UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

UMA SISTEMÁTICA BASEADA NO REUSO DE ATIVIDADES PARA ADAPTAÇÃO DE PROCESSOS DE SOFTWARE UTILIZANDO LINHAS DE PROCESSOS DE SOFTWARE

DISSERTAÇÃO DE MESTRADO

Wagner Gadêa Lorenz

Santa Maria, RS, Brasil

2014

UMA SISTEMÁTICA BASEADA NO REUSO DE ATIVIDADES PARA ADAPTAÇÃO DE PROCESSOS DE SOFTWARE UTILIZANDO LINHAS DE PROCESSOS DE SOFTWARE

Wagner Gadêa Lorenz

Dissertação apresentada ao Curso de Mestrado Programa de Pós-Graduação em Informática (PPGI), Área de Concentração em Computação, da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do grau de

Mestre em Ciência da Computação

Orientadora: Profª. Drª. Lisandra Manzoni Fontoura

Santa Maria, RS, Brasil

2014

Gadêa Lorenz, Wagner

UMA SISTEMÁTICA BASEADA NO REUSO DE ATIVIDADES PARA ADAPTAÇÃO DE PROCESSOS DE SOFTWARE UTILIZANDO LINHAS DE PROCESSOS DE SOFTWARE / por Wagner Gadêa Lorenz. – 2014.

94 f.: il.; 30 cm.

Orientadora: Lisandra Manzoni Fontoura

Dissertação (Mestrado) - Universidade Federal de Santa Maria, Centro de Tecnologia, Programa de Pós-Graduação em Informática, RS, 2014.

1. Linhas de Processos de Software. 2. Adaptação de Processos. 3. Processos de Software. 4. Arquiteturas de Processos. 5. Contexto de Projeto. I. Manzoni Fontoura, Lisandra. II. Título.

E-mail: wagnerglorenz@gmail.com

Universidade Federal de Santa Maria
Centro de Tecnologia
Programa de Pós-Graduação em Informática

A Comissão Examinadora, abaixo assinada,
aprova a Dissertação de Mestrado

UMA SISTEMÁTICA BASEADA NO REUSO DE ATIVIDADES PARA ADAPTAÇÃO DE PROCESSOS DE SOFTWARE UTILIZANDO LINHAS DE PROCESSOS DE SOFTWARE

elaborada por
Wagner Gadêa Lorenz

como requisito parcial para obtenção do grau de
Mestre em Ciência da Computação

COMISSÃO EXAMINADORA:

Lisandra Manzoni Fontoura, Drª.
(Presidente/Orientadora)

Fabricia C. Roos Frantz, Drª. (Unijui)

Luís Alvaro de Lima Silva, Dr. (UFSM)

Santa Maria, 1 de Setembro de 2014.

Dedico este trabalho a minha noiva Ariéli de Andrade dos Santos e aos meus pais Werner Lorenz e Senaira Gadêa Lorenz, por todo amor, carinho, compreensão e incentivo, sem os quais esta conquista seria muito árdua e penosa.

AGRADECIMENTOS

A minha orientadora, Prof. Dra. Lisandra Manzoni Fontoura, pelas ideias, pela disponibilidade em esclarecer dúvidas e por dividir seu conhecimento.

A minha noiva Ariéli de Andrade dos Santos, por apoiar todas as minhas decisões e estar presente em todos os momentos, sempre atenciosa e carinhosa, nunca poupando esforços para que eu alcançasse meus objetivos.

A minha família, especialmente meus pais Werner Lorenz e Senaira Gadêa Lorenz, por apoiarem minhas decisões.

Aos meus amigos Miguel Bauermann Brasil, Guilherme Vaz Pereira, Ruan Carlo Bonilha Pozzebon e Ricardo Fröhlich da Silva, pelas diversas horas de conversa e apoio em todo o trabalho realizado e caminho percorrido durante o mestrado.

A CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior), pelo apoio financeiro a este trabalho.

"Ninguém nasceu sábio, diz um provérbio; e tem razão. Todos os entes humanos vêm a este mundo terrestre com o fim de aprenderem; e, quem deve aprender, não é, naturalmente, ainda sábio". (Francisco Valdomiro Lorenz)

RESUMO

Dissertação de Mestrado
Programa de Pós-Graduação em Informática
Universidade Federal de Santa Maria

UMA SISTEMÁTICA BASEADA NO REUSO DE ATIVIDADES PARA ADAPTAÇÃO DE PROCESSOS DE SOFTWARE UTILIZANDO LINHAS DE PROCESSOS DE SOFTWARE

AUTOR: WAGNER GADÊA LORENZ
ORIENTADORA: LISANDRA MANZONI FONTOURA
Local da Defesa e Data: Santa Maria, 1 de Setembro de 2014.

Definição de processos de software requer escolher os elementos de processo que apropriadamente satisfazem os requisitos de adaptação, tais como a prevenção de riscos ou para satisfazer as metas de qualidade. A seleção dos elementos de processos adequados é geralmente feita manualmente, tornando este processo complexo, demorado e sujeito a erros. O principal objetivo é definir uma abordagem sistemática para adequar o processo de software e uma ferramenta de apoio para simplificar e apoiar o processo de adaptação, melhorar a seleção dos elementos de processos reutilizáveis. Foi desenvolvido uma abordagem sistemática para adequar o processo de software com base em arquiteturas de processo de software e linhas. A abordagem seleciona os elementos de processos mais adequados de acordo com os requisitos de adaptação. Uma ferramenta web foi desenvolvida para suportar o uso da abordagem proposta. Visando exemplificar a abordagem proposta neste trabalho, foram realizados um estudo de caso e um experimento. O estudo de caso descreve projetos com diferentes características, e, consequentemente são gerados diferentes processos adaptados. Com o experimento pode-se notar que as Linha de Processos de Software auxiliaram principalmente no sequenciamento do processo, otimizando os recursos e melhorando o gerenciamento do processo. A utilização da arquitetura proporcionou a recuperação de um conjunto de atividades que foram priorizadas de acordo com a caracterização do projeto, definindo elementos obrigatórios e opcionais e o acréscimo de requisitos de adaptação. Facilitando, desta forma, a adequação do processo utilizado para diferentes contextos de projetos. Conclui-se que a abordagem auxilia o engenheiro de processos a tomar decisões na seleção do conjunto de elementos de processos mais adequados as exigências de adaptação e contexto do projeto.

Palavras-chave: Linhas de Processos de Software. Adaptação de Processos. Processos de Software. Arquiteturas de Processos. Contexto de Projeto.

ABSTRACT

Master's Dissertation
Post-Graduate Program in Informatics
Federal University of Santa Maria

A SYSTEMATIC REUSE-BASED ACTIVITIES FOR TAILORING SOFTWARE PROCESS USING SOFTWARE PROCESS LINES

AUTHOR: WAGNER GADÊA LORENZ
ADVISOR: LISANDRA MANZONI FONTOURA
Defense Place and Date: Santa Maria, September 1^{1th}, 2014.

Software process definition requires choosing the process elements that appropriately fulfil the tailoring requirements, such as to prevent risks or to satisfy quality goals. The selection of appropriate process elements is usually done manually, making this process complex, time-consuming and error-prone. Our main objective is to define a systematic approach to tailor software process and a tool support to simplify and to support the tailoring process, improving the selection process of reusable process elements. A systematic approach was developed to tailor the software process based on software architectures and process lines. The approach selects the most appropriate elements of processes according to the tailoring requirements. A web tool was developed to support the use of the proposed approach. Aiming to illustrate the approach proposed in this paper, an case study and an experiment were conducted. The case study describe projects with different characteristics and consequently are different generated tailoring processes. With experiment can be noted that the Software Process Line helped mainly in the sequencing process, optimizing resources and improving management process. The use of architecture provided the retrieval of a set of activities that have been prioritized according to the characterization of the project, defining required and optional elements and the accretion of tailoring requirements. Facilitating thus the adequacy of the process used for different contexts of projects. We concluded that the approach aids process engineer to make decisions for selecting a set of process elements suitable to the tailoring requirements and to the project context.

Keywords: *Software Process Lines, Process Tailoring, Software Process, Software Process Architecture, Project Context*.

LISTA DE FIGURAS

Figura 2.1 – *Octopus Model* 28
Figura 3.1 – Sistemática para adaptação de processos. 32
Figura 3.2 – MfTP – Metamodelo para Adaptação de Processos. 33
Figura 3.3 – Elementos de processo instanciados para representar a atividade "*Analyze the Problem*". 35
Figura 3.4 – Exemplo de arquitetura definida para a disciplina de requisitos para a LPrS. . 39
Figura 3.5 – Detalhes da atividade "*Analyze the Problem*". 40
Figura 3.6 – Exemplo de Linha de Processo de Software para a disciplina de Requisitos. . 43
Figura 4.1 – MfTPt – Módulo principal da ferramenta. 46
Figura 4.2 – MfTPt - Primeiro passo para criar uma atividade. 47
Figura 4.3 – MfTPt – Segundo passo para criar uma atividade, seleção das tarefas. 48
Figura 4.4 – MfTPt - Terceiro passo, tarefas selecionadas e *link* para associação dos artefatos. 48
Figura 4.5 – MfTPt – Terceiro passo, associação entre atividades, tarefas e artefatos. 49
Figura 4.6 – MfTPt – Definição do contexto, critério de adaptação e associação da atividade a uma arquitetura. 49
Figura 4.7 – MfTPt - Contextualização da atividade. 50
Figura 4.8 – MfTPt - Definição dos requisitos de adaptação para a atividade. 51
Figura 4.9 – MfTPt - *Create an architecture*. 52
Figura 4.10 – MfTPt - *Define Architecture*. 52
Figura 4.11 – MfTPt - *Activities in Architecture*. 53
Figura 4.12 – Módulo SPrL - *Project Context: Fill Project Values*. 54
Figura 4.13 – Módulo SPrL – *Tailoring Requirements: Select the requirements*. 55
Figura 4.14 – Módulo SPrL – *Prioritization of Activities: Select activities for your process*. 55
Figura 4.15 – Módulo SPrL – *Software Process Line: Fill all details*. 57
Figura 5.1 – Arquitetura definida para a disciplina de requisitos contendo atividades para prevenção de riscos em projetos. 59
Figura 5.2 – MfPTt – Caracterização do Projeto "*Your Destiny*". 60
Figura 5.3 – MfTPt – Requisitos de adaptação e arquitetura. 61
Figura 5.4 – MfTPt – Priorização das atividades. 61
Figura 5.5 – MfTPt – Linha de Processo de Software adaptada para o Projeto "*Your Destiny*". 62
Figura 5.6 – MfTPt – Linha de Processo de Software adaptada para o Projeto "*Movie rental*". 65
Figura 7.1 – Arquitetura de processos para o STIC. 71
Figura 7.2 – Priorização e seleção das atividades para o processo do STIC. 72
Figura 7.3 – Linha de Processos de Software adaptada para o STIC. 73
Figura A.1 – Exemplo de arquitetura definida para a disciplina de Teste para a LPrS. 84
Figura A.2 – Linha de Processos de Software para a disciplina de Testes. 85
Figura A.3 – Exemplo de arquitetura definida para a disciplina de Análise e Projeto para a LPrS. 86
Figura A.4 – Linha de Processos de Software para a disciplina de Análise e Projeto. 87
Figura A.5 – Exemplo de arquitetura definida para a disciplina de Implementação para a LPrS. 88
Figura A.6 – Linha de Processos de Software para a disciplina de Implementação. 89

Figura A.7 – Exemplo de arquitetura definida para a disciplina de Gerencia de Projetos a LPrS. 90
Figura A.8 – Linha de Processos de Software para a disciplina de Gerencia de Projetos. . . 91

LISTA DE TABELAS

Tabela 3.1 – Exemplos de caracterização de projetos utilizando contexto planejado e ágil. 37
Tabela 3.2 – Associação entre riscos e atividades. 37
Tabela 3.3 – Conjunto de atividades recuperadas e priorizadas. 42
Tabela 5.1 – Conjunto de atividades recuperadas e priorizadas utilizando o método AHP. . 64
Tabela 6.1 – Comparação entre as abordagens. 68
Tabela 7.1 – Caracterização do projeto do STIC. 70

LISTA DE ANEXOS

ANEXO A – Arquiteturas e Linha de Processos de Software 84
ANEXO B – Questionário de avaliação aplicado no IF Farroupilha – Campus São Vicente do Sul 92

LISTA DE ABREVIATURAS E SIGLAS

AHP	Analytic Hierarchy Process
CMMI	Capability Maturity Model Integration
DSDM	Dynamic Systems Development Method
ISO/IEC	International Organization for Standardization
JPA	Java Persistence API
JSF	Java Server Faces
LPrS	Linhas de Processos de Software
MDE	Model Driven Engineering
MfTP	Metamodel for Tailoring Process
MPS-BR	Melhoria de Processo do Software Brasileiro
PSPO	Processo de Software Padrão da Organização
RUP	Rational Unified Process
SEI	Software Engineering Institute
SME	Situational Method Engineering
SPEM	Systems Process Engineering Meta-Model
SPL	Software Product Lines
SPrL	Software Process Lines
STIC	Setor de Tecnologia da Informação e Comunicação
UP	Unified Process
XP	eXtreme Programming

SUMÁRIO

1 INTRODUÇÃO 16
1.1 Definição do Problema 19
1.2 Escopo da Pesquisa 20
1.3 Estrutura da Dissertação 21
2 PROCESSOS DE SOFTWARE 22
2.1 Processos ágeis 23
2.2 Processos planejados 24
2.3 Processos híbridos 24
2.4 Abordagens de adaptação de processos 25
2.5 Caracterização de processos 26
2.6 Linha de Processos de Software (LPrS) 28
2.7 Arquiteturas de Processos 30
3 USANDO LINHA DE PROCESSOS DE SOFTWARE E ARQUITETURAS DE PROCESSOS PARA ADAPTAÇÃO DE PROCESSOS 32
3.1 Metamodelo 33
3.2 Contextualização de Projetos 36
3.3 Requisitos de Adaptação 37
3.4 Arquiteturas de Processos 38
3.5 *Analytic Hierarchy Process* (AHP) 41
3.6 Linha de Processos de Software adaptada 42
4 FERRAMENTA DE APOIO MFTPT 45
4.1 Módulo principal 45
4.2 Módulo para adaptação de processos de software baseado em Linhas de Processos de Software 53
5 ESTUDO DE CASO 58
5.1 Descrição do Projeto 1 58
5.2 Descrição do Projeto 2 63
5.3 Análise entre os estudos de caso 64
6 TRABALHOS RELACIONADOS 66
7 EXPERIMENTOS 69
7.1 Experimento – IF Farroupilha - Campus São Vicente do Sul 69
7.2 Avaliação do experimento 74
8 CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS 76
9 REFERÊNCIAS 78
ANEXOS 83

1 INTRODUÇÃO

Segundo o CMMI (*Capability Maturity Model Integration*), processos de software são formados por outros processos (subprocessos) ou elementos de processos (CHRISSIS; KONRAD; SHRUM, 2011). Um "elemento de processo"é uma unidade de definição de processo fundamental que descreve atividades e tarefas para que o trabalho possa ser consistentemente executado (CHRISSIS; KONRAD; SHRUM, 2011).

Adaptação de processos visa customizar um modelo de processo de desenvolvimento para atender as necessidades específicas de um projeto ou organização. A adaptação consiste em identificar os elementos de processo que devem ser usados para compor o processo e definir regras de conexão entre esses elementos. A escolha dos elementos de processo não é uma tarefa simples, devido a fatores como a complexidade no desenvolvimento de software e a variedade de modelos existentes (BENCOMO, 2005).

Algumas abordagens de adaptação têm como base processos de software genéricos, enquanto outras abordagens se baseiam em processos específicos. A adaptação a partir de processos genéricos é mais custosa porque envolve a definição de elementos de processos a partir de instanciação de elementos descritos em um metamodelo, que são agrupados para a criação de processos, mas por outro lado podem atender às necessidades de diferentes organizações (HENDERSON-SELLERS; RALYTÉ, 2010). Processos específicos facilitam a atividade de adaptação, pois são elaborados a partir de *frameworks* de processos, como o *Rational Unified Process* (RUP) (IBM COPORATION, 2007). A adaptação de processos consiste na alteração ou exclusão de elementos do *framework* para gerar os processos adaptados (BENCOMO, 2005). Porém, nem sempre o processo adaptado atende as necessidades organizacionais.

Além disso, algumas abordagens impõem regras bastante rígidas sobre o processo de uma organização, ou seja, o processo definido contém todos os elementos julgados necessários para a sua execução, mas como nem tudo pode ser previsto, deve existir uma flexibilidade, possibilitando a adequação do processo as características organizacionais (ROMBACH, 2006). Existe uma complexidade evidenciada na adaptação dos processos de software, sendo necessário caracterizar, usar e gerenciar as similaridades e diferenças entre os processos de forma a flexibilizar o reuso (ARMBRUST et al., 2009).

Considerando que a adaptação baseada em processos genéricos pode atender às necessidades de diferentes projetos e organizações, esta dissertação propõe uma abordagem de adap-

tação baseada em processos genéricos. O processo de software adaptado é elaborado a partir de componentes de processo reutilizáveis armazenados em um repositório tornando-o mais fácil e menos dispendioso. Este repositório de componentes é uma biblioteca de ativos de processos organizacionais que visa apoiar a aprendizagem organizacional e melhoria do processo, permitindo compartilhar as melhores práticas e lições aprendidas na organização ou entre organizações.

Muitas abordagens de adaptação são limitadas apenas à seleção dos componentes de processo e compõem o seu processo com base nas características do projeto, e não abordam a compatibilidade (coerência) e consistência dos processos gerados. O uso de linhas de processo de software tem como objetivo desenvolver processos consistentes uma vez que muitas abordagens de adaptação são limitadas apenas para a seleção dos elementos do processo com base em características do projeto, mas não se preocupam com o sequenciamento e a consistência dos processos gerados. Autores como (JAUFMAN; MUNCH, 2005; WASHIZAKI, 2006a) propõem o uso de *Software Processos Line* (SPrL), como forma de viabilizar a reutilização de componentes e contextualização do processo.

Esta dissertação propõe uma abordagem de adaptação de processos de software a partir de Linhas de Processos de Software e informações sobre as características dos projetos. A abordagem proposta utiliza o reuso de elementos de processo, previamente definidos, gerando processos planejados e ágeis. A Linha de Processo de Software foi elaborada a partir de um conjunto de elementos de processos (atividades, papéis, artefatos, entre outros) instanciados a partir de um metamodelo que descreve conceitos propostos por SPEM (*Systems Process Engineering Meta-Model*)(OMG, 2008).

O metamodelo define um conjunto de conceitos que são utilizados para representar modelos de processos de software e os relacionamentos entre esses conceitos. Os elementos de processo, instanciados a partir do metamodelo proposto, são armazenados em um repositório, que consiste em um conjunto de componentes, elaborados a partir de diferentes modelos de processos de software, e deve evoluir com o tempo e com a maturidade da organização.

A recuperação dos elementos de processo mais adequados para um determinado projeto inicia com a seleção e priorização de atividades a partir do repositório utilizando o método AHP (*Analytic Hierarchy Process*) (SAATY, 2008). Foi desenvolvida uma ferramenta *Web* para apoiar o uso da abordagem proposta.

A ferramenta contém um conjunto de funcionalidades necessárias para adaptação de

processos de software baseado em Linhas de Processo de Software, incluindo funcionalidades para criação do repositório de elemento de processos, como: registro de artefatos, tarefas, papéis, atividades, contextualização de atividades; como funcionalidades relacionadas a definição de arquiteturas e de Linhas de Processos de Software; e a adaptação das linhas para projetos específicos.

Visando exemplificar a abordagem proposta neste trabalho, foram realizados um estudo de caso fictício e um experimento. O estudo de caso descreve projetos com diferentes características, e consequentemente são gerados diferentes processos adaptados. Em ambos os casos foram inclusas atividades para prevenção de riscos em projetos. Para contextualização dos projetos apresentados nos estudos de caso, foi utilizado o *Octopus Model*, proposto por Kruchten (KRUCHTEN, 2013).

O Projeto 1 apresenta características adequadas a um processo planejado, estipuladas segundo o *Octopus Model*. De acordo com o contexto do projeto e arquitetura utilizada, foram recuperadas atividades contendo características provenientes de processos planejados, desta forma, proporcionando maior planejamento e documentação do software. A partir da arquitetura e da recuperação das atividades foi elaborada a Linha de Processo de Software para o projeto.

O Projeto 2 apresenta um contexto situacional ágil de acordo com fatores estipulados pelo *Octopus Model*. De acordo com o contexto do projeto, as atividades recuperadas visam proporcionar maior agilidade ao processo gerado. A partir da arquitetura e das atividades recuperadas foi elaborada a Linha de Processos de Software para um projeto envolvendo contexto ágil.

Para os projetos em questão, foram recuperadas diferentes atividades pela ferramenta de apoio para cada estudo de caso, de acordo com a contextualização do projeto as atividades são recuperadas da base de conhecimento. O mecanismo de priorização e seleção de atividades não elimina o papel do engenheiro de processo, mas facilita a identificação das atividades mais relevantes de acordo com os requisitos de adaptação e contexto situacional do projeto.

O experimento foi aplicado no Setor de Tecnologia da Informação e Comunicação (STIC), do Instituto Federal Farroupilha – Campus São Vicente do Sul, com a equipe de desenvolvimento de software. O experimento foi conduzido em duas etapas: na primeira etapa, foi utilizada pelo setor de desenvolvimento a ferramenta proposta nesta dissertação, cadastrando e definindo todos os elementos necessários para o seu processo; na segunda etapa, o processo

foi executado utilizando a sistemática proposta nesta dissertação para adaptação de processos de software utilizando Linha de Processos de Software, sendo executada as cinco etapas e os quatro passos da sistemática. Com o experimento pode-se notar que as Linha de Processos de Software auxiliaram principalmente no sequenciamento do processo, otimizando os recursos e melhorando o gerenciamento do processo.

A abordagem proposta garante a consistência do processo em relação ao sequenciamento de componentes (atividades), através da arquitetura e da Linha de Processo de Software. Além disso, reduz a probabilidade de uma adaptação de processo de software inadequada.

1.1 Definição do Problema

Hoje em dia, muitas empresas e organizações baseiam-se essencialmente em software. O software agrega um valor significativo para muitos produtos e serviços e permite a diferenciação competitiva no mercado. Com a importância crescente de software e o surgimento de novos paradigmas de desenvolvimento de software, o mercado impõem demandas e desafios nos processos de desenvolvimento, operação e manutenção.

Sistemas de software e sistemas de software intensivo são desenvolvidos por centenas ou milhares de pessoas divididas em equipes. Elas desempenham um grande número de atividades diferentes, os chamados processos.

As organizações de desenvolvimento de software atualmente estão interessadas em aumentar sua competitividade e níveis de qualidade. Para atingir determinadas metas, elas precisam ter processos bem definidos. Um modelo de processo de software é uma representação das atividades do mundo real de um processo de produção de software.

Existem diferentes formas de modelar e avaliar modelos de processos de software, tais como CMMI (CHRISSIS; KONRAD; SHRUM, 2011) e ISO/IEC 15504 (ISO/IEC 15504, 2003). Assim, qualquer modelo de processo de software tem que modelar adequadamente o processo do mundo real e deve atender às exigências especificadas em cada fase do processo.

Neste sentido o problema definido para a pesquisa é:

Como adaptar processos de software, que sejam caracterizados, gerenciados e elaborados para atender as necessidades específicas de um projeto ou organização, mantendo a consistência no processo adaptado?

1.2 Escopo da Pesquisa

O escopo da pesquisa visa responder a pergunta: Como adaptar processos de software, que sejam caracterizados, gerenciados e elaborados para atender as necessidades específicas de um projeto ou organização, mantendo a consistência no processo adaptado?

Acredita-se que usando uma abordagem de adaptação de processos de software a partir de Linha de Processos de Software e da reutilização de elementos de processo previamente caracterizados é possível ser alcançado.

Esta dissertação apresenta como principal contribuição a proposta de uma abordagem sistemática para adaptação de processos de software a partir de Linha de Processos de Software e informações sobre as características dos projetos. A abordagem proposta utiliza o reuso de elementos de processos, previamente definidos, gerando processos planejados e ágeis. A Linha de Processos de Software adaptada é elaborada a partir da recuperação dos elementos previamente cadastrados nas arquiteturas de processos, ou seja, são recuperadas atividades contextualizadas de acordo com os requisitos de adaptação e o contexto do projeto.

Contribuições do trabalho incluem:

- Um metamodelo para definição de atividades, permitindo a instanciação de componentes de natureza ágil e planejada, representando os seus conceitos e relacionamento com elementos de processo.
- Definição de arquiteturas de processos com componentes obrigatórios, opcionais, concretos e abstratos;
- Mecanismo para criação de arquiteturas de processos de software.
- Associação de atividades com diferentes critérios de adaptação, diferentes contextos e diferentes arquiteturas de processos;
- Utilização de diferentes formas de caracterização de processos para situações específicas em projetos de software, para utilização na abordagem de adaptação de processos a partir de Linha de Processos de Software;
- Construção de Linha de Processos de Software a partir das arquiteturas de processos.

1.3 Estrutura da Dissertação

Este trabalho está organizado como descrito a seguir. No Capítulo 2 são descritos conceitos sobre processos de software, abordagens de adaptação de processos de software, caracterização de processos, conceitos sobre Linha de Processos de Software e arquiteturas de processos. No Capítulo 3 é apresentado o metamodelo com todas as suas especificidades e descrições, também é descrito a abordagem sistemática proposta de utilização de Linha de Processos de Software e Arquiteturas de Processos para adaptação de processos. No Capítulo 4 é apresentado a ferramenta de apoio *Metamodel for Tailoring Process* (MfTP), contendo o módulo principal e o módulo para adaptação de processos baseado em Linha de Processos de Software. No Capítulo 5, um estudo de caso, contendo dois projetos com contexto distintos é descrito. No Capítulo 6 são apresentados os trabalhos relacionados e uma comparação destes com a abordagem proposta neste trabalho. No Capítulo 7 é apresentado o experimento realizado para validação da proposta. Por fim, no Capítulo 8 são apresentadas as considerações finais e trabalhos futuros.

2 PROCESSOS DE SOFTWARE

As organizações de desenvolvimento de software vem a muito tempo investindo em qualidade para aumentar o seu nível de competitividade. Para atingir determinadas metas, elas precisam ter processos bem definidos. Um modelo de processo de software é uma representação de atividades do mundo real de um processo de produção de software.

Existem diferentes formas de avaliar modelos de processos de software, tais como *Capability Maturity Model Integration* (CMMI) (CHRISSIS; KONRAD; SHRUM, 2011) e ISO/IEC 15504 (ISO/IEC 15504, 2003) . Assim, qualquer modelo de processo de software tem que modelar adequadamente o processo do mundo real e deve atender às exigências especificadas em cada fase do processo.

Os processos necessitam ser padronizados ou formalizados de forma a melhorar o entendimento e funcionamento, possibilitando um melhor treinamento, adequação a propostas de melhorias, entre outros fatores. Um modelo de processo deve especificar os pré-requisitos e consequências de cada tarefa (BEN-SHAUL; KAISER, 1994).

Processo de software é o conjunto completo e ordenado de atividades de engenharia de software necessárias para transformar os requisitos do usuário em um software (HUMPHREY, 1989). O *Software Engineering Institute* (SEI) (OMG, 2008), por meio do CMMI, propõe que seja definido um Processo de Software Padrão da Organização (PSPO). Esse processo contém elementos essenciais para todos os projetos e define que os projetos sejam adaptados de acordo com as necessidades de cada um. Assim, cria-se um processo específico para cada projeto.

A adequação de um processo de software depende das características do projeto, da organização e do produto, e estas evoluem continuamente. Cada projeto possui suas próprias características e necessita de técnicas e estratégias de desenvolvimento particulares (ALEGRÍA et al., 2011).

Adaptação de processos visa customizar um modelo de processo de desenvolvimento para atender as necessidades específicas de um projeto ou organização. A adaptação de processos não é uma tarefa simples, devido a fatores como a complexidade no desenvolvimento de software e a variedade de modelos existentes (BENCOMO, 2005).

Os projetos de uma organização apresentam diferenças em relação a fatores como: equipe, cliente, características do software, riscos envolvidos, metas de qualidade, entre outros. Ou seja, cada projeto possui suas próprias características que requerem um conjunto particular

e completo de atividades, técnicas e estratégias, organizadas de forma ordenada e coesa e que devem estar contidas no processo de software para o projeto (ALEGRÍA et al., 2011; FONTOURA, 2006; XU; RAMESH, 2008a). A adaptação de processos é a atividade que se propõe a ajustar o processo organizacional para atender as características específicas de um projeto.

Contudo, segundo Alegría *et al*. (ALEGRÍA et al., 2011), cada projeto possui características distintas, como por exemplo, o tamanho da equipe, a disposição geográfica do time de desenvolvimento, modelo de negócios adotado, criticidade do sistema, taxa de mudança dos requisitos do sistema, entre outros. Estas características dividem-se em características do projeto, da organização e do produto, e evoluem continuamente. Cada projeto possui suas próprias características e necessita de técnicas e estratégias de desenvolvimento particulares, por isso os processos precisam ser adaptados para atender a essas necessidades.

Devido a importância da definição de processos de software, muitos modelos e métodos foram propostos, tais como: RUP (IBM COPORATION, 2007), XP (BECK, 2004), SCRUM (SCHWABER; BEEDLE, 2003). Existe ainda a ideia de que um processo de desenvolvimento padrão deva ser adaptado para atender as necessidades específicas de cada projeto e/ou organização, por meio da avaliação de processos, utilizada a partir de modelos de avaliação como: ISO/IEC 12207 (SINGH, 1996), CMMI-DEV (CHRISSIS; KONRAD; SHRUM, 2011) e MPS-BR (WEBER; ARAÚJO; MACHADO, 2005).

As seções seguintes descrevem, respectivamente, conceitos relacionados a processos ágeis, processos planejados, processos híbridos, abordagens de adaptação de processos de software, caracterização de processos de software, conceitos sobre Linha de Processos de Software e arquiteturas de processos

2.1 Processos ágeis

Na literatura existem diferentes abordagens para elaboração de processos de desenvolvimento de software, as duas principais são ágeis e planejados. O termo métodos ágeis foi proposto por vários metodologias em uma reunião em 2001 em Utah, nos Estados Unidos, essa reunião teve o propósito de discutir similaridades entre suas abordagens. Métodos ágeis, propostos através do manifesto ágil (http://www.agilealliance.org/) surgiu em resposta ao peso da burocracia, à desumanização das abordagens planejadas com foco no detalhamento, e as rápidas mudanças no ambiente de tecnologia da informação (BOEHM, 2002).

