.

5.2. Definição dos especialistas que conduzem o julgamento das alternativas

Um ponto importante para a execução da atividade de instanciação do processo de desenvolvimento de software é a equipe que irá executá-la. O conhecimento que a equipe possui é essencial para que os julgamentos sejam feitos de forma coerente. Muitas vezes a instanciação é realizada somente pelo gerente do projeto, sem que a opinião de outros membros da equipe seja levada em consideração. Isso pode representar um erro crítico, pois certamente existem fatores envolvidos no projeto que o gerente do projeto não domina completamente. Portanto, o ideal é que sejam envolvidos na execução dessa atividade especialistas que possuem conhecimentos com focos diferentes, de forma a complementarem o conhecimento um do outro.

A partir do exposto no parágrafo anterior, é proposto aqui que o grupo que executará a atividade de instanciação do processo de software seja formado minimamente pelos seguintes papéis:

Quadro 3 - Equipe base para a execução do método

Papel	**Descrição/Objetivo**
Gerente do projeto	O gerente de projeto é o principal responsável pelo monitoramento e controle do projeto. Dessa forma, é natural que ele possua experiência na execução de atividades com essas finalidades. A opinião do gerente do projeto, portanto, será essencial para julgar atividades com essa finalidade.
Líder Funcional	O Líder Funcional é um membro do projeto que possui grande conhecimento do processo de negócio do cliente e de arquiteturas de desenvolvimento. Sua opinião, portanto, representa o lado técnico do projeto.
Membro do QA	O membro do QA é alguém que não pertence ao projeto, mas que possui grande conhecimento do processo padrão da empresa. Normalmente ele participou da criação do processo e também é o responsável pela execução de auditorias de qualidade nos projetos da organização. Ele conhece a importância de cada uma das atividades descritas do processo, independente das características de cada.
Process Knowledge Persona	O *Process Knowledge Persona* é uma entidade virtual proposta por este trabalho, formada pelo conhecimento adquirido em projetos anteriores. Tal entidade não participa diretamente do processo de julgamento das atividades, mas suporta os outros papéis em suas tomadas de decisão. (Maiores detalhes sobre serão fornecidos na próxima seção)

Fonte: Elaborado pelo Autor

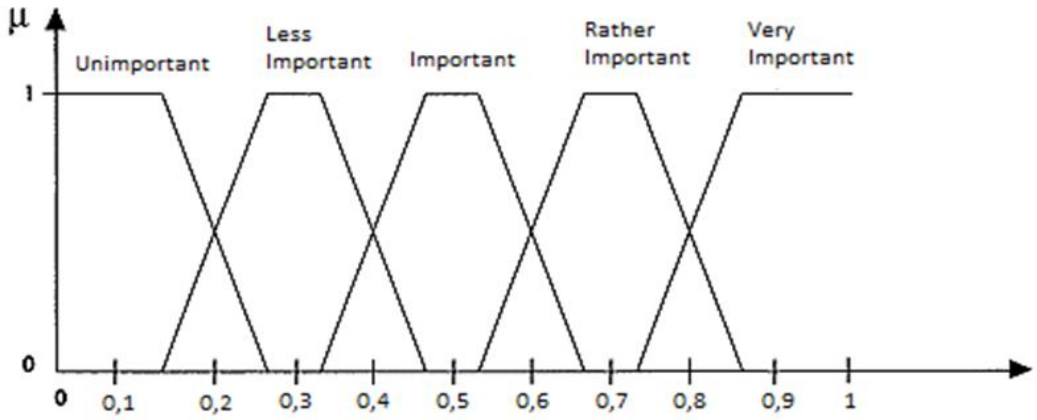
5.3. Definição dos termos linguísticos e da função de pertinência

Como já mencionado, a função de pertinência possui como objetivo mapear a opinião dos especialistas em cada conjunto *fuzzy*. Também foi visto que para cada conjunto *fuzzy*, pode ser associado um termo linguístico e que os especialistas podem utilizar estes termos linguísticos para estabelecerem suas estimativas *fuzzy*.

O método proposto neste trabalho utiliza os mesmos termos linguísticos definidos em (WANG et al, 2001) e apresentados na Figura 18. A escolha se deve aos bons resultados

obtidos naquele trabalho e pelas semelhanças que ele possui com este, uma vez que, em (WANG et al, 2001), é apresentado uma proposta de uso da decisão de multicritérios em grupo para tratar o problema de gestão de configuração.

Figura 18 - Termos linguísticos para julgamento das atividades do processo



Fonte: Elaborado pelo Autor

Em nível organizacional, podem ser definidos grupos de escalas linguísticas além da apresentada aqui como padrão. Cada grupo pode ser associado a uma função de pertinência específica. Ao iniciar o projeto de desenvolvimento de software, os especialistas terão a liberdade de escolher com qual grupo de termos linguísticos irão trabalhar, tendo em vista as características de cada grupo.

5.4. Agregação de Preferências e Metodologia de consenso

No método apresentado neste trabalho, optou-se por estabelecer como padrão a agregação baseada no operador mínimo, seguindo o mesmo procedimento utilizado nos exemplos apresentados em 4.6.2. Novamente, a escolha se baseou no sucesso que a utilização deste operador teve em trabalhos com contexto semelhante de aplicação, como em (WANG et al, 2001) e (PEDRYCZ et al., 2011).

O operador mínimo (*min*) é tradicionalmente usado na teoria dos conjuntos *fuzzy* para agregar as condições às implicações de uma regra. Na agregação das relações de preferências, o operador *min* preserva a apreciação mais pessimista dentre todas as pessoas, considerando suas respectivas importâncias.

Nos procedimentos de agregação, é possível estabelecer a priorização de critérios e também de pessoas. Essa priorização é realizada pela atribuição de coeficientes de

importância (usualmente denominados pesos), que distinguem a importância, o impacto, a influência e/ou dominância de determinados critérios e/ou pessoas sobre os demais. A aplicação dos pesos pode ser realizada utilizando as expressões (13) e (14).

Como resultado desse procedimento, tem-se, portanto, um único valor numérico que representa a opinião agregada dos especialistas com relação a aquele item. Uma vez que esse procedimento tenha sido realizado para todos os itens julgados, é possível obter a matriz Rc, assim como mostrado na seção 4.6.2.

A tomada de decisão objetiva, em primeira instância, encontrar a melhor solução num conjunto de alternativas possíveis. Na modalidade de grupo, em particular, surgem algumas questões decorrentes da participação e interação entre diversas pessoas. A questão mais importante e também a mais complexa, é a condução dos decisores a uma convergência de posições visando a construção de uma situação de consenso.

O termo consenso, tradicionalmente significa um acordo estrito e unânime entre todas as pessoas envolvidas num processo de tomada de decisão. NESS (NESS, 1998) definem consenso como "...uma decisão que deve ser alcançada, quando a maioria dos membros de uma equipe concorda com uma opção clara e os poucos que se opõe a ela, pensam que tiveram oportunidade razoável para influenciar na escolha. Todos os membros da equipe concordam em apoiar a decisão".

As técnicas para a construção de consenso exigem, inicialmente, a definição de um nível mínimo de concordância de grupo, que deve ser atingido por uma ou mais alternativas. Dessa forma, durante o processo de tomada de decisão em grupo, os valores de consenso de cada alternativa e de grupo, são monitorados e as alterações necessárias são promovidas até que o nível desejado seja alcançado.

O consenso sobre uma alternativa, quando comparado ao consenso sobre a alternativa preferida pelo grupo, é dado por:

$$C_i = \sum_{l=1}^{m} \left[\left(1 - \frac{\left| O^G(X_i) - O^{E_l}(X_i) \right|}{n-1} \right) \times \omega_{E_l} \right], \tag{15}$$

onde C_i indica o nível de consenso sobre a i-ésima alternativa; ω_{E_l} alternativa; iva; consenso sob da l-ésima pessoa; $O^G(X_i)$ é a ordem da i-ésima alternativa na classificação de

grupo; $O^{E_l}(X_i)$ é a ordem da i-ésima alternativa na classificação da l-ésima pessoa; e n representa o número de alternativas no conjunto de soluções possíveis.

Considerando o exemplo abaixo:

Tabela 2 - Exemplo da aplicação do método de consenso

Especialista	Ordem das preferências	Ordem das alternativas			
		$O(X_1)$	$O(X_2)$	$O(X_3)$	$O(X_4)$
E_1	$X_4 \succ X_2 \succ X_1 \succ X_3$	3	2	4	1
E_2	$X_3 \succ X_1 \succ X_2 \succ X_4$	2	3	1	4
E_3	$X_4 \succ X_3 \succ X_1 \succ X_2$	3	4	2	1
E_4	$X_3 \succ X_4 \succ X_1 \succ X_2$	3	4	1	2
E_5	$X_3 \succ X_1 \succ X_2 \succ X_4$	2	3	1	4
Grupo	$X_3 \succ X_1 \succ X_4 \succ X_2$	**2**	**4**	**1**	**3**

Fonte: Elaborado pelo Autor

Aplicando (15) para cada uma das alternativas é obtido o consenso do grupo sobre cada uma delas. Para a alternativa X_1, por exemplo, temos o seguinte resultado:

$$C_1 = \left(1-\frac{|2-3|}{3-1}\right)\times 0{,}2 + \left(1-\frac{|2-2|}{3-1}\right)\times 0{,}2 + \left(1-\frac{|2-3|}{3-1}\right)\times 0{,}2 + \left(1-\frac{|2-3|}{3-1}\right)\times 0{,}2 + \left(1-\frac{|2-2|}{3-1}\right)\times 0{,}2 = 0{,}8 \quad (16)$$

O resultado final da execução para todas as alternativas é apresentado na tabela abaixo:

Tabela 3 - Resultado da aplicação do método de consenso

	C_1	C_2	C_3	C_4
Consenso sobre cada alternativa	0,8	0,7333	0,7333	0,5333
Consenso de grupo	C^G=73,33%			

Fonte: Elaborado pelo Autor

Se fosse estabelecido por exemplo que o mínimo aceitável de consenso entre o grupo fosse de 80% para cada uma das alternativas, aos resultados obtidos em C2, C3 e C4 não seriam aceitáveis. O grupo precisaria conversar para alinhar melhor os conhecimentos e repetir o procedimento de julgamento novamente.

5.5. Definição do critério de corte

Com o desenvolvimento da matriz R_c obtida na seção anterior a partir da aplicação da correlação $\eta\{\mu_{R_r}^{ND}(X_i) = 1 - \sup \mu_R(X_i, X_j)$ assim como mostrado na seção 4.6.1, obtém-se um vetor de valores que representam a importância que cada atividade de avaliação possui, considerando a opinião do grupo como um todo. Um exemplo deste vetor é apresentado abaixo (ordenado de X_1 a X_6):

$$\mu_{R_C^r}^{ND}(X_k) = \begin{bmatrix} 0 & 0{,}85 & 0{,}5 & 1 & 0{,}45 & 0{,}8 \end{bmatrix} \quad (17)$$

A partir desta situação, o grande desafio é decidir o que será utilizado e o que não será utilizado. É possível, por exemplo, dizer que X_4, X_2 e X_6 são itens de grande importância e que devem ser selecionados, porém esse julgamento já se torna mais difícil com relação ao item X_3 já que ele possui um valor mediano. Novamente ocorre um problema no qual a subjetividade em excesso pode prejudicar a decisão final.

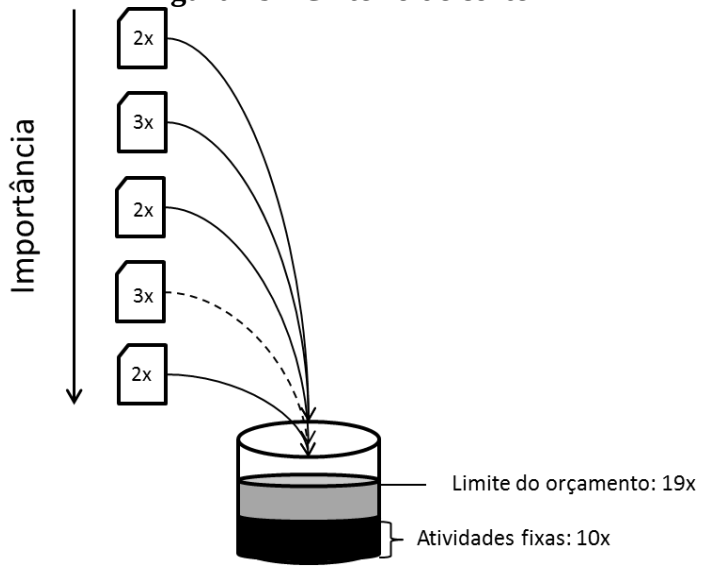
Para tornar essa decisão mais confiável, o método proposto neste trabalho estabelece a seguinte metodologia:

a) Para cada atividade estabelecida no processo padrão da organização, estabelecer um custo padrão. Este custo pode ser estabelecido em horas ideais ou em um valor financeiro. É de extrema importância que esses valores sejam revisados periodicamente e que a gestão de conhecimento da empresa ajude neste processo. Não será escopo deste trabalho definir a metodologia para estabelecer estes valores, porém, durante esse processo, é necessário estabelecer variáveis que dependam das características do projeto (número de requisitos, tamanho da equipe, etc.). Dessa, forma, para um projeto pequeno, uma determinada atividade do processo pode ter um custo "x" enquanto para um projeto classificado como médio, a mesma atividade pode ter um custo "2x".
b) Estabelecer um conjunto de atividades do processo padrão que são consideradas fixas, ou seja, elas devem ser executadas independente da opinião dos especialistas ou de seus custos. Fazem parte desse conjuntos atividades essenciais tais como algumas relacionadas á analise de requisito, criação do plano do projeto, etc..

c) Tendo o custo que cada atividade representa para aquele projeto e o orçamento estabelecido para o projeto em contrato, o gerente do projeto pode realizar o seguinte procedimento para realizar o corte: Por ordem de importância (maior para menor), o gerente soma os custos de cada atividade. Assim que o custo somado atingir o orçamento estabelecido para o projeto, aquele ponto é considerado o ideal para realizar o corte. As atividades classificadas como obrigatórias devem ser somadas previamente antes de iniciar esse processo. A figura 13 exemplifica esse procedimento. Antes desse momento, todas as atividades que couberem no orçamento considerando essa regra devem ser obrigatoriamente incluídos no processo instanciado do projeto. Atividades que ficaram fora devem ser avaliadas pelo gerente, porém não deve ser obrigatória a sua inclusão.
d) Discrepâncias muito grandes na capacidade de absorver atividades do projeto tendo em vista essa metodologia podem indicar um erro na elaboração de seu orçamento. Isso pode indicar a necessidade de rever o planejamento financeiro do projeto.

A figura abaixo mostra como a metodologia funciona:

Figura 19 - Critério de corte



Fonte: Elaborado pelo Autor

5.6. Process Knowledge Persona

Como mencionado, a atividade de instanciação do processo organizacional tem como objetivo realizar sua customização de forma a atender as características do projeto. Esta atividade é normalmente realizada pelo gerente de projeto (ou por um grupo representativo) que utiliza seus conhecimentos individuais para executar tal tarefa. Porém, assim como já citado, em processos de alta maturidade, existe uma tendência de se ter um grande conjunto

de ativos e uma grande quantidade de informação. Esses fatores tornam a atividade de instanciação mais complexa devido à dificuldade de interação entre o conhecimento do gerente e o conhecimento agregado pelo processo.

Outro problema nesta atividade é o fato de que, provavelmente, um novo projeto possui características peculiares, não familiares gerente de projeto, e, nessa situação, seu conhecimento real sobre o que é realmente importante ou não no processo organizacional torna-se mais uma vez obscura. Em resposta a esse contexto, o gerente de projeto irá escolher atividades pelo seu instinto, e esta escolha subjetiva poderia prejudicar fortemente a qualidade do projeto.

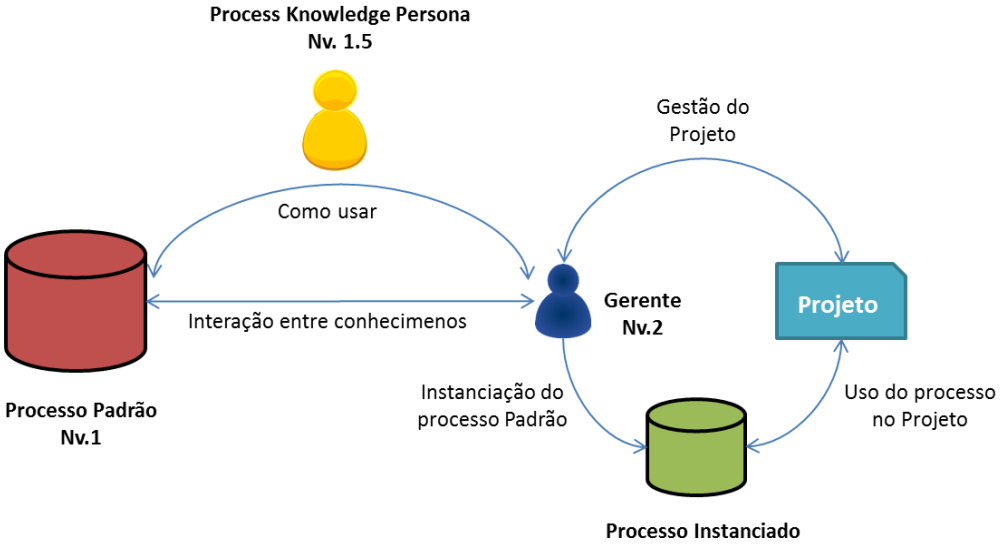
É importante também considerar que, provavelmente, outro gerente de projeto da empresa possui experiência com essa característica de projeto que os outros não possui. Em outras palavras, se eles pudessem compartilhar seus conhecimentos sobre as atividades do processo padrão, certamente, a atividade de instanciação do processo, em geral, se tornaria muito mais segura. No entanto, apesar de ser o ideal, é extremamente difícil garantir a presença do(s) gerente(s) que possui(em) esse conhecimento no momento da instanciação.

Para lidar com os problemas expostos, a gestão de conhecimento pode ser usada como uma poderosa ferramenta. Todo o conhecimento de uso do processo de desenvolvimento de software que os gerentes daquela organização possuem pode ser armazenado em um único local e compilados de forma que se agreguem. Essa nova base de informação pode se comportar como uma entidade que representa a agregação do conhecimento de todos os usuários do processo padrão daquela empresa sobre o uso do processo: um persona dos gerentes.

Como definido por Nonaka (2011), "Persona é um termo dado para descrever uma versão de ser que todos os membros de um grupo possuem. Comportamentos comuns são selecionados de acordo com o objetivo desejado e, com esta informação, uma Persona (que representa os comportamentos comuns) pode ser criada".

Com base no exposto acima e nos problemas motivadores desta pesquisa, é proposta a criação do *Process Knowledge Persona*, uma base de conhecimento que terá como objetivo armazenar conhecimento de como usar o processo padrão e suportar a interação entre o usuário e esse processo. A figura 20 ilustra como ficaria o modelo da instanciação do processo padrão com a presença do *Process Knowledge Persona*. O gerente do projeto utiliza a experiência armazenada no *Process Knowledge Persona* para tomar suas decisões tornando-as mais eficientes e seguras.

Figura 20 - Atuação do *Process Knowledge Persona*



Fonte: Elaborado pelo Autor

A criação dessa nova base de desenvolvimento envolve alguns desafios. É necessário, por exemplo, definir como o conhecimento gerado organizacionalmente sobre como se usar o processo pode ser agregado no *Process Knowledge Persona* e como este vai disponibilizar tal conhecimento para os usuários do processo padrão. Tais itens serão discutidos na próxima seção.

5.7. Criação do *Process Knowledge Persona*

A principal ideia por traz do *Process Knowledge Persona* é que ele se comporte como uma personificação dos usuários do processo. Ele deve ser capaz de representar as opiniões dos usuários do processo nas diferentes situações pelas quais eles já passaram.

Considerando o exposto acima, é natural concluir que a entrada de conhecimento que vai construir o *Process Knowledge Persona* venha dos próprios usuários do processo. Porém, uma vez que esse conhecimento precisa ser compilado e agregado em uma única personificação, é necessário estabelecer regras e padrões para a entrada desse conhecimento. Nesse sentido, é proposto aqui a criação de um *template* que deverá ser preenchido pelo usuário do processo padrão para informar ao *Process Knowledge Persona* as experiências adquiridas com o seu uso.

A ideia é que, no final de um projeto, o gerente, por exemplo, realize uma reunião com sua equipe e discuta os pontos positivos e negativos do uso do processo durante todo o desenvolvimento do projeto. Esta prática já é de certa forma comum e é inclusive incentivada

por modelos de maturidade, como o CMMI. O diferencial aqui é que estas informações serão armazenadas em um documento específico, que será uma instancia do *template* apresentado a seguir.

Figura 21 - *Template* para entrada de conhecimento do *Knowledge Persona*

Nome do Projeto:
Cliente:
Domínio de atuação do Cliente: << Mineração, Bebidas, Alimentos, etc...>>
Tipo do projeto: <<Análise, Desenvolvimento, Análise e Desenvolvimento, etc..>>
Tamanho do Projeto: <<Pequeno, Médio, Grande>>
Data:

1. Lista de atividades utilizadas e não adaptadas do processo padrão.
   - 1.1. <<Área de Processo 1>>
     - 1.1.1. <<Atividade 1>>
       - 1.1.1.1. <<Feedback>>
     - 1.1.2. <<Atividade 2>>
       - 1.1.2.1. <<Feedback>>
   - 1.2. <<Área de Processo 2>>
     - 1.2.1. <<Atividade 1>>
       - 1.2.1.1. <<Feedback>>
     - 1.2.2. <<Atividade 2>>
       - 1.2.2.1. <<Feedback>>
   - 1.3. <<Área de Processo N>>
     - 1.3.1.1. ...
2. Lista de atividades utilizadas e adaptadas do processo padrão.
   - 2.1. <<Área de Processo 1>>
     - 2.1.1. <<Atividade 1>>
       - 2.1.1.1. <<Adaptação>>
       - 2.1.1.2. <<Motivo>>
       - 2.1.1.3. <<Feedback>>
     - 2.1.2. <<Atividade 2>>
       - 2.1.2.1. <<Adaptação>>
       - 2.1.2.2. <<Motivo>>
       - 2.1.2.3. <<Feedback>>
     - 2.1.3. <<Atividade N>>
       - 2.1.3.1. <<Adaptação>>
       - 2.1.3.2. <<Motivo>>
       - 2.1.3.3. <<Feedback>>
   - 2.2. <<Área de Processo N>>
     - 2.2.1.1. ...
3. Lista de atividades não utilizadas do processo padrão.
   - 3.1. <<Área de Processo 1>>
     - 3.1.1. <<Atividade 1>>
       - 3.1.1.1. <<Motivo>>
       - 3.1.1.2. <<Feedback>>
     - 3.1.2. <<Atividade N>>
       - 3.1.2.1. <<Motivo>>
       - 3.1.2.2. <<Feedback>>
   - 3.2. <<Área de Processo N>>
     - 3.2.1.1. ...

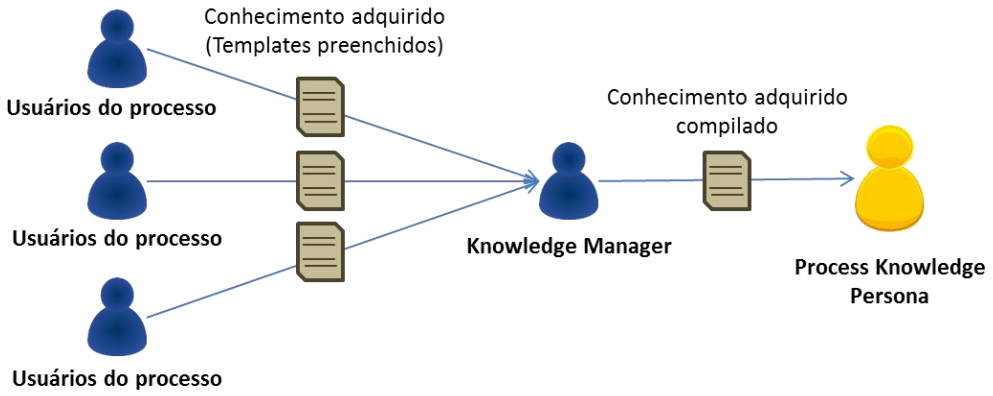
Fonte: Elabordo pelo Autor

O *template* apresentado acima deve ser preenchido durante a própria instanciação do processo padrão realizado pelo projeto com exceção dos itens relativos à *feedback* que devem ser preenchidos no final do projeto. O projeto deve fornecer um feedback sobre as escolhas realizadas (adaptação, exclusão ou uso da atividade) neste ultimo momento.

O preenchimento do *template* apresentado na Figura 21 permite que os usuários do processo registrem o conhecimento adquirido por eles com relação ao uso, adaptação ou exclusão de cada uma das atividades propostas pelo processo padrão. Uma vez que esses conhecimentos estiverem registrados, o desafio para a formação do *Process Knowledge Persona* passa a ser a agregação desses conhecimentos em um só local. A principio, é proposto aqui que esta atividade de agregação de conhecimento seja executada por um papel importante dentro do contexto da gestão de conhecimento: o *Knowledge Manager*.

O *Knowledge Manager* é um papel organizacional que possui como principal responsabilidade o gerenciamento do conhecimento organizacional gerado. Ele é responsável, por exemplo, por organizar treinamentos organizacionais e garantir que o conhecimento adquirido ali se multiplique e permaneça na empresa. Ele deve ser capaz de identificar e avaliar criticamente o valor do conhecimento manipulado dentro da organização. Devido a isso, esse papel se torna ideal para realizar a agregação do conhecimento informado pelos usuários do processo, uma vez que ele já está acostumado com tal procedimento.

Figura 22 - Processamento dos *templates*



Fonte: Elaborado pelo Autor

Utilizando o mesmo *template* apresentado na Figura 21, o *Knowledge Manager* deve analisar as informações fornecidas pelos usuários e, de forma simplificada, extrair as decisões tomadas. Ele deve agrupar o conhecimento de uso do processo por características do projeto que o utilizou, tais como tamanho do projeto, cliente e domínio de aplicação. Os *templates* preenchidos pelo *Knowledge Manager* são então inseridos em uma ferramenta específica ou

diretamente dentro do processo padrão. O conjunto das informações presentes nesses documentos representa o conhecimento agregado da organização em como utilizar o processo padrão nas mais variadas situações e formam o *Process Knowledge Persona*.