As abordagens ágeis visam encontrar o equilíbrio, provendo apenas o processo suficiente

para obter retorno, focam na comunicação entre os membros da equipe e com o cliente e são menos centradas em documentação, assumindo que parte da documentação é o próprio código-fonte. Outras características marcantes são que estas não funcionam bem com equipes grandes, a característica adaptativa em relação às mudanças e a orientação a pessoas (FOWLER, 2001).

Dentre os métodos ágeis podemos destacar os mais populares como: *eXtreme Programming* (XP) (BECK, 2004), *Dynamic Systems Development Method* (DSDM) (STAPLETON, 1999), Scrum (SCHWABER; BEEDLE, 2003) e Crystal (COCKBURN, 2004).

2.2 Processos planejados

Os processos tradicionais, também conhecidos como processos planejados, definem determinadas disciplinas no desenvolvimento de software através de um planejamento extensivo das atividades e processos de codificação e um rigoroso reuso para tornar o desenvolvimento mais eficiente e previsível (BOEHM, 2002). As abordagens tradicionais envolvem um processo detalhado com forte ênfase em planejamento e inspirado em outras disciplinas de engenharia.

Embora sejam usados há anos, esses processos de desenvolvimento ainda apresentam problemas como estimativas imprecisas de prazos e custos, baixa qualidade dos produtos, insatisfação dos clientes, entre outros.

Como exemplos de processos de software planejados, podemos citar: RUP (IBM CORPORATION, 2007), OPEN (HENDERSON-SELLERS, 2000), IBM Global Services Method (NICHOLS, 2005), e processos elaborados para atender normas ou modelos tais como o CMMI (CHRISSIS; KONRAD; SHRUM, 2011) e ISO/IEC 12207 (BALDASSARRE et al., 2009).

2.3 Processos híbridos

Existem projetos para os quais uma abordagem planejada é muito custosa para ser aplicada e ao mesmo tempo uma abordagem ágil não contém todos os elementos e a formalização desejada. Segundo Boehm (2002) existem projetos para os quais pode-se utilizar os princípios das abordagens ágeis ou planejadas, podendo ser uma muito melhor em relação à outra. Porém, se esse não for o caso, uma abordagem híbrida é preferível.

Quando agilidade pura e planejamento puro não satisfazem as necessidades do projeto, uma mistura entre elas é desejável. Com isso, a utilização de práticas de abordagens ágeis e planejadas podem ser mescladas, dando origem a abordagens híbridas de acordo com o contexto

de cada projeto.

2.4 Abordagens de adaptação de processos

A adaptação de um processo está relacionada com a customização de um processo padrão de desenvolvimento, voltado para atender as necessidades de uma organização e/ou projeto. Dentre as atividades de adaptação tem-se adicionar, excluir e/ou modificar elementos de um processo padrão, como por exemplo, atividades, artefatos, papéis, entre outros (GINSBERG; QUINN, 1995; XU; RAMESH, 2008b; XU, 2005).

Pode-se através da adaptação de processos balancear o uso de práticas planejadas e ágeis, originando os processos híbridos (BOEHM; TURNER, 2003). A adaptação de processos tem sido proposta por modelos e normas de referência, como: ISO 15504 (ISO/IEC 15504, 2003) e CMMI (CHRISSIS; KONRAD; SHRUM, 2011).

Vários trabalhos com propostas de abordagens sobre adaptação de processos têm sido descritas na literatura como *Unified Process* (UP) (JACOBSON; BOOCH; RUMBAUGH, 1999), *Situational Method Engineering* (SME) (HENDERSON-SELLERS; RALYTÉ, 2010), *Model Driven Engineering* (MDE), CASPER (PEREIRA; BASTOS; OLIVEIRA, 2007). Esses trabalhos propõem diferentes formas para adaptação de processos.

Unified Process é voltado para projetos com contexto planejados. *Situational Method Engineering* propõe a construção de um método de desenvolvimento específico para cada projeto, considerando o contexto situacional, mas não se preocupa com o sequenciamento de seus componentes. CASPER propõem um metamodelo em conformidade com o RUP e regras de boa formação orientando o processo de adaptação.

Um dos problemas em adaptação de processos de software é a consistência do processo, pequenas mudanças podem involuntariamente quebrar a sua consistência, por exemplo, a remoção de uma característica, embora podendo ser válida a partir do ponto de vista de uma sintaxe, pode invalidar outras características relacionadas (GUO; WANG, 2010). Normalmente, a inconsistência vem de restrições contraditórias, que impedem a produção de qualquer produto válido (MAB, VON DEREN; LICHTER, 2004).

Devido à complexidade evidenciada na adaptação dos processos de software tem-se a necessidade de caracterizar, usar e gerenciar as similaridades e diferenças entre os processos (SUTTON; OSTERWEIL, 1996).

Linha de Processos de Software é uma forma de adaptação de processos que possui os

seguintes objetivos: i) aumento da qualidade e adequação dos processos gerados; ii) representação de variabilidades e semelhanças entre processos para potencializar a reutilização; e iii) diminuição dos riscos de uma adaptação inadequada do processo (JAUFMAN; MUNCH, 2005; ROMBACH, 2006).

2.5 Caracterização de processos

Existe um consenso de que os processos ágeis e os processos planejados são adequados a contextos específicos, que pode ser determinado por características da organização ou projeto. (BOEHM; TURNER, 2003; COCKBURN, 2004; KRUCHTEN, 2013). Com isso tem-se a necessidade de caracterizar processos de acordo com as características dos projetos.

Autores como Boehm e Turner (2003) propõem cinco características para definição do contexto, que são: **a)** tamanho da equipe; **b)** criticidade do sistema; **c)** dinamismo do ambiente; **d)** habilidades da equipe; e **e)** cultura organizacional. Na família de processos da Crystal, proposta por Alistair Cockburn (2004) são definidos processo baseados em: **a)** tamanho, **b)** criticidade, e **c)** habilidades.

Outra proposta para definir as características do projeto é o *Octopus Model*, proposto por Kruchten (2013).Pode-se visualizar na Figura 2.1, as oito características descritas pelo *Octopus Model* que afetam significativamente o desenvolvimento de software, que são:

a) Tamanho (*Size*): este fator está relacionado ao tamanho da equipe ou tempo de desenvolvimento ou do orçamento (KRUCHTEN, 2013). Neste trabalho o tamanho foi utilizado para caracterizar o tamanho da equipe e pode ser definido como: pequeno, médio ou grande. Como definido por Kruchten (2013).

b) Arquitetura estável (*Stable Architecture*): este fator está relacionado se os projetos são novos o suficiente para exigir um grande esforço da arquitetura (*middleware*, linguagem de programação, etc.), ou seguem padrões comumente aceitos em seus respectivos domínios (KRUCHTEN, 2013). Neste trabalho a arquitetura estável foi utilizada para caracterizar esforço quanto à mudança em seu domínio e pode ser definida como: estável, modificada ou móvel. Como definido por Kruchten (2013).

c) Modelo de negócio (*Business Model*): este fator está relacionado ao desenvolvimento do sistema, ou seja, se está sendo desenvolvido um sistema interno, um produto comercial, um sistema sobre medida para um cliente, ou um componente de um grande sistema que envolve muitas partes diferentes (KRUCHTEN, 2013). Neste trabalho o modelo de negócios

foi utilizado para caracterizar a abordagem de desenvolvimento empregada, definindo-a como: sob medida para um cliente, comercial ou componente de um grande sistema. Como definido por Kruchten (2013).

d) Distribuição da equipe (*Team Distribution*): este fator está relacionado à dimensão do projeto, quantas equipes estão envolvidas e se estão alocadas juntas (KRUCHTEN, 2013). Neste trabalho a distribuição da equipe foi utilizada para caracterizar a alocação do time e existência de equipes diferentes no mesmo projeto e pode ser definida como: mesmo local, equipes diferentes ou distribuídas geograficamente. Como definido por Kruchten (2013).

e) Taxa de mudança (*Rate of Change*): este fator está relacionado à estabilidade do ambiente de negócios e requisitos do sistema (KRUCHTEN, 2013). Neste trabalho a taxa de mudança foi utilizada para caracterizar a porcentagem da estabilidade do ambiente e dos seus requisitos e pode ser definida como: mais que 30%, entre 10% e 30% ou menos que 10%. Como definido por Kruchten (2013).

f) Idade do sistema (*Age of the System*): este fator está relacionado à evolução do sistema (KRUCHTEN, 2013). Neste trabalho a idade do sistema foi utilizada para caracterizar quanto à evolução e pode ser definida como: novo desenvolvimento, manutenção ou evolução de sistema legado. Como definido por Kruchten (2013).

g) Criticidade (*Criticity*): este fator está relacionado a perdas que podem ocorrer em caso de falhas do sistema (KRUCHTEN, 2013). Neste trabalho a criticidade foi utilizada para caracterizar quanto a perdas em caso de falhas do sistema e pode ser definida como: perda de conforto, perda de dinheiro ou mortes. Como definido por Kruchten (2013).

h) Controle (*Governance*): este fator está relacionado à como o projeto de software é iniciado e gerenciado (KRUCHTEN, 2013). Neste trabalho o controle foi utilizado para caracterizar o tipo de gerenciamento adotado pelo projeto de software e pode ser definido como: dinâmico flexível, regras simples ou mecânico formal. Como definido por Kruchten (2013).

Entre as várias propostas para caracterização de projetos mencionadas, pode-se notar que existem semelhanças entre as propostas de (BOEHM; TURNER, 2003) e (KRUCHTEN, 2013). Pode-se notar que “tamanho da equipe” e “criticidade” estão presentes em ambas as propostas.

Além das características acima mencionadas, projetos de software têm requisitos que podem ser considerados em uma adaptação, como prevenção de riscos, o cumprimento das metas de qualidade, entre outros. Por exemplo, se há o risco de falta de entendimento dos

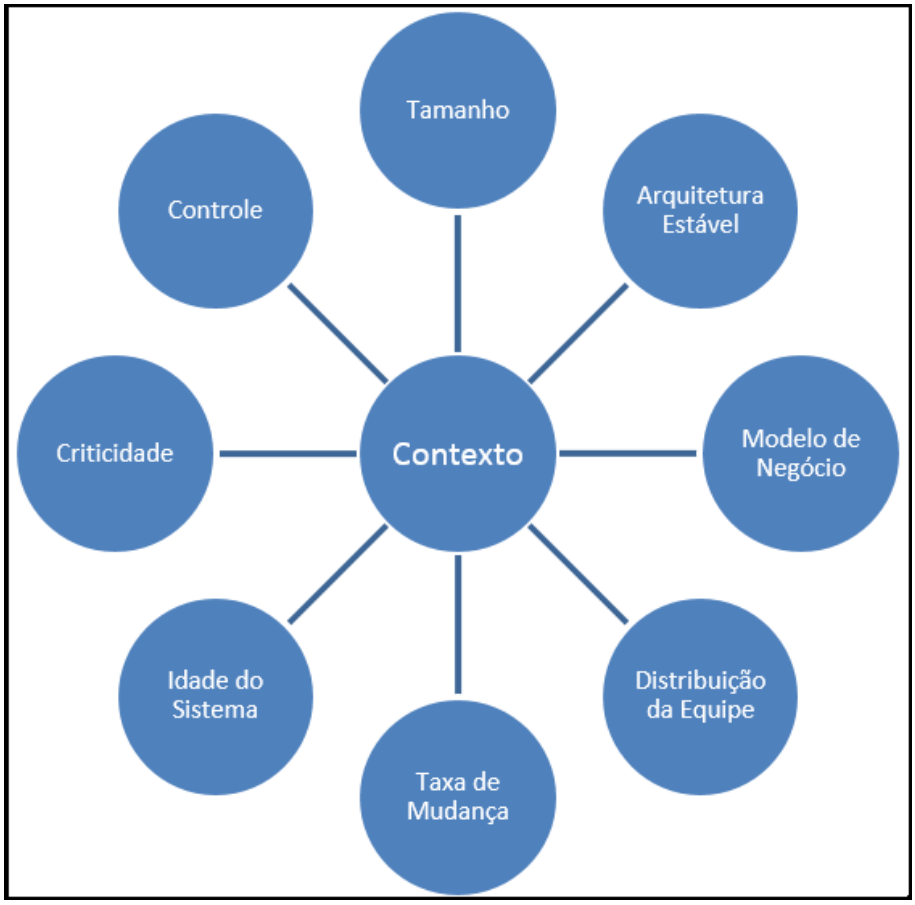

Figura 2.1 – *Octopus Model*
Fonte: Adaptado de (KRUCHTEN, 2013)

requisitos de um projeto, é possível adicionar atividades no processo para criar e avaliar os protótipos, a fim de evitar este risco.

Nesta dissertação, pode-se utilizar mais de uma abordagem, como as citadas acima para contextualização de atividades e projetos. Para cada tipo de contextualização, pode-se definir para as atividades e para o projeto características de abordagens planejadas, ágeis ou híbridas.

2.6 Linha de Processos de Software (LPrS)

Na última década, as organizações encontraram maneiras de oferecer produtos diversificados, tentando reduzir o seu custo de desenvolvimento e tempo de colocação no mercado. Na comunidade científica e industrial, acreditou-se que o reuso de software era um poderoso meio para tentar melhorar a produtividade e a qualidade no desenvolvimento de software (GRISS, 1997; PARNAS, 1976).

Software Product Lines (SPL), também conhecidas como Linhas de Produtos de Soft-

ware, surgiram como uma forma promissora de reutilização de software, e visam abordar a reutilização sistemática em todas as fases de desenvolvimento de software. Linha de produtos de software permite a reutilização em larga escala modular por meio de uma arquitetura de software abordando um núcleo múltiplo e diferentes características (FIGUEIREDO et al., 2008).

Este paradigma de desenvolvimento de software tem aumentado a produtividade das indústrias relacionadas com a área de TI, reduzindo o tempo de entrada do produto no mercado e permitindo o desenvolvimento de produtos diversificados (CLEMENTS; NORTHROP, 2001; VAN DER LINDEN; SCHMID; ROMMES, 2007). Uma das primeiras utilizações de linhas de produtos de software foi na aplicação da customização em massa para a produção de software (BENAVIDES; SEGURA; RUIZ-CORTÉS, 2010; BENAVIDES, 2007).

Uma Linha de Produtos de Software é um conjunto de produtos de software que compartilham um conjunto comum de características que satisfazem às necessidades específicas de um segmento de mercado em particular e são desenvolvidos a partir de artefatos de forma prescrita (NORTHROP, 2002). A sistemática baseada em reutilização e adaptação, proposta em Linhas de Produtos de Software, pode ser aplicada a processos de software, melhorando as abordagens de adaptação existentes, em sua maioria *ad-hoc* (MONTERO; PEÑA; RUIZ-CORTÉS, 2007).

A definição ou adaptação de processos exige conhecimentos específicos de processos de software, normas e modelos de qualidades. Não é trivial encontrar profissionais com esse perfil em equipes de software.

Abordagens baseadas em reuso estão sendo bastante utilizadas em adaptação de processos, tais como: componentes, arquiteturas e linhas de processo. *Software Process Lines* (SPrL), também conhecidas como Linhas de Processos de Software (LPrS) surgiram a partir de Linhas de Produtos de Software, que são elementos de software que compõem o produto gerado (JAUFMAN; MUNCH, 2005; WASHIZAKI, 2006a).

Técnicas de reutilização foram adaptadas a partir do desenvolvimento de produtos de software tradicional para o contexto de processo de software para realizar a reutilização do conhecimento de processos de software (KELLNER, 1996; WASHIZAKI, 2006a). Conceitos como componentes, arquiteturas e linhas de produtos têm sido utilizadas para definir e melhorar os processos de software existentes.

De acordo com Washizaki (2006a), uma Linha de Processo pode ser definida como “um conjunto de processos para um determinado domínio de problema ou com um determinado

propósito, que tem características em comum e é construído baseado em ativos reutilizáveis de processos". Portanto, acredita-se que os processos de software podem usufruir de benefícios obtidos pela utilização de Linhas de Produto de Software, principalmente o reuso (ARMBRUST et al., 2009).

O uso de Linha de Processos visa à elaboração de processos consistentes, pois muitas abordagens de adaptação se limitam a seleção de elementos de processos com base em características dos projetos, mas não se preocupam com o sequenciamento dos processos gerados, o que pode acarretar em inconsistências. Autores como Barreto *et al*. (2011) e Magdaleno (2010b) propõem o uso de Linha de Processo de Software (LPrS) como forma de possibilitar a reutilização de componentes e contextualização do processo.

Linha de Processos de Software é uma forma de adaptação de processos que possui os seguintes objetivos (JAUFMAN; MUNCH, 2005; ROMBACH, 2006):

- aumento da qualidade e adequação dos processos gerados;
- representação de variabilidades e semelhanças entre processos para potencializar a reutilização; e
- diminuição dos riscos de uma adaptação inadequada do processo.

Pode-se considerar uma linha de processos de software como uma espécie de arquitetura de processos de software, mas que possui elementos reutilizáveis, gerando processos de software diferentes a partir da mesma estrutura.

Uma linha de processos de software pode conter em sua arquitetura, componentes obrigatórios e opcionais. Componentes obrigatórios são usados para definir os elementos que devem estar presentes em todos os processos organizacionais, isto é, o conjunto mínimo de elementos a serem executados em um processo. Componentes opcionais definem elementos que podem ou não compor o processo organizacional.

2.7 Arquiteturas de Processos

Arquitetura de Processos de Software é definida como "uma estrutura de processos que reflete semelhanças e variabilidades em um conjunto de processos que compõem uma Linha de Processos a partir da perspectiva de otimização global" (WASHIZAKI, 2006b). Ou seja,

ela define a estrutura principal que o processo deve conter e determina os seus componentes principais e como é o relacionamento entre eles.

O reuso é realizado com base em repositórios nos quais é possível efetuar o armazenamento, a busca e a recuperação de componentes que são utilizados para a adaptação de processos. Os componentes armazenados no repositório são definidos a partir da instanciação de elementos de um metamodelo de definição de processos, elaborado a partir de SPEM (OMG, 2008). Os componentes são caracterizados por meio de fatores de contexto e requisitos de adaptação. A Seção 3.2 detalha a definição e caracterização de componentes. A proposta é que o conhecimento de engenheiros experientes seja representado por meio da definição e da caracterização de cada componente de processo, auxiliando engenheiros menos experientes nas suas tarefas de definição de processos.

É por meio das características do componente que as atividades serão recuperadas do repositório usando uma técnica multicritério chamada de *Analytic Hierarchy Process* (AHP) para a partir da arquitetura de processos construir a Linha de Processos de Software adaptada. Essa técnica é utilizada para apoio à tomada de decisão em ambientes complexos. Com sua utilização pode-se estabelecer uma ordem relativa de importância para projetos de melhoria de processos com base em um processo executado de forma estruturada (DAVID; SAATY, 2007). AHP é explicada na Seção 3.5.

Neste capítulo foi apresentado o referencial teórico utilizado nesta dissertação, abordando processos de software, abordagens para adaptação, caracterização de processos e linha de processos de software. O capítulo 3 aborda como são utilizadas as Linhas de Processos de Software e Arquiteturas de Processos para adaptação de processos, o metamodelo utilizado e a sistemática de adaptação de processos de software utilizando Linha de Processos de Software.

3 USANDO LINHA DE PROCESSOS DE SOFTWARE E ARQUITETURAS DE PROCESSOS PARA ADAPTAÇÃO DE PROCESSOS

Este trabalho propõe uma sistemática para adaptação de processo de software por meio do uso de Linhas e Arquiteturas de Processos. Os elementos de processo utilizados para definir processo de software são instanciados e armazenados em um repositório a partir do metamodelo que será apresentado na Seção 3.1. O repositório utilizado nos estudos de caso deste trabalho é formado por elementos descritos nas metodologias: RUP (IBM COPORATION, 2007), SCRUM (SCHWABER; BEEDLE, 2003), XP (BECK, 2004); ou Padrões de Processo e Organizacionais (COPLIEN; ALEXANDER, 1996); ou elementos elaborados a partir das diretrizes de modelos de avaliação de processo, como CMMI (CHRISSIS; KONRAD; SHRUM, 2011) e MPS.BR (MPS et al., 2012). Padrões de processo e organizacionais descrevem soluções para problemas recorrentes e podem ser utilizados na construção e adaptação de processo de software.

Pode-se visualizar na Figura 3.1, a sistemática de adaptação de processos. A sistemática é composta por quatro passos (formulários) contendo cinco etapas que estabelecem uma sequência para a adaptação, são elas: **i)** definição das características do projeto; **ii)** seleção dos requisitos de adaptação; **iii)** seleção da arquitetura de processos; **iv)** priorização das atividades; e **v)** criação da linha de processo de software adaptada.

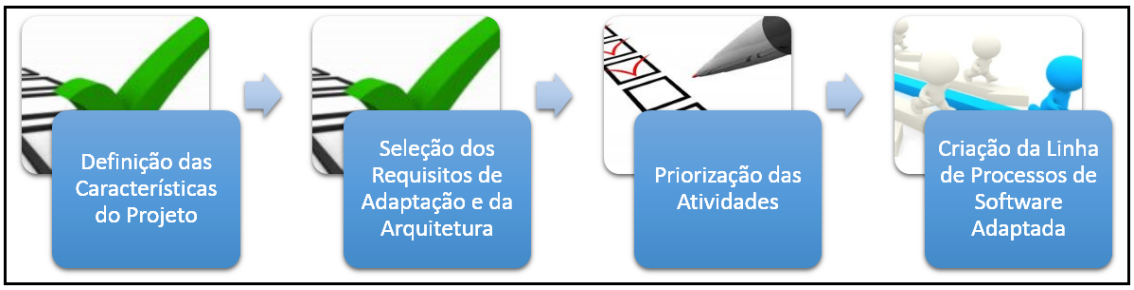

Figura 3.1 – Sistemática para adaptação de processos.

No primeiro passo, são definidas as características para o projeto, no segundo passo são selecionados os requisitos de adaptação e a(s) arquiteturas para o processo, no terceiro são recuperadas e priorizadas as atividades presentes na arquitetura, no quarto e último passo é criada a linha de processo de software adaptada de acordo com as atividades selecionadas no passo anterior. Cada uma das etapas será abordada em detalhes nas Seções que seguem.

3.1 Metamodelo

O *Metamodel for Tailoring Process* (MfTP) representa um conjunto de conceitos que são usados na modelagem de processos de software. O MfTP foi elaborado a partir do SPEM 2.0 (OMG, 2008) e RUP *version* 7.2 (IBM COPORATION, 2007), e pode ser visualizado na Figura 3.2.

Dentre os conceitos utilizados, o SPEM foi escolhido por ser um metamodelo proposto pela OMG para descrição de um processo de desenvolvimento de software, ele aparece como uma proposta de unificação entre as diferentes metodologias propostas para modelagem de processos, permitindo acomodar uma grande variedade de processos de desenvolvimento de software. Ambos os metamodelos usam a UML como uma notação e adotam uma abordagem orientada a objetos.

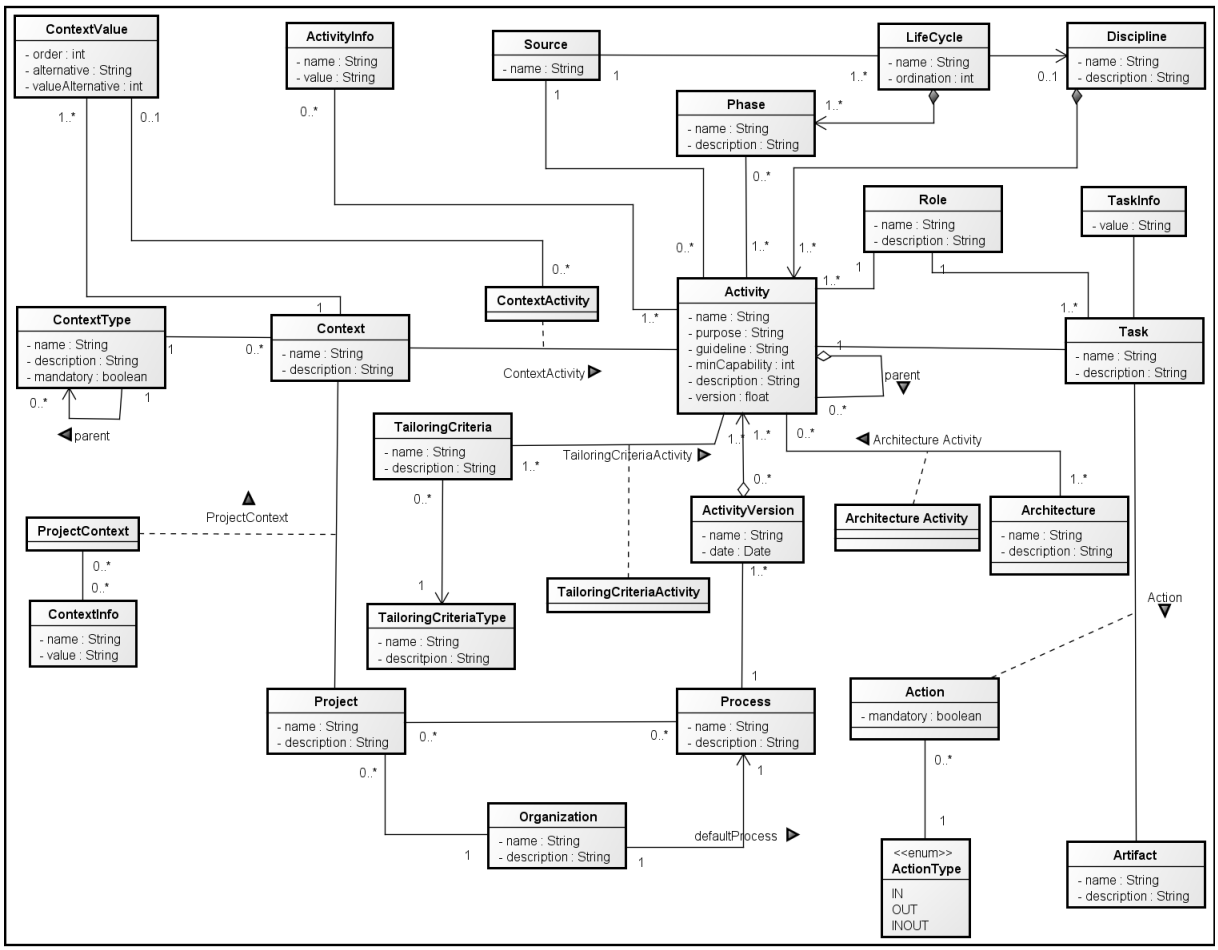

Figura 3.2 – MfTP – Metamodelo para Adaptação de Processos.

O metamodelo permite a instanciação de elementos do processo, que são utilizados na definição de processos planejados, ágeis e híbridos, bem como na customização de processos para o uso em projetos específicos. Os elementos de processos instanciados são armazenados em um repositório e combinados para produzir diferentes modelos de processos de software.

Do ponto de vista de gestão, o ciclo de vida (*LifeCycle* class) é composto por uma série de fases (*Phase* class) e cada atividade possui uma origem (*Source* class) completado por um marco. Do ponto de vista técnico, o ciclo de vida é composto de várias disciplinas (*Discipline* class).

Disciplinas são formadas por um conjunto de atividades (*Activity* class) e associadas a um conjunto de papéis (*Role* class). As atividades são formadas por tarefas (*Task* class) que os papéis executam.

Artefatos (*Artifact* class) são produtos de trabalho gerados durante a execução de tarefas associadas com o desenvolvimento do software. Eles podem ser modelos, planos, *releases* de software, relatórios, entre outros. Para cada tarefa, ações são definidas (*Action* class) representando o tipo de interação entre artefato e tarefa. As ações podem ser: *in*, *out* e *in/out*. Um tipo de ação "*in*"indica a leitura de um artefato, enquanto que uma ação do tipo "*out*"significa a criação de um artefato. Um tipo de ação "*in/out*"indica que ocorreram modificações no artefato.

A adaptação de processos considera a caracterização das atividades e requisitos de adaptação. Pode-se definir um ou mais contextos (*Context* class) para cada atividade. Em tipo de contexto (*ContextType* class) são definidos vários contextos associados a uma atividade e os valores de cada contexto (*ContextValue* class). Pode-se, por exemplo, contextualizar a partir do *Octopus Model* (KRUCHTEN, 2013), fatores de Boehm e Turner (2003), entre outras formas de contextualização.

Requisitos de adaptação (*TailoringCriteria* class) definem os requisitos que podem ser satisfeitos pelo processo da instanciação de uma determinada atividade. Pode-se definir um ou mais tipos de requisitos de adaptação (*TailoringCriteriaType* class) para cada atividade.

Para cada projeto (*Project* class) existe uma organização associada (*Organization* class) e um ou mais processos (*Process* class). Cada projeto possui seu próprio contexto situacional associado. Cada processo possui o seu versionamento das atividades (*ActivityVersion* class) devidamente caracterizadas e contextualizadas.

Arquiteturas (*Architecture* class) são formadas por um conjunto de atividades. Cada atividade é obrigatória ou opcional e concreta ou abstrata, sendo que cada atividade pertence a

uma ou mais arquiteturas (*ArchitectureActivity* class).

Na Figura 3.3, pode-se visualizar a associação entre a atividade (*Activity* class) "*Analyze the problem*" com os elementos de processo usados para descrevê-la. Esta atividade está associada com a fase "*Inception*", a disciplina "*Requirements*", do ciclo de vida do RUP. Esta atividade tem entre uma de suas tarefas "*Develop Vision*". "*Develop Vision*" tem como seus artefatos de entrada "*Iteration Plan*", "*Stakeholder Requests*" e "*Business Case*". Como artefato de saída "*Requirements Attributes*". "*Vision*" é um artefato de entrada e saída, ou seja, ocorreu uma mudança neste artefato. Todas as tarefas são executadas pelo papel do "*System Analyst*". A atividade "*Analyze the Problem*" foi contextualizada usando um contexto situacional, conforme definido pelo *Octopus Model*. Esta atividade pode ser usada para evitar o risco de "*Misunderstanding the Requirements*".