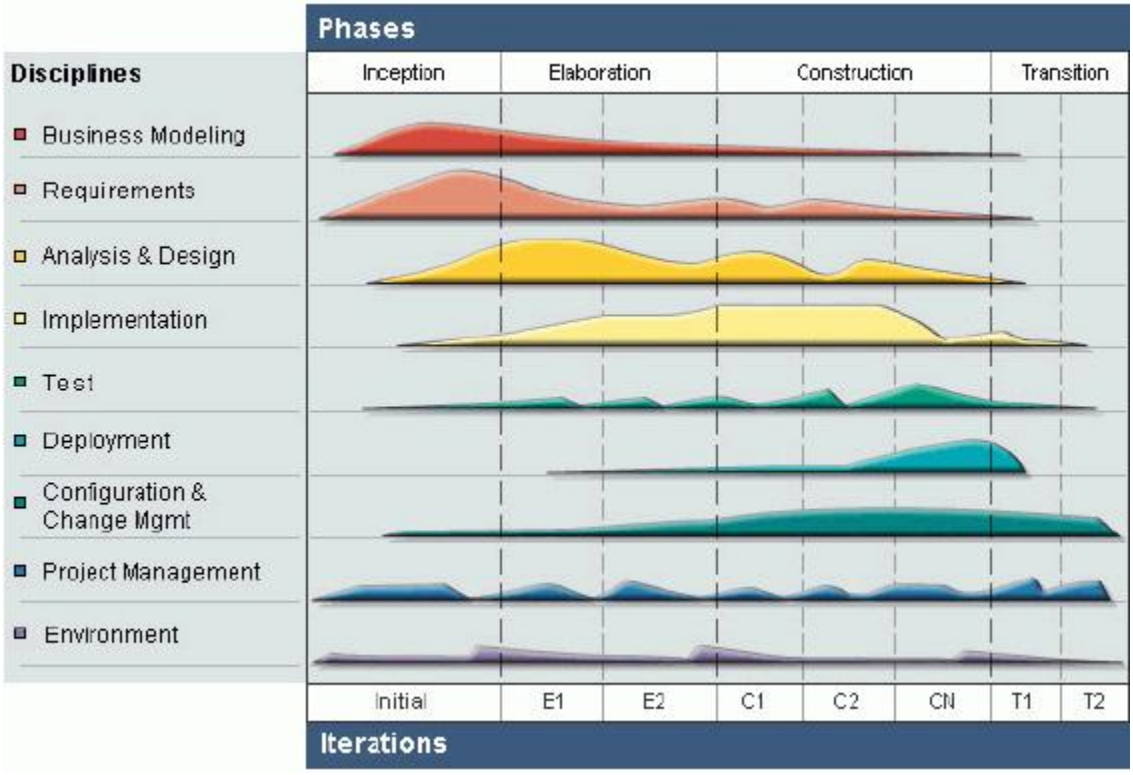
A forma como o conhecimento presente no *Process Knowledge Persona* é fornecido aos usuários é um fator importante dentro do contexto de todo o processo. Este conhecimento deve ser claro e facilmente acessível. Uma forma simples de atingir tal objetivo é separar tal conhecimento por atividade do processo padrão e agregar esses trechos na descrição das próprias atividades no processo. Assim, quando o usuário estiver lendo sobre a atividade, ele terá também as informações históricas de seu uso dentro da organização classificadas de acordo com as características dos projetos.

5.8. Exemplo de Uso

Com o intuito de esclarecer melhor como o *Knowledge Persona* funcionará dentro da atividade de instanciação do processo padrão, é apresentado aqui um exemplo de uso da proposta. Para tal, será utilizado aqui o *Rational Unified Process* como o processo padrão base.

O *Rational Unified Process* é um processo proprietário de engenharia de software adquirido pela IBM, que usa a abordagem da orientação a objetos em sua concepção e é projetado e documentado utilizando a anotação UML. De forma geral, é um processo altamente conhecido no meio acadêmico e utilizado como referência em várias pesquisas científicas, como (POLLICE et al., 2003) (HEIJSTEK et al., 2008) e (BELANI et al., 2009).

Figura 23 - *Rational Unified Process*

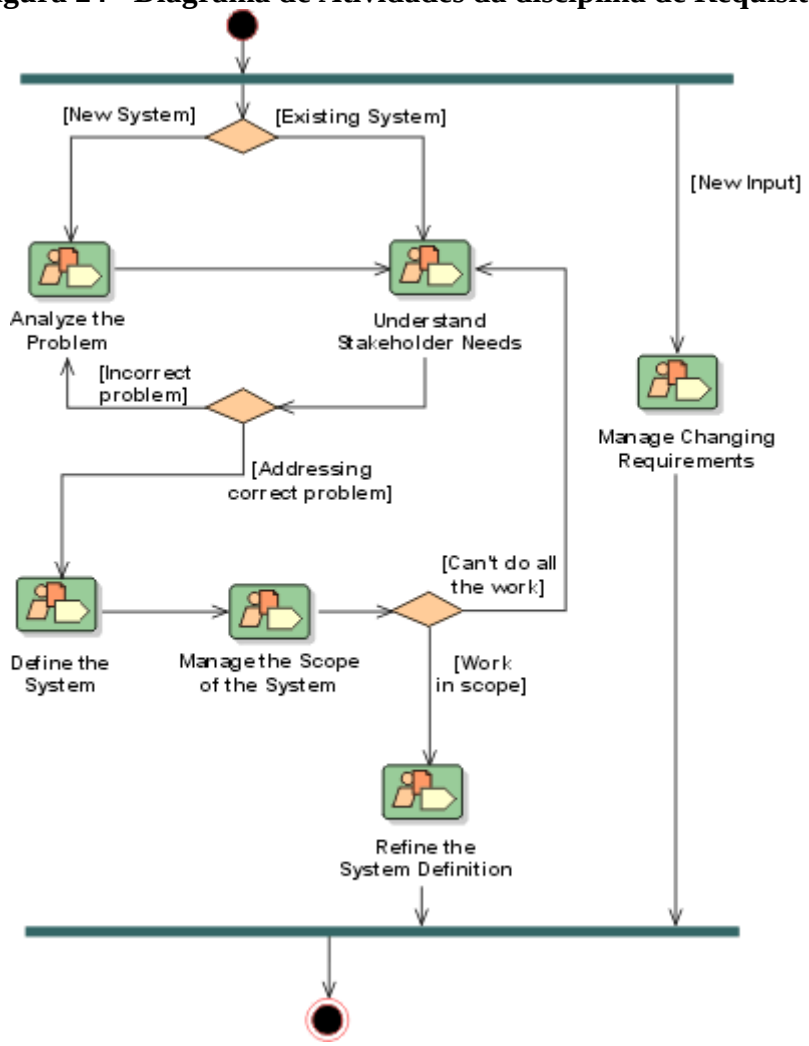


Fonte: HEIJSTEK et al., 2008

Cada disciplina do RUP contém um conjunto de atividades que idealmente devem ser executadas para garantir a qualidade do produto sendo construído. A disciplina de requisitos, por exemplo, que será utilizada nesse caso de teste, explica como levantar pedidos dos *stakeholders* e transformá-los em um conjunto de requisitos que os produtos devem atender.

Uma das formas que o RUP representa as atividades que precisam ser executadas em uma disciplina é através de um diagrama de atividade. O diagrama de atividade relativo à disciplina de requisitos pode ser visualizado na Figura 24.

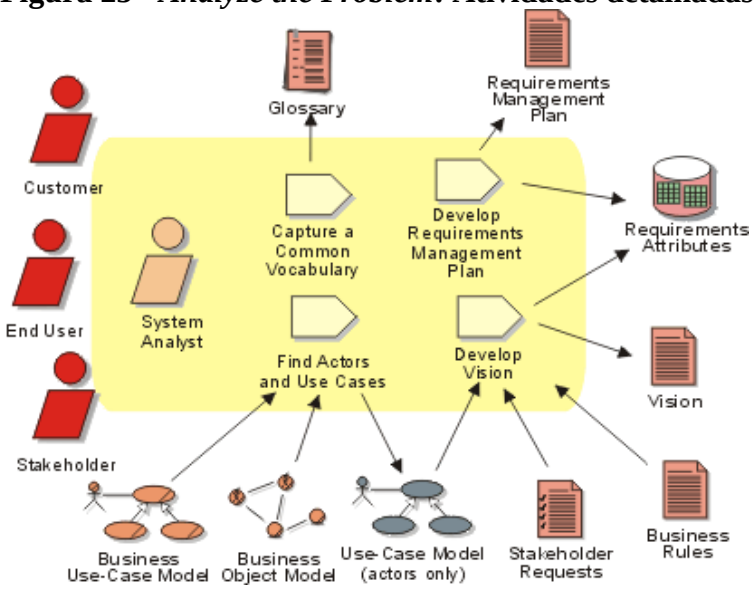
Figura 24 - Diagrama de Atividades da disciplina de Requisitos



Fonte: HEIJSTEK et al., 2008

Como pode ser visto na Figura 24, existem ao todo seis atividades macros envolvendo a disciplina de requisitos. Cada uma dessas atividades macros pode ainda se expandir em um conjunto de atividades de maior grau de detalhe como, por exemplo, as apresentadas na Figura 25.

Figura 25 - *Analyze the Problem*: Atividades detalhadas



Fonte: HEIJSTEK et al., 2008

Ao todo, a disciplina de requisitos possui 23 atividades detalhadas, como mostrado na Tabela 6. Apesar de algumas atividades terem o mesmo nome, elas estão associadas a contextos diferentes e, portanto, envolvem níveis de detalhes distintos e possuem diferentes produtos de trabalho.

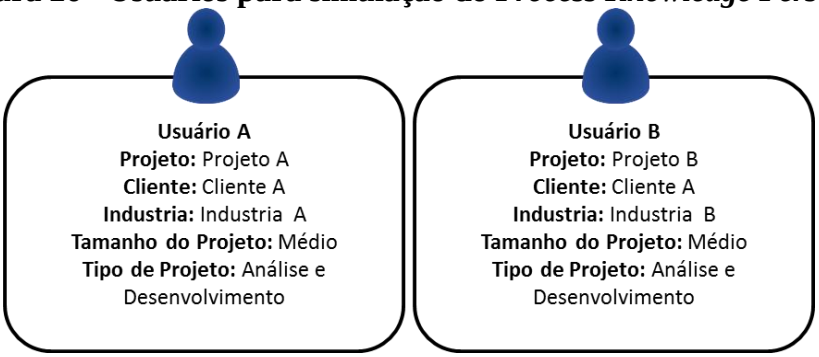
Quadro 4 - Atividades da disciplina de Requisitos do RUP

Disciplina	Atividades Macros	Atividades Detalhadas
Requisitos	Analisar o problema	Capturar um vocabulário comum
		Elaborar o plano de gestão de requisitos
		Encontrar atores e casos de uso
		Desenvolver visão
	Entender as necessidades dos envolvidos	Capturar um vocabulário comum
		Desenvolver visão
		Elicitar requisitos dos envolvidos
		Encontrar atores e caso de usos
		Gerenciar dependências
	Definir o sistema	Desenvolver visão
		Gerenciar dependências
		Capturar um vocabulário comum
		Encontrar atores e casos de uso
	Gerenciar o escopo do sistema	Priorizar casos de uso
		Desenvolver visão
		Gerenciar dependências
	Refinar a definição do sistema	Detalhar casos de uso
		Detalhar requisitos do software
		Modelar interfaces com usuários
		Prototipar as interfaces com o usuário
	Gerenciar mudanças nos requisitos	Estruturar o modelo de caso de uso
		Gerenciar Dependências
		Revisar Requisitos

Fonte: Elaborado pelo Autor

Suponha que quatro usuários do processo RUP tenham utilizado a disciplina de requisitos para os requisitos do produto objetivo de seus projetos. Os dados desses quatros usuários são exibidos abaixo:

Figura 26 - Usuários para simulação do *Process Knowledge Persona*



Fonte: Elaborado pelo Autor

No início do seu projeto, o usuário A visualiza o processo definido pelo RUP e realiza uma instância das atividades da disciplina de requisitos para seu projeto. O *template* definido por esse trabalho é utilizado como input da atividade de instanciação e o seu preenchimento parcial é um dos produtos de trabalho dessa atividade. O *template* parcialmente preenchido pelo usuário A pode ser visualizado na Figura 27:

Figura 27 - *Template* parcialmente preenchido

Nome do Projeto:	Projeto A
Cliente:	Cliente A
Domínio de atuação do Cliente:	Indústria A
Tipo do projeto:	Análise e Desenvolvimento
Tamanho do Projeto:	Médio
Data:	11/03/2012

1. Lista de atividades utilizadas e não adaptadas do processo padrão.
   - *1.1. Requirements*
     - *1.1.1. Analyze the Problem: Develop Requirements Management Plan*
     - *1.1.2. Analyze the Problem: Find Actors and Use Cases*
     - *1.1.3. Analyze the Problem: Develop Vision*
     - *1.1.4. Understand Stakeholder Needs: Develop Vision*
     - *1.1.5. Understand Stakeholder Needs: Elicit Stakeholder Requests*
     - *1.1.6. Understand Stakeholder Needs: Manage Dependencies*
     - *1.1.7. Define the System: Develop Vision*
     - *1.1.8. Define the System: Manage Dependencies*
     - *1.1.9. Manage the Scope of the System: Develop Vision*
     - *1.1.10. Manage the Scope of the System: Manage Dependencies*
     - *1.1.11. Refine the System Definition: Detail the Software Requirements*
     - *1.1.12. Refine the System Definition: Model the User-Interface*

1.1.13. Refine the System Definition: Prototype the User-Interface

2. Lista de atividades utilizadas e adaptadas do processo padrão.
   2.1. Requirements
      2.1.1. Manage the Scope of the System: Prioritize Use Cases
         2.1.1.1. Motivo: Como não será desenvolvido casos de uso, não haverá priorização de casos de uso e sim de requisitos.
3. Lista de atividades não utilizadas do processo padrão.
   3.1. Requirements
      3.1.1. Analyze the Problem: Capture a Common Vocabulary
         3.1.1.1. Motivo: A equipe do projeto já está acostumada a trabalhar com esse cliente e conhece os termos envolvidos com as funcionalidades requeridas. Sendo assim, não é necessário que essa atividade seja executada.
      3.1.2. Understand Stakeholder Needs: Capture a Common Vocabulary
         3.1.2.1. Motivo: A equipe do projeto já está acostumada a trabalhar com esse cliente e conhece os termos envolvidos com as funcionalidades requeridas. Sendo assim, não é necessário que essa atividade seja executada.
      3.1.3. Define the System: Capture a Common Vocabulary
         3.1.3.1. Motivo: A equipe do projeto já está acostumada a trabalhar com esse cliente e conhece os termos envolvidos com as funcionalidades requeridas. Sendo assim, não é necessário que essa atividade seja executada.
      3.1.4. Define the System: Find Actors and Use Cases
         3.1.4.1. Motivo: Optou-se por não utilizar casos de uso durante a definição dos requisitos deste projeto. Acredita-se que essa atividade agregue pouco valor visto que entende-se que os requisitos são simples e claros. Será realizado apenas o detalhamento do requisito.
      3.1.5. Understand Stakeholder Needs: Find Actors and Use Cases
         3.1.5.1. Motivo: Optou-se por não utilizar casos de uso durante a definição dos requisitos deste projeto. Acredita-se que essa atividade agregue pouco valor visto que entende-se que os requisitos são simples e claros. Será realizado apenas o detalhamento do requisito.
      3.1.6. Refine the System Definition: Detail a Use Case
         3.1.6.1. Motivo: Optou-se por não utilizar casos de uso durante a definição dos requisitos deste projeto. Acredita-se que essa atividade agregue pouco valor visto que entende-se que os requisitos são simples e claros. Será realizado apenas o detalhamento do requisito.