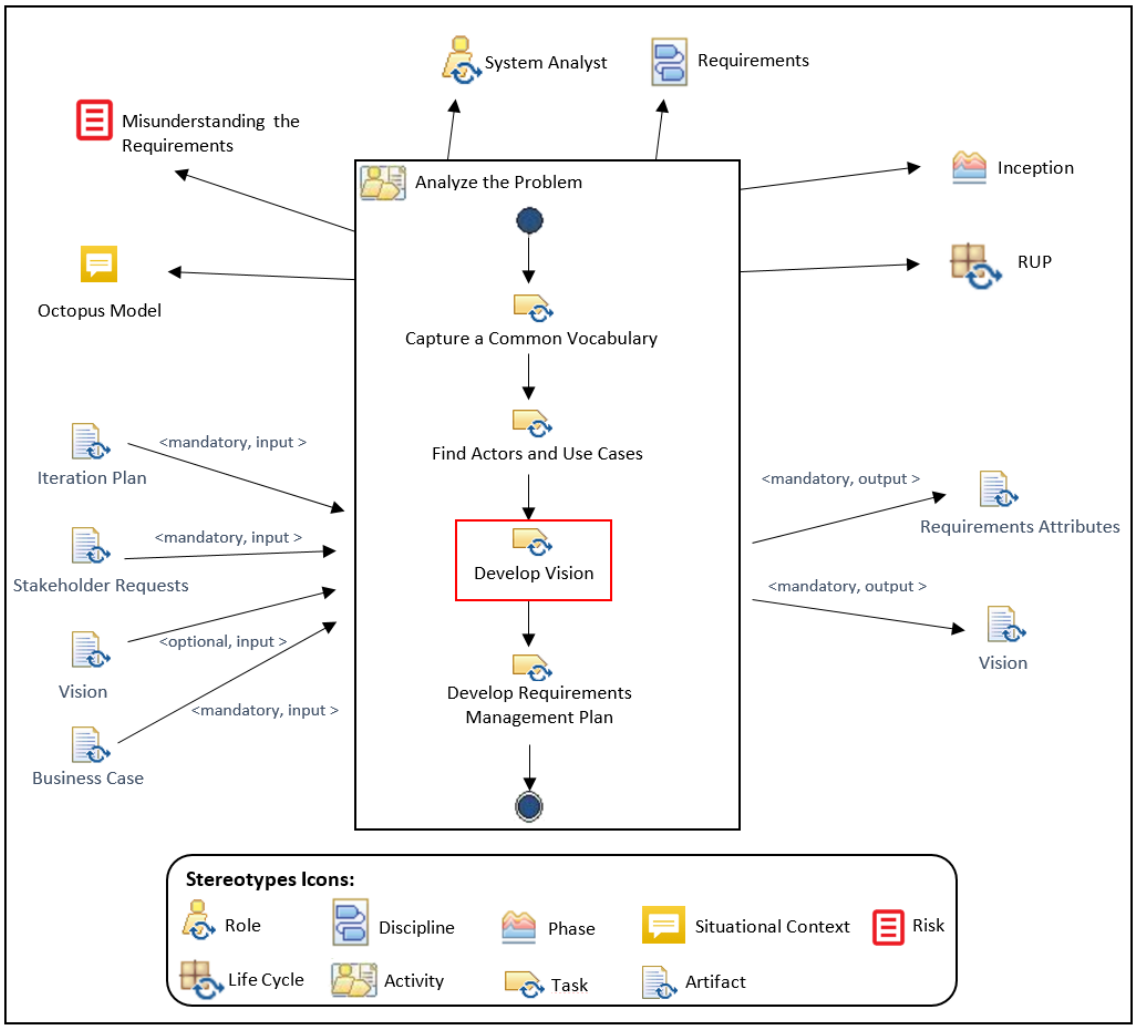

Figura 3.3 – Elementos de processo instanciados para representar a atividade "*Analyze the Problem*".

Os elementos do metamodelo descritos na Figura 3.2, foram instanciados na Figura 3.3. Eles são baseados nos estereótipos do SPEM 2.0 (OMG, 2008). Como o SPEM não contém estereótipos para Riscos e Contexto Situacional foram criados dois estereótipos com a finalidade de representá-los.

3.2 Contextualização de Projetos

A contextualização de projetos é utilizada para definir as principais características dos projetos que precisam ser considerados na adaptação de processos de software. O objetivo da contextualização é definir se elementos de processo ágeis ou planejados são mais indicados ao projeto em questão. O contexto do projeto é comparado ao contexto de cada elemento de processo armazenado na base de conhecimento, visando identificar elementos de processo com contexto mais similar ao contexto do projeto que satisfazem um determinado critério.

Nos estudos de caso realizados optou-se por utilizar o *Octopus Model* (KRUCHTEN, 2013), para definição das características do projeto. O *Octopus Model* descreve oito fatores que afetam significantemente o desenvolvimento de software, e para cada fator são listados atributos que o caracterizam.

Os fatores do *Octopus Model* e respectivos atributos são: a) ***size*** (*small, medium, large*); b) ***stable architecture*** (*stable, changed, new*); c) ***business model*** (*in house, commercial, component of a large system*); d) ***team distribution*** (*collocated, different teams, geographically distributed*); e) ***rate of change*** (*% in a month*) (*over 30, between 10 and 30, less than 10*); f) ***age of system*** (*new development, maintenance, evolution of legacy system*); g) ***criticality*** (*loss of comfort, loss of money, deaths*) e h) ***control*** (*dynamic/flexible, simple rules, mechanical/formal*). O primeiro atributo listado indica o melhor contexto para projetos ágeis e o último para projetos planejados, o atributo do meio indica que pode ser indicado para ambos os projetos.

Na Tabela 3.1, pode-se visualizar dois exemplos de caracterização de projetos, um com contexto planejado e outro ágil. Por exemplo, para projetos de tamanho grande, com distribuição geográfica da equipe, com necessidade de regras formais para comunicação, tem-se uma caracterização do projeto voltada para o contexto planejado.

Já em equipes pequenas, com distribuição local da equipe, com utilização de regras simples para comunicação, tem-se uma caracterização do projeto voltada para o contexto ágil. Esses fatores de caracterização são sugestões e podem ser adequados a realidade da organização.

Tabela 3.1 – Exemplos de caracterização de projetos utilizando contexto planejado e ágil.

Fatores	Atributos	
	Contexto planejado	Contexto ágil
Tamanho	Grande	Pequena
Criticidade	Perda de dinheiro	Perda de conforto
Arquitetura estável	Nova	Estável
Modelo de Negócios	Componente de um grande sistema	Sob medida para um cliente
Distribuição do Time	Distribuição geográfica	Local
Taxa de mudanças (% no mês)	Menos de 10	Mais que 30
Idade do sistema	Evolução de um sistema legado	Novo desenvolvimento
Governança	Mecânica/Formal	Regras simples

3.3 Requisitos de Adaptação

Normas e modelos de processo propõem que os processos de software sejam adaptados para satisfazer às necessidades específicas de projetos (CHRISSIS; KONRAD; SHRUM, 2011). Neste trabalho as necessidades específicas de um projeto podem ser derivadas de riscos, qualidade ou segurança.

Outra fonte de necessidades específicas pode ser riscos de um projeto. Independentemente do tipo de necessidade específica, a ideia é selecionar elementos de processo que venham a satisfazer essas necessidades específicas e incorporá-los ao processo.

Na Tabela 3.2, pode-se visualizar exemplos de riscos e atividades associadas para prevenção dos mesmos. A associação entre as atividades e os riscos são sugestões propostas por (FONTOURA; PRICE, 2008; FONTOURA, 2006; HARTMANN; FONTOURA; PRICE, 2005). Diferentes atividades podem ser associadas a um risco.

Tabela 3.2 – Associação entre riscos e atividades.

Riscos de Projeto	Atividades
Misunderstanding the Requirements	Scenarios Define Problem (COPLIEN; ALEXANDER, 1996)
	Implied Requirement (COPLIEN; ALEXANDER, 1996)
	Build Prototype (COPLIEN; ALEXANDER, 1996)
	Software Configuration Management (CHRISSIS; KONRAD; SHRUM, 2011)
	Early and Regular Deliver RUP (IBM COPORATION, 2007)
	Planning Game (BECK, 2004)
	Constant Refactoring (BECK, 2004)
	Write User Story (BECK, 2004)
	Divide User Story (BECK, 2004)
Requirements Instability	Implied Requirement (COPLIEN; ALEXANDER, 1996)
	Scenarios Define Problem (COPLIEN; ALEXANDER, 1996)
	Build Prototype (COPLIEN; ALEXANDER, 1996)
	Early and Regular Deliver RUP (IBM COPORATION, 2007)
	Early and Regular Deliver XP (COPLIEN; ALEXANDER, 1996)
	Simple Design (BECK, 2004)
	On Site Customer (BECK, 2004)

Por exemplo, para prevenção do risco "falta de compreensão dos requisitos" (*Misunderstanding the Requirements*), são recuperadas as seguintes atividades: "*Scenarios Define Problem*", "*Implied Requirement*", "*Build Prototype*", "*Software Configuration Management*", "*Early and Regular Deliver RUP*", "*Planning Game*", "*Constant Refactoring*", "*Write User*

Story" e "*Divide User Story*". De acordo com a contextualização do projeto, para cada atividade é atribuída uma probabilidade, indicando desta forma, as mais adequadas para a caracterização do projeto em questão, ou seja, se o projeto possui caracterização voltada para abordagens planejadas, as atividades recuperadas com características de abordagens planejadas terão uma probabilidade maior em relação as atividades recuperadas com características de abordagens ágeis.

3.4 Arquiteturas de Processos

Nesta dissertação a arquitetura de processos é descrita por meio de componentes opcionais "*optional*" (OP) ou obrigatórios "*mandatory*" (M) e concretos "*concrete*" (C) ou abstratos "*abstract*" (AB). Componentes obrigatórios são utilizados para definir os elementos (atividades, tarefas, papéis e artefatos) que devem estar presentes em todos os processos organizacionais, isto é, descrevem um conjunto mínimo de elementos, que segundo a literatura, devem existir em um processo de software para que ele seja viável. Componentes opcionais definem elementos que podem ou não ser utilizados para compor o processo do projeto.

Seguindo a proposta de Barreto *et al.* (2011), pode-se ter componentes concretos e abstratos, aumentando assim a variabilidade para a Linha de Processos de Software. Um componente concreto não permite qualquer tipo de variabilidade. Um componente abstrato é uma maneira de representar as variabilidades, sendo considerado um ponto de variação. Ele pode ser diretamente substituído por componentes concretos (variantes) ou componentes abstratos, formando hierarquias de componentes abstratos, que em algum momento precisam ser substituídos por componentes concretos.

Neste trabalho, uma arquitetura é definida a partir do RUP (IBM COPORATION, 2007), Pressman (PRESSMAN, 2010) e padrões de processo (COPLIEN, JAMES O AND ALEXANDER, 1996). Para cada disciplina do RUP, são definidos componentes contendo sua finalidade e características situacionais, bem como seus relacionamentos. Para cada componente definido na arquitetura são recuperadas atividades que possuem caracterização semelhantes. Pode-se também definir uma arquitetura a partir de abordagens ágeis, ou uma arquitetura contendo elementos de abordagens planejadas e ágeis, ou seja, uma arquitetura híbrida.

Um exemplo de uma arquitetura definida para a disciplina de requisitos pode ser visualizada na Figura 3.4.

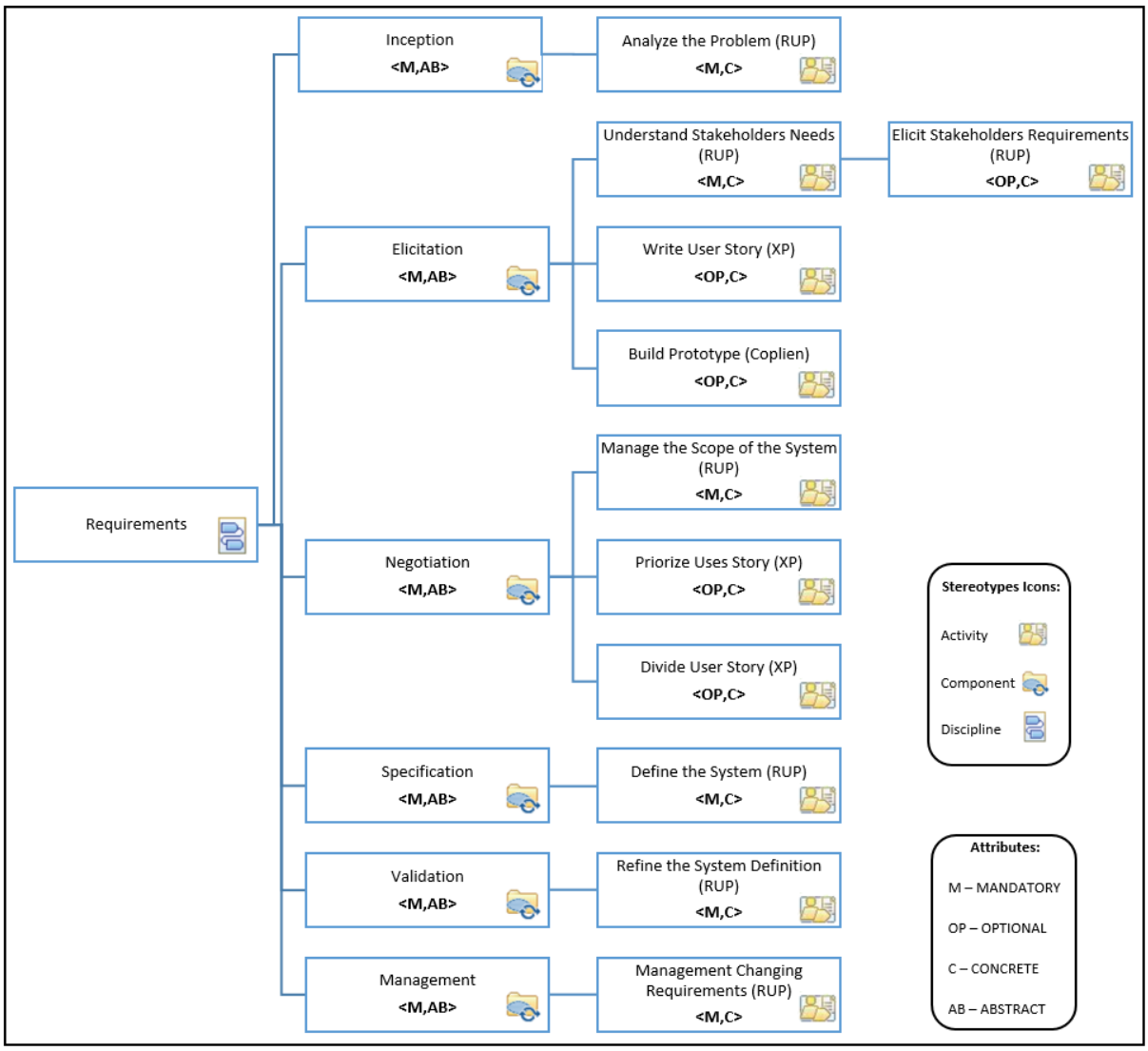

Figura 3.4 – Exemplo de arquitetura definida para a disciplina de requisitos para a LPrS.

Essa arquitetura deve ser utilizada no desenvolvimento de novos sistemas que utilizem uma abordagem planejada. É possível observar a variabilidade na estrutura a partir de uso de componentes obrigatórios (*mandatory*) (M), opcionais (*optional*) (OP), concretos (*concrete*) (C) e abstratos (*abstract*) (AB).

Por exemplo, para o componente abstrato "*Elicitation*" podem ser recuperadas as atividades "*Understand Stakeholder Needs*", "*Elicit Stakeholder Requirements*", definidas a partir do RUP, ou "*Write User Story*"e "*Build Prototype*", definidas a partir do XP. "*Build Prototype*"e "*Write User Story*"são componentes opcionais e concretos. "*Understand Stakeholder Needs*"é um componente obrigatório e concreto, isto é, deve ser utilizado no processo e não existe ne-

nhuma atividade com propósito e contexto similar a este componente."*Elicit Stakeholder Requirements*"é um componente opcional e concreto, pode-se ou não utiliza-lo no processo. Este componente tem como objetivo compreender as necessidades do cliente. Assim, o engenheiro de processos pode selecionar os componentes (atividades) para compor o processo de software mais adequado às necessidades do projeto.

Pode-se visualizar na Figura 3.5, a atividade "*Analyze the Problem*", um exemplo de componente de arquitetura da Figura 3.4, proporcionando uma representação mais detalhada para a atividade.

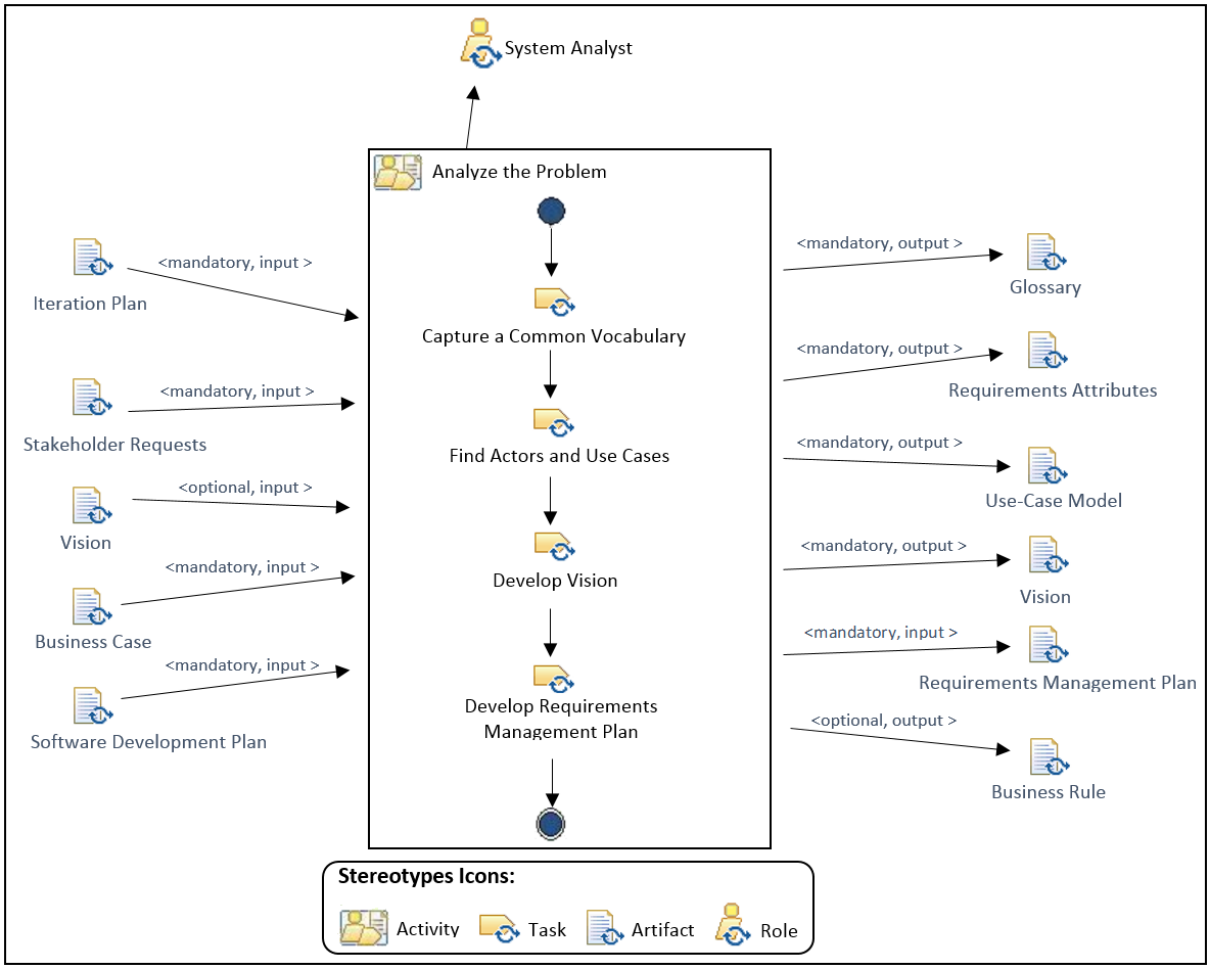

Figura 3.5 – Detalhes da atividade "*Analyze the Problem*".

Pode-se observar que a atividade contém suas tarefas, papel e artefatos de entrada e saída. Por exemplo, a atividade "*Analyze the Problem*" contém a tarefa "*Find Actors and Use Cases*" realizada pelo "*System Analyst*", com o artefato de entrada obrigatório "*Iteration Plan*" e "*Stakeholders Request*" e artefato de entrada opcional "*Vision*". Como artefato de saída obrigatório "*Use Case Model*" e "*Requirements Attributes*". A obrigatoriedade dos artefatos de

entrada e saída é definida pela literatura. Para realizar uma determinada tarefa, é necessário que um artefato de entrada obrigatório seja utilizado, obtendo um artefato de saída obrigatório ou opcional. No caso do artefato ser opcional, pode-se ou não utilizar o artefato como entrada ou saída. Quando um artefato é de entrada e saída, significa que o artefato foi modificado. O objetivo desta atividade é obter o acordo sobre o problema a ser resolvido. Análise do problema consiste em identificar as partes envolvidas, definindo os limites e identificando as restrições impostas pelo sistema.

Uma das questões críticas das abordagens de adaptação de processos de software está relacionada com a escolha das atividades mais apropriadas de acordo com as características do projeto. Na abordagem apresentada neste trabalho, os componentes previamente definidos para a arquitetura são recuperados de acordo com os requisitos de adaptação para criar a linha de processo de software. Com isso, é priorizado por meio de um algoritmo de priorização baseado na técnica AHP (*Analytic Hierarchy Process*).

A arquitetura e as linhas de processos para as demais disciplinas serão descritas no anexo A. Todas as arquiteturas definidas são sugestões deste trabalho e da literatura. Elas contêm componentes contendo sua finalidade e características, bem como os seus relacionamentos e precedências. Para cada uma destas arquiteturas, tem-se a recuperação do repositório, atividades que possuem caracterizações semelhantes.

3.5 *Analytic Hierarchy Process* (AHP)

A AHP (*Analytic Hierarchy Process*) é um método matemático para apoiar a teoria de decisão que trabalha com alternativas e com um objetivo global, ou seja, a probabilidade numérica de cada alternativa é calculada e quanto maior for esta probabilidade, maior é a chance de uma alternativa satisfazer a meta global (VARGAS, 2010). Neste trabalho às alternativas correspondem as atividades armazenadas no repositório de componentes, enquanto a meta global é priorizar as atividades mais adequadas para satisfazer os requisitos de adaptação, definidos na Seção 3.3.

A priorização de atividades utiliza um algoritmo que organiza as atividades previamente definidas em ordem decrescente. A classificação mais elevada significa que a atividade é mais adequada em relação ao contexto do projeto. Para priorizar as atividades, primeiramente, o engenheiro de processo estabelece pesos relativos aos critérios de comparação, a fim de determinar a importância relativa entre eles. A escala de importância relativa utilizada neste trabalho

foi proposta por Saaty (2005). Os seus valores variam de 1 (igualmente preferida) até 9 (extremamente preferida). Após isso as atividades candidatas são comparadas duas a duas para cada um dos fatores de *Octopus Model*.

O envolvimento do engenheiro de processo é definir os pesos relativos entre os critérios de comparação. A partir disso o algoritmo calcula a probabilidade relativa de cada uma das atividades em relação a cada um dos fatores dos *Octopus Model*. A probabilidade relativa significa a chance que a atividade tem de cumprir com êxito o objetivo geral para cada critério de comparação. Depois disso, a probabilidade final é calculada para cada atividade em relação ao objetivo geral, definindo, assim, uma lista de atividades priorizadas.

Na Tabela 3.3, pode-se visualizar como um exemplo, um conjunto de atividades recuperadas e priorizadas. Pode-se observar a probabilidade definida para cada atividade, utilizando o método AHP. Para cada atividade a probabilidade de acordo com o contexto situacional é definida e calculada para o projeto.

Tabela 3.3 – Conjunto de atividades recuperadas e priorizadas.

Requisitos de Adaptação	Atividades para prevenir riscos	Probabilidade
Misunderstanding the Requirements	Scenarios Define Problem (COPLIEN; ALEXANDER, 1996)	18.682%
	Implied Requirement (COPLIEN; ALEXANDER, 1996)	18.682%
	Build Prototype (COPLIEN; ALEXANDER, 1996)	18.682%
	Early and Regular Deliver RUP (IBM COPORATION, 2007)	15.318%
	Constant Refactoring (BECK, 2004)	2.261%
	Write User Story (BECK, 2004)	2.261%
	Divide User Story (BECK, 2004)	2.261%
Requirements Instability	Implied Requirement (COPLIEN; ALEXANDER, 1996)	18.682%
	Scenarios Define Problem (COPLIEN; ALEXANDER, 1996)	18.682%
	Build Prototype (COPLIEN; ALEXANDER, 1996)	18.682%
	Early and Regular Deliver RUP (IBM COPORATION, 2007)	15.318%
	Early and Regular Deliver XP (COPLIEN; ALEXANDER, 1996)	2.261%
	On Site Customer (BECK, 2004)	2.261%

3.6 Linha de Processos de Software adaptada

A partir da arquitetura de processos definida para este trabalho apresentada na Seção 3.4 e das atividades recuperadas, priorizadas e selecionadas com o método matemático AHP

apresentado na Seção 3.5, tem-se a elaboração da Linha de Processo de Software adaptada de acordo com as características situacionais do projeto. Pode-se visualizar na Figura 3.6, a Linha de Processos de Software para a disciplina de Requisitos.

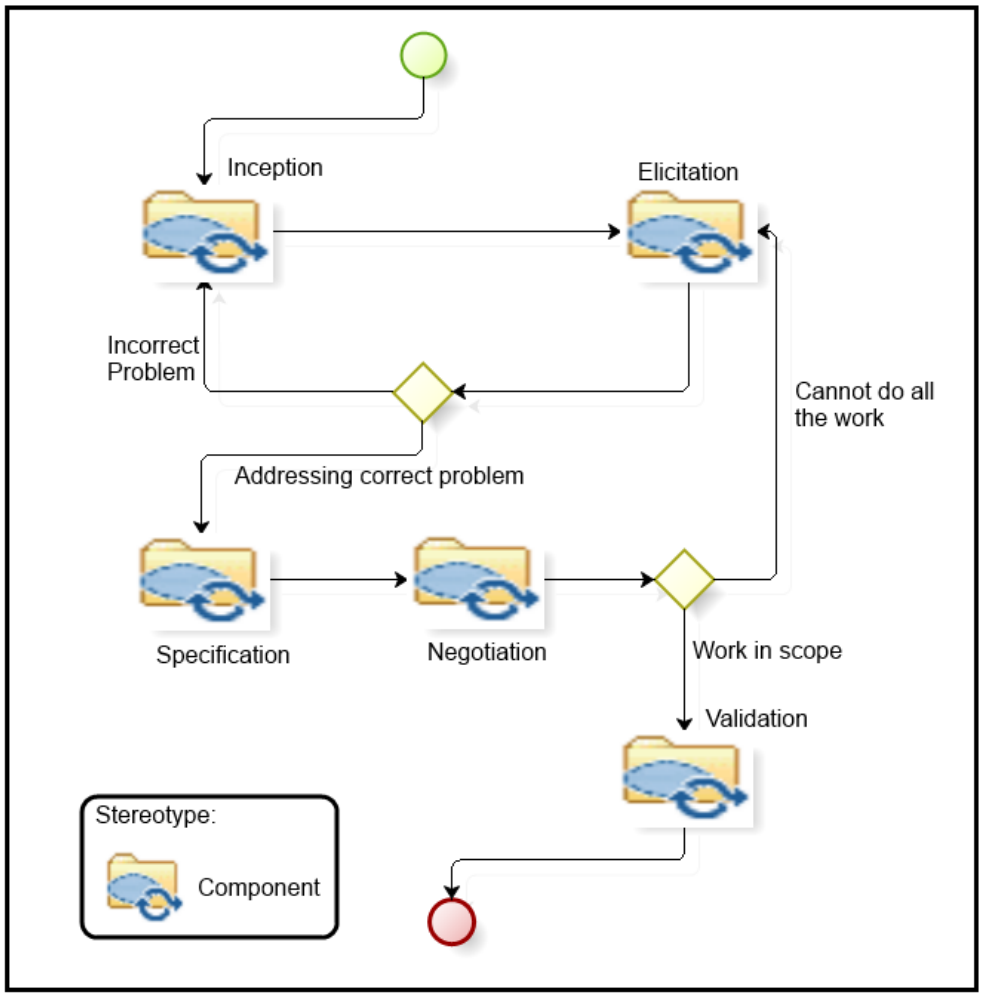

Figura 3.6 – Exemplo de Linha de Processo de Software para a disciplina de Requisitos.

Pode-se observar que ela inicia no componente "*Inception*", passa pelo componente "*Elicitation*", "*Specification*", "*Negotiation*"e "*Validation*. Para cada um destes componentes é selecionada uma ou mais atividades que contenham contexto situacional similar, como já ilustrado na Figura 3.4 da Seção 3.4.

As Linhas de Processos de Software elaboradas a partir das arquiteturas previamente definidas para as demais disciplinas são apresentadas no anexo A.

Neste capítulo foi apresentado o metamodelo elaborado para esta dissertação e a sistemática proposta para adaptação de processos utilizando Linha de Processos de Software, com as etapas de contextualização de projetos e requisitos de adaptação, definições de utilização das

arquiteturas de adaptação com seus componentes obrigatórios, opcionais, abstratos e concretos e como são recuperados os elementos para compor as Linhas de Processos de Software adaptadas. O capítulo 4 abordará a ferramenta de apoio para este trabalho, com seu módulo principal e o módulo para adaptação de processo de software utilizando Linha de Processos de Software.

4 FERRAMENTA DE APOIO MFTPT

Com o intuito de validar a proposta deste trabalho, foi desenvolvida uma ferramenta para apoiar a sistemática proposta para adaptação de processos de software. A ferramenta contém dois módulos, que são: o módulo principal e o módulo para adaptação de processos de software.

O módulo principal contém todas as funcionalidades para adaptação de processos, desde o registro dos artefatos, tarefas, papéis, atividades, requisitos de adaptação, tipos de contextualização, contextualização das atividades, contextualização do projeto, até a definição das arquiteturas de processo. O módulo para adaptação de processos de software é dividido em duas abordagens distintas, denominadas *Quality Validate Strategy* (QaVaS) e *Software Process Lines* (SPrL).

A primeira foi desenvolvida com base em conceitos de *Situational Method Engineering* e tem como objetivo garantir a consistência e a completude dos fragmentos de processo instanciados a partir do metamodelo MfTP. Essa abordagem faz parte de outro trabalho de dissertação elaborada por Miguel Bauermann Brasil (BRASIL, 2014).