Fonte: Elaborado pelo Autor

Durante toda a fase de escopo "*Analyze the Problem*" do projeto, o usuário "A" utiliza o processo instanciado acima para desenvolver as atividades. No final dessa fase, ele pode facilmente perceber a adequação de suas escolhas com relação ao processo e realizar assim um feedback com relação a elas. Dessa forma, ele já é capaz de preencher o *template* por completo como mostrado na Figura 28.

Figura 28 - *Template* totalmente preenchido

Nome do Projeto:	Projeto A
Cliente:	Cliente A
Domínio de atuação do Cliente:	Indústria A
Tipo do projeto:	Análise e Desenvolvimento

Tamanho do Projeto: Médio
Data: 11/03/2012

4. Lista de atividades utilizadas e não adaptadas do processo padrão.
   4.1. Área do processo: *Requirements*
      4.1.1. Analyze the Problem: Develop Requirements Management Plan
      4.1.2. Analyze the Problem: Find Actors and Use Cases
      4.1.3. Analyze the Problem: Develop Vision
      4.1.4. Understand Stakeholder Needs: Develop Vision
      4.1.5. Understand Stakeholder Needs: Elicit Stakeholder Requests
      4.1.6. Understand Stakeholder Needs: Manage Dependencies
      4.1.7. Define the System: Develop Vision
      4.1.8. Define the System: Manage Dependencies
      4.1.9. Manage the Scope of the System: Develop Vision
      4.1.10. Manage the Scope of the System: Manage Dependencies
      4.1.11. Refine the System Definition: Detail the Software Requirements
      4.1.12. Refine the System Definition: Model the User-Interface
      4.1.13. Refine the System Definition: Prototype the User-Interface
5. Lista de atividades utilizadas e adaptadas do processo padrão.
   5.1. Requirements
      5.1.1. Manage the Scope of the System: Prioritize Use Cases
         5.1.1.1. Motivo: Como não será desenvolvido casos de uso, não haverá priorização de casos de uso e sim de requisitos.
         5.1.1.2. Feedback: A priorização baseada em requisitos é possível e representou muito bem as necessidades de urgência do cliente. No entanto é importante que os requisitos estejam claros para que isso funcione.
6. Lista de atividades não utilizadas do processo padrão.
   6.1. Requirements
      6.1.1. Analyze the Problem: Capture a Common Vocabulary
         6.1.1.1. Motivo: A equipe do projeto já está acostumada a trabalhar com esse cliente e conhece os termos envolvidos com as funcionalidades requeridas. Sendo assim, não é necessário que essa atividade seja executada.
         6.1.1.2. *Feedback:* De fato, quando a equipe já está mais acostumado com o contexto do processo de negócio do cliente, essa atividade não é necessária. A não execução dessa atividade não impactou na qualidade do projeto.
      6.1.2. Understand Stakeholder Needs: Capture a Common Vocabulary
         6.1.2.1. Motivo: A equipe do projeto já está acostumada a trabalhar com esse cliente e conhece os termos envolvidos com as funcionalidades requeridas. Sendo assim, não é necessário que essa atividade seja executada.
         6.1.2.2. *Feedback:* De fato, quando a equipe já está mais acostumado com o contexto do processo de negócio do cliente, essa atividade não é necessária. A não execução dessa atividade não impactou na qualidade do projeto.
      6.1.3. Define the System: Capture a Common Vocabulary
         6.1.3.1. Motivo: A equipe do projeto já está acostumada a trabalhar com esse cliente e conhece os termos envolvidos com as funcionalidades requeridas. Sendo assim, não é necessário que essa atividade seja executada.
         6.1.3.2. *Feedback:* De fato, quando a equipe já está mais acostumado com o contexto do processo de negócio do cliente, essa atividade não é necessária. A não execução dessa atividade não impactou na qualidade do projeto.
      6.1.4. Define the System: Find Actors and Use Cases
         6.1.4.1. Motivo: Optou-se por não utilizar casos de uso durante a definição dos requisitos deste projeto. Acredita-se que essa atividade agregue pouco valor visto que entende-se que os requisitos são simples e claros. Será realizado apenas o detalhamento do requisito.

6.1.4.2. *Feedback:* Alguns requisitos possuem complexidade maior e, devido a isso, são mais difíceis de se entenderem e de serem desenvolvidos. Devido a isso, os desenvolvimentos de casos de uso podem ajudar a torna-los mais claros. Para próximos projetos, essa atividade pode ser adaptada de forma que só se aplique a técnica de casos de uso para requisitos mais complexos.

6.1.5. Understand Stakeholder Needs: Find Actors and Use Cases

6.1.5.1. Motivo: Optou-se por não utilizar casos de uso durante a definição dos requisitos deste projeto. Acredita-se que essa atividade agregue pouco valor visto que entende-se que os requisitos são simples e claros. Será realizado apenas o detalhamento do requisito.

6.1.5.2. *Feedback:* Alguns requisitos possuem complexidade maior e, devido a isso, são mais difíceis de se entenderem e de serem desenvolvidos. Devido a isso, os desenvolvimentos de casos de uso podem ajudar a torna-los mais claros. Para próximos projetos, essa atividade pode ser adaptada de forma que só se aplique a técnica de casos de uso para requisitos mais complexos.

6.1.6. Refine the System Definition: Detail a Use Case

6.1.6.1. Motivo: Optou-se por não utilizar casos de uso durante a definição dos requisitos deste projeto. Acredita-se que essa atividade agregue pouco valor visto que entende-se que os requisitos são simples e claros. Será realizado apenas o detalhamento do requisito.

6.1.6.2. *Feedback:* Alguns requisitos possuem complexidade maior e, devido a isso, são mais difíceis de se entenderem e de serem desenvolvidos. Devido a isso, os desenvolvimentos de casos de uso podem ajudar a torna-los mais claros. Para próximos projetos, essa atividade pode ser adaptada de forma que só se aplique a técnica de casos de uso para requisitos mais complexos.

Fonte: Elaborado pelo Autor

O *Knowledge Manager* recebe o documento acima preenchido e faz uma análise dele extraindo os seguintes conhecimentos desse documento:

Figura 29 - Análise do *Knowledge Manager*

Para projetos do cliente A

- O cliente possui maturidade suficiente para trabalhar com níveis de especificações de requisitos mais enxutos. É possível utilizar casos de usos apenas para requisitos de maior complexidade, sendo necessário, para os outros, apenas um texto descritivo.
  - Atividades relacionadas a essa adaptação: *Find Actors and Use Cases, Detail a Use Case*.

Para projetos da Indústria A

- Desde que a equipe já possui conhecimento sobre o processo de negócio do cliente, é possível deixar mais informal as atividades relacionadas ao levantamento de vocabulário.
  - Atividades relacionadas a essa adaptação: Capture a Common *Vocabulary*

Fonte: Elaborado pelo Autor

O usuário B está para começar seu projeto. O cliente do projeto é o mesmo cliente do projeto A que já foi finalizado. Porém, trata-se de um domínio de atuação diferente. É a

primeira vez que o usuário B trabalha com esse cliente e essa indústria e, devido a isso, ele tem dúvidas sobre quais adaptações seriam as melhores no processo. Ao consultar o Knowledge Persona, ele descobre que o cliente A é maduro o suficiente para trabalhar com um nível menos detalhado de requisitos e, ao mesmo tempo, descobre que aquele é o primeiro projeto da indústria B e que, portanto, o ideal é que todas as atividades de levantamento de vocabulário sejam executadas. A partir de tais conhecimentos, ele faz as seguintes customizações no processo padrão:

Figura 30 - *Template* preenchido (Projeto B)

Nome do Projeto:	Projeto B
Cliente:	Cliente A
Domínio de atuação do Cliente:	Indústria B
Tipo do projeto:	Análise e desenvolvimento
Tamanho do Projeto:	Médio
Data:	21/07/2012

1. Lista de atividades instanciadas não adaptadas do processo padrão.
    1.1. Requirements
        1.1.1. Analyze the Problem: Capture a Common Vocabulary
        1.1.2. Analyze the Problem: Develop Requirements Management Plan
        1.1.3. Analyze the Problem: Find Actors and Use Cases
        1.1.4. Analyze the Problem: Develop Vision
        1.1.5. Understand Stakeholder Needs: Capture a Common Vocabulary
        1.1.6. Understand Stakeholder Needs: Develop Vision
        1.1.7. Understand Stakeholder Needs: Elicit Stakeholder Requests
        1.1.8. Understand Stakeholder Needs: Manage Dependencies
        1.1.9. Define the System: Capture a Common Vocabulary
        1.1.10. Define the System: Develop Vision
        1.1.11. Define the System: Manage Dependencies
        1.1.12. Manage the Scope of the System: Develop Vision
        1.1.13. Manage the Scope of the System: Manage Dependencies
        1.1.14. Refine the System Definition: Detail the Software Requirements
        1.1.15. Refine the System Definition: Model the User-Interface
        1.1.16. Refine the System Definition: Prototype the User-Interface
2. Lista de atividades instanciadas e adaptadas do processo padrão.
    2.1. Requirements
        2.1.1. Manage the Scope of the System: Prioritize Use Cases
            2.1.1.1. Motivo: Como não será desenvolvido casos de uso, não haverá priorização de casos de uso e sim de requisitos.
3. Lista de atividades não instanciadas do processo padrão.
    3.1. Requirements
        3.1.1. Define the System: Find Actors and Use Cases
            3.1.1.1. Motivo: Optou-se por não utilizar casos de uso durante a definição dos requisitos deste projeto. Acredita-se que essa atividade agregue pouco valor visto que entende-se que os requisitos são simples e claros. Será realizado apenas o detalhamento do requisito.
        3.1.2. Understand Stakeholder Needs: Find Actors and Use Cases
            3.1.2.1. Motivo: Optou-se por não utilizar casos de uso durante a definição dos requisitos

deste projeto. Acredita-se que essa atividade agregue pouco valor visto que entende-se que os requisitos são simples e claros. Será realizado apenas o detalhamento do requisito.

3.1.3. Refine the System Definition: Detail a Use Case

3.1.3.1. Motivo: Optou-se por não utilizar casos de uso durante a definição dos requisitos deste projeto. Acredita-se que essa atividade agregue pouco valor visto que entende-se que os requisitos são simples e claros. Será realizado apenas o detalhamento do requisito.

Fonte: Elaborado pelo Autor

Como pode ser visualizado, o *Process Konwledge Persona* pode ajudar usuários do processo padrão a customizá-lo para seus projetos com base no conhecimento adquirido em projetos anteriores. O usuário B conseguiu customizar seu processo com mais eficiência, graças ao conhecimento disponibilizado pelo *Process Knowledge Persona*.

A forma com que o conhecimento é disponibilizado para os usuários do processo é um fator importante. Deve ser possível pesquisar pelo conhecimento através de parâmetros tais como domínio de atuação, cliente, tamanho e outros que podem ser estabelecidos de acordo com as necessidades das organizações.

6. AVALIAÇÃO DO MÉTODO PROPOSTO

Com o intuito de avaliar o método proposto por este trabalho, um projeto de desenvolvimento de software real foi utilizado como experimento. Por envolver questões confidenciais, a empresa na qual o projeto se desenvolveu não permitiu que seu nome fosse revelado. De forma semelhante, detalhes relacionados ao projeto e ao processo também foram omitidos por questões de confidencialidade. Dessa forma, será apresentado aqui somente uma visão geral sobre o processo de desenvolvimento de software da empresa e do projeto utilizado para avaliação do método, com ênfase em detalhes relevantes para a avaliação do método.

A avaliação teve como escopo a análise dos seguintes itens:

a) Impacto da utilização do método no contexto de custo.
b) Impacto da utilização do método no contexto de tempo/cronograma
c) Impacto da utilização do método no contexto de qualidade do produto
d) Levantamento de requisitos de usabilidade para arquitetura da ferramenta
e) *Feedback* subjetivo dos envolvidos com relação ao uso do método

A análise dos três primeiros itens será realizada pelo acompanhamento de métricas específicas do projeto, como será melhor explicado nas próximas seções. Os dois últimos itens serão analisados em uma reunião a ser executada no final do projeto na qual será realizado um processo de análise crítica e discutição sobre os resultados.