A abordagem SPrL é uma proposta desta dissertação para adaptação de processos baseados em Linhas de Processos de Software. O módulo de adaptação das Linhas de Processo de Software (SPrL) é organizado de acordo com o conjunto de etapas, que são: **i)** definição das características do projeto; **ii)** seleção dos requisitos de adaptação; **iii)** seleção da arquitetura de processos; **iv)** priorização das atividades; e **v)** criação da linha de processos de software adaptada.

Para que a adaptação seja realizada é necessário que os elementos de processo tenham sido cadastrados/definidos e armazenados no repositório. Os elementos de processo são instâncias das classes descritas no metamodelo MfTP, apresentado na Seção 3.1.

A ferramenta foi desenvolvida utilizando-se a linguagem Java para especificação das regras de negócio, *Java Persistence* API (JPA) e *Hibernate* para persistência e acesso a dados; *Java Server Faces* (JSF), HTML, CSS3 e JQuery para compor a camada de apresentação e o banco de dados MySQL para a base de dados da ferramenta.

4.1 Módulo principal

O módulo principal permite o acesso as funcionalidades necessárias para adaptação de processos de software. Pode-se registrar artefatos, tarefas, papéis, atividades, requisitos de

adaptação, tipos de contextualização e seus valores, contextualização das atividades, contextualização do projeto e a definição de arquiteturas para processos.

Pode-se visualizar na Figura 4.1 o menu contendo as funcionalidade da ferramenta (à esquerda). As funcionalidade da ferramenta MfTPt são subdivididas em categorias, que são: *Getting Started*, *Organization*, *Knowledge Base*, *Contextualization*, *Tailoring Criteria*, *Architecture* e *Settings*. A seguir uma breve descrição de cada uma delas:

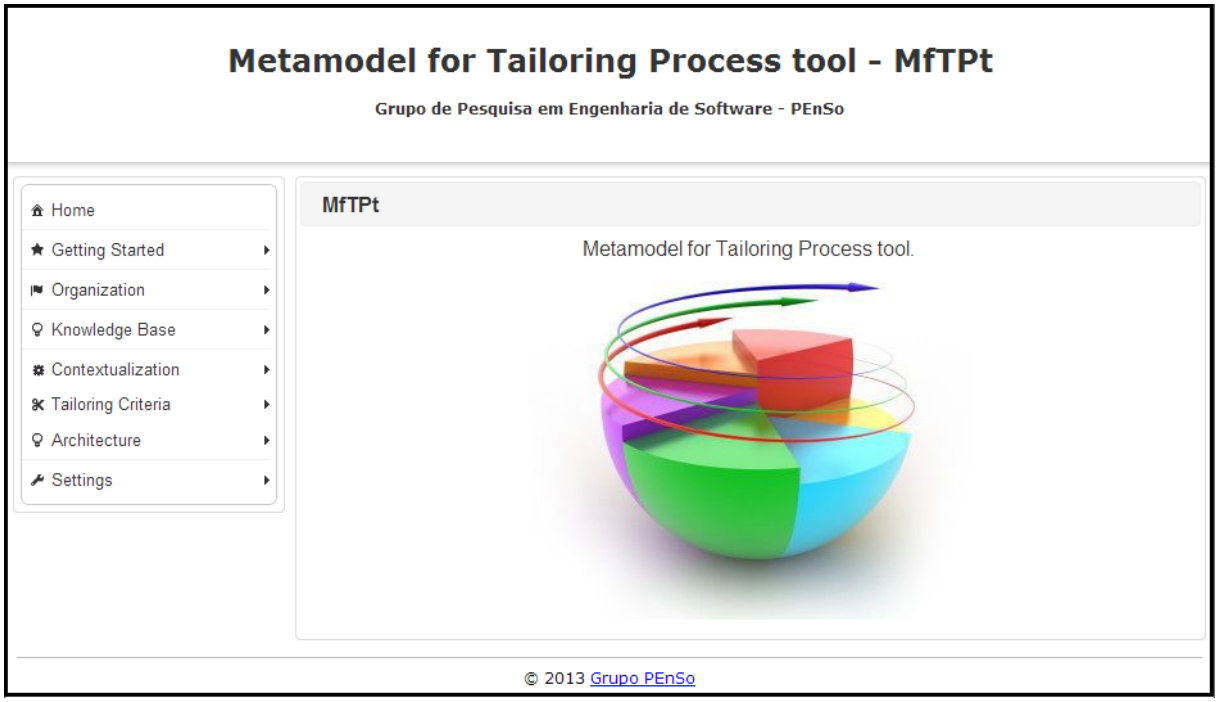

Figura 4.1 – MfTPt – Módulo principal da ferramenta.

Getting Started: descreve as funcionalidade relacionadas a adaptação do processo, ou seja, o usuário pode selecionar uma das duas abordagens para adaptação de processo de software contidas no ferramenta MfTPt. Este trabalho apenas aborda a proposta SPrL, abordagem proposta para adaptação de processos de software baseada em Linha de Processos de Software.

Organization: descreve as funcionalidades relacionadas à organização, tais como: cadastrar nova organização, criar um processo, cadastrar projeto, contextualizar o projeto e gerenciar versionamento dos processos cadastrados.

Na opção ***Organization***, o usuário pode realizar o cadastro da organização com seu nome e descrição. Em *Project*, o usuário pode fazer o cadastramento do projeto, atribuindo uma organização para o mesmo e efetuando a caracterização do projeto. Em *Process*, pode-se visualizar os processos que foram criados e estão relacionados com uma determinada organização e projeto.

Knowledge Base: armazena o conhecimento organizacional utilizado na adaptação dos processos. Descreve as funcionalidades envolvendo a definição das *Action*, *Activity*, *Artifact*, *Discipline*, *Life Cycle*, *Phase*, *Role*, *Source* e *Task*.

Na opção *Activity*, o usuário pode cadastrar as atividades e seus atributos como: nome, descrição, propósito, orientações, origem, fases, disciplina e papel. A opção "*Parent*" é utilizada para definir se a atividade a qual está sendo cadastrada é proveniente de outra atividade cadastrada. Pode-se visualizar na Figura 4.2, o primeiro passo para o cadastramento das atividades.

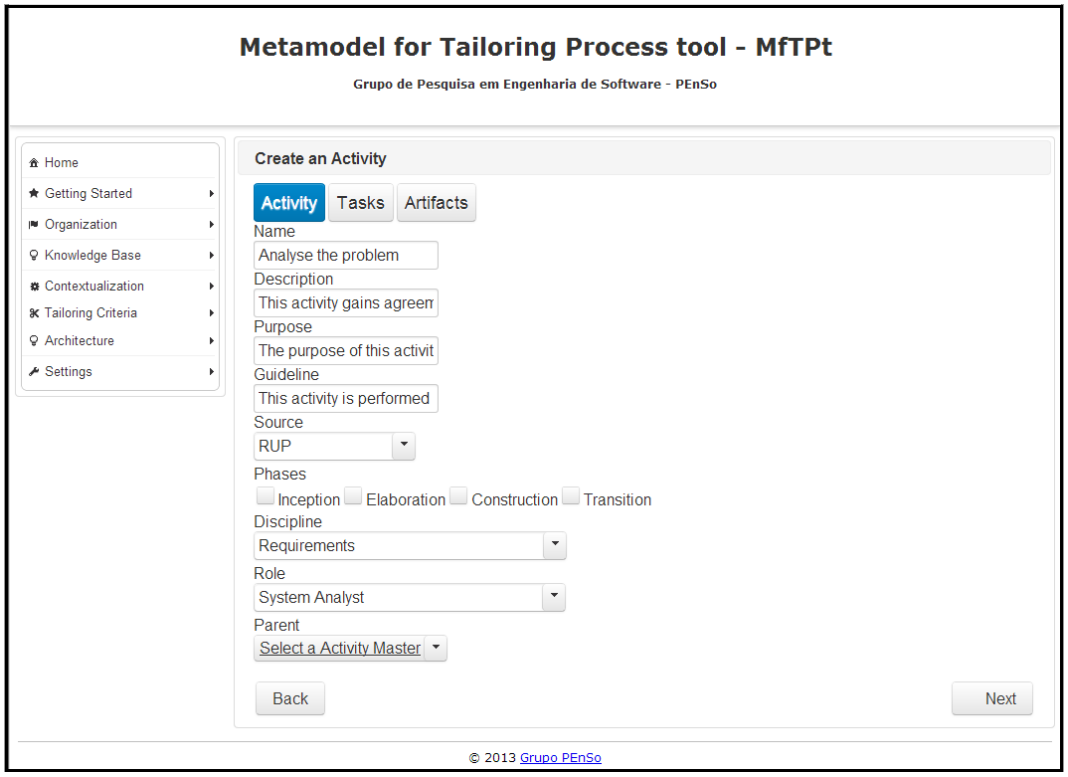

Figura 4.2 – MfTPt - Primeiro passo para criar uma atividade.

Pode-se visualizar na Figura 4.3 o segundo passo para o cadastramento das atividades. Neste passo são selecionadas as tarefas "*Tasks*" que pertencem a respectiva atividade. Por exemplo, para a atividade "*Analyze the problem*", são selecionadas as tarefas "*Tasks*": "*Capture a Common Vocabulary*", "*Develop Requirements Management Plan*", "*Develop Vision*" e "*Find Actors and Use Cases*".

Pode-se visualizar na Figura 4.4 o terceiro passo para o cadastramento das atividades. Neste passo são exibidas as tarefas selecionadas para a atividade em questão, e em seguida associadas aos seus respectivos artefatos "*Artifacts*".

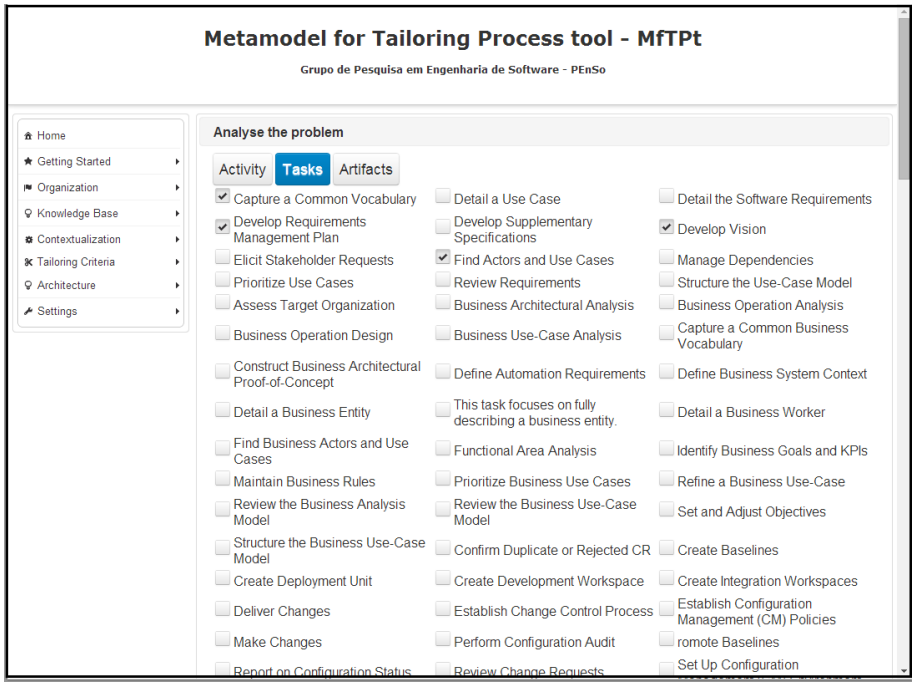

Figura 4.3 – MfTPt – Segundo passo para criar uma atividade, seleção das tarefas.

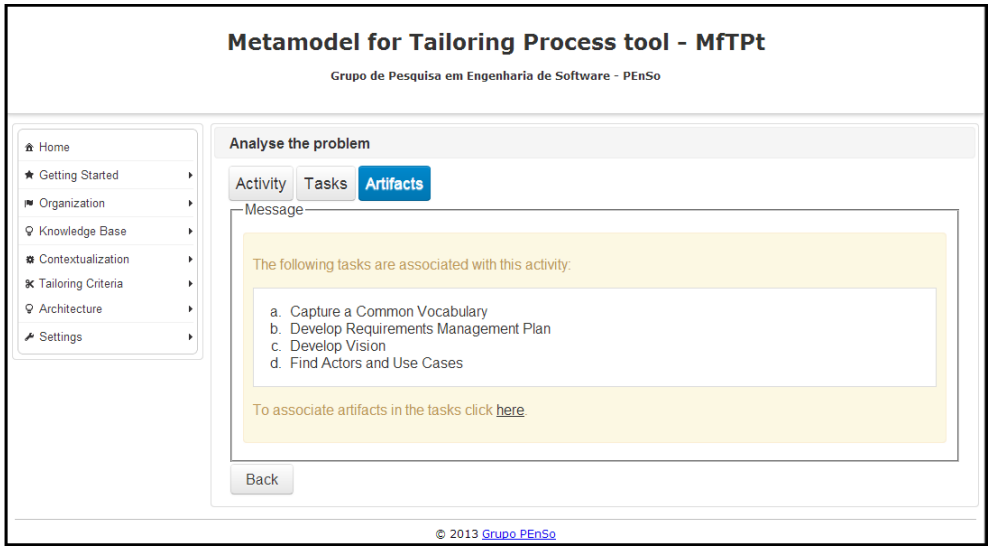

Figura 4.4 – MfTPt - Terceiro passo, tarefas selecionadas e *link* para associação dos artefatos.

Na Figura 4.5 pode-se visualizar, a atividade selecionada com suas respectivas tarefas e seus artefatos. Cada atividade tem uma ou mais tarefas associadas e cada tarefa possuí um ou mais artefatos de entrada "*IN*", de saída "*OUT*" ou modificado "*INOUT*". Cada artefato ainda é definido como obrigatório "*Mandatory*" ou opcional "*Optional*". Por exemplo, para a atividade "*Analyze the problem*" tem-se a tarefa "*Capture a Common Vocabulary*", o qual tem como artefato opcional "*Optional*" de entrada "*Business Analysis Model*".

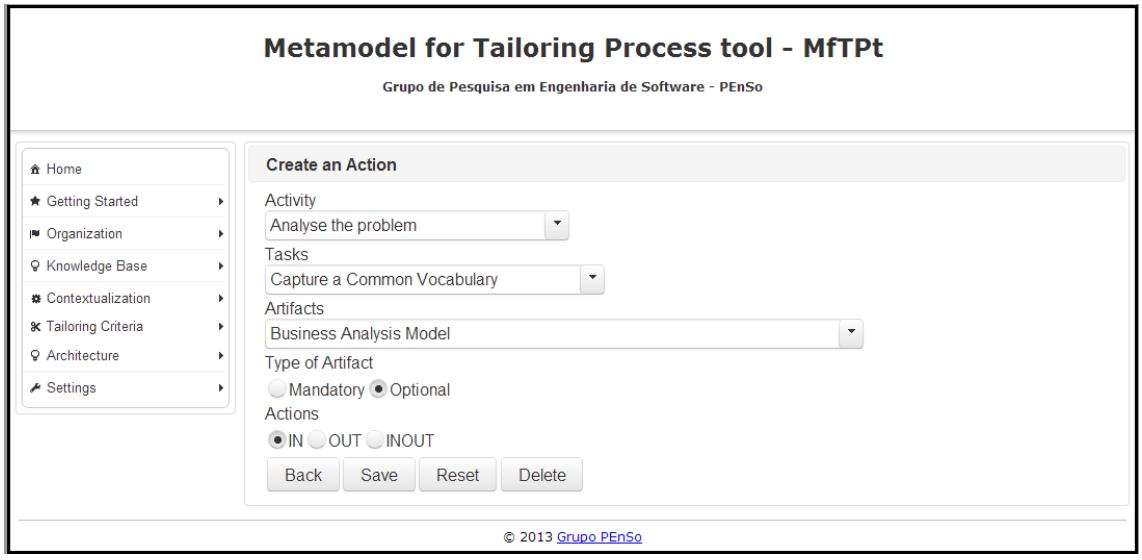

Figura 4.5 – MfTPt – Terceiro passo, associação entre atividades, tarefas e artefatos.

Para cada atividade pode-se definir um ou mais contextos e um ou mais requisitos de adaptação e define-se a qual arquitetura a atividade pertence, sendo que a atividade pode pertencer a mais de uma arquitetura. Pode-se visualizar na Figura 4.6, as opções "*Edit context this activity*", "*Edit tailoring criteria this activity*" e "*Associate activity to an architecture*".

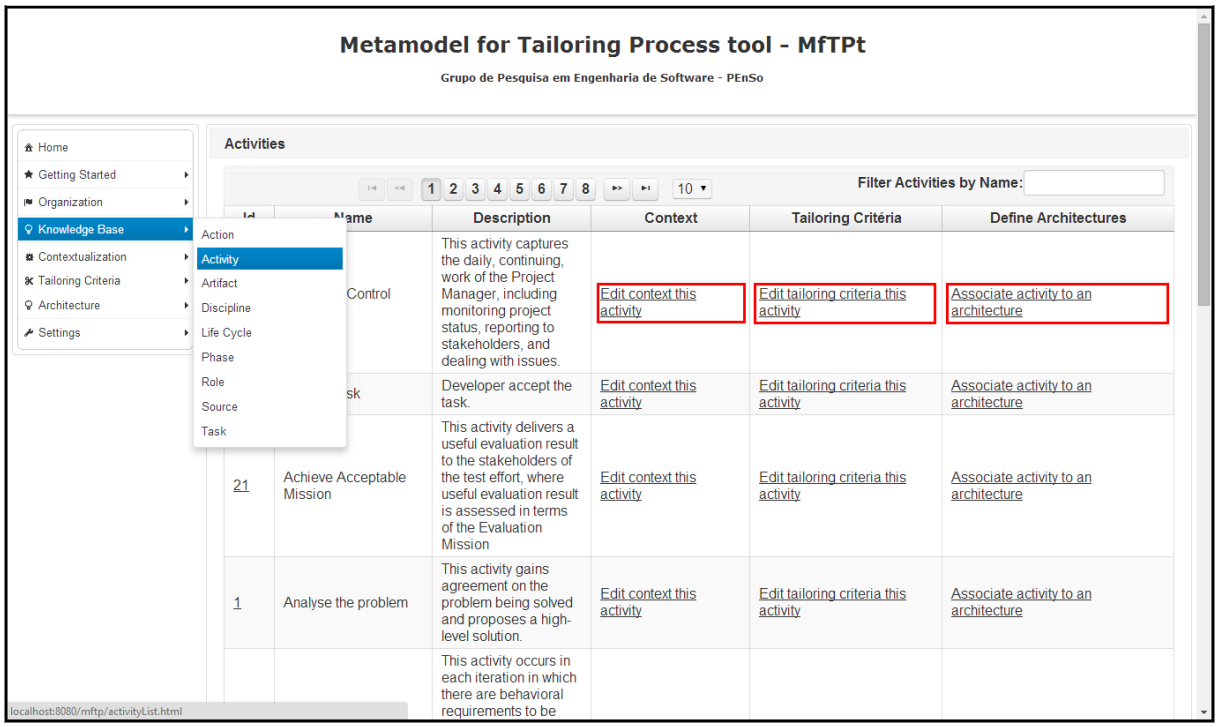

Figura 4.6 – MfTPt – Definição do contexto, critério de adaptação e associação da atividade a uma arquitetura.

Na opção "*Edit context this activity*", destacada em vermelho na Figura 4.6, define-se um ou mais contextos para cada atividade, por exemplo, seleciona-se a contextualização utilizando o *Octopus Model*, definindo valores para cada um dos seus oito fatores para a atividade em questão. Pode-se visualizar na Figura 4.7 a contextualização da atividade.

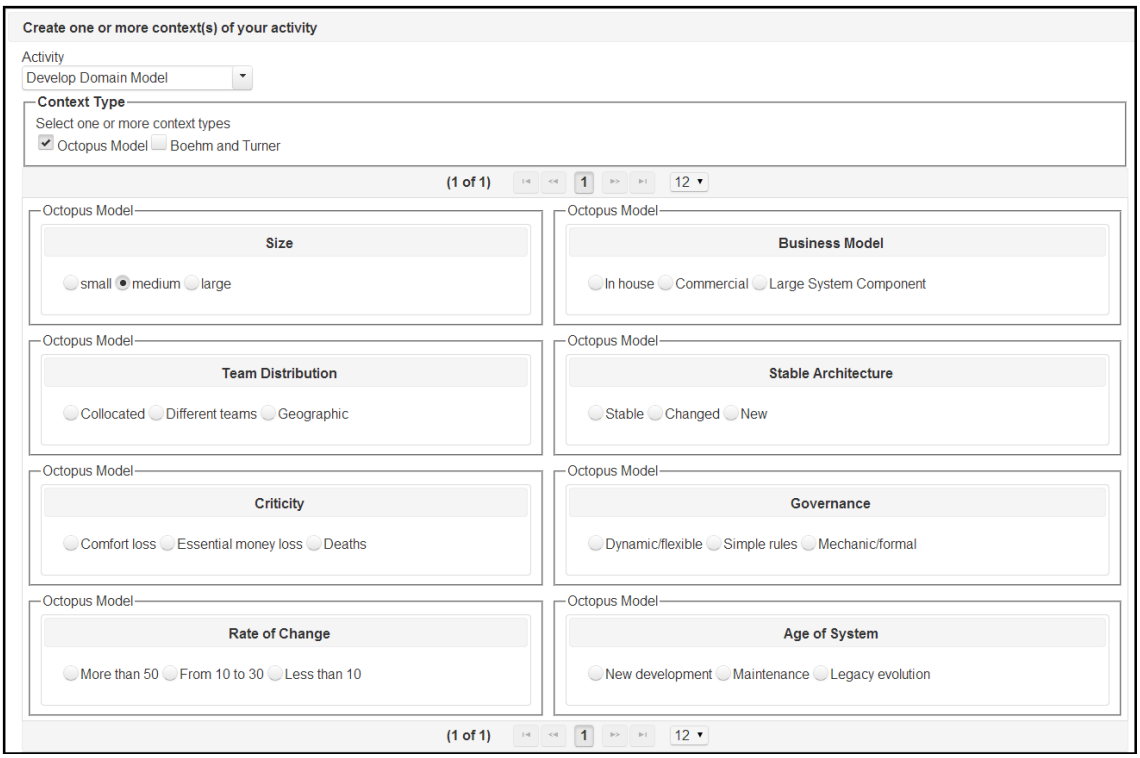

Figura 4.7 – MfTPt - Contextualização da atividade.

Na opção "*Edit tailoring criteria this activity*", destacada em vermelho na Figura 4.6, define-se um ou mais requisitos de adaptação que a atividade ajuda a satisfazer, por exemplo, a atividade "*Develop Domain Model*" ajuda a prevenir o risco "*Misunderstanding the Requirements*". Pode-se visualizar na Figura 4.8, a definição dos requisitos de adaptação associados a atividade.

Na opção "*Associate activity to an architecture*", destacada em vermelho na Figura 4.6, é associado a atividade para uma arquitetura. Nessa opção são também definidos os atributos da atividade considerando a arquitetura na qual ela é associada. Esses atributos indicam se a atividade é obrigatória ou opcional e concreta ou abstrata. Pode-se atribuir uma atividade para mais de uma arquitetura.

Contextualization: descreve as funcionalidades relacionadas a caracterização do projeto, como tipos de caracterização e seus valores relacionados. O submenu *Organization* contém as opções *Context*, *Context Value* e *Context Type*.

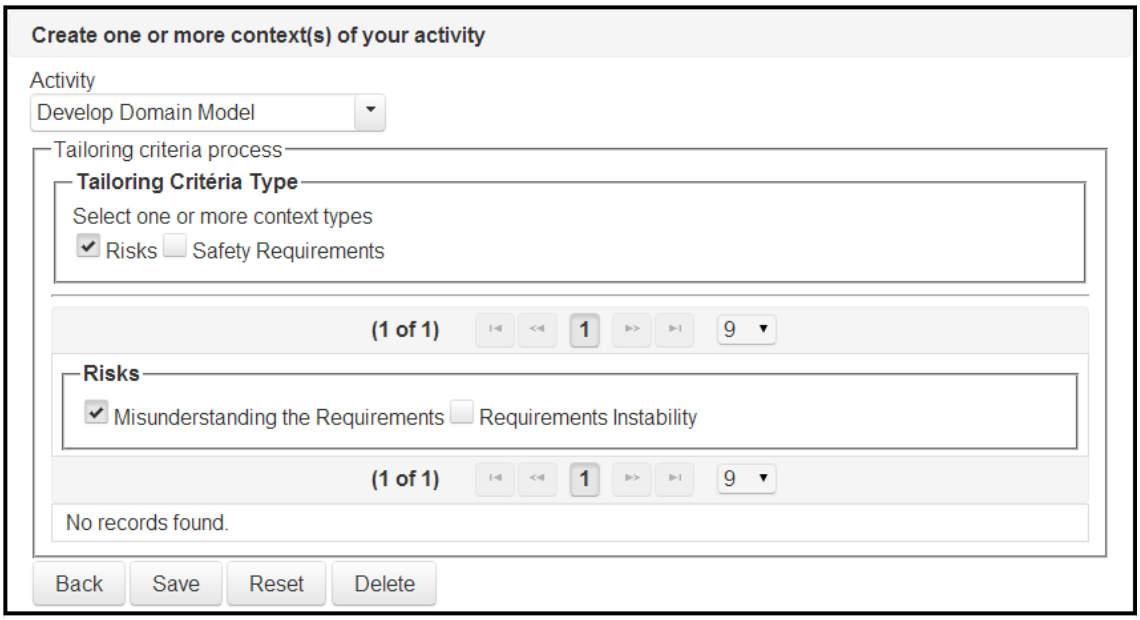

Figura 4.8 – MfTPt - Definição dos requisitos de adaptação para a atividade.

Em *Context Type*, o usuário pode realizar o cadastro dos tipos de contexto, por exemplo, *Octopus Model* ou Boehm e Turner. Tem-se em *Context* a definição dos fatores relacionados a cada tipo de contextualização, por exemplo, "*size*", "*stable architecture*", "*team distribution*", estes são fatores para o tipo de contexto cadastrado, no caso, para o *Octopus Model*. *Context Value* define os valores para cada fator do *Context*, por exemplo, "*size*" contém os valores *small*, *medium* e *large*.

Tailoring Criteria: descreve as funcionalidades relacionadas aos tipos de requisitos de adaptação. O submenu *Tailoring Criteria*, contém as opções *Criteria* e *Criteria Type*.

Na opção *Criteria Type*, pode-se cadastrar diferentes tipos de requisitos de adaptação, por exemplo, riscos, requisitos de segurança, qualidade, entre outros. *Criteria* define os diferentes ocorrências para cada tipo de requisitos de adaptação, por exemplo, para riscos tem-se "*Misunderstanding the Requirements*", "*Requirements Instability*", entre outros.

Architecture: descreve diferentes arquiteturas definidas com seus componentes obrigatórios ou opcionais e concretos ou abstratos. Na opção *Architecture*, o usuário tem a listagem de todas as arquiteturas cadastradas, e a opção de cadastramento de uma nova arquitetura. Pode-se visualizar na Figura 4.9, a interface onde o cadastramento de uma arquitetura pode ser realizado.

Na opção *Define*, tem-se a definição das atividades que irão compor a arquitetura selecionada. Pode-se visualizar na Figura 4.10, uma arquitetura cadastrada com suas respectivas atividades.

Figura 4.9 – MfTPt - *Create an architecture*.

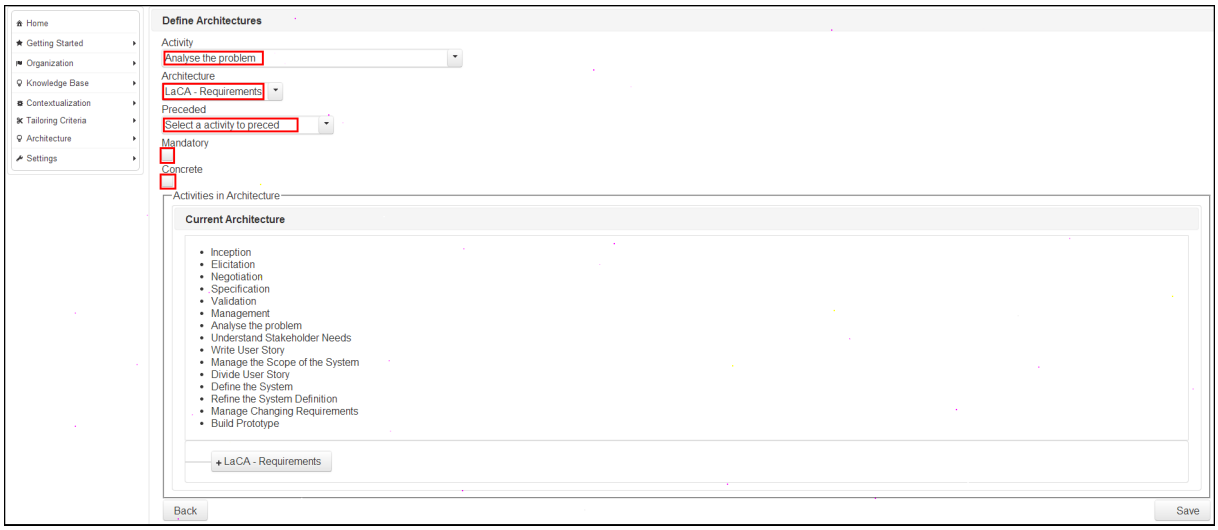

Figura 4.10 – MfTPt - *Define Architecture*.

Pode-se observar que no campo "*Activity*" é selecionada a atividade em questão, por exemplo, "*Analyze the problem*", no campo "*Architecture*" é selecionada para qual arquitetura a atividade selecionada irá pertencer, no campo "*Preceded*" é selecionada a atividade que irá preceder a atividade selecionada, caso não exista nenhuma, não é necessário selecionar. Nos campos "*Mandatory*" é definido se a atividade selecionada é obrigatória "*Mandatory*" ou opcional "*Optional*" para esta arquitetura; no campo "*Concrete*" é definido se a atividade selecionada é concreta "*Concrete*" ou abstrata "*Abstract*" para a arquitetura em questão.

Pode-se observar na Figura 4.11, um exemplo de arquitetura definida com suas respectivas atividades.