6.1. O Processo de desenvolvimento de software da empresa

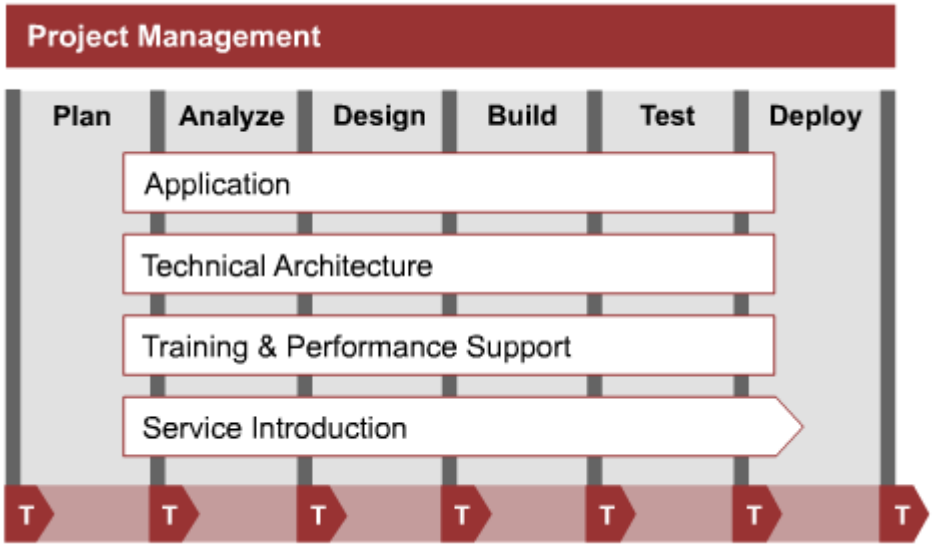
O processo de desenvolvimento de Software da empresa instanciado pelo projeto é um processo de alta maturidade que foi avaliado no CMMI nível 5. O rigor para se obter tal certificação não deixa dúvidas com relação à completude e eficiência de tal processo, porém o torna também mais complexo de ser instanciado, uma vez que ele apresenta uma grande quantidade de atividades, ferramentas e metodologias que fica disponibilizado para uso dos projetos. Tal fato o torna adequado para a utilização da metodologia proposta neste trabalho, já que o julgamento de todos esses itens passa a ser ainda mais trabalhoso e muitos critérios e pontos de vista têm que ser considerados.

Todas as informações apresentadas aqui foram criteriosamente acompanhadas e revisadas por uma equipe da empresa responsável pela manutenção e evolução do processo organizacional.

6.1.1. Ciclo de vida

A figura abaixo representa o ciclo de vida do processo de desenvolvimento de software da empresa. Ele prevê fases e atividades desde o planejamento do projeto até a implantação do sistema desenvolvido.

Figura 31 - Representação do processo de desenvolvimento de software instanciado pelo projeto utilizado para avaliação do método



Fonte: Processo da Empresa

O processo possui seis fases:

a) Planejamento do Projeto: O foco dessa fase é estabelecer o planejamento geral do projeto. É um momento em que se realiza, por exemplo, a instanciação do processo de desenvolvimento de software. Além disso, define-se o planejamento de escopo, financeiro e o cronograma macro do projeto. É uma fase de grande importância para qualquer projeto uma vez que é nela que se define a metodologia que o projeto será utilizada ao longo de seu desenrolar. Em especial, é nesta fase que o método proposto por esse trabalho é aplicado.

b) Análise: O foco desta fase é no detalhamento dos requisitos no contexto funcional. O processo de negócio do cliente é investigado a fundo e requisitos detalhados são estabelecidos utilizando técnicas como casos de uso.

c) Desenho: Nessa fase, os requisitos levantados na fase anterior são estudados e estabelece-se então a solução técnica para a implementação deles. Ao contrário da fase de Análise, que possui um caráter mais funcional, a fase de Desenho possui um caráter mais técnico.
d) Teste: A fase de teste tem como foco o teste individual das funcionalidades implementadas, assim como o teste de seu funcionamento integrado. Um dos principais objetivos dessa fase é garantir que as expectativas do cliente foram atendidas.
e) Implantação: A última fase do ciclo de vida do processo. Essa fase descreve atividades para se realizar a entrega do produto desenvolvido ao cliente. O sistema desenvolvido é então colocado no seu ambiente de execução e passa-se então para um processo de operação assistida, na qual o projeto acompanha/apoia o funcionamento do sistema no cliente por um tempo até que se tenha certeza de seu funcionamento correto.

Apenas considerando as fases expostas acima, tem-se a impressão de que se trata de um processo em cascata, porém existe um diferencial: enquanto cada fase do ciclo de vida estabelece uma execução sequencial, o processo estabelece quatro fluxos horizontais contínuos que se estendem entre as fases. A ideia é que se execute cada uma das fases tendo como base quatro contextos diferentes: Aplicação, Arquitetura Técnica, Suporte a Treinamento e *Performance* e, por fim, Introdução do Serviço. Esses quatro fluxos são executados em paralelo dentro de cada uma das fases.

No final de cada fase, o processo estabelece pontos de decisão que possuem critérios para seguir para a próxima fase.

Em paralelo a todas as fases, tem-se ainda um fluxo horizontal de gestão do projeto. Esse fluxo estabelece atividades, ferramentas e técnicas para que se acompanhe a evolução do projeto, riscos e a qualidade.

6.2. Métricas para avaliação

Por já ter sido alvo de avaliação CMMI nível 5, o processo de desenvolvimento de software em questão estabelece métricas de projeto que, por sua vez, alimentam métricas organizacionais. Essas últimas permitem ter uma visão geral da qualidade dos projetos de desenvolvimento de software como um todo. É estabelecido, por exemplo, métricas como

velocidade média, números de *bugs* médio, entre outras. Essas métricas são ainda estratificadas por tamanho de projeto, podendo este ser pequeno, médio ou grande, dependendo da quantidade e complexidade dos requisitos.

Algumas das métricas organizacionais se estabelecem como parâmetros para se medir a qualidade de um projeto dentro da empresa. Assim que ele é finalizado, ou durante durante sua execução, é possível extrair as mesmas métricas do projeto e comparar com as médias definidas para a organização.

Como a hipótese defendida neste trabalho é a de que a aplicação do método proposto durante a instanciação do processo de desenvolvimento de software permite melhorar a qualidade do processo instanciado para o projeto, espera-se como resultado final que ocorram melhorias significativas em métricas do projeto. Dessa forma, as métricas organizacionais se estabelecem como um parâmetro ideal para validar o método proposto, uma vez que é possível medir a diferença entre elas e as métricas extraídas de um projeto que aplicou a metodologia.

Tendo como base os três primeiros itens do escopo de avaliação estabelecido inicialmente e apresentado no início deste capitulo (análise do impacto da utilização do método no contexto de custo, análise do impacto da utilização do método no contexto de tempo/cronograma e análise do impacto da utilização do método no contexto de qualidade do produto), foi selecionado um conjunto de métricas organizacionais existentes na empresa que serão utilizadas para realizar as análises requeridas. Os valores absolutos das métricas em nível organizacional não serão apresentados por questões confidenciais, porém, como o importante na análise é identificar o impacto do uso do método nessas métricas, será apresentada a diferença percentual dos valores das métricas do projeto e as organizacionais na seção de resultados.

Foi selecionada pelo menos uma métrica para os critérios tempo, qualidade e custo.

Quadro 5 - Métricas organizacionais escolhidas para avaliação do método

Métrica	Objetivo	Descrição	Fórmula
Deliverable Timeliness	Medir a eficiência do processo. Será utilizada para análise do impacto na utilização do método no contexto de tempo/cronograma. Critério associado: Tempo	Essa métrica indica a capacidade do projeto/empresa de entregar seus entregáveis na data planejada.	**Projeto:** # de entregáveis submetidos na data / Total de entregáveis. Entregaveis aqui não se relaciona somente ao que é entregue a clientes, mas a qualquer entregavel planejado. **Empresa:** média dos projetos finalizados.
Peer Review Problem Detection	Medir a qualidade da condução do projeto com base no número de erros e defeitos. Será utilizada para análise do impacto na utilização do método no contexto de qualidade. Critério associado: Qualidade	Determina a qualidade dos produtos de trabalho que passam pela revisão. Todas os entregaveis desenvolvidos (requisitos, desenhos, código, testes e etc.) devem passar pelo processo de revisão em pares.	**Projeto:** Número de problemas detectados em revisão por par / tamanho do entregável. **Empresa:** Média dos projetos concluídos.

Velocidade de desenvolvimento	Capacidade de produção Será utilizada para análise do impacto na utilização do método no contexto de tempo/cronograma. Critério associado: tempo	Essa métrica indica a velocidade do projeto para produzir seus entregáveis. Baseia-se em pontos de função.	Projeto: Média do número de pontos de função construídos no dia pela equipe / 8 horas. Empresa: média dos projetos concluídos.
Variação do custo em porcentagem	Qualidade financeira do projeto Será utilizada para análise do impacto na utilização do método no contexto de custo. Critério associado: Custo	Essa métrica indica a variação do custo do projeto com relação ao planejado. Sendo assim ela reflete a saúde financeira do projeto.	Projeto: (*BCWP (Budgeted Cost of Work Performed*) - *ACWP (Actual Cost of Work Performed*)) /*BCWP* Empresa: Media dos projetos.

Fonte: Elaborado pelo Autor com base no processo da empresa

6.3. O projeto alvo da avaliação

O projeto onde será utilizado o método em questão é considerado de porte médio e envolve ao todo 24 funcionários da empresa que desempenham papéis variados como analista funcional, gerente, desenvolvedor e *tester*. O Projeto possui como objetivo o desenvolvimento de um MES (*Manufactury Execution System*). Tais sistemas são soluções tecnológicas que têm o objetivo de auxiliar no gerenciamento de todas as etapas de produção. A importância destes sistemas vem da lacuna que normalmente existe entre o ERP (*Entreprise Resource Planning*) e os softwares específicos da linha de produção.

O MES pode importar dados do ERP e integrá-los com o dia-a-dia da produção, gerenciando e sincronizando as tarefas produtivas com o fluxo de materiais. Considerando que na cadeia de suprimento o maior valor agregado costuma estar na produção, faz todo sentido investir em sistemas que otimizem o fluxo, controle e qualidade do material.

Estas são algumas das funções que os sistemas MES costumam ter:

a) Importação de dados do sistema ERP: itens, estações de trabalho, armazenagem, estoque, planos da qualidade, dados de funcionários, etc;
b) Importação de parâmetros para a produção, como pedidos e prioridades de manufatura;
c) Emissão automatizada de instruções para que o armazém entregue o material nas células de trabalho;
d) Exibição da fila de trabalho, instruções e documentação específica para a célula de trabalho, em função das prioridades definidas anteriormente;
e) Armazenamento das informações de atividades da produção: tempos de operação (por operador), tempos de máquinas, componentes usados, material desperdiçado, etc;
f) Instruções para reposição de material na linha de produção;
g) Armazenamento e divulgação dos dados de qualidade;
h) Instruções para que a continuidade do fluxo de materiais pela linha;
i) Monitoramento da produção em tempo real e ajustes em todas as etapas, conforme seja necessário;
j) Análise de métricas e desempenho da produção.

Os principais benefícios que podem ser obtidos na implementação do MES são:

a) Redução do desperdício (excesso de produção, tempos de espera, inventário desnecessário, defeitos);
b) Redução dos tempos de produção;
c) Redução dos custos de mão de obra e treinamento;
d) Apoio à manufatura enxuta;
e) Apoio à melhoria contínua;
f) Melhoria da confiabilidade do produto final obtendo melhor qualidade;
g) Aumento da visibilidade das atividades do chão de fábrica, assim como dos custos do processo de manufatura.

6.4. A aplicação do método

A aplicação do método foi orientada pelas fases apresentadas na Figura 17 exibido na seção 5. A equipe responsável por fazer a manutenção no processo de desenvolvimento de software da empresa foi a principal intermediadora da aplicação do método e acompanhamento dos resultados. Foi demostrado um grande interesse por parte desta equipe pelo fato de estarem de acordo com o problema motivador deste trabalho.

Como preparação preliminar, a equipe responsável pelo projeto estimou, em horas ideias, o custo de se executar cada uma das atividades propostas pelo processo. Essa estimativa levou em consideração fatores históricos da organização. A estimativa também se baseou nas características já estabelecidas pela organização para projetos de porte médio. Esse procedimento é essencial para se aplicar o critério de corte definido pelo método proposto neste trabalho.

O processo de desenvolvimento de software inicia-se com a fase de planejamento, que possui como uma das primeiras atividades a instanciação do processo em si. É exatamente essa atividade que é alvo do método proposto neste trabalho.

Em um primeiro momento foi realizado o mapeamento dos papéis presentes no projeto com os propostos pelo método.

Quadro 6 - Mapeamento dos papéis propostos pelo método e os existentes no projeto

Papel proposto pelo método	Papel que desempenhou a função
Gerente do projeto	Gerente Sênior do projeto.
Líder Funcional	Gerente da equipe funcional
Membro do QA	Membro sênior da equipe responsável pela manutenção do Processo
Process Knowledge Persona	Gerente Sênior que não faz parte do projeto.

Fonte: Elaborado pelo Autor

A aplicação do *Process Knowledge Persona,* da forma como é proposta por este trabalho, exige a elaboração de uma recurso virtual e automatizado capaz de armazenar informações em forma de dicas e fornecê-las durante para cada tópico de decisão. A elaboração para tal recurso na empresa levaria tempo e não faz parte do escopo deste trabalho. Porém a validação da teoria proposta era de grande importância e, por isso, optou-se pela

simulação do *Process Knowledge Persona* por meio da participação de um membro na empresa que tivesse um bom conhecimento e experiência sobre projetos passados, que já haviam sidos finalizados. A ideia foi que, a cada atividade em julgamento. essa pessoa relatasse a importância que essa atividade teve em projetos que ela já presenciou.

A equipe decidiu utilizar os mesmos critérios de julgamento e termos linguísticos propostos por este trabalho. Porém, foi manifestada a vontade de se alterar os critérios para melhor se adaptar aos parâmetros da empresa. Tal adaptação às características da empresa é extremamente importante e é incentivada por este trabalho. Por exemplo, a empresa pode escolher trabalhar com variáveis de controle de gestão definidos por outra metodologia.

Os critérios de corte e de consenso também foram preservados conforme o padrão apresentado neste trabalho.

Assim como proposto pelo método, cada uma das atividades do processo foram então analisadas e julgadas perante cada um dos critérios. Ao todo, foram julgadas 111 atividades. Uma versão inicial do programa desenvolvido neste trabalho para suportar o uso desse método foi utilizado para armazenar e manipular as matrizes *fuzzy*. Ao todo foram gerados 3916 produtos cartesianos semelhantes aos mostrados na seção 4.6.1 para cada par de critério e especialista. Com a aplicação de (2) e (3), obteve-se então as matrizes de preferência *fuzzy* de ordem quadrática 111, que mostra a relação do preferência entre cada um dos itens julgados.

O procedimento de agregação foi executado e então o processamento *fuzzy* foi finalizado com a aplicação da correlação (11). O resultado da compilação *fuzzy* dessas matrizes, assim como mostrado no capítulo 4, é um vetor que apresenta, em valor matemático, a preferência consolidada de cada item julgado. Esse resultado, assim como a estimativa de horas para cada item pode ser visto na figura 32 a seguir.

Figura 32 - Resultado da aplicação do método

Legenda	Item
	Estimativa (horas ideais)
	Preferência Fuzzy

1	2	3	4	5	6	7	8	9	10
132,08	1.149,12	94,83	31,41	21,98	32,24	32,24	111,46	163,81	741,71
1	1	1	1	1	1	1	1	1	1

11	12	13	14	15	16	17	18	19	20
103,07	162,75	277,21	219,61	1.917,73	851,63	533,10	1.066,20	575,83	287,77
1	1	1	1	1	1	1	1	1	1

21	22	23	24	25	26	27	28	29	30
184,81	39,62	185,13	108,00	88,71	23,91	42,43	38,57	6,17	26,24

1	0,75	0,75	0,75	0,75	0,75	0,75	0,75	0,75	0,75

31	32	33	34	35	36	37	38	39	40
21,98	29,31	8,79	138,61	89,76	68,76	32,46	25,60	36,00	11,75
0,75	0,75	0,75	0,75	0,75	0,75	0,75	0,75	0,75	0,75

41	42	43	44	45	46	47	48	49	50
15,67	11,75	15,67	39,98	58,64	58,64	39,98	53,31	15,99	23,91
0,75	0,75	0,75	0,75	0,75	0,75	0,75	0,75	0,75	0,75

51	52	53	54	55	56	57	58	59	60
31,09	76,73	107,42	24,55	73,66	24,55	323,42	5,33	5,33	251,62
0,75	0,75	0,75	0,75	0,75	0,75	0,75	0,75	0,75	0,75

61	62	63	64	65	66	67	68	69	70
21,32	13,56	27,12	17,99	53,96	88,71	88,71	26,24	31,41	31,41
0,75	0,75	0,75	0,75	0,75	0,5	0,5	0,5	0,5	0,5

71	72	73	74	75	76	77	78	79	80
31,41	31,41	5,95	3,81	31,46	25,60	25,60	25,60	25,60	58,64
0,5	0,5	0,5	0,5	0,5	0,5	0,5	0,5	0,5	0,5

81	82	83	84	85	86	87	88	89	90
36,00	36,00	23,91	6,40	25,24	38,57	23,91	31,41	36,00	63,10
0,5	0,5	0,5	0,5	0,5	0,25	0,25	0,25	0,25	0,25

91	92	93	94	95	96	97	98	99	100
25,24	63,10	25,24	50,48	60,08	42,92	103,00	17,17	42,92	85,84
0	0	0	0	0	0	0	0	0	0

101	102	103	104	105	106	107	108	109	110
42,92	25,75	77,25	120,17	42,92	34,33	60,08	77,25	8,58	17,17
0	0	0	0	0	0	0	0	0	0

111
504,80
0

Fonte: Elaborado pelo Autor

Por fim, com base nas estimativas estabelecidas em cada uma das atividades do processo, foi realizado o corte. Os itens em com borda vermelha na figura 32 foram os selecionados com base no orçamento disponível do projeto. Como resultado final, apenas 67 atividades ao todo foram selecionadas para serem utilizadas no projeto.

6.5. Resultados

Os resultados apresentados aqui se dividem em duas partes: Análise dos resultados obtidos referentes às métricas expostas na subseção 6.2 e *feedbacks* coletados na reunião de análise critica do projeto com relação ao método utilizado.

6.5.1. Análise das Métricas

Como comentado anteriormente, quatro métricas do projeto foram acompanhadas com o intuito de realizar a avaliação do método proposto por esse trabalho: *Deliverable Timeliness, Peer Review Problem Detection,* Velocidade de desenvolvimento e Variação do custo em porcentagem.

A tabela abaixo apresenta a diferença percentual entre o valor da métrica organizacional (média dos projetos já finalizados) e o valor das métricas do projeto no seu término.

Tabela 4 - Diferença em percentual dos valores das métricas a nível organizacional e os obtidos pelo projeto

Métrica	Resultado
Deliverable Timeliness	Aumento em 19% na eficiência de entrega.
Peer Review Problem Detection	7% mais defeitos detectados nas revisões em pares.
Velocidade de desenvolvimento	Ganho de 3% na velocidade de desenvolvimento
Variação do custo em porcentagem	Variação 21% menor

Fonte: Elaborado pelo Autor

Como pode ser visto, obteve-se uma melhoria significativa com relação á capacidade de entrega no prazo do projeto e na capacidade do projeto em manter seu planejamento financeiro inicial. A velocidade de desenvolvimento não apresentou ganhos significativos, por outro lado o número de problemas encontrados durante as revisões em pares foi 7% maior.

Com relação à métrica *Deliverable Timeliness, da* lista de 78 entregaveis envolvidos com as atividades julgadas, 21 foram descartados devido às escolhas realizadas. Isso possibilitou que o projeto focasse seu esforço somente em produtos de trabalho que realmente foram julgados importantes no ponto de vista da equipe de especialistas que executou a instanciação do processo. Por outro lado, atividade de acompanhamento de cronograma e de monitoramento das responsabilidades do projeto forma mantidas garantindo que o projeto tivesse condições de manter seu compromisso com os entregaveis planejados. Tais fatos provavelmente justificam o aumento da eficiencia da entraga do projeto em 19%.

O aumento no número de problemas encontrados durante as revisões em pares do projeto indicam uma pequena queda na qualidade dos produtos de trabalho. Observou-se que houve uma concentração de detecção desses problemas principalmente em revisões relacionadas ao código desenvolvido. Ao confrontar tal fato com relação ás escolhas de

atividades realizadas durante a instanciação, foi possivel perceber que algumas atividades de suporte ao desenvolvimento foram cortadas. Por exemplo, não foi desenvolvido para este projeto um documento que apresentasse os padrões de desenvolvimento de código com as boas práticas definidas pelo projeto. Por outro lado, todas as atividades relacionadas a peer review foram mantidas permitindo que o projeto detectasse tal problema e o contornasse..

As atividades relacionadas à gestão financeira foram mantidas durante a instanciação e a diminuição do número de atividades instanciadas e, consequentemente, no número de entregaveis permitiu que o projeto tivesse um maior controle com relação ao fator financeiro. Mesmo tendo problemas com relação ao números de defeitos, as atividades de teste selecionadas permitiram que os mesmos fossem detectados em fases iniciais trazendo pouco impacto financeiro para o projeto.

De forma geral, as diferenças percentuais obtidas apontam numericamente para um ganho significativo na gestão da qualidade do projeto como um todo quando comparado com as médias organizacionais.

6.5.2. Reunião de *Feedback*

Após a finalização do projeto, foi realizado uma reunião de análise crítica com os envolvidos para análise da aplicação do método proposto como um todo. A reunião teve como objetivo realizar uma análise crítica sobre a utilização do método, tendo como foco a análise de seus benefícios, melhorias detectadas e requisitos de usabilidade.

A seguir são apresentados os principais *feedbacks* fornecidos pelos membros que participaram da instanciação do processo de desenvolvimento de software.

a) A mecânica de funcionamento do método proposto faz com que se analise com maior grau de critério cada uma das atividades propostas pelo processo. Isso ajuda não só a tomar decisões melhores, mas também obriga aos membros participantes a refletir sobre como o projeto irá agir perante problemas/casos específicos que serão enfrentados. No início do projeto, é muitas vezes difícil ter essa visão. Passar pelas atividades do processo acaba ajudando a refletir melhor em como o projeto será conduzido.
b) O método certamente aumenta a assertividade das escolhas das atividades do processo, porém isso tem um custo. É claramente perceptível que a atividade de instanciação acaba consumindo um tempo maior do que gasta quando não

se utiliza o método. A existência de uma ferramenta apropriada pode melhorar esse tempo.

c) O diferencial do método é que ele permite que as opiniões sejam expressas com incertezas. Fica mais natural trabalhar assim e as incertezas são expressas na composição do resultado final, permitindo que se execute uma ordenação de importância mais real.

d) O fato de se analisar cada uma das atividades do processo frente a diferentes critérios permite que os membros reflitam sobre pontos importantes que todos os projetos deveriam refletir na fase inicial. Muitas vezes pensar em qualidade, orçamento, tempo de entrega, escopo e outros critérios, tudo ao mesmo tempo, torna-se difícil. O método permite que se realize essa análise em etapas, sem que se perca um resultado agregado no final.

A necessidade de uma ferramenta com boa usabilidade foi claramente colocado como um fator importante para a aplicação do método. Tal fato já era previsto por esse trabalho. Com a exposição de tal necessidade, foi conduzido então um processo de levantamento de requisitos para a elaboração da arquitetura de tal ferramenta. Tal levantamento será apresentado na próxima seção deste trabalho.

6.6. Considerações Finais

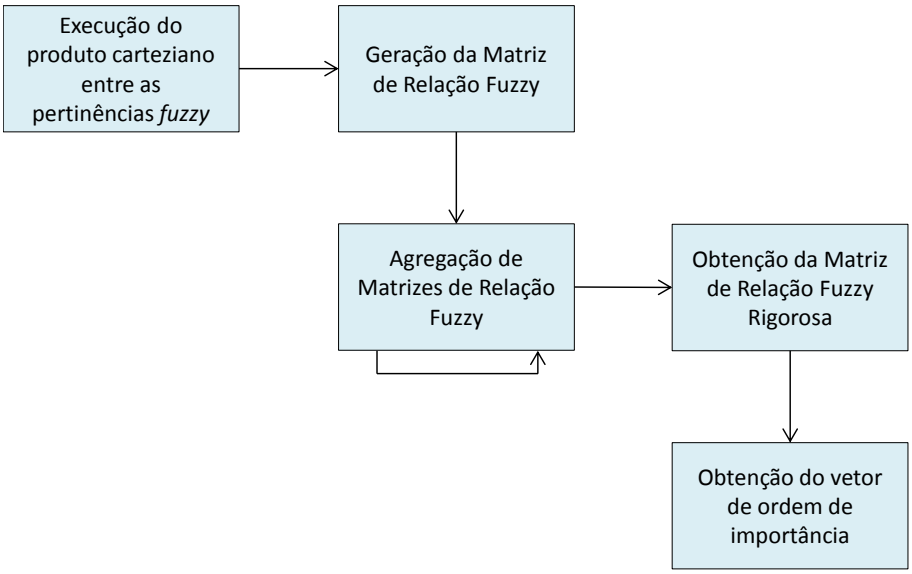
É importante ressaltar que apenas esse estudo de caso não permite afirmar que os objetivos do método foram alcançados, porém o *feedback* fornecido e as métricas recolhidas apontam para uma direção condizente com os objetivos. Para se chegar a uma conclusão mais precisa, seria necessário replicar esse teste em vários projetos e acompanhar a sua evolução de forma a poder realizar as medições necessárias. Infelizmente tal procedimento requer um tempo consideravelmente maior de experimento e disponibilidade de uma gama maior de projetos.

A elaboração de uma ferramenta mais avançada para a aplicação do método também se monstrou um fator importante e será considerado com um dos trabalhos futuros prioritários e deve se basear nos requisitos aqui coletados. A análise desses requisitos é apresentada na próxima seção deste trabalho.

7. VISÃO GERAL DA FERRAMENTA PROPOSTA

Como exposto no capítulo 4, a aplicação do método proposto neste trabalho envolve uma série de operações matemáticas até que se chegue ao vetor resultante que apresenta as alternativas ordenas por importância. A figura abaixo basicamente resume as operações executadas considerando como entrada a opinião de dois especialistas.

Figura 33 - Processo para obtenção do vetor resultante da técnica de decisão multicritério em grupo apoiada pela lógica *fuzzy*



Fonte: Elaborado pelo Autor

A execução dessas operações pode se tornar algo complexo, ainda mais considerando o número de critérios e ativos julgados dentro do contexto de instanciação do processo de desenvolvimento de software. Devido a isso, foi ainda desenvolvido neste trabalho uma ferramenta com o objetivo de automatizar a execução dessas operações utilizando a linguagem de programação Java como mostrado na figura 34.

Figura 34 - Ferramenta para suporte nas operações *fuzzy*

Fonte: Elaborado pelo Autor

Cada uma das operações desenvolvidas na ferramenta recebe como entrada um arquivo de extensão TXT com dados obtidos na fase anterior e executa o processamento matemático de forma a gerar como saída um novo arquivo TXT com a operação realizada. A Tabela 10 a seguir descreve com maior detalhe a entrada e a saída de cada operação.