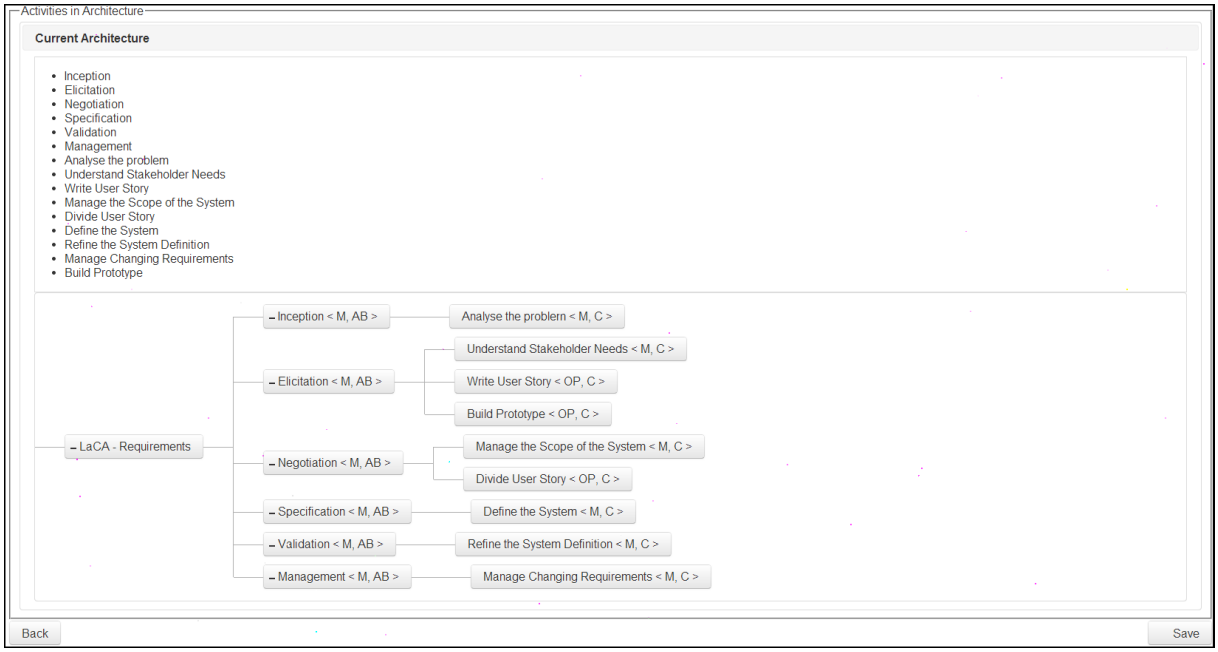

Figura 4.11 – MfTPt - *Activities in Architecture*.

Settings: descreve as funcionalidades relacionadas ao cadastramento dos usuários e suas autorizações.

4.2 Módulo para adaptação de processos de software baseado em Linhas de Processos de Software

A adaptação de processos de software utilizando as Linhas de Processos de Software é realizada em quatro passos (formulários), contendo o conjunto de etapas da sistemática para adaptação descrita. O módulo para adaptação de processos é organizado de acordo com o conjunto de etapas, que são: **i)** definição das características do projeto; **ii)** seleção dos requisitos de adaptação; **iii)** seleção da arquitetura de processos; **iv)** priorização das atividades; e **v)** criação da linha de processos de software adaptada.

Definição das características do projeto (***Definitions of project context***): a primeira etapa da sistemática para adaptação de processos é a definição das características do projeto, ou seja, o contexto situacional é definido/recuperado para o projeto. Pode-se visualizar na Figura 4.12, a contextualização para o projeto com seus respectivos valores.

Pode-se observar que existem diferentes projetos cadastrados, cada projeto possui a sua contextualização própria. Por exemplo, o projeto LaCA foi contextualizado utilizando fatores do *Octopus Model*, como pode ser visualizado. Este projeto foi contextualizado com caracte-

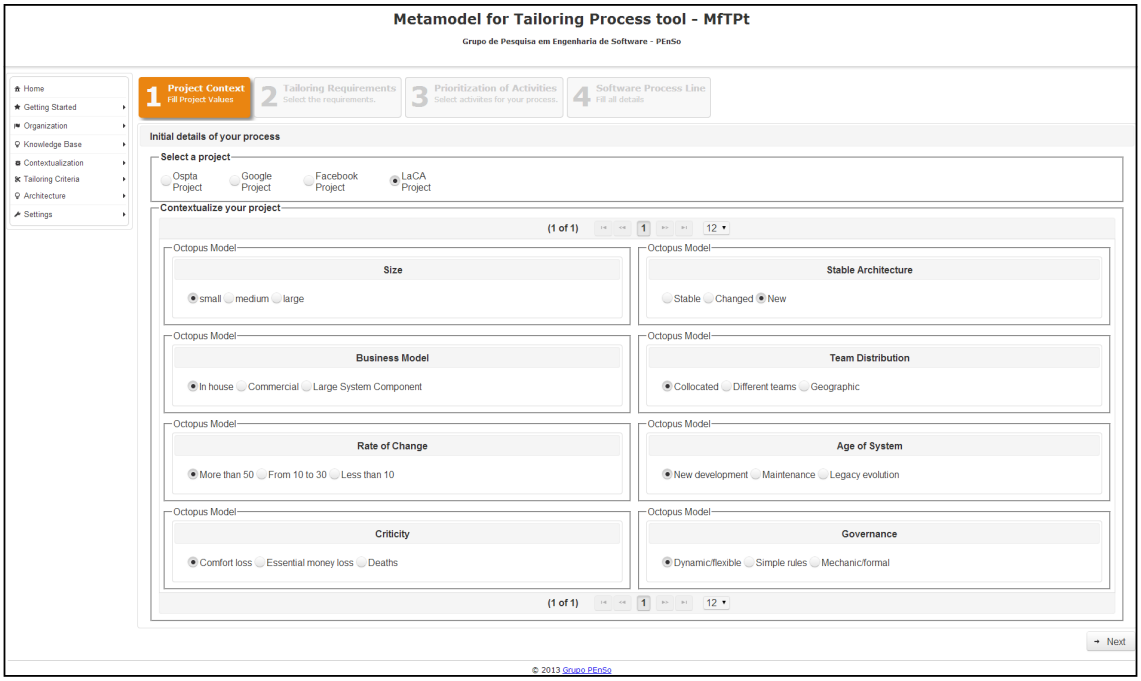

Figura 4.12 – Módulo SPrL - *Project Context: Fill Project Values*.

rísticas voltadas para metodologias ágeis.

Seleção dos requisitos de adaptação (***Selection of the tailoring requirements***): a segunda etapa é a seleção dos requisitos de adaptação que devem ser considerados no processo adaptado. No mesmo passo é realizada a terceira etapa, que é a **seleção da arquitetura de processos** (***Selection of the architecture of processes***). Essa arquitetura precisa ser previamente definida. Pode-se visualizar na Figura 4.13, a seleção da arquitetura para o processo e a seleção dos requisitos de adaptação.

Pode-se observar que para o projeto em questão, foi selecionada a arquitetura "LaCA – Requirements", e selecionado como critério de adaptação para prevenção de riscos em projetos, os riscos "*Misunderstanding the Requirements*" e "*Requirements Instability*". Para estes critérios de adaptação, existe na arquitetura as atividades "*Divide User Story*", "*Write User Story*" e "*Build Prototype*" que podem ser utilizadas para suprir estes critérios.

Priorização das atividades (***Prioritization of the activities***): a quarta etapa e terceiro passo é a recuperação e priorização das atividades da arquitetura utilizando o método AHP para compor a Linha de Processos de Software. Pode-se visualizar na Figura 4.14, as atividades que são opcionais para o processo e possuem critério de adaptação priorizadas.

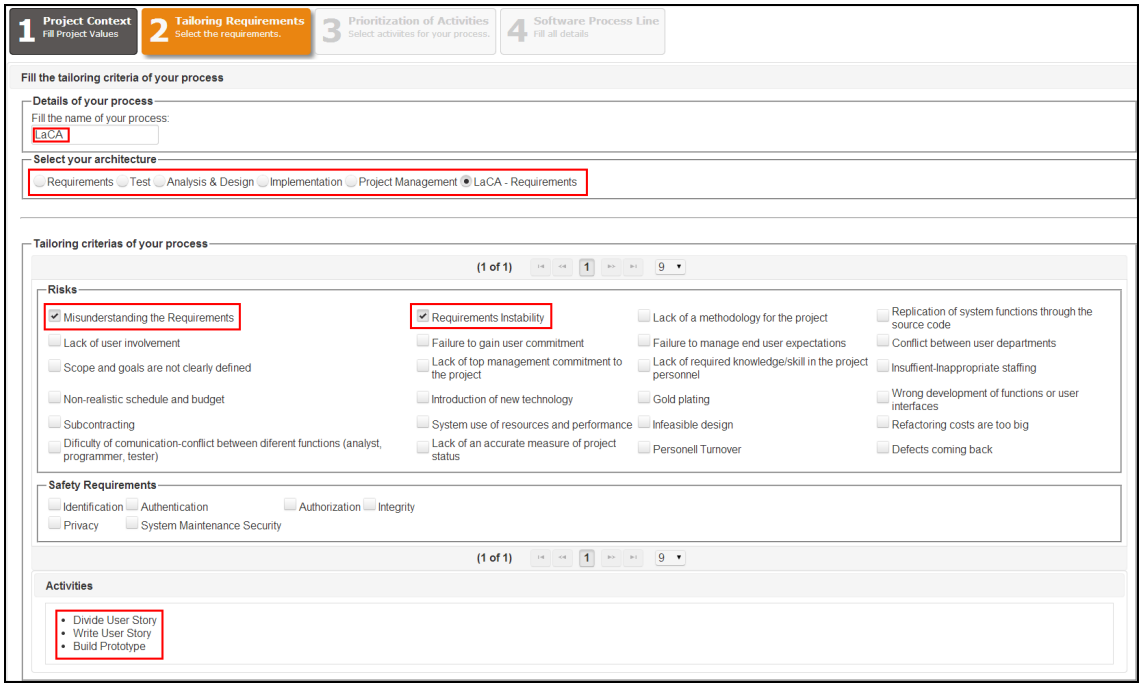

Figura 4.13 – Módulo SPrL – *Tailoring Requirements: Select the requirements.*

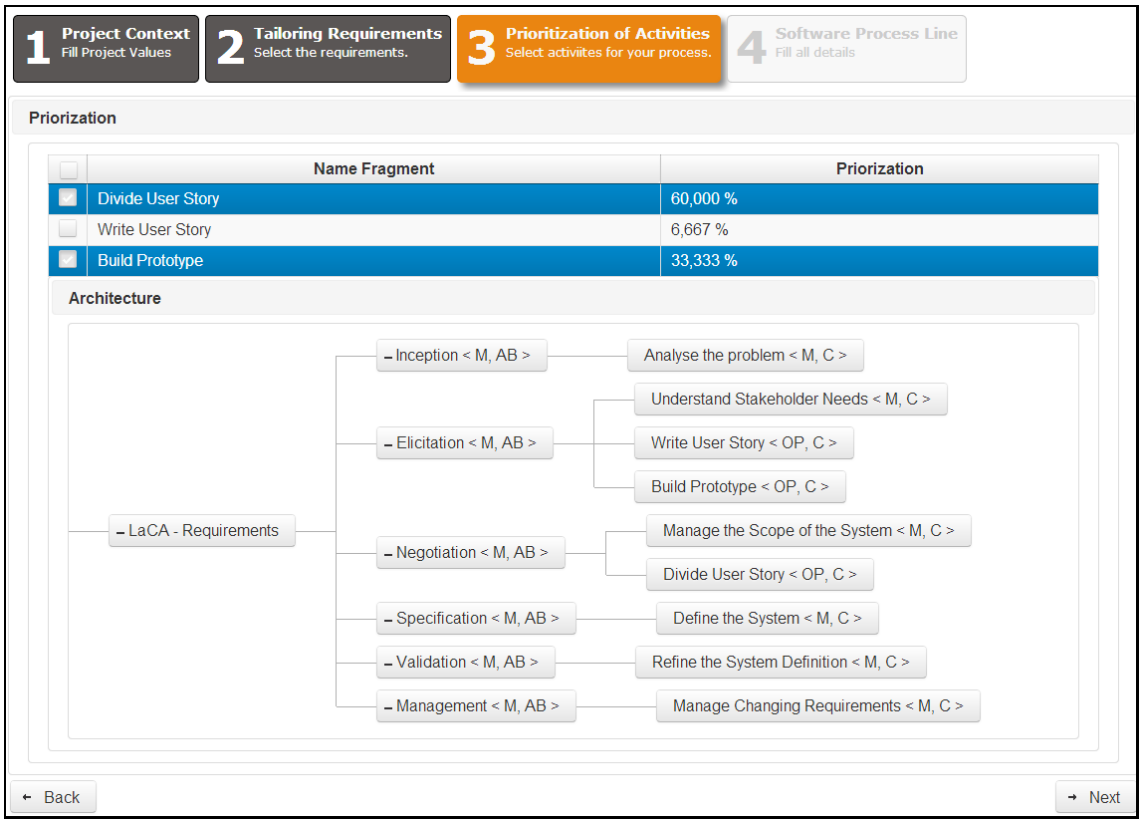

Figura 4.14 – Módulo SPrL – *Prioritization of Activities: Select activities for your process.*

Pode-se observar que foram priorizadas as atividades "*Divide User Story*", "*Write User Story*" e "*Build Prototype*", contendo respectivamente 60%; 6,66% e 33,33%, ou seja, o método AHP compara o contexto do projeto com o contexto das atividades e compara as atividades duas a duas para obter os valores de priorização, indicando quanto por cento cada atividade possui em relação ao projeto. Para auxiliar o engenheiro de processo de software, foi mantida a arquitetura de processos selecionada, auxiliando na visualização dos elementos priorizados que serão selecionados para o processo.

Para o projeto em questão foram selecionadas as atividades "*Divide User Story*" e "*Build Prototype*", estas atividades irão compor a Linha de Processo de Software adaptada, juntamente com todos os componentes obrigatórios.

Criação da linha de processos de software adaptada (***Creation of the tailored software processes line***): o quarto passo que corresponde a quinta etapa da sistemática é a criação da Linha de Processo de Software, contendo todas as atividades recuperadas, priorizadas e selecionadas para o processo. Pode-se visualizar na Figura 4.15, a Linha de Processo de Software adaptada.

A Linha de Processo de Software é composta por todos os seus elementos obrigatórios e pelos elementos opcionais selecionados que possuem critério de adaptação. Dentre as atividades priorizadas para a Linha de Processo de Software em questão, estão as atividades "*Divide User Story*" e "*Build Prototype*", mantidas na linha de processo adaptada. As outras atividades opcionais não selecionadas para o processo são removidas da linha de processos final.

Pode-se observar na Figura 4.15, o fluxo da Linha de Processo de Software, iniciando no componente "*Inception*", passando pelos componentes "*Elicitation*", "*Negotiation*", "*Specification*", "*Validation*" e finalizando no componente "*Management*". Para cada componente existe um conjunto de atividades que fazem parte da sua Linha de Processos. Por exemplo, para o componente "*Elicitation*" tem-se as atividade "*Understand Stakeholders Needs*" e "*Build Prototype*".

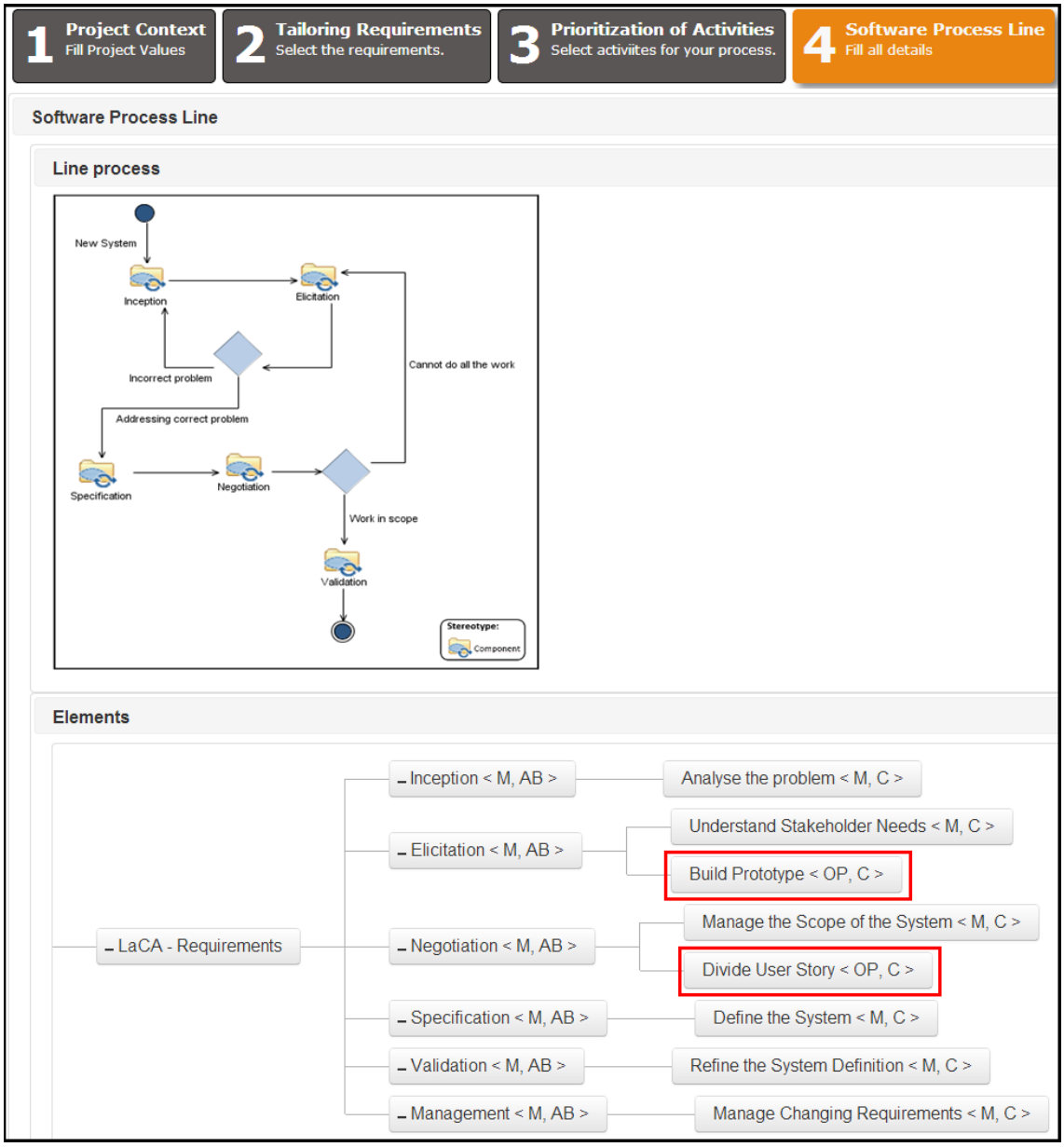

Figura 4.15 – Módulo SPrL – *Software Process Line: Fill all details*.

Neste capítulo foi apresentada a ferramenta de apoio MfTP. Nesta ferramenta, pode-se encontrar o módulo principal para cadastramento e definição dos principais elementos para adaptação de processos, e o módulo principal para adaptação de processos de software utilizando Linha de Processos de Software. O capítulo 5 abordará um estudo de caso, descrevendo dois projetos com contextos diferenciados, um voltado para as características planejadas e outro para características ágeis, descrevendo como são adaptados utilizando a sistemática para adaptação de processos de software utilizando Linha de Processos de Software.

5 ESTUDO DE CASO

Um estudo de caso foi realizado com o objetivo de validar a abordagem proposta e a ferramenta, exemplificando como processos de software para diferentes contextos podem ser gerados a partir de uma mesma arquitetura e linha de processo. Por razões de espaço o estudo de caso limita-se a discutir o processo gerado para a disciplina de requisitos.

Foi definida uma arquitetura para o desenvolvimento de novos sistemas, que pode ser visualizada na Figura 5.1. A arquitetura contém atividades contextualizadas para as metodologias ágeis e planejadas e atividades para prevenção de riscos: "*Implied Requirements*", "*Write User Story*", "*Build Prototype*", "*On Site Customer*", "*Planning Game*", "*Early and Regular Deliver RUP*", "*Early and Regular Deliver XP*", "*Divide User Story*", "*Scenarios Define Problem*", "*Simple Design*", "*Constant Refactoring*" e "*Software Configuration Management*".

5.1 Descrição do Projeto 1

O primeiro projeto foi desenvolvido para uma empresa chamada "*Your Destiny*". O processo da organização de software foi definido a partir do RUP. A organização precisa desenvolver software para a venda de passagens aéreas pela Internet. O sistema será desenvolvido por equipes que trabalham no mesmo lugar, mas não há a necessidade de regras formais de comunicação. Não haverá um envolvimento frequente e direto dos clientes durante o desenvolvimento. O software envolve transações financeiras e possível perda substancial de dinheiro. Regras formais para a gestão da equipe e do projeto são adotadas pela organização.

Para este projeto, a empresa quer adicionar requisitos para a prevenção de riscos no projeto. Os riscos a serem evitados são: falta de entendimento dos requisitos e requisitos instáveis.

A primeira atividade da sistemática de adaptação é a definição das características do projeto, ou seja, é definido o contexto situacional para o projeto. As informações de contexto do projeto são utilizadas para efetuar e comparar com o contexto das atividades armazenadas no repositório, com o objetivo de selecionar as atividades mais adequadas às características do projeto.

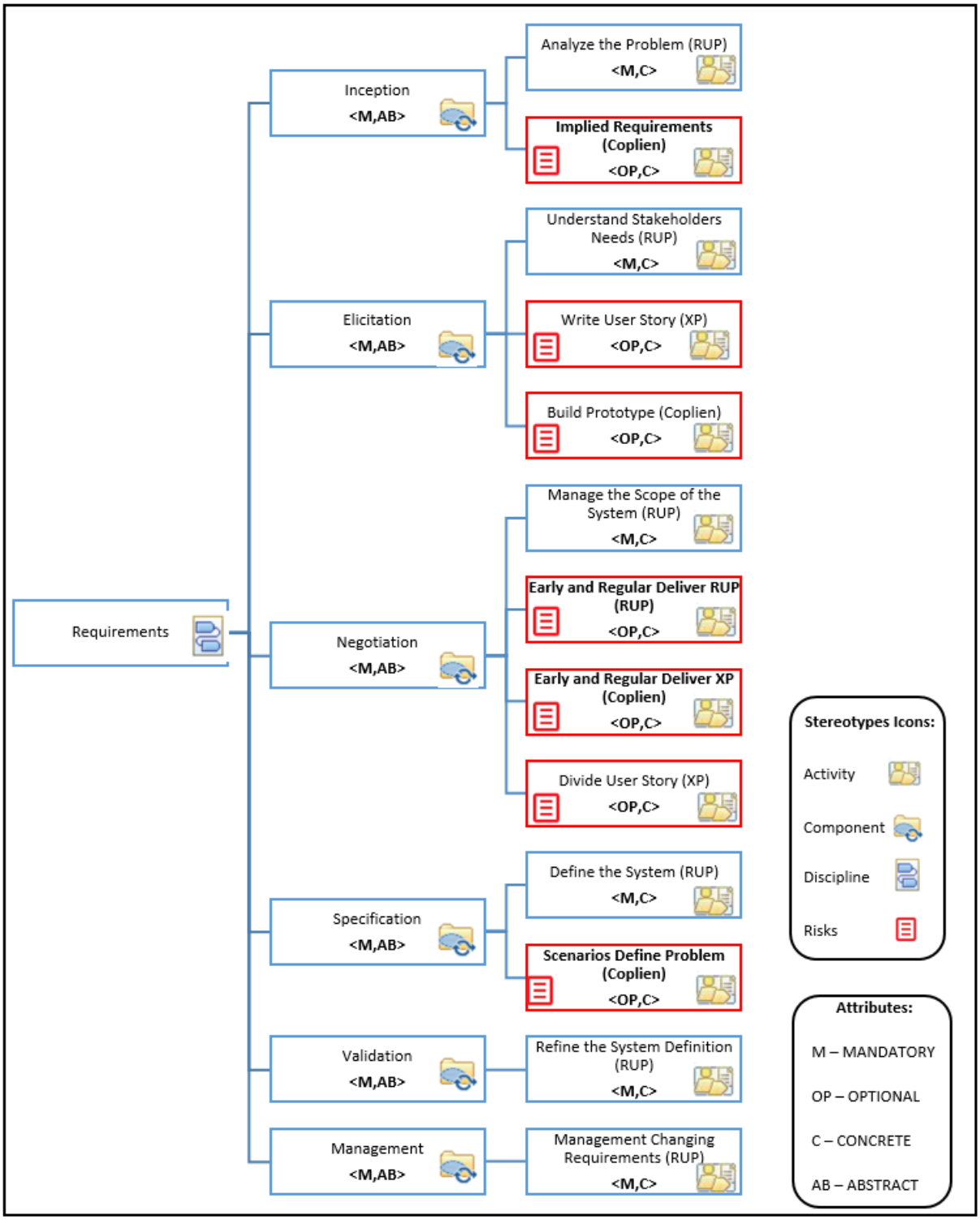

Figura 5.1 – Arquitetura definida para a disciplina de requisitos contendo atividades para prevenção de riscos em projetos.

Pode-se visualizar na Figura 5.2, os valores de contexto situacional definidos para o projeto a partir do *Octopus Model*.

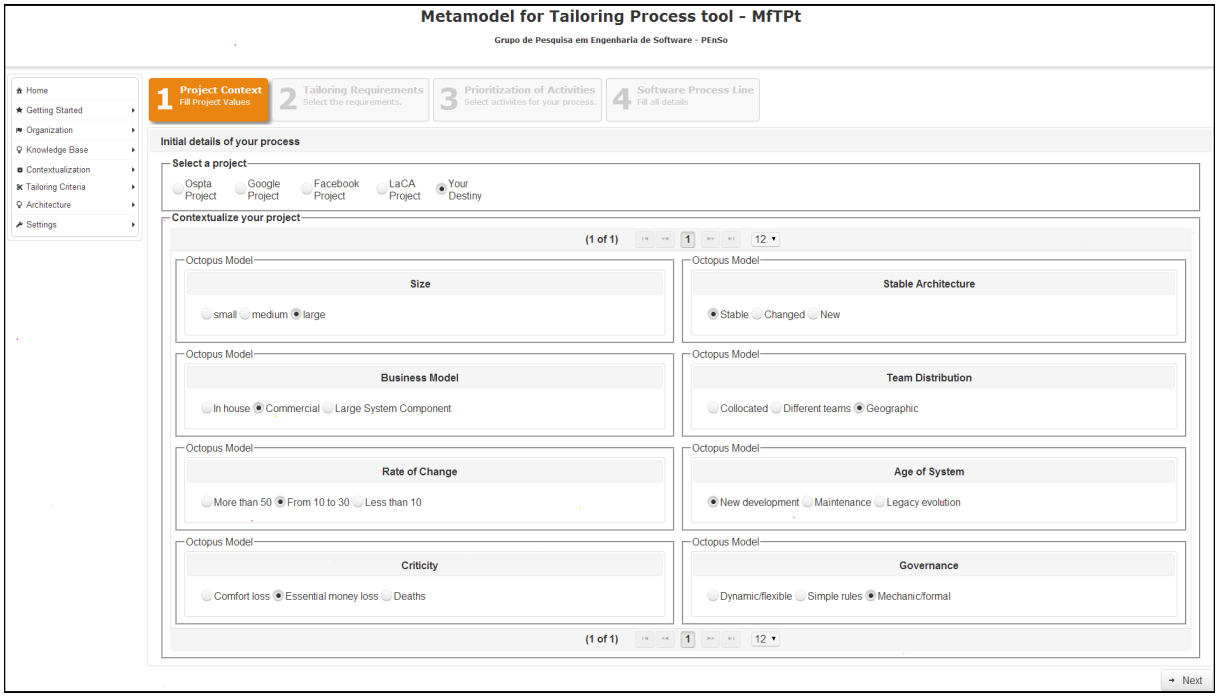

Figura 5.2 – MfPTt – Caracterização do Projeto "*Your Destiny*".

O projeto foi caracterizado utilizando os seguintes valores para os atributos do *Octopus Model*: **a)** *Size* (*Large*); **b)** *Team Distribution* (*Geographic*); **c)** *Criticality* (*Essential Money loss*); **d)** *Stable Architecture* (*Stable*); **e)** *Rate of Change* (*From* 10 to 30); **f)** *Governance* (*Mechanical/Formal*); **g)** *Business Model* (*Commercial*); e **h)** *Age of System* (*New development*).

A segunda etapa é a seleção dos requisitos de adaptação que devem ser considerados durante a adaptação. Para o projeto em questão, foram selecionados os riscos "*Misunderstanding the Requirements*" e "*Requirements Instability*". No mesmo passo é realizada a terceira etapa, que é a seleção da arquitetura de processos que melhor se adequa as características situacionais do projeto. Para o projeto em questão tem-se a seleção da arquitetura "*Requirements*". Pode-se visualizar na Figura 5.3, os requisitos de adaptação selecionados e a arquitetura para o processo, no caso foi selecionada a arquitetura "*Requirements*".

Após a seleção da arquitetura e dos requisitos de adaptação, a quarta etapa da sistemática e terceiro passo é a recuperação e priorização das atividades para a arquitetura. A recuperação e priorização das atividades são realizadas a partir do método AHP apresentado na Seção 3.5. De acordo com os riscos selecionados "*Misunderstanding the Requirements*" e "*Requirements Instability*", um conjunto de atividades é recuperado para evitar esses riscos. Pode-se visualizar na Figura 5.4, as atividades selecionadas com suas respectivas prioridades.

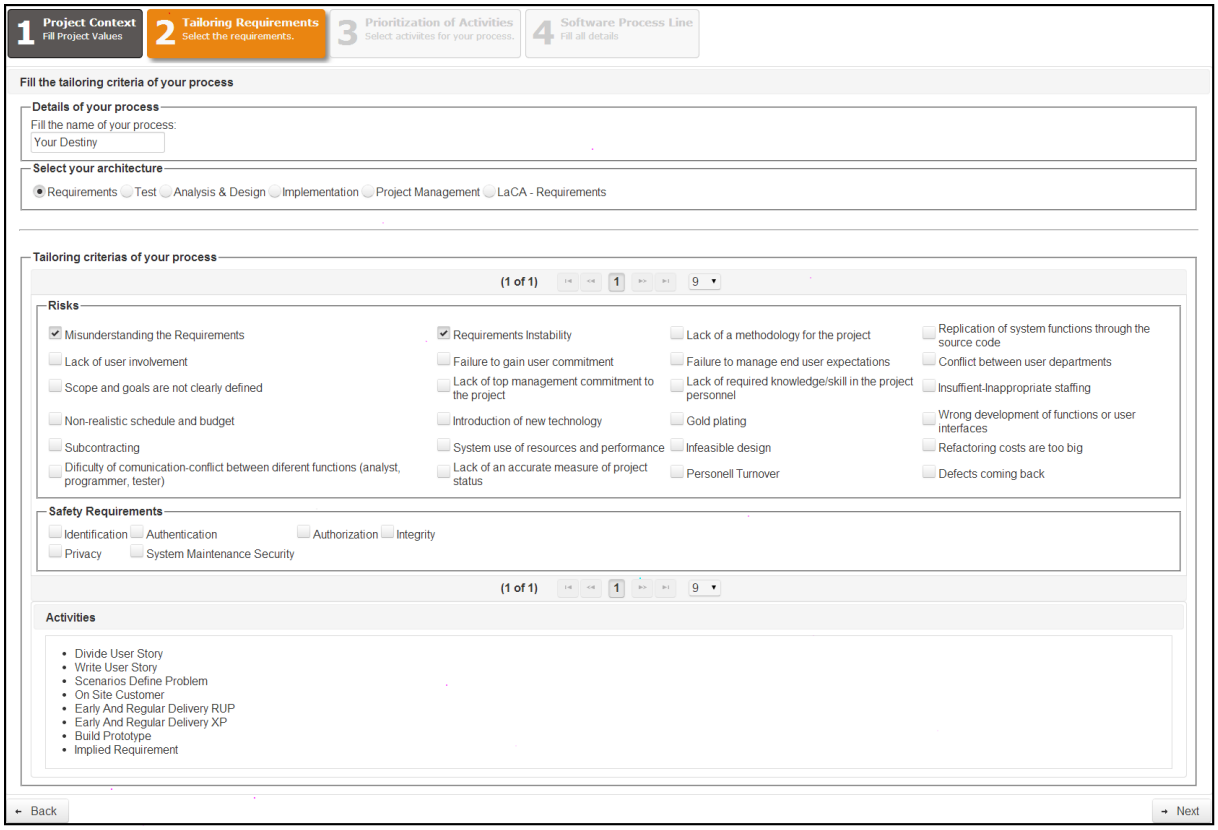

Figura 5.3 – MfTPt – Requisitos de adaptação e arquitetura.

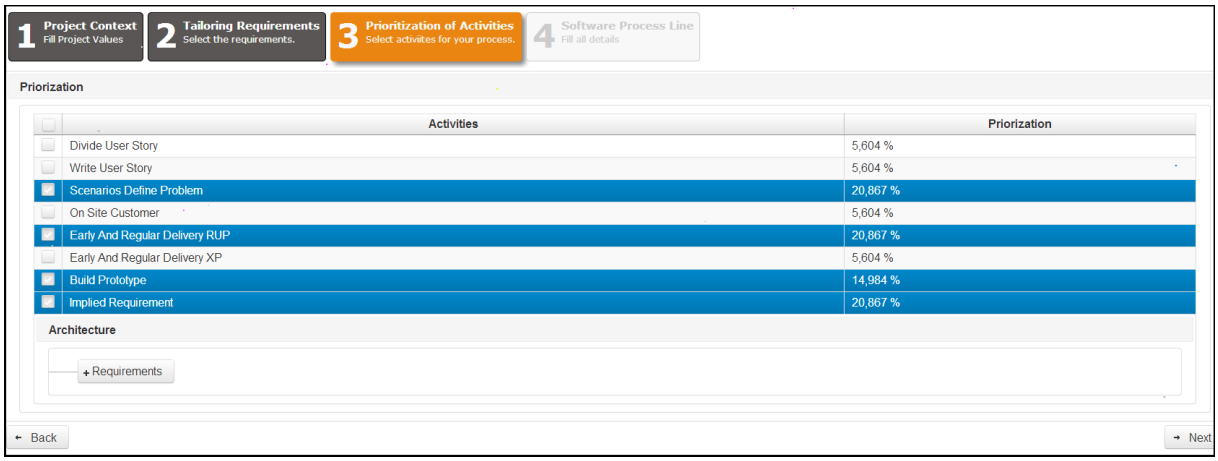

Figura 5.4 – MfTPt – Priorização das atividades.

O engenheiro de processos seleciona as atividades que serão utilizadas para criar a Linha de Processos de Software. Neste caso, "*Scenarios Define Problem*" probabilidade de 20,86%, "*Early and Regular Deliver RUP*", com probabilidade de 20,86%, "*Build Prototype*", com probabilidade de 14,98% e "*Implied Requirements*", com probabilidade de 20,86%. Salienta-se que essas atividades são priorizadas com as probabilidades mais altas segundo o método. Isto

ocorre porque o contexto dessas atividades é o mais similar ao contexto do projeto.

A quinta etapa da sistemática e quarto passo é a criação da Linha de Processo de Software adaptada, contendo as atividades recuperadas, priorizadas e selecionadas para o processo. Estas atividades formam a Linha de Processo de Software adaptada. Pode-se visualizar na Figura 5.5, a Linha de Processo de Software para a disciplina de Requisitos.

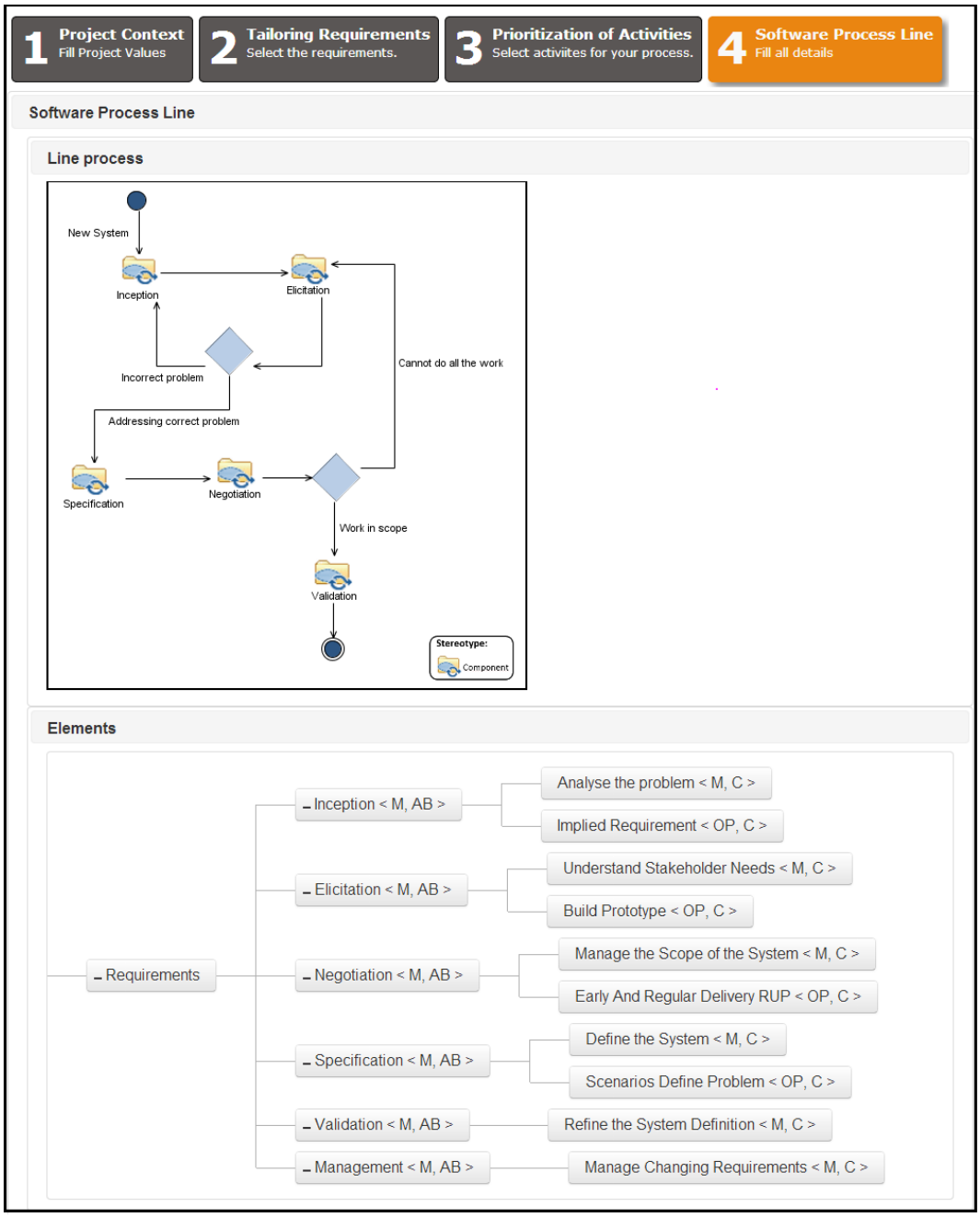

Figura 5.5 – MfTPt – Linha de Processo de Software adaptada para o Projeto "*Your Destiny*".

Pode-se notar que os elementos obrigatórios, os opcionais priorizados e selecionados compõem a Linha de Processos de Software. Os elementos não priorizados e não selecionados são removidos. Para cada disciplina do projeto é elaborada uma Linha de Processo de Software.

5.2 Descrição do Projeto 2

A organização precisa desenvolver um software para gestão de aluguel de filmes ("*Movie rentals*"). O sistema será desenvolvido por equipes que trabalham no mesmo lugar, mas não há a necessidade de regras formais de comunicação. Haverá um envolvimento frequente e direto com os clientes durante o desenvolvimento. Regras simples para o gerenciamento da equipe e do projeto são adotadas pela organização.

De acordo com o contexto do projeto, a empresa deseja adicionar requisitos de adaptação para a prevenção de riscos em seu projeto, com isso, tem-se a adição do risco "*Misunderstanding the Requirements*".

O projeto foi caracterizado por meio dos seguintes atributos: **a)** *Size (Small)*; **b)** *Team Distribution (Collocated)*; **c)** *Criticality (Comfort Loss)*; **d)** *Stable Architecture (Stable)*; **e)** *Rate of Change (More than 50)*; **f)** *Governance (Simple rules)*; **g)** *Business Model (In house)* e **h)** *Age of System (New development)*.

A arquitetura utilizada para esse projeto foi a mesma definida para o Projeto 1. Após a seleção da arquitetura para disciplina de requisitos e do critério de adaptação, foram recuperadas e priorizadas as atividades da arquitetura. Para o risco selecionado "*Misunderstanding the Requirements*", um conjunto de atividades é recuperado para a prevenção dos riscos. As atividades selecionadas com as probabilidades para a disciplina de Requisitos podem ser visualizadas na Tabela 5.1. O engenheiro de processo seleciona as atividades que serão utilizadas para compor o processo de software. O engenheiro de processo seleciona as atividades que serão utilizadas para compor o processo de software. Neste caso, foram selecionadas as atividades "*Divide User Story*", "*Write User Story*", "*On Site Customer*"e "*Early and Regular Delivery XP*". Essas atividades são propostas por metodologias ágeis.

Tabela 5.1 – Conjunto de atividades recuperadas e priorizadas utilizando o método AHP.

Requisitos de Adaptação	Atividades para prevenir riscos	Probabilidade
Misunderstanding the Requirements	Divide User Story	16,450 %
	Write User Story	16,450 %
	On Site Customer	16,450 %
	Early And Regular Delivery XP	16,450 %
	Build Prototype	12,453 %
	Scenarios Define Problem	7,249 %
	Implied Requirement	7,249 %
	Early And Regular Delivery RUP	7,249 %

Pode-se visualizar na Figura 5.6, a Linha de Processo de Software para a disciplina de requisitos elaborada para o Projeto 2, contendo os elementos selecionados para compor o processo.

5.3 Análise entre os estudos de caso

Os estudos de caso foram realizados com diferentes características/contextos, gerando diferentes processos adaptados. Ambos os casos incluem atividades para prevenção de riscos de projeto.

O Projeto 1, apresenta um contexto situacional planejado, contextualizado utilizando o *Octopus Model*. De acordo com o contexto do projeto e arquitetura utilizada, foram recuperadas atividades contendo características provenientes de processos planejados, desta forma, proporcionando maior planejamento e documentação do processos de software. A partir da arquitetura e da recuperação das atividades foi elaborada a Linha de Processo de Software para o projeto.

O Projeto 2, apresenta um contexto situacional ágil de acordo com fatores estipulados pelo *Octopus Model*. De acordo com o contexto do projeto, as atividades recuperadas visam proporcionar maior agilidade ao processo gerado. A partir da arquitetura e das atividades recuperadas foi elaborada a Linha de Processos de Software para um projeto envolvendo contexto ágil.

Finalmente, foram recuperadas diferentes atividades pela ferramenta de apoio para cada estudo de caso, devido a contextualização do projeto, as atividades são recuperadas da base de conhecimento. O mecanismo de priorização e seleção das atividades não elimina o papel do engenheiro de processo, mas facilita a identificação das atividades mais relevantes de acordo com os requisitos de adaptação e contexto situacional do projeto.

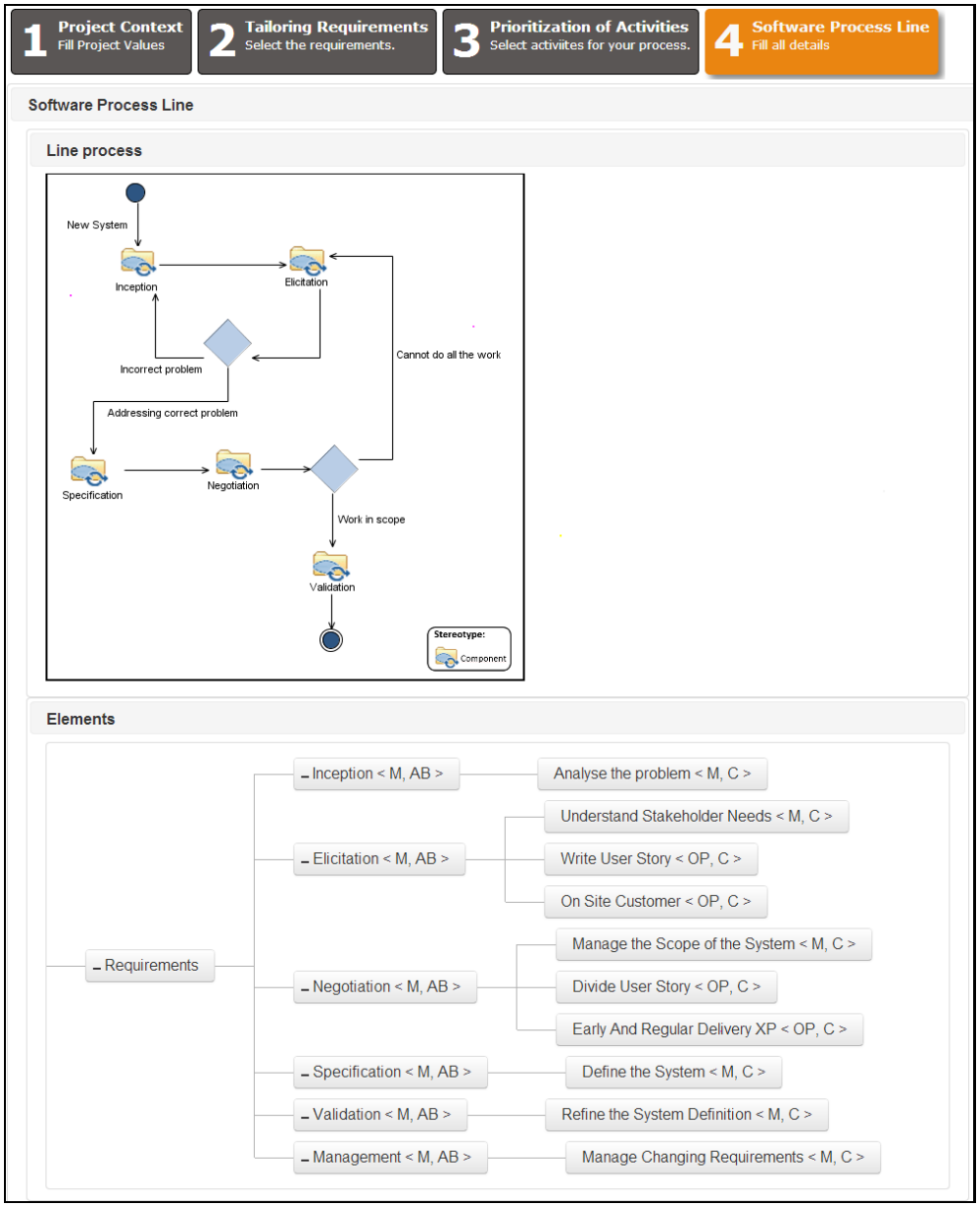

Figura 5.6 – MfTPt – Linha de Processo de Software adaptada para o Projeto "*Movie rental*".

A abordagem proposta garante a consistência do processo em relação ao sequenciamento de componentes (atividades), por meio da arquitetura e da Linha de Processo de Software. Além disso reduz a probabilidade de uma adaptação de processo de software inadequada e facilita o reuso, por meio da recuperação das atividades de um repositório.

Neste capítulo foi apresentado um estudo de caso fictício, com dois projetos diferentemente contextualizados e adaptados utilizando a sistemática para adaptação de processos de software utilizando Linha de Processos de Software. O capítulo 6 abordará os trabalhos relacionados com suas características e comparações com esta dissertação.

6 TRABALHOS RELACIONADOS

A abordagem de Münch (2004) apresenta uma técnica baseada em uma ferramenta para personalização baseada na transformação de modelos de processo formais para restrições de projetos. Tem-se a criação de um modelo de processo sob medida descrevendo os processos apropriados para um contexto de projeto específico, entradas para a atividade de customização são alternativas de processos, regras de personalização, e uma caracterização do projeto. Obtém-se como saída um modelo de processo adaptado com o conteúdo apropriado. O autor refere-se que o *feedback* a partir da aplicação de modelos de processo deve ser utilizado para melhorar os modelos, os critérios de personalização e o processo de personalização.

A abordagem de Münch (2005) centra-se na composição de padrões de processos de uma forma orientada para o objetivo. Esboça um método para compor os padrões de processos de uma forma orientada para os objetivos, mostrando os resultados da avaliação inicial. De acordo com o autor, a aplicação do método mostra que é possível fornecer mecanismos para a aplicação sistemática e a utilização de padrões de processos durante o planejamento do projeto. Além disso, o método fornece mecanismos para verificar se uma determinada combinação de padrões de processo cumpre uma meta do projeto em um contexto específico. O autor ainda menciona que, com todos os modelos, a qualidade dos resultados depende fortemente da validação (ou seja, principalmente a base empírica) dos modelos.

A abordagem de Jaufman e Münch (2005) é baseada em conceitos de adaptação do processo padrão para projetos específicos baseando-se em linhas de processos. Os autores utilizam conceitos de caracterização de processos apenas para um domínio específico e não fazem uso de um método para a recuperação e priorização. O método consiste em duas etapas principais: i) gerar uma linha de processo específico de um domínio de uma forma *top-down* e ii) revisar o processo gerado com base nos dados coletados durante sua execução.

A abordagem de Magdaleno (2010a) propõe como adequar os processos de desenvolvimento de acordo com as necessidades dos projetos e organizações (contexto), concentrando-se, principalmente, na questão da cooperação e disciplina como caminhos-chave.

Barreto *et al.* (2011) apresentam uma definição de processo com base na reutilização de componentes de processos legados, definidas e executadas nas organizações, para criar as linhas de processo. Esta abordagem propõe o uso de ciclo de vida, arquiteturas e modelos de processos, e faz uso dos conceitos "concretos"e "abstratos"para representar a variabilidade.

Além disso, utiliza modelos de domínio para a caracterização do processo e usa uma base de conhecimento, mas apenas recuperar os elementos, não se preocupando com a priorização.

Ruiz e Hurtado (2012) apresentam uma abordagem de Linha de Processos baseada no Processo Unificado (JACOBSON; BOOCH; RUMBAUGH, 1999). A abordagem utiliza algumas regras de contextualização para o processo, compondo assim as linhas de processo. Esta abordagem utiliza elementos definidos como obrigatórios, opcionais e alternativos para implementação da variabilidade. A abordagem proposta recupera um conjunto de componentes sem priorizá-los.

Este trabalho propõe uma abordagem para o desenvolvimento de processos adaptados utilizando conceitos de Linhas de Processo e contextualização de projetos. Sua diferença está na elaboração de uma Linha de Processo de Software adaptado a partir da reutilização de atividades devidamente caracterizadas. A Linha de Processos de Software adaptada utiliza uma arquitetura de processos para os quais são recuperadas e priorizadas atividades para compor a Linha de Processos de Software. A variabilidade em uma Linha de Processos de Software é implementada por componentes obrigatórios, opcionais, concretos e abstratos. Assim, é possível adaptar a Linha de Processo de Software para contextos ágeis, planejados e híbridos, de acordo com a caracterização do projeto e de suas atividades.

Outra diferença é a possibilidade de inclusão de requisitos de adaptação na Linha de Processo de Software e o uso de um método multicritério para priorização de atividades que irão compor o processo adaptado.

Pode-se visualizar na Tabela 6.1, o comparativo entre a abordagem proposta neste trabalho e os trabalhos relacionados, de acordo com os critérios que segue: utiliza conceitos de adaptação, faz uso de arquitetura de processos, utiliza variabilidade, utiliza conceitos de caracterização de processos e utiliza método de recuperação e priorização.

A abordagem proposta garante a consistência do processo em relação ao sequenciamento de componentes (atividades), por meio da Arquitetura e da Linha de Processo de Software. Além disso garante a variabilidade do processo usando tipos de componentes (obrigatórios ou opcionais e concretos ou abstratos), evitando assim o retrabalho no processo e a possibilidade de uma adaptação inadequada, devido à utilização das Linha de Processo de Software. A abordagem oferece diferentes critérios para adaptação, ou seja, o processo pode ser adaptado para atender a diferentes objetivos como: riscos, qualidade, segurança. Também proporciona a utilização de múltiplos critérios para caracterização do contexto e das atividades utilizadas.

Tabela 6.1 – Comparação entre as abordagens.

Contribuição	**Abordagem de Münch** (MUNCH, 2004)	**Abordagem de** (MÜNCH, 2005)	**Abordagem de Jaufman and Münch** (JAUFMAN ; MUNCH, 2005)	**Abordagem de Magdaleno** (MAGDALENO, 2010a)	**Abordagem de Barreto *et al.*** (BARRETO; MURTA; ROCHA, 2011)	**Abordagem de Ruiz and Hurtado** (RUIZ, PABLO H AND HURTADO , 2012)	**Abordagem proposta**
Utiliza conceitos de adaptação	Sim, a partir de regras de personalização.	Sim, na composição de padrões de processos de uma forma orientada para o objetivo	Faz uso a partir do processo padrão	Sim, focadas no contexto.	Preocupa-se com a reutilização, focando na componentização do processo.	Sim, foca apenas no Processo Unificado.	Sim, processos Planejados e Ágeis.
Faz uso de arquiteturas de processo	Não	Não	Não	Não	Propõem que pode-se usar modelos de ciclo de vida, arquiteturas ou modelos de processos.	Utiliza modelo de características.	Sim
Utiliza variabilidade	Sim, atividade de customização são alternativas de processos.	Sim, para a composição de padrões de processo: combinação sequencial combinação em paralelo, combinação condicional e combinação iterativa.	Não	Não cita	Propõem o conceito de componentes Concretos e Abstratos.	Utiliza o conceito de Obrigatório, Opcional e Alternativo.	Propõem o uso dos conceitos de componentes Obrigatórios, Opcionais, Concretos e Abstratos.
Utiliza conceitos de caracterização de processos	Sim	Sim	Apenas utiliza para um domínio específico	Sim, foca principalmente na questão da colaboração e disciplina como principais caminhos.	Sim, multiorganizacional (modelos de domínio).	Faz uso de algumas regras de contextualização para o processo utilizado.	Múltipla forma de caracterização dos componentes (Atividades).
Utiliza método de recuperação e priorização	Não	Não	Não, apenas sequencia manualmente	Propõem que será criado um módulo para recuperação.	Utiliza uma base de conhecimento, apenas recupera.	Apenas recupera os componentes definidos.	Utiliza uma base de conhecimento, recupera e prioriza.

Neste capítulo foi apresentado os trabalhos relacionados e a comparação com esta dissertação. O capítulo 7 abordará um experimento realizado no Instituto Federal Farroupilha – Campus São Vicente do Sul.

7 EXPERIMENTOS

Os processos de software são, em grande parte, de base humana e, consequentemente, não-determinístico. Além disso, são fortemente dependentes do contexto, estudos de diferentes tipos são necessários para entender e determinar os efeitos dos processos e analisar os riscos ao mudar processos ou a introduzir novos (MÜNCH et al., 2012).

Em um experimento do tipo controlado, pretende-se controlar completamente o ambiente experimental. No campo da engenharia de software, inclui o número e a experiência da pessoa de teste, e as tarefas que lhes são atribuídas. As tarefas analisadas neste experimento variam de problemas reais retirados de projetos de software anteriores a situações completamente imaginárias. Problemas adaptados também podem ser utilizados para a comparação de vários novos métodos (MÜNCH et al., 2012).

O experimento apresentado nesta dissertação foi baseado em um processo de projetos de software real. Para o projeto em questão tem-se a inserção de um cenário de riscos em projetos de software, ou seja, foram adicionados critérios de adaptação ao projeto utilizado pela instituição.

7.1 Experimento – IF Farroupilha - Campus São Vicente do Sul

O experimento foi aplicado no Setor de Tecnologia da Informação e Comunicação (STIC), do Instituto Federal Farroupilha – Campus São Vicente do Sul. A equipe de desenvolvimento de software que participou do experimento é composta por cinco pessoas, sendo um dos integrantes classificado como muito experiente (mais de cinco anos de experiência) e os demais como pouco experientes (menos de dois anos de experiência).

O processo é utilizado pela instituição há quatro anos e se baseia no RUP. O RUP foi utilizado por causa da familiaridade dos integrantes da equipe e por suas características com foco em documentação, visto que, a equipe mudava muito. O processo utilizado pelo STIC é dividido em cinco disciplinas, sendo elas: Abertura do Projeto, Requisitos, Análise e Projeto, Implementação, Teste e Implantação e Manutenção.

O experimento foi conduzido em duas etapas: na primeira etapa, foi utilizado pelo setor de desenvolvimento a ferramenta MfTP, cadastrando e definindo todos os elementos necessários para o seu processo como: atividades, tarefas, artefatos, papéis, arquitetura, linha de processo, critérios para contextualização do processo e requisitos de adaptação. Na segunda etapa foi

executada a sistemática proposta nesta dissertação para adaptação de processos.

Pode-se observar na Tabela 7.1, os valores estabelecidos pela equipe do STIC para o contexto do projeto usando o *Octopus Model*.

Tabela 7.1 – Caracterização do projeto do STIC.

Fatores	Atributos
Tamanho	Pequeno
Criticidade	Perda de conforto
Arquitetura estável	Estável
Modelo de Negócios	Sob demanda pra um cliente
Distribuição do Time	Local
Taxa de mudanças (% no mês)	Entre 10 e 30
Idade do sistema	Novo desenvolvimento
Governança	Mecânica/Formal

Os requisitos de adaptação identificados pela equipe do STIC foram riscos a serem prevenidos, sendo eles: "*Misunderstanding the Requirements*", "*Requirements Instability*" e "*Lack of user involvement*".

A partir do processo utilizado pelo STIC foi definida uma arquitetura de processos contendo todas as atividades contidas no processo utilizado. A estrutura da arquitetura contém todas as atividades julgadas obrigatórias, opcionais, concretas e abstratas. Para o experimento em questão foi criado apenas uma arquitetura de processos, por ser um processo que contém poucas atividades em cada disciplina, com isso, justificando a criação de apenas uma arquitetura e não uma arquitetura para cada disciplina. Nesta arquitetura foram adicionadas atividades para prevenir riscos em projetos já citadas acima. Pode-se observar na Figura 7.1, a arquitetura definida para o processo utilizado no experimento.

Na segunda etapa do experimento, foi executado pelo STIC a sistemática para adaptação de processos de software utilizando Linha de Processos de Software.

Primeiramente foi selecionado o contexto para o projeto, como já descrito na Tabela 7.1. No segundo passo da sistemática, tem-se a seleção dos requisitos de adaptação e seleção da arquitetura. Como requisitos de adaptação foram definidos os riscos: "*Misunderstanding the Requirements*", "*Requirements Instability*" e "*Lack of user involvement*". Foi utilizada a arquitetura definida para o processo do STIC.