Quadro 7 - Funcionalidades da ferramenta desenvolvida

<table>
<tr><th>Operação</th><th>Entrada</th><th>Saída</th></tr>
<tr><td>Gerar Produto Cartesiano</td><td>Arquivo txt contendo duas linhas. Cada uma das linhas é na verdade um vetor que representa o conjunto de funções de pertinência fuzzy dado o julgamento do especialista. Os valores deve ser separados por espaço.
Ex.:
0,5 0,75 0,25 0,5
0,25 1 0,5 0,75</td><td>A saída da operação é um novo arquivo txt contendo o produto cartesiano entre os dois vetores levando em consideração o operador min. Para o exemplo de entrada, seria obtido algo semelhante ao exibido abaixo.
<table>
<tr><td></td><td>0,5</td><td>0,75</td><td>0,25</td><td>0,5</td></tr>
<tr><td>0,25</td><td>0,25</td><td>0,25</td><td>0,25</td><td>0,25</td></tr>
<tr><td>1</td><td>0,5</td><td>0,75</td><td>0,25</td><td>0,5</td></tr>
<tr><td>0,5</td><td>0,5</td><td>0,5</td><td>0,25</td><td>0,5</td></tr>
<tr><td>0,75</td><td>0,5</td><td>0,75</td><td>0,25</td><td>0,5</td></tr>
</table></td></tr>
</table>

Gerar Matriz de Relação Fuzzy	Arquivo txt contendo os produtos cartesianos de cada um dos pares de comparação dos itens julgados. Esse arquivo txt pode ser montado agregando-se manualmente os resultados gerados a partir da operação anterior.	Novo arquivo txt contendo os valores da matriz de relação *fuzzy*. Cada linha do arquivo txt representa uma linha da matriz. Os números são separados por espaço.
Agregar Matrizes de Relação Fuzzy	Na operação anterior, obteve-se a matriz de relação *fuzzy* para um especialista. No entanto, é necessário que se repita o procedimento para cada especialista e depois que se realize a agregação das matrizes geradas. A entrada dessa operação é um arquivo txt contendo várias matrizes de relação *fuzzy* escritas no mesmo formato gerado na operação anterior. Esse arquivo txt pode ser montado agregando-se manualmente os resultados gerados utilizando a operação anterior.	O resultado dessa operação é um novo arquivo txt com a matriz agregada. A operação de agregação utilizará o operador *mim*.
Gerar Matriz de relação Fuzzy Rigorosa	Como visto no capítulo 4, a transformação da Matriz de relação *fuzzy* em sua versão rigorosa é um importante passo no processo. Essa funcionalidade da ferramenta permite executar esse procedimento. A entrada da operação é um arquivo txt contendo a matriz em sua forma não rigorosa.	O resultado dessa operação é um novo arquivo txt contendo a matriz de relação fuzzy escrita na sua forma rigorosa conforme procedimento mostrado no capítulo 4.

Obter Vetor de Importância	A partir do arquivo txt contendo a matriz de relacionamento *fuzzy* rigorosa, essa funcionalidade executa as operações matemáticas necessárias para se obter o vetor final de onde se pode verificar a ordem de importância dos elementos julgados com base na opinião dos especialistas.	Um arquivo txt contendo o vetor final, de onde se pode verificar a ordem de importância dos elementos julgados com base na opinião dos especialistas.

Fonte: Elaborado pelo Autor

Apesar de ser útil para executar o procedimento e verificar os resultados passo a passo, a ferramenta desenvolvida não apresenta uma interface adequada para ser utilizada por um usuário leigo. Para o usuário final, não é importante, por exemplo, verificar o produto cartesiano formado ou a matriz de relação *fuzzy*. Somente o resultado final é realmente relevante. No entanto, a partir da elicitação de requisitos realizada durante a reunião de análise crítica do método, foi possível identificar outras funcionalidades desejáveis que poderiam ser agregadas à ferramenta. Nas próximas seções será apresentada uma breve análise desses requisitos, com o objetivo de subsidiar uma futura melhoria da ferramenta.

7.1. Elicitação dos requisitos

No capitulo 7, foi apresentada uma lista de necessidades levantadas junto a possíveis usuários. Este trabalho propõe que tais requisitos sejam subdivididos nos módulos apresentados na Tabela 11:

Quadro 8 - Módulos de requisitos

Módulo	Descrição
Dados Mestres	São requisitos relacionados a informações que devem ser fornecidas ao nível organizacional, ou seja, dados básicos que são manipulados para gerar outras informações (Dados Mestres). Todas as funcionalidades são praticamente de cadastro e, no momento em que o projeto utiliza o software, tais informações já devem estar disponíveis para manipulação.

Execução	São requisitos relacionados a execução do método dentro do projeto. Na sua maioria são requisitos relacionados ao julgamento e processamento matemático das atividades do processo.
Relatórios	São requisitos relacionados a funcionalidades que serão utilizadas após a execução do método. Possuem como objetivo dar visibilidade ao resultado final, fornecendo um *feedback* sobre os resultados.

Fonte: Elaborado pelo Autor

Tais requisitos podem ser divididos entre funcionais e não funcionais. A Tabela 12 apresenta a lista dos requisito e seus respectivos modulo separados entre funcionais e não-funcionais:

Quadro 9 - Mapeamento dos requisitos e Módulos

Tipo	Descrição	Modulo
Funcinonal	Deve ser possível cadastrar, editar e excluir processos de desenvolvimentos de software.	Dados Mestres
Funcinonal	Deve ser possível cadastrar, editar e excluir atividades do processo.	Dados Mestres
Funcinonal	Deve ser possível cadastrar/editar e excluir critérios de julgamento	Dados Mestres
Funcinonal	Deve ser possível cadastrar, editar e excluir grupos de termos linguísticos.	Dados Mestres
Funcinonal	O Sistema deve exibir as atividades do processo em uma espécie de grafo e o usuário poderá navegar nesse grafo para selecionar a atividade que deseja julgar.	Dados Mestres
Funcinonal	Deve ser possível cadastrar, editar e excluir descrições sobre as atividades do processo.	Dados Mestres
Funcinonal	Cada membro da equipe deve ser capaz de informar individualmente suas opiniões sobre cada uma das atividades do processo.	Execução
Funcinonal	Deve ser possível cadastrar, editar e excluir membros da equipe que irão participar do julgamento	Execução

Funcinonal	O sistema deve apresentar um controle de estados indicando se todos os membros da equipe já realizaram seus julgamentos. Somente quando todos tiverem finalizado, o sistema deve liberar a funcionalidade de compilação das opiniões	Execução
Funcinonal	No grafo de atividades, deve existir um sistema de status que indica se a atividade já foi julgada ou não pelo usuário. Os estados devem estar relacionados ao usuário logado (status individual).	Execução
Funcinonal	No momento do julgamento de uma atividade, o usuário deve selecionar com facilidade o critério que deseja julgar e o seu julgamento em si através dos termos linguísticos.	Execução
Funcinonal	Deve ser apresentado um status de julgamento por critério	Execução
Funcinonal	O Sistema deve permitir a seleção de métodos de consenso	Execução
Funcinonal	O Sistema deve permitir a inserção do orçamento disponível pelo projeto para gasto com processo.	Execução
Funcinonal	O resultado da compilação deve ser disponibilizado para todos os envolvidos por e-mail.	Execução
Funcinonal	O sistema deve executar automaticamente o algoritmo de consenso para cada item de julgamento logo após este ser julgado por todos os envolvidos. Deve existir um status de consenso.	Execução
Funcinonal	Toda a matemática *fuzzy* deve ser transparente para o usuário. Porém, as matrizes geradas devem ser armazenadas temporariamente para questões de verificação e manutenção do sistema.	Execução
Funcinonal	Deve ser possível realizar o cadastro de projetos. O Sistema deve apresentar um tutorial aos usuários de novos projetos indicando quais os próximos passos devem ser executados.	Execução
Funcinonal	O sistema deve gerar um relatório final informando quais atividades deverão ser utilizadas e quais não serão utilizadas.	Encerramento
Funcinonal	O Sistema deve permitir que os envolvidos cadastrem *feedback* de atividades do processo. Esses *feedbacks* devem ficar disponíveis para consulta no momento do julgamento	Encerramento

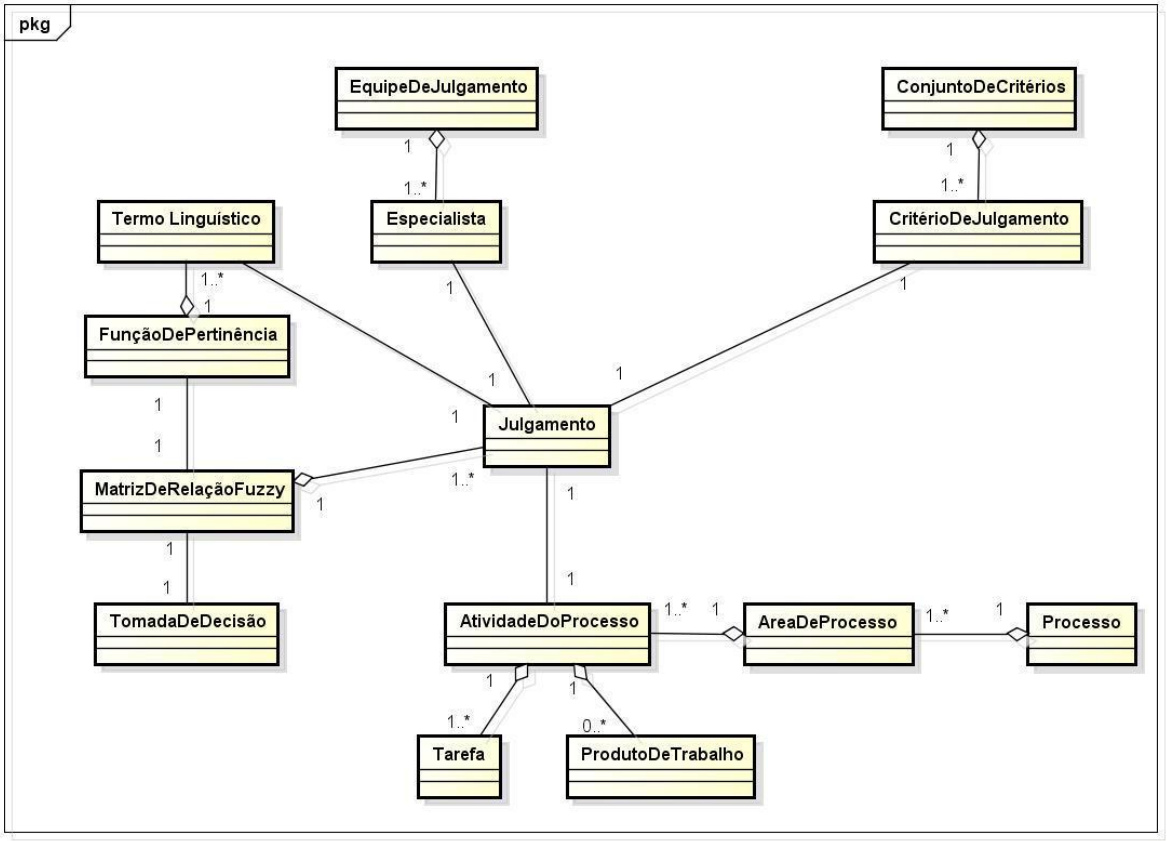
Não Funcionais	É importante que o sistema seja WEB e que possa ser acessado via um controle de usuários cadastrados	Todos
Não Funcionais	O sistema deve ser rápido, mesmo durante os processamentos matemáticos	
Não Funcionais	Deve existir a opção de linguagem inglês e português no sistema	
Não Funcionais	É importante que o sistema seja amigável para o usuário, ou seja, as informações devem ser de fácil acesso e deve ser fácil e intuitivo executar as atividades.	

Fonte: Elaborado pelo Autor

7.2. Arquitetura de Implementação

Considerando as funcionalidades associadas ao sistema, o diagrama de classe apresentado na Figura 35 é proposto para a implementação do software em questão:

Figura 35 - Diagrama de classe: relacionamento entre as entidades do sistema



Fonte: Elaborado pelo Autor

Basicamente, o elemento chave do sistema seria o objeto Julgamento. Este objeto seria formado por outros quatro objetos: Termo Linguístico, Especialista, CritérioDeJulgamento e AtividadeDoProcesso. A agregação de todos os julgamentos forma as matrizes manipuláveis de Relações *Fuzzy.* A partir daí, todo o processo matemático pode ser executado para se obter o resultado final.

O usuário poderá escolher um projeto, um processo, um conjunto de critérios, um método de Consenso e, por fim, um conjunto de termos linguísticos para inicializar o processo de julgamento. Para tal, a ferramenta deve permitir o CRUD (*Create, Retreave, Update, Delete*) de cada um desses elementos.

Uma vez que essas escolhas tenham sido feitas, o julgamento de cada um dos envolvidos é iniciado. Toda a lógica de julgamento deverá ser implementada pela classe "Julgamento". Essa classe também seria responsável por garantir a integridade dos julgamentos realizados, tendo as operações do método de consenso implementadas.

Basicamente, a ferramenta já implementada neste trabalho corresponde à classe MatrizDeRelaçãoFuzzy e parte da classe TomadaDeDecisão. Em outras palavras, grande parte do módulo matemático presente neste projeto de sistema já se encontra implementado.

8. CONCLUSÕES

Este trabalho teve como objetivo desenvolver um método para a execução da atividade de instanciação do processo de desenvolvimento de software, baseado na técnica de decisão multicritério, na lógica *fuzzy* e na gestão de conhecimento, com o intuito de aumentar a justificativa e eficiência real da escolha da configuração das atividades relacionadas ao processo de software organizacional.