Após a seleção da arquitetura e dos requisitos de adaptação, o terceiro passo é a recuperação e priorização das atividades para a arquitetura. De acordo com os riscos selecionados "*Misunderstanding the Requirements*", "*Requirements Instability*" e "*Lack of user involvement*"um conjunto de atividades é recuperado visando prevenir a ocorrência desses riscos. As atividades:

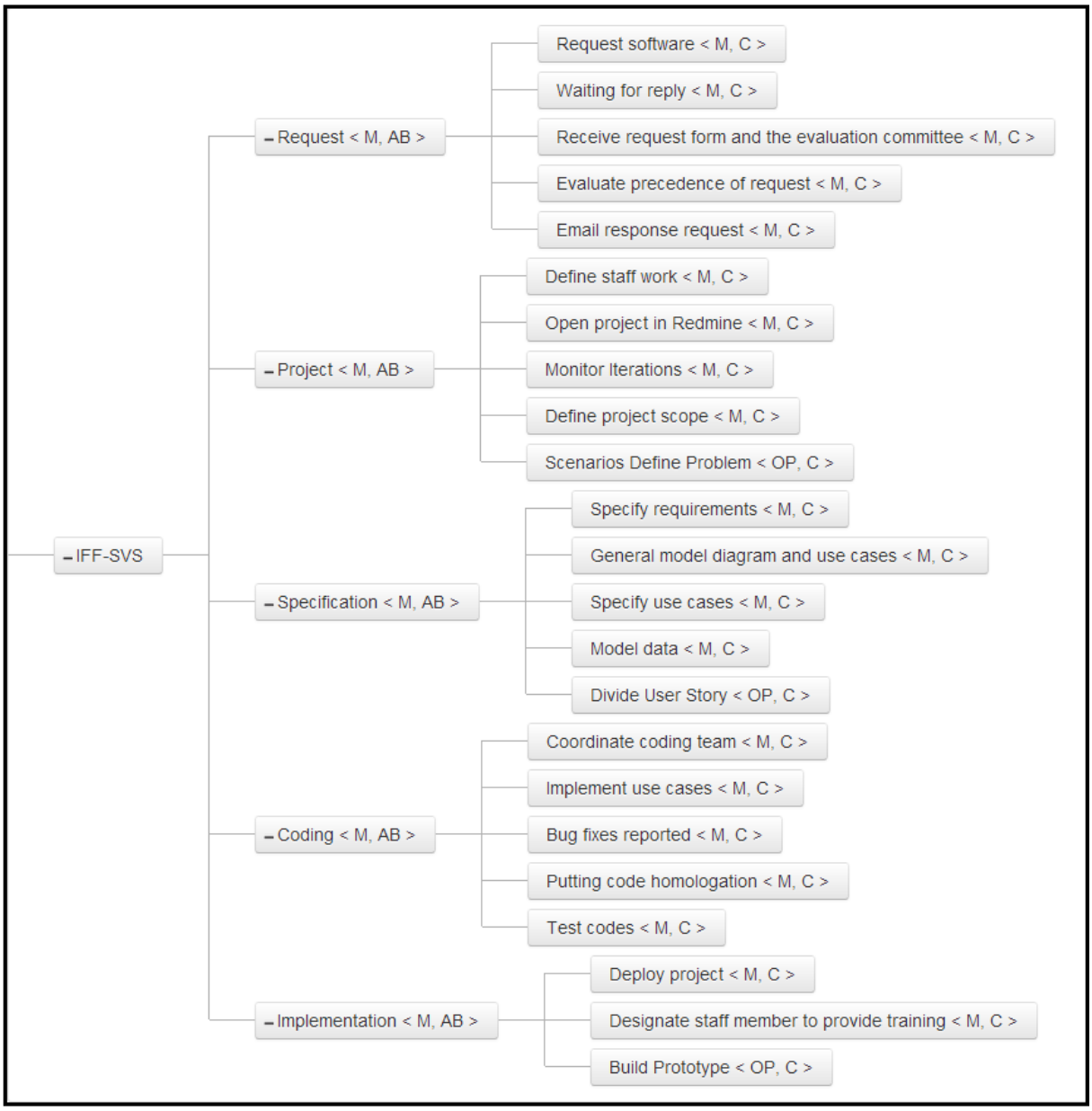

Figura 7.1 – Arquitetura de processos para o STIC.

"*Divide User Story*", "*Write User Story*", "*Scenarios Define Problem*", "*On Site Customer*", "*Early And Regular Delivery RUP*", "*Build Prototype*" e "*Implied Requirement*"" descrevem ações corretivas a esses riscos. De acordo com a contextualização do projeto, tem-se atividades com maior probabilidade, as quais foram selecionadas para compor a Linha de Processo de Software adaptada. Pode-se visualizar na Figura 7.2, as atividades selecionados com suas respectivas prioridades.

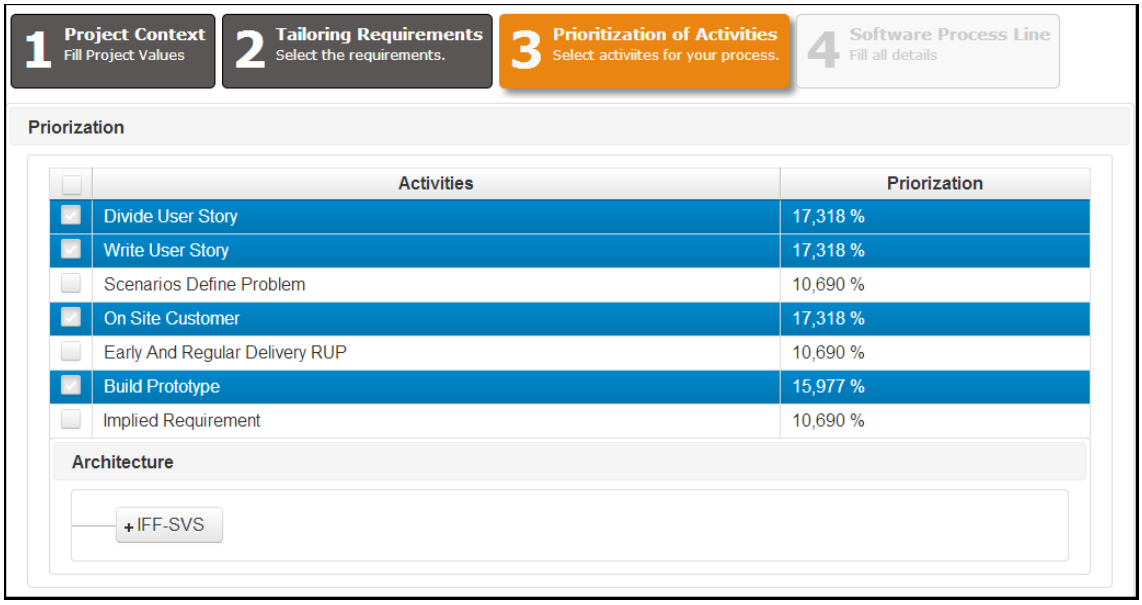

Figura 7.2 – Priorização e seleção das atividades para o processo do STIC.

O quarto passo é a criação da Linha de Processo de Software adaptada, contendo todas as atividades recuperadas, priorizadas e selecionadas para o processo. Pode-se visualizar na Figura 7.3, a Linha de Processo de Software adaptada para o processo do STIC, contendo os elementos obrigatórios e selecionados para compor o processo.

Pode-se notar que os elementos obrigatórios e os opcionais priorizados e selecionados compõe a Linha de Processos de Software. Os elementos não priorizados e não selecionados são removidos. Apesar de o processo de software do STIC ser baseado no RUP, a contextualização do projeto utilizado é voltado para características ágeis, desta forma, as atividades com características ágeis possuem uma maior priorização.

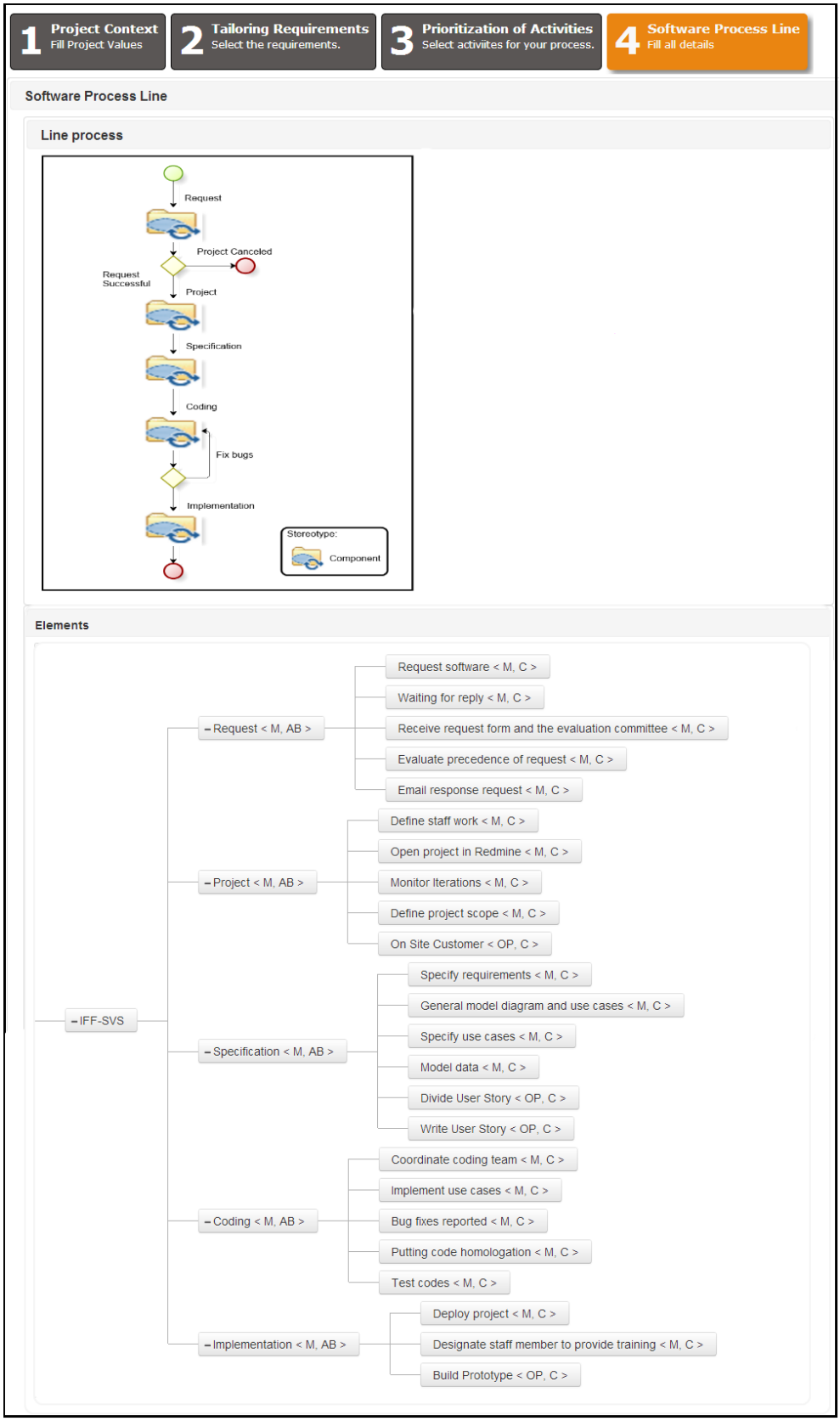

Figura 7.3 – Linha de Processos de Software adaptada para o STIC.

7.2 Avaliação do experimento

Para o experimento foi aplicado um questionário com o intuito de conhecer mais sobre o processo de software utilizado pela instituição, avaliar se as Linha de Processos de Software auxiliam o processo de desenvolvimento de software e avaliar a usabilidade da ferramenta proposta e do módulo para adaptação de processos de software utilizando Linha de Processos de Software. O questionário encontra-se no Anexo B.

Para o questionário aplicado após o experimento, pode-se destacar alguns pontos importantes:

- Foi julgado importante ter um conjunto de elementos de processo para que possam ser utilizados e reutilizados para definir novos processos ou adaptar processos existentes. Manter um repositório contendo elementos para serem utilizados e/ou adaptados em processos novos, evita o retrabalho.

- A utilização de Linhas de Processos de Software auxilia no sequenciamento do processo, no gerenciamento do processo e otimiza a alocação dos recursos, por meio da definição do que é realmente obrigatório e opcional para o processo. Uma das maiores vantagens é a criação da abstração proporcionada por elementos abstratos, porque com a utilização da abstração pode-se definir agrupamentos de atividades e definir quais serão executados, facilitando a organização e o entendimento.

- A visualização em árvore foi suma importância para identificar o sequenciamento do processo e onde cada elementos do processo deve ser instanciado.

- A utilização de elementos obrigatórios, opcionais, concretos e abstratos permite não sobrecarregar a comparação entre as atividades e priorização dos elementos que devem ou não serem executados.

- A caracterização das atividades e da equipe foi julgada muito importante, principalmente no aspecto de recuperação de atividades que melhor se adequam ao contexto do projeto.

- O processo gerado utilizando a Linha de Processos de Software foi julgado totalmente satisfatório em relação ao processo utilizado pelo STIC. Primeiramente, pelo auxílio na definição de algumas lacunas, ou seja, elementos que não estavam claros no processo de

software utilizado. Em segundo, auxiliando no amadurecimento do processo, mais especificamente na utilização de atividades que melhor se adequam ao contexto do projeto, principalmente no sequenciamento de algumas atividades que deveriam estar presentes no processo.

- O módulo para adaptação de processos utilizando Linha de Processos de Software foi de fácil utilização, principalmente pela orientação proporcionada pela sistemática para adaptação, facilitando definir o contexto do projeto, os seus requisitos de adaptação, seleção da arquitetura desejada para o processo, seleção das atividades melhor priorizadas e criação da Linha de Processos de Software contendo todos os elementos julgados importantes para o processo.

- Dentre as dificuldades encontra-se o cadastramento e definição dos elementos em primeiro momento, até se obter um repositório satisfatório para a reutilização dos elementos, auxiliando na definição de novas atividades e novos processos.

- Para a ferramenta tem-se como sugestão o acréscimo de documentação para cada processo e a criação de um módulo para gestão de processos e acompanhamento.

Neste capítulo foi apresentado um experimento realizado no Instituto Federal Farroupilha – Campus São Vicente do Sul, no qual foi aplicado a sistemática proposta nesta dissertação e um questionário para validação da aplicação do experimento. O capítulo 8 abordará as considerações finais e os trabalhos futuros.

8 CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

Este trabalho apresenta uma Linha de Processos de Software modelada de acordo com o metamodelo MfPT (*Metamodel for Process Tailoring*), propõe uma abordagem para o desenvolvimento de processos adaptados utilizando conceitos de Linha de Processos de Software e contextualização de projetos. A Linha de Processos de Software foi criada a partir de um conjunto de atividades definidas pelo metamodelo, o qual é baseado no RUP e SPEM, tornando-se possível caracterizar, usar e gerenciar as similaridades e diferenças existentes entre os processos. As atividades previamente caracterizadas são recuperadas a partir de uma base de métodos, de acordo com as características do projeto e priorizadas através do método *Analytic Hierarchy Process*.

O principal diferencial deste trabalho em relação aos citados está na elaboração de uma Linha de Processos de Software adaptada a partir da reutilização de atividades devidamente caracterizadas. A Linha de Processos de Software adaptada utiliza uma arquitetura de processos para os quais são recuperadas e priorizadas atividades para compor a Linha de Processos de Software. A variabilidade em uma Linha de Processos de Software é implementada por componentes obrigatórios, opcionais, concretos e abstratos. Assim, é possível adaptar a Linha de Processo de Software para contextos ágeis e planejados, de acordo com a caracterização do projeto e de suas atividades. Outra diferença é a possibilidade de inclusão de requisitos de adaptação na Linha de Processo de Software e o uso de um método multicritério para priorização de atividades que irão compor o processo adaptado.

Visando exemplificar a sistemática para adaptação de processos proposta, foi realizado um estudo de caso. O estudo de caso descreve dois projetos com diferentes características, e com isso, são gerados diferentes processos adaptados. Em ambos os projetos foram inseridos requisitos de adaptação, no caso, riscos a serem prevenidos em projetos. Para cada projeto foi demonstrado as cinco etapas da sistemática para adaptação de processos de software utilizando Linha de Processos de Software, são elas: caracterização do projeto, seleção de requisitos de adaptação, seleção da arquitetura de processos, priorização e seleção das atividades de acordo com as características do projeto e Linha de Processos de Software adaptada.

Para os projetos em questão, foram recuperadas diferentes atividades pela ferramenta de apoio para cada estudo de caso, devido a contextualização do projeto, as atividades são recuperadas da base de conhecimento. O mecanismo de priorização e seleção de atividades

não elimina o papel do engenheiro de processo, mas facilita a identificação das atividades mais relevantes de acordo com os requisitos de adaptação e contexto situacional do projeto. Portanto, existe uma Linha de Processo de Software adaptada de acordo com o contexto situacional de cada projeto.

Um experimento foi aplicado no Setor de Tecnologia da Informação e Comunicação, do Instituto Federal Farroupilha – Campus São Vicente do Sul, com a equipe de desenvolvimento de software. O experimento foi conduzido em duas etapas: na primeira etapa, foi utilizado pelo setor de desenvolvimento a ferramenta MfTP, cadastrando e definindo todos os elementos necessários para o seu processo como: atividades, tarefas, artefatos, papéis, arquitetura, linha de processo, critérios para contextualização do processo e requisitos de adaptação. Na segunda etapa, o processo foi executado utilizando a sistemática proposta nesta dissertação para adaptação de processos.

Com o experimento pode-se notar que as Linha de Processos de Software auxiliaram principalmente no sequenciamento do processo, otimizando os recursos e melhorando o gerenciamento do processo. A utilização da arquitetura proporcionou a criação de elementos abstratos, para o quais são definidos conjuntos de atividades que podem ser recuperadas e priorizadas de acordo com sua caracterização, definições de obrigatoriedade ou opcionalidade e requisitos de adaptação. Com isso, facilitando a adequação do processo utilizado para diferentes contextos de projetos.

Utilizando atividade caracterizadas na Linha de Processo de Software, pode-se obter uma maior consistência do processo em relação ao sequenciamento de componentes (atividades) no processo de desenvolvimento de software. Essa consistência é por meio do sequenciamento e gerenciamento de uma arquitetura de processos, contendo componentes obrigatórios, opcionais, concretos e abstratos, representando a variabilidade na Linha de Processos de Software.

Como outra forma de validação, um artigo contendo a sistemática para adaptação de processos de software utilizando Linhas de Processos de Software baseada no reuso de atividades apresentada neste trabalho foi aceito em dois de Maio de 2014 no *International Journal of Software Engineering and Knowledge Engineering*.

Trabalhos futuros incluem a verificação da consistência dos processos formados, um módulo de avaliação de processos adaptados, criação dinâmica do fluxo da linha de processos de software de acordo com suas caraterísticas situacionais e critérios de adaptação.

9 REFERÊNCIAS

ALEGRÍA, J. A. H. et al. An MDE approach to software process tailoring. In: **Proceedings of the 2011 International Conference on Software and Systems Process**. New York, New York, USA: ACM Press, 2011. p. 43–52.

ARMBRUST, O. et al. Scoping software process lines. **Software Process: Improvement and Practice**, v. 14, n. 3, p. 181–197, maio 2009.

BALDASSARRE, M. T. et al. **Comparing ISO/IEC 12207 and CMMI-DEV: Towards a mapping of ISO/IEC 15504-7 2009.** ICSE Workshop on Software Quality. Anais. IEEE, maio 2009.

BARRETO, A.; MURTA, L.; ROCHA, A. R. Software Process Definition: a Reuse-based Approach. **Journal Of Universal Computer Science**, v. 17, n. 13, p. 1765–1799, 2011.

BECK, K. **Programação Extrema (XP) Explicada: Acolha as Mudanças**. Brasil: Bookman, 2004.

BENAVIDES, D. **On the automated analysis of software product lines using feature models: a framework for developing automated tool support.** [s.l.] Universidad de Sevilla, 2007.

BENAVIDES, D.; SEGURA, S.; RUIZ-CORTÉS, A. Automated analysis of feature models 20 years later: A literature review. **Information Systems**, v. 35, n. 6, p. 615–636, set. 2010.

BENCOMO, A. Extending the RUP, Part 1: process modeling. **IBM**, p. http://www–128.ibm.com/developerworks/rational/lib. 2005.

BEN-SHAUL, I.; KAISER, G. A paradigm for decentralized process modeling and its realization in the Oz environment. **Proceedings of the 16th international conference on Software engineering**, p. 179–188, 1994.

BOEHM, B. Get ready for agile methods, with care. **Computer**, v. 35, n. 1, p. 64–69, 2002.

BOEHM, B.; TURNER, R. Using Risk to Balance Agile and Plan-Driven Methods. **IEEE Computer Society**, n. June, p. 57–66, 2003.

BRASIL, M. A. B. Uma estratégia para validação da completude e consistência em processos de software. 2014.

CHRISSIS, M. B.; KONRAD, M.; SHRUM, S. **CMMI for Development: Guidelines for Process Integration and Product Improvement?; [CMMI-DEV, Version 1.3]**. [s.l.] Addison Wesley Professional, 2011. p. 892

CLEMENTS, P.; NORTHROP, L. Software product lines: practices and patterns. In: **SEI Series in Software Engineering**. Massachusetts, USA: Addison-Wesley Reading, 2001.

COCKBURN, A. **Crystal clear: a human-powered methodology for small teams**. [s.l.] Pearson Education, 2004. v. 48

COPLIEN, J. O.; ALEXANDER, A. W. O. **Software Patterns**. [s.l.] Citeseer, 1996.

DAVID, J.; SAATY, D. Use analytic hierarchy process for project selection. **ASQ Six Sigma Forum Magazine**, p. 22–29, 2007.

FIGUEIREDO, E. et al. Evolving software product lines with aspects. In: **Software Engineering, 2008. ICSE'08. ACM/IEEE 30th International Conference on**. [s.l.] IEEE, 2008. p. 261–270.

FONTOURA, L. **PRiMA: project risk management approach**. [s.l.] Universidade Federal do Rio Grande do Sul (UFRG), 2006.

FONTOURA, L. M.; PRICE, R. T. Systematic Approach to Risk Management in Software Projects through Process Tailoring. **International Conference on Software Engineering and Knowledge Engineering (SEKE)**, 2008.

FOWLER, M. The new methodology. **Wuhan University Journal of Natural Sciences**, v. 6, n. 1-2, p. 12–24, 2001.

FUGGETTA, A. Software process: a roadmap. **Proceedings of the Conference on the Future of Software Engineering**, p. 25–34, 2000

GINSBERG, M.; QUINN, L. Process Tailoring and the the Software Capability Maturity Model. 1995.

GRISS, M. L. Software reuse architecture, process, and organization for business success. In: **Computer Systems and Software Engineering, 1997., Proceedings of the Eighth Israeli Conference on**. [s.l: s.n.]. p. 86–89.

GUO, J.; WANG, Y. Towards consistent evolution of feature models. In: **Software Product Lines: Going Beyond**. Jeju Island, South Korea: Springer, 2010. v. LNCS 6287p. 451–455.

HARTMANN, J.; FONTOURA, L.; PRICE, R. Using Risk Analysis and Patterns to Tailor Software Processes. **Simpósio Brasileiro de Engenharia de Software - SBES**, v. 19,

2005.

HENDERSON-SELLERS, B. **Object-oriented methods and processes**. Proceedings International Conference on Software Methods and Tools. SMT 2000. Anais...IEEE Comput. Soc, 2000.

HENDERSON-SELLERS, B.; RALYTÉ, J. Situational Method Engineering: State-of-the-Art Review. **J. UCS**, v. 16, n. 3, p. 424–478, 2010.

HUMPHREY, W. S. **Managing the Software Process**. [s.l.] Addison-Wesley Professional, 1989. v. 16p. 494

IBM COPORATION. IBM Rational Unified Process. **Version**, v. 7, p. 2000, 2007.

ISO/IEC 15504. Information Technology – Software Process Assessment (parts 1–9). **International Organization for Standardization**, 2003.

JACOBSON, I.; BOOCH, G.; RUMBAUGH, J. **The unified software development process**. [s.l.] Addison-Wesley Reading, 1999.

JAUFMAN, O.; MUNCH, J. Acquisition of a project-specific process. In: BOMARIUS, F.; KOMI-SIRVIÖ, S. (Eds.). **Product Focused Software Process Improvement**. LNCS. [s.l.] Springer Berlin Heidelberg, 2005. v. 3547p. 328–342.

KELLNER, M. I. Connecting reusable software process elements and components. **Proceedings 10th International Software Process Workshop**, 1996

KRUCHTEN, P. Contextualizing agile software development. **Journal of Software: Evolution and Process**, v. 25, n. 4, p. 351–361, 2013.

MAB, VON DEREN, T.; LICHTER, H. Deficiencies in feature models. In: **workshop on software variability management for product derivation-towards tool support**. Boston, MA, USA: [s.n.].

MAGDALENO, A. Balancing collaboration and discipline in software development processes. In: **Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 2**. New York, New York, USA: ACM Press, 2010a. v. 2p. 331–332.

MAGDALENO, A. M. An Optimization-based Approach to Software Development Process Tailoring. **2nd International Symposium on Search Based Software Engineering**, p. 40–43, set. 2010b.

MONTERO, I.; PEÑA, J.; RUIZ-CORTÉS, A. Business Family Engineering: Does it make sense? **Business Family Engineering: Does it make sense**, 2007.

MPS, M. et al. **MPS . BR - Melhoria de Processo do Software Brasileiro Guia Geral**

Sumário. [s.l: s.n.].

MUNCH, J. Transformation-based creation of custom-tailored software process models. n. ProSim, p. 50–56, 2004.

MÜNCH, J. Goal-oriented composition of software process patterns. **International Workshop on Software Process Simulation and Modeling (ProSim 2005)**, n. ProSim, p. 164–168, 2005.

MÜNCH, J. et al. **Software Process Definition and Management**. [s.l.] Springer, 2012

NORTHROP, L. SEI's software product line tenets. **Software, IEEE**, n. August, 2002.

OMG. **Software & Systems Process Engineering Meta-Model Specification Verson 2.0**. [s.l.] Continuous Representation, Pittsburgh: Software Engineering Institute, Carnegie Mellon University, 2008. p. 236

PARNAS, D. L. On the design and development of program families. **Software Engineering, IEEE Transactions on**, v. 1, p. 1–9, 1976.

PEREIRA, E. B.; BASTOS, R. M.; OLIVEIRA, T. C. A systematic approach to process tailoring. In: **Systems Engineering and Modeling, 2007. ICSEM'07. International Conference on**. [s.l.] IEEE, 2007. p. 71–78.

PRESSMAN, R. S. **Software Engineering**. Seventh ed. [s.l.] McGraw-Hill, 2010. p. 930

ROMBACH, D. Integrated software process and product lines. In: **Unifying the Software Process Spectrum**. [s.l.] Springer, 2006. p. 83–90.

RUIZ, PABLO H AND HURTADO, J. A. A software process line based on the Unified Process. In: **Computing Congress (CCC), 2012 7th Colombian**. [s.l.] IEEE, 2012. p. 1–6.

SAATY, T. L. **Theory e Applications of the Analytic Network Process: Decision Making with Benefits, Opportunities, Costs, and Risks**. [s.l.] Pittsburgh: RWS Publications, 2005.

SAATY, T. L. Decision making with the analytic hierarchy process. **International journal of services sciences**, v. 1, n. 1, p. 83–98, 2008.

SCHWABER, K.; BEEDLE, M. **Agile Software Development with Scrum**. [s.l.] Prentice Hall, 2003.

SINGH, R. International Standard ISO/IEC 12207 software life cycle processes. **Software Process Improvement and Practice**, v. 2, n. 1, p. 35–50, 1996.

STAPLETON, J. Dsdm: Dynamic systems development method. In: **Technology of Object-Oriented Languages and Systems, 1999. Proceedings of**. [s.l.] IEEE, 1999. p. 406–406.

SUTTON, S.; OSTERWEIL, L. Product families and process families. **Software Process Workshop, 1996. Process Support of Software Product Lines., Proceedings of the 10th International**, p. 109–111, 1996.

VAN DER LINDEN, F. J.; SCHMID, K.; ROMMES, E. **Software product lines in action**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.

VARGAS, R. V. Utilizando a programação multicritério (Analytic Hierarchy Process-AHP) para selecionar e priorizar projetos na gestão de portfólio. In: **PMI Global Congress**. [s.l: s.n.]. p. 1–22.

WASHIZAKI, H. Building software process line architectures from bottom up. In: MÜNCH, J.; VIERIMAA, M. (Eds.). **Product-Focused Software Process Improvement**. Lecture Notes in Computer Science. [s.l.] Springer Berlin Heidelberg, 2006a. v. 4034p. 415–421.

WASHIZAKI, H. Deriving project-specific processes from process line architecture with commonality and variability. **Industrial Informatics, 2006 IEEE International Conference on**, p. 1301–1306, 2006b.

WEBER, K.; ARAÚJO, E.; MACHADO, C. Modelo de Referência e Método de Avaliação para Melhoria de Processo de Software–versão 1.0 (MR-MPS e MA-MPS). **IV Simpósio Brasileiro de Qualidade de Software**. Porto Alegre-RS: Anais do SBQS, v. 0, p. 14, 2005

XU, P. **Knowledge Support in Software Process Tailoring**. Proceedings of the 38th Annual Hawaii International Conference on System Sciences. Anais...IEEE, 2005.

XU, P.; RAMESH, B. Impact of knowledge support on the performance of software process tailoring. **Journal of Management Information Systems**, v. 25, n. 3, p. 277–314, 2008a.

XU, P.; RAMESH, B. Using process tailoring to manage software development challenges. **IT Professional**, v. 10, n. 4, p. 39–45, jul. 2008b.

ANEXOS

ANEXO A – Arquiteturas e Linha de Processos de Software

Todas as arquiteturas definidas são sugestões deste trabalho e da literatura. Elas contêm componentes contendo sua finalidade e características, bem como os seus relacionamentos e precedências. Para cada uma destas arquiteturas, tem-se a recuperação do repositório, atividades que possuem caracterizações semelhantes.

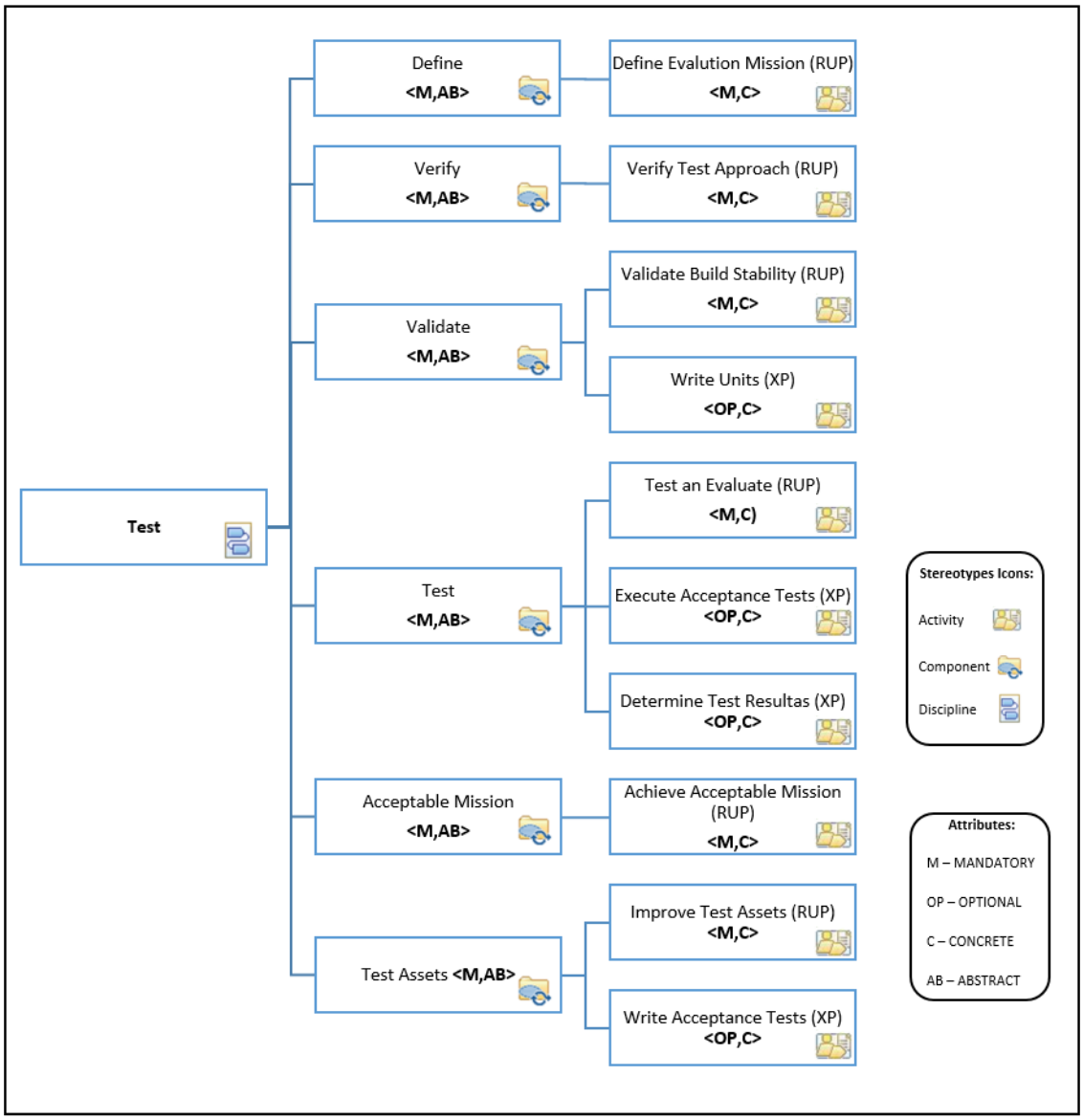

Figura A.1 – Exemplo de arquitetura definida para a disciplina de Teste para a LPrS.