Para alcançar tal objetivo, foi realizado um levantamento bibliográfico sobre processos de desenvolvimento de software, modelos de maturidade de processos, o problema de decisão multicritério em grupo, lógica *fuzzy* e, por fim, gestão de conhecimento. Estes foram os pilares para a construção da metodologia apresentada nesse trabalho. O estudo sobre processos de desenvolvimento de software e modelos de maturidade de processos tiveram como objetivo embasar com maior rigor o conhecimento sobre o contexto para o qual o método seria desenvolvido. Já o estudo sobre a problemática da decisão multicritério em grupo, a lógica *fuzzy* e a gestão de conhecimento tiveram como objetivo embasar o método em si.

Após o levantamento bibliográfico, a construção do método foi iniciada. O processo para ser aplicado foi dividido em dois níveis: organizacional e projeto. No nível organizacional, é necessário realizar adaptações para que a aplicação do método no nível de projeto seja efetivo. Essas adaptações estão principalmente relacionadas à gestão de conhecimento e disponibilização de ferramentas para que o projeto possa executar o método. No nível de projeto, o método é aplicado durante a instanciação do processo de desenvolvimento de software organizacional seguindo as diretrizes estabelecidas pela organização e características específicas do projeto.

O método elaborado foi aplicado e validado em um projeto real de desenvolvimento de software. A empresa noa qual o método foi validado possui um conjunto de métricas organizacionais que são as médias das métricas internas dos projetos já executados. Dessa forma, a empresa consegue avaliar a *performance* de novos projetos. Ao compar as métricas organizacionais com os resultados do projeto no qual o método foi executado, foi possível identificar um aumento em 19% na eficiência de entrega, ganho de 3% na velocidade de desenvolvimento, uma variação 21% menor no custo do projeto com relação ao planejado e 7% mais defeitos detectados nas fases de teste.

Apesar do fato dessa única experiência não ser suficiente para afirmar com certeza que os objetivos do método foram alcançados, o *feedback* fornecido pelos envolvidos e as métricas coletadas durante a validação apontam para uma direção condizente com os objetivos propostos.

8.1. Principais Contribuições

Dentre as principais contribuições deste trabalho, podem ser citadas:

a) Aumento da justificativa das escolhas e da eficiência das decisões durante a atividade de instanciamento de projeto;
b) Evolução no processo de gestão de conhecimento organizacional com a elaboração da metodologia do *Process Knowledge Persona*;
c) Associação da lógica *fuzzy* com instanciação de processo;
d) Elaboração de uma ferramenta capaz de suportar as operações *fuzzy* relacionadas com o processo de tomada de decisão;
e) Levantamento de requisitos funcionais e não funcionais para o desenvolvimento de uma ferramenta avançada de apoio à aplicação do método.

8.2. Trabalhos Futuros

Como visto no Capítulo 7, a aplicação da metodologia proposta vai se tronando mais complexa quanto maior for o conjunto de alternativas que se deseja julgar assim como quanto maior for o número de critérios que se deseja utilizar. Para lidar com esse problema, o desenvolvimento de uma ferramenta com interface mais amigavel com usuário para auxiliar na aplicação do método proposto seria o principal trabalho futuro. A base para o desenvolvimento dessa ferramenta (requisitos, arquitetura e protótipo) foram propostos por este trabalho.

Além disso, o método apresentado pode ser estendido para outros problemas relacionados à decisão presentes no ciclo de vida do processo padrão de desenvolvimento de software de uma organização. Em especial, nota-se um grande potencial para os processos de gestão de configuração e gestão de mudanças.

Outro fundamental trabalho futuro é a realização de novas aplicações do método proposto, a fim de validar sua eficácia e contribuir para o seu melhoramento.

REFERÊNCIAS

ABECKER, A. et al. Toward a Technology for Organizational Memories, **IEEE Intelligent Systems**, v. 13 n. 3, p. 40-48, May/June 1998.

ALAVI, M.; LEIDNER, D. Review: Knowledge management and knowledge management systems: conceptual foundations and research issues. **MIS Quarterly**, v. 25. n. 1, p. 107-136, March, 2001.

BAAS, Sjoerd M.; KWAKERNAAK, Huibert. Rating and ranking of multiple-aspect alternatives using fuzzy sets. **Automatica**, v. 13 p. 1:47--58, 1977.

BASILI, V.; ROMBACH, H.D., The TAME Project: Towards Improvement-Oriented Software Environments, **IEEE Transactions on Software Engineering**, v. 14 p. 6, June 1988.

BELANI, H. et al. RUP-based process model for security requirements engineering in value-added service development**,** Software Engineering for Secure Systems. **ICSE Workshop**, v.23 , n. 4, p 54-60, May 2009

BERGER, Patricia M., **Instanciação de Processos de Software em Ambientes Configurados na Estação TABA**. 2003. Dissertação (Mestrado) - Universidade Federal do Rio de Janeiro, Programa de Pós-Graduação em Educação, Rio de Janeiro.

BROOMÉ, M.; RUNESON, P., Technical Requirements for the Implementation of an Experience Base, in Proc. 11th Int. Conference on Software Engineering and Knowledge Engineering , **SEKE'99**, Kaiserslautern, Germany, 1999.

CERPA, Narciso; VERNER, June M. Why did your project fail?, **ACM 52**, v. 12, n.1 p. 130-134, December 2009.

CHEN, Shu-Jen J.; HWANG, C. L.. **Fuzzy Multiple Attribute Decision Making:** Methods and Applications. 1 ed., New York: Springer-Verlag, 1992.

CHENG, C.; Simple fuzzy group decision making method**, IEEE International Conference Fuzzy Systems**, v. 2, p. 66-71, June 1999.

CHICLANA, F.; HERRERA, F.; VIEDMA, E.. Integrating three representation models in fuzzy multipurpose decision making based on fuzzy preference relations. ***Fuzzy Sets Syst***, v. 97, n.1, p. 33 – 48, July 1998.

CHRISSIS, Mary Beth; KONRAD, Mike; SHRUM, Sandy. **CMMI Guidlines for Process Integration and Product Improvement**., 3ed. Boston: Addison-Wesley Longman Publishing Co., Inc., 2003.

COELHO, C.C., **MAPS - Um Modelo para Instanciação de Processos de Software**, 2002. Dissertação (Mestrado), Universidade Federal de Pernambuco, Centro de Informática, Recife.

CORBIN, R.; DUNBAR, C.; ZHU, Q. A three-tier knowledge management scheme for software engineering support and innivation. **Journal of Systems and Software**, v. 80, n. 9, p. 1494-1505, 2007.

COSTA, Anderson J. S., Gerência Flexível de Processos de Software com o Ambiente WebAPSEE, In Simpósio Brasileiro de Engenharia de Software, 20, 2006, Florianópolis.: **Anais de Informática.** Florianópolis**: UFSC**, 2006, p. 10-17.

DIENG, R. et al., Methods and Tools for Corporate Knowledge Management, **Int. Journal of Human-Computer Studies**, v. 51, n. 3, p. 567-598, 1999.

DUBOIS, Didier. ***Fuzzy Sets and Systems: Theory and Applications*.** 2 ed., Orlando, FL, USA: Academic Press, Inc., 1997.

EKEL, P. ; PARREIRAS, R. O.. **Esquema de Consenso para Decisão em Grupo por Meio de Avaliações Lingüísticas**. In: XL Simpósio Brasileiro de Pesquisa Operacional, João Pessoa - Pb. Anais do XL Simpósio Brasileiro de Pesquisa Operacional. Rio de Janeiro - RJ, 2008. p. 1-12.

EKEL, P.. Methods of decision making in fuzzy environment and their application, **IEEE Transmission and Distribution Conference**, v. 2, p. 979-990, 2001.

EKEL, P; PEDRYCZ, W.; SCHINZINGER, R.. Methods of multcriteria decision making in fuzzy environment and their applications, In: International Conference of the North American, **Fuzzy Information Processing Society,** v. 1, p. 625-629, 1999.

FALBO, R. A., **Integração de Conhecimento em um Ambiente de Desenvolvimento de Software**, 1998. Tese (Doutorado), Universidade Federal do Rio de Janeiro, Programa de Pós-Graduação em Educação, Rio de Janeiro.

FALBO, R.; BORGES, L.; VALENTE, F. Using knowledge management to improve software process performance in a CMM level 3 organization. In: Proceedings of the 4th International Conference on Quality Software. **IEEE Computer Society**, Braunschweig, Germany. Washington, 2004, p. 162-169.

FAYAD, Mohamed E.. Software development process: a necessary evil, ACM 40, v. 9, p. 101-103, September 1997.

FUGGETTA, Alfonso**.** Software process: a roadmap. In: Conference on The Future of Software Engineering, 4, 2000, New York. **ICSE '00**, New York: ACM , 2000. p. 25-34.

GOTTSCHALK, P. **Strategic Knowledge Management Technology**. 2 ed. London: Idea Group, 2005.

HEIJSTEK, Werner and CHAUDRON, Michel R.V., Exploring effort distribution in RUP projects. In: Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement, **ACM ESEM '08**, New York, NY, USA. 2008.

HERRERA, F.; HERRERA-VIEDMA, E.; VERDEGAY, J. L.. Direct Approach Processes in Group Decision Making Using Linguistic OWA Operators. **Fuzzy Sets and Systems**, v. 79, p. 175--190, 1994.

ISO. **ISO/IEC 12207:2008 Systems and software engineering - Software life cycle processes**., 2008. www.iso.org

KUKKO, M.; HELANDER, N.; VIRTANEN, P.; , Knowledge Management in Renewing Software Development Processes, In Hawaii International Conference on System Sciences, **41st Annual** , v. 1, n. 1, p. 7-10, Jan 2008.

LEE, H.; CHOI, B. Knowledge management enablers, processes and organizational performance: an integrative view and empirical examination. **Journal of Management Infromation Systems**, v. 20, n. 1, p. 179-228, 2003.

LU, Jie; ZHANG, Guangquan; RUAN, Da. **Multi-Objective Group Decision Making:** Methods, Software and Applications with Fuzzy Set Techniques**. 6 ed.** London, UK: Imperial College Press, 2007.

MACHADO, L.F.C., **Modelo para Definição, Especialização e Instanciação de Processos de software na Estação TABA**, 2000. Dissertação (Mestrado) - Universidade Federal do Rio de Janeiro, Programa de Pós-Graduação em Educação, Rio de Janeiro.

MONTANA. J. The legal system and knowledge management. **The Information Management Journal**, v. 34, n. 3, p. 54-57, 2001.

MONTONI, Mariano Angel; ROCHA, Ana Regina; WEBER, Kival Chaves. MPS.BR: a successful program for software process improvement in Brazil. **Software Process**, v. 14, n. 5, p. 289-300, September 2009.

NESS, J. and HOFFMAN, C., **Putting Sense Into Consensus:** Solving the Puzzle of Making Team Decisions. 1 ed. Tacoma: VISTA Associates, 1998.

NONAKA, I.; UMEMOTO, K.; SENOO, D. From information processing to knowledge creation: a paradigm shift in business management. **Technology in Society**, v. 18, n. 2, p. 204-218, 1996.

O'LEARY, D. E., Enterprise Knowledge Management, **IEEE Computer**, v. 31, n. 3, p. 54-61, March 1998.

ORLOVSKY, S.A.. Decision-making with a fuzzy preference relation. **Fuzzy Sets and Systems**, v. 1, n. 3, p. 155 - 167, June 1978.

Goetsch, David L. and Davis, Stanley, **Quality Management for Organizational Excellence:** Introduction to Total Quality. 7 ed. New York: Prentice Hall, 2012.

PEDRYCZ W.; EKEL, P.; PARREIRAS, R.. **Fuzzy Multicriteria Decision-Making:** Models, Methods, and Applications. 1 ed. New York/Chichester/Brisbane: John Wiley & Sons, 2011.

POLLICE, Gary; AUGUSTINE, Liz; LOWE, Chris; MADHUR, Jas. **Software Development for Small Teams:** A Rup-Centric Approach. 1 ed. Redwood City, CA, USA: Addison-Wesley Professional, 2003.

PRESSMAN, Roger. **Software Engineering:** A Practitioner's Approach. 7 ed. New York, USA: McGraw-Hill, 2009.

Project Management Institute, **A Guide to the Project Management Body of Knowledge (PMBOK Guide):** An American National Standard**,** 4 ed. Newtown Square, PA: Project Management Institute, 2009.

ROUT, Terence P.; TUFFLEY, Angela. Harmonizing ISO-IEC 15504 and CMMI. **Software Process Jornal**, v. 12, n. 4, p. 361-371, July 2007.

RUPPRECHT, C. et al., Capture and Dissemination of Experience about the Construction of Engineering Processes. **12th International Conference on Advanced Information Systems Engineering,** UK, p. 294-308.

SEI – Software Engineering Institute. Disponivel em: < http://www.sei.cmu.edu/>. Acesso em: 08 maio. 2011.

SOMMERVILLE, I. **Software Engineering**. 8. ed. Harlow: Pearson Education, 2007.

Standish Group International. **CHAOS: Project failure and success report report**. Disponível em: <http://www.pm2go.com/sample_research/chaos_2010_2.asp>, Acesso em: 08 maio. 2011.

WANG, J.; LIN, Y.; A Fuzzy Multicriteria group decision making approach to select configuration items for software development, **Fuzzy Sets Systems Jornal**, v. 134, n. 3, p.343-363, 2001.

WONG, K. Critical success factors for implementing knowledge management in small and medium enterprises. **Industrial management and data systems**, v. 105, n. 3, p. 261 - 279, 2005.

YAGER, R.R.. Non-numeric multi-criteria multi-person decision making**, Group Decision and Negotiation**, v. 2, n. 1, p. 81-93, March 1993.

ZADEH, Lotfi Asker. **Fuzzy Sets, Fuzzy Logic, and Fuzzy Systems:** Selected Papers by Lotfi A. Zadeh. 6 ed. River Edge, NJ, USA: World Scientific, 1996.