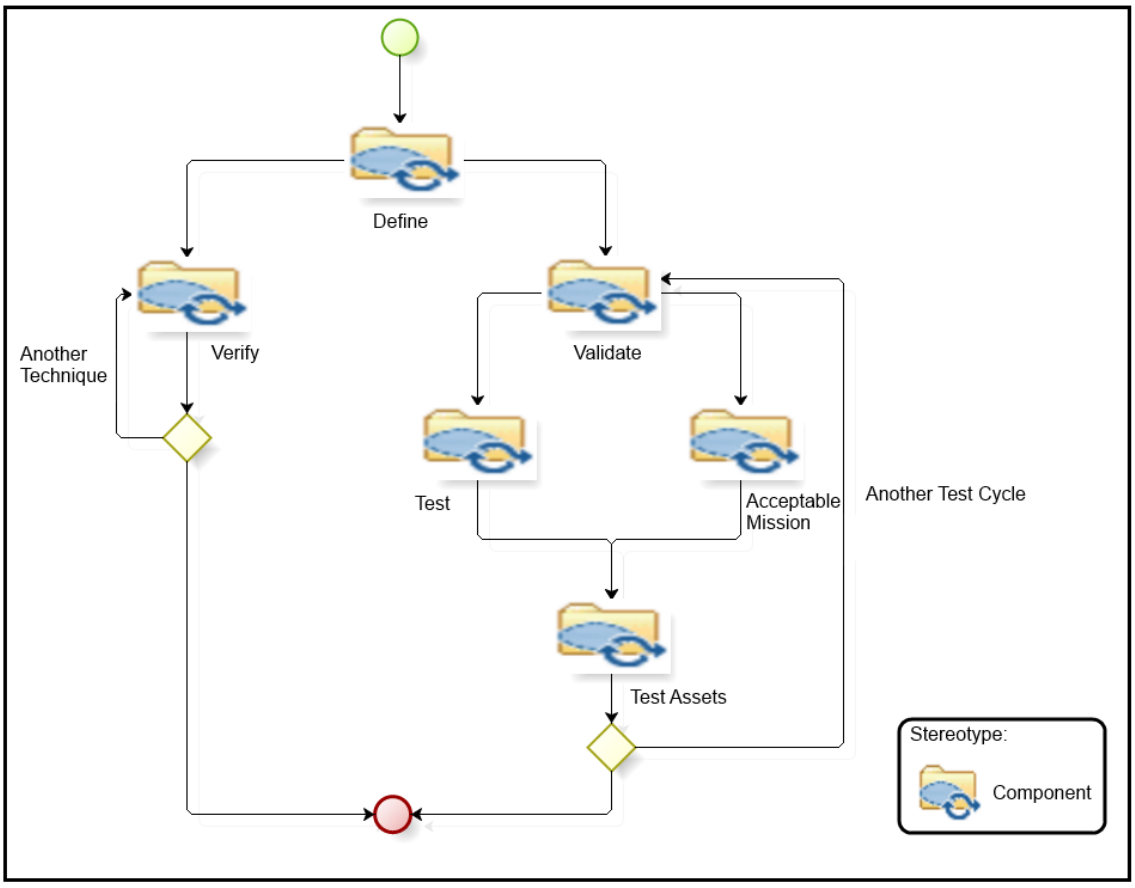

Figura A.2 – Linha de Processos de Software para a disciplina de Testes.

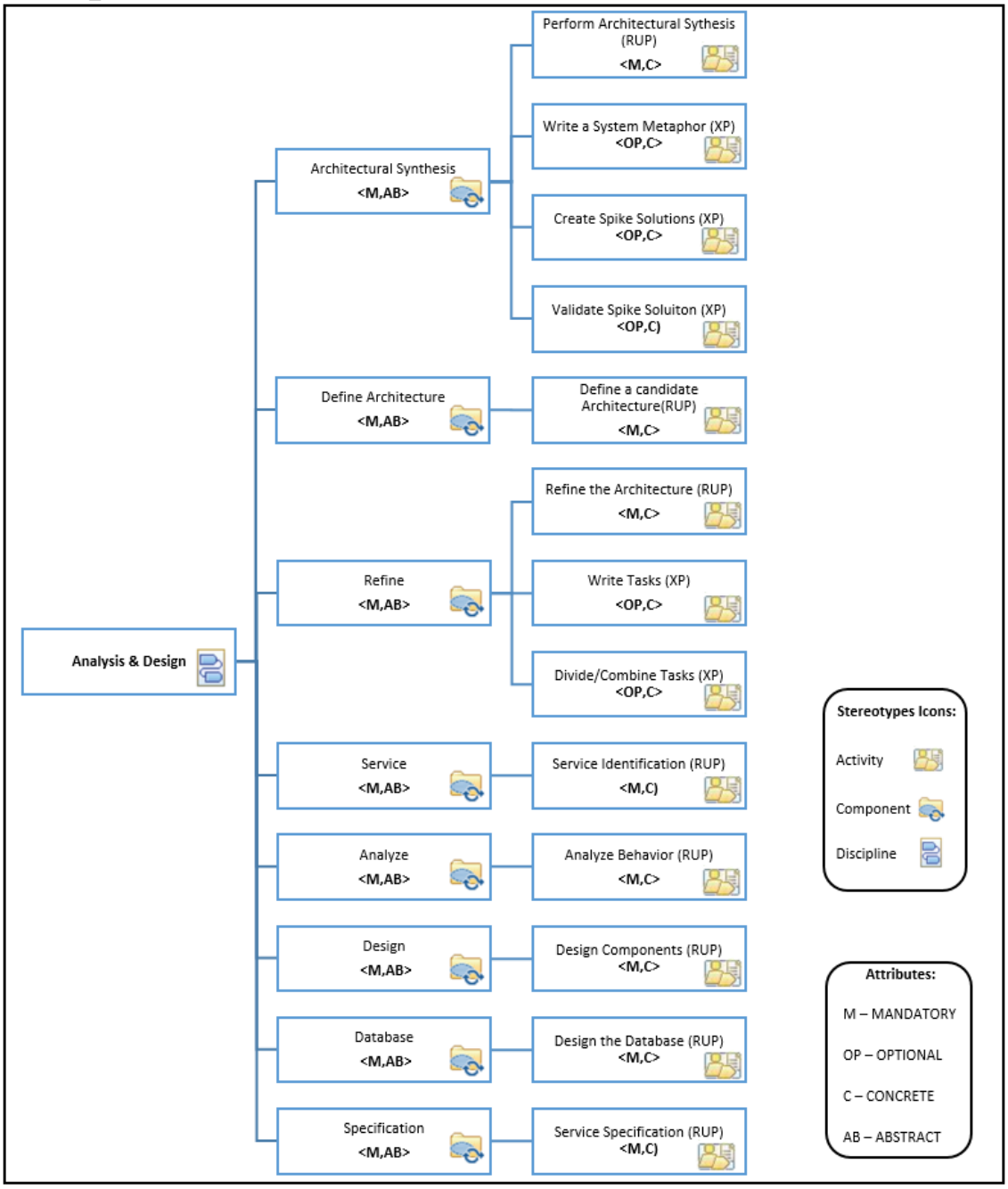

Figura A.3 – Exemplo de arquitetura definida para a disciplina de Análise e Projeto para a LPrS.

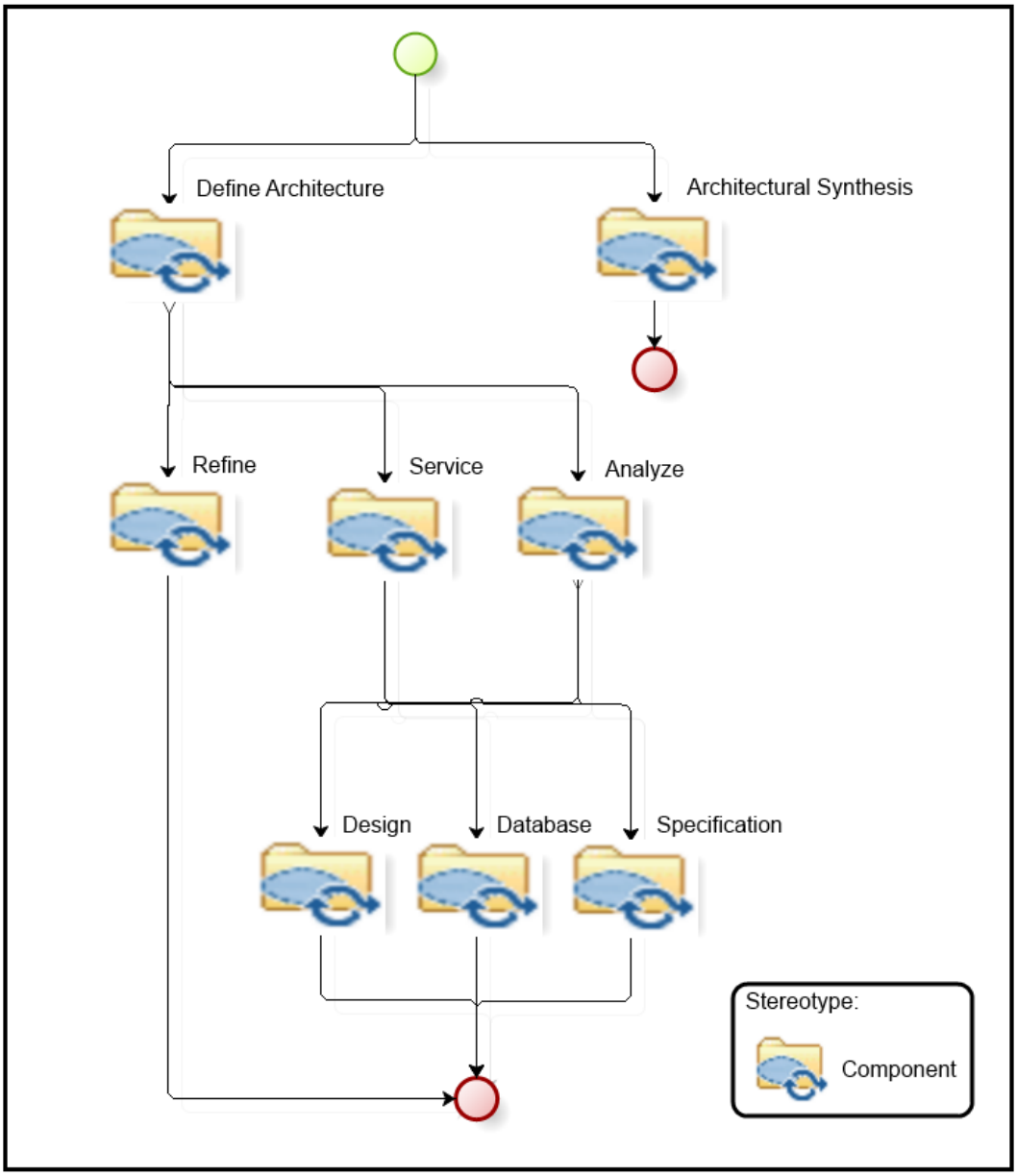

Figura A.4 – Linha de Processos de Software para a disciplina de Análise e Projeto.

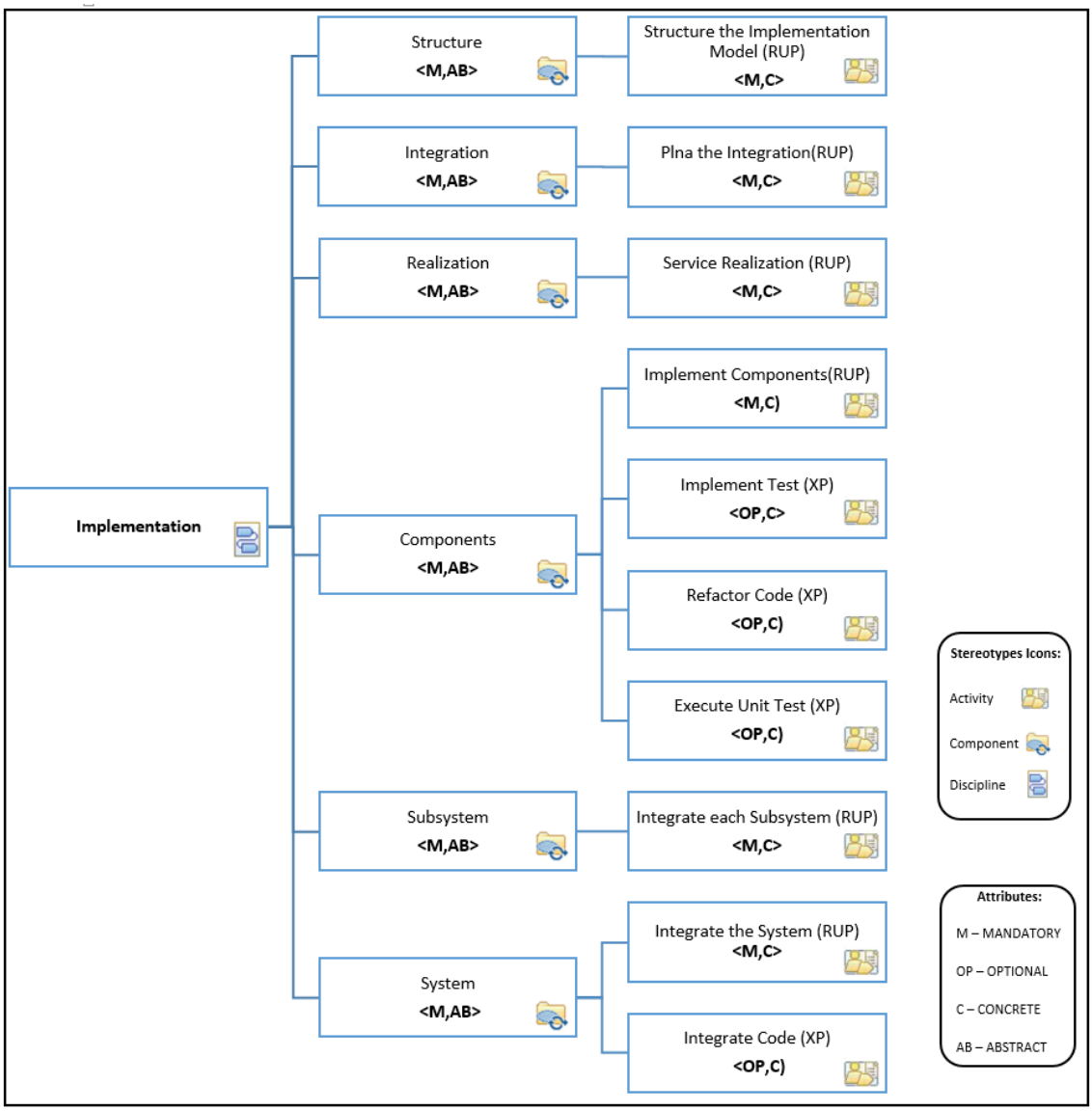

Figura A.5 – Exemplo de arquitetura definida para a disciplina de Implementação para a LPrS.

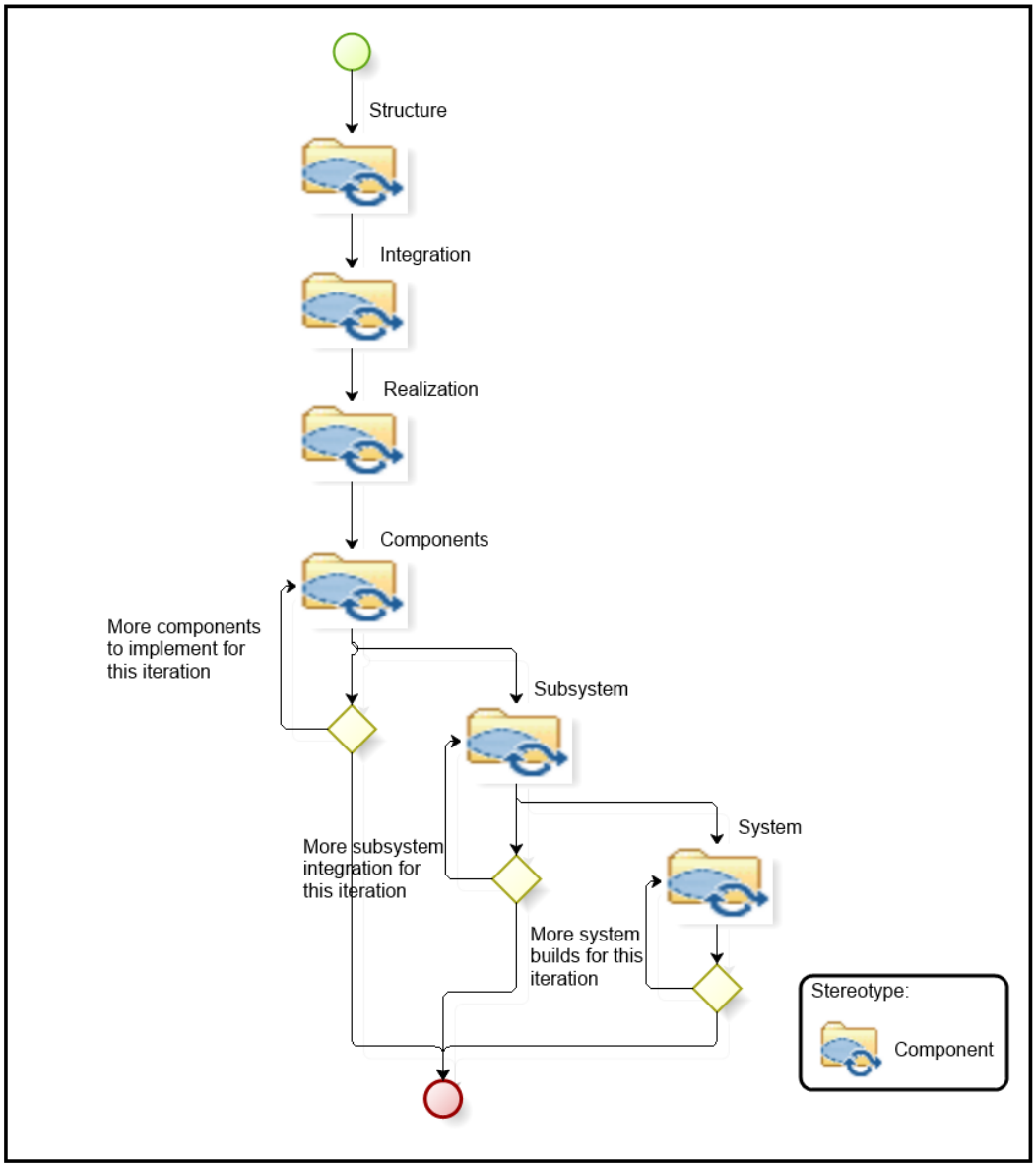

Figura A.6 – Linha de Processos de Software para a disciplina de Implementação.

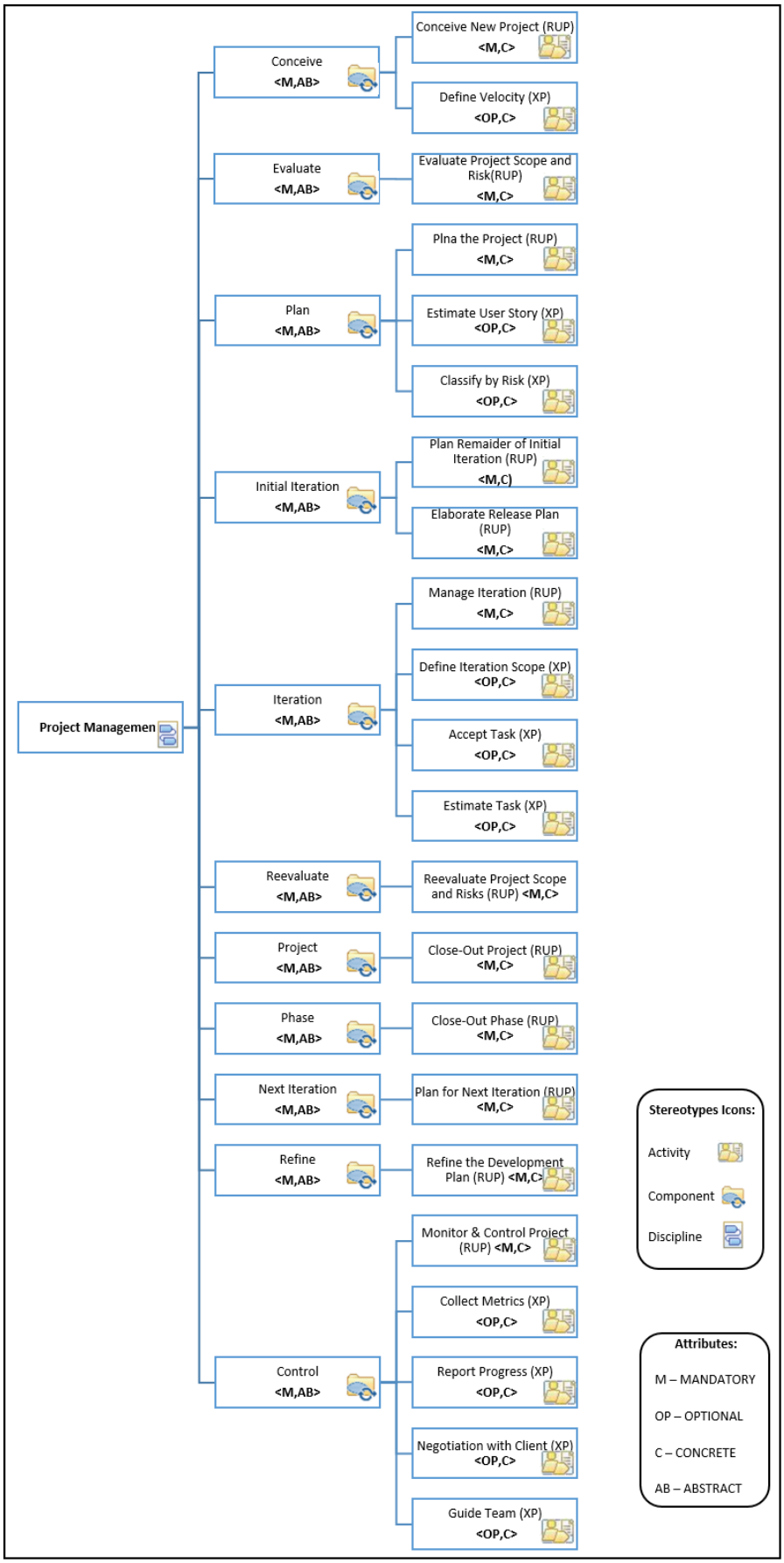

Figura A.7 – Exemplo de arquitetura definida para a disciplina de Gerencia de Projetos a LPrS.

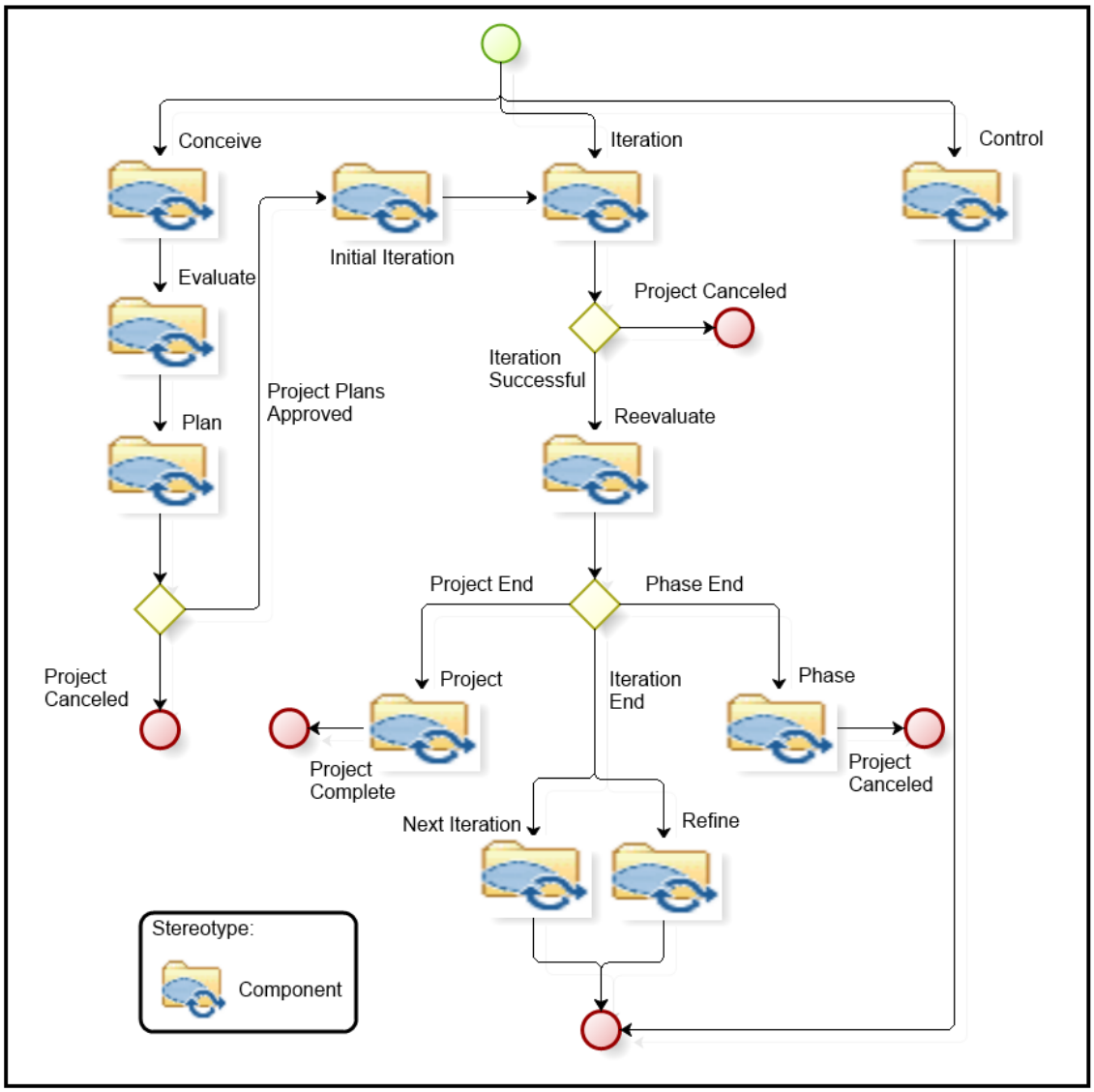

Figura A.8 – Linha de Processos de Software para a disciplina de Gerencia de Projetos.

ANEXO B – Questionário de avaliação aplicado no IF Farroupilha – Campus São Vicente do Sul

Questionário referente a utilização da ferramenta MfTP Tool para adaptação de processos de Software utilizando Linha de Processos de Software:

I. Processo de Software

• Objetivos:

o Medir a experiência/conhecimento dos participantes envolvidos no experimento;

o Avaliar a importância da adaptação de processos de software no desenvolvimento de projetos;

1. Qual seu nível de experiência em desenvolvimento de software?

( ) Muito experiente (mais de 5 anos)

( ) Experiente (entre 2 e 5 anos)

( ) Pouco experiente (menos de 2 anos)

( ) Não tenho experiência

Quantas pessoas em cada nível de classificação.

2. Como foi elaborado o processo usado atualmente?

3. Em todos os projetos é utilizado o mesmo processo.

a. Caso a resposta seja sim, o processo atende as diferentes características de todos os projetos?

b. Caso a resposta seja não, como é efetuada a adaptação?

II. Linha de Processos de Software:

• Objetivos:

o Avaliar se a Linha de Processos de Software auxiliam o processo de desenvolvimento de software;

o Avaliar se a Linha de Processos de Software auxiliam na consistência do processo gerado (nível de sequenciamento).

1. A Linha de Processos de Software auxiliam a definição de um processo de desenvolvimento? (facilitam em comparação com outras abordagens).

2. Na sua opinião qual é a maior vantagem do uso de Linhas de Processos de Software?

3. É importante considerar características da equipe e do projeto na definição das atividades a serem executadas?

4. É importante ter um conjunto de elementos de processo que possam ser reusado para definir novos ou adaptar processos existentes (importância do repositório)?

5. O processo gerado utilizando a Linha de Processo de Software foi satisfatório?

III. Ferramenta MfTP Tools SPrL

• Objetivos:

o Avaliar a usabilidade da ferramenta;

o Avaliar a usabilidade da abordagem proposta (SPrL).

1. A utilização da ferramenta é de fácil entendimento e utilização?

2. O módulo para adaptação de processos utilizando Linha de Processos de Software é de fácil entendimento e utilização?

3. O módulo para adaptação de processos utilizando Linha de Processos de Software permite gerar um processo adequado às necessidades da empresa?

4. Pontos fortes e oportunidades de melhoria da ferramenta?

IV. Críticas e sugestões: