SARP – Uma Ferramenta para Gerenciamento de Projetos Baseada na Geração de Cenários

Nádia Borges

UNIVERSIDADE DO VALE DO RIO DOS SINOS
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
SISTEMAS DE INFORMAÇÃO

Nádia Borges

SARP – Uma Ferramenta para Gerenciamento de Projetos Baseada na Geração de Cenários

São Leopoldo,
2007

Nádia Borges

SARP – Uma Ferramenta para Gerenciamento de Projetos Baseada na Geração de Cenários

Monografia apresentada à Universidade do Vale do Rio dos Sinos como requisito parcial para a obtenção do título de Bacharel em Sistemas de Informação.

Orientador: D. Sc. Sérgio Crespo

São Leopoldo
2007

Foram muitos, os que me ajudaram a concluir este trabalho.

Meus sinceros agradecimentos...

...à minha família, por todos os momentos de ausência nos últimos tempos,
pelas oportunidades geradas e
pelo eterno amor que teu tenho por eles;

...ao Alexandre, por seus memoráveis momentos de silêncio,
pela revisão gramatical feita durante o jogo
em que o Grêmio eliminava o Santos ☺ na Libertadores e
pelo amor e confiança que tem por mim;

...aos meus preciosos amigos Cleiton, Éverton, Gelson, Irajá, Karin e Mayra
pelas longas conversas via MSN,
pelas inúmeras panquecas e
pela amizade que vai além dos portões da Unisinos;

...ao mestre Sérgio Crespo, pelas valiosas sugestões
feitas durante a execução desse trabalho;

...à Tuca e ao Deco, pelos carinhos, afagos e amor incondicional.

Tên përdu, jhamâi së rëcôbro.
Tempo perdido jamais se recupera.
(Provérbio occitano medieval)

RESUMO

Os requisitos de *software* não são imutáveis. Os sistemas devem ser desenvolvidos tendo em vista que, em um futuro não muito distante, modificações serão necessárias para atender as necessidades dos usuários finais. Tendo em vista essa característica, os gerentes de projeto devem estar atentos à forma como irão calcular o impacto dessas modificações. Isso pode implicar na forma de alocação da sua equipe, no seu cronograma e, inclusive, nos prazos do projeto. O presente trabalho propõe uma ferramenta, SARP, para auxiliar os gerentes de projeto na tomada de decisão, quando deparados com alterações de requisitos, permitindo que eles visualizem o tamanho do impacto causado por essas alterações, principalmente na forma de alocação de sua equipe.

Palavras-chave: gerenciamento de projeto, volatilidade de requisitos de *software*, gestão de tempo, gestão de equipes.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1. INTRODUÇÃO

A disciplina de gerenciamento de projeto vem se solidificando no mercado desde a década de 60. Nascida na indústria bélica e aeroespacial americana, logo foi propagada para a construção civil e outras áreas da engenharia. Segundo Martins (2006), o mercado atual impõe um aumento na eficácia de todas as atividades executadas e isso está possibilitando a propagação e a utilização dos conceitos dessa disciplina.

Segundo Kerzner (2006), mudanças de diversos aspectos da vida humana são refletidas na forma como os projetos são conduzidos e nos resultados obtidos. Como conseqüência disso, a necessidade de gerenciar projetos de uma forma mais eficiente é um dos grandes desafios dos executivos modernos.

O gerenciamento de projetos demanda a criação de uma sinergia entre as necessidades de custo, tempo, escopo e qualidade. Em projetos de desenvolvimento de sistemas de informação muitos esforços estão voltados ao gerenciamento do escopo do projeto e no escopo do produto. Gerenciar o escopo do produto em projetos dessa natureza está diretamente relacionado com a gerência de requisitos. Muitos autores afirmam que a principal causa de fracasso em projetos de desenvolvimento de *softwares* está na má administração dos requisitos do sistema.

Existem inúmeras iniciativas de ferramentas que visam auxiliar o gerenciamento de projetos e outras tantas focadas na gerência de requisitos. Entretanto, poucas integram essas duas disciplinas tão necessárias para a obtenção do sucesso em projetos de desenvolvimento de *softwares*.

Dessa forma, o presente trabalho apresenta uma proposta, o SARP, que visa unir o gerenciamento de projetos com a gerência de requisitos no intuito de prover uma ferramenta integrada para a gestão de projetos de desenvolvimento de *softwares*.

Esta monografia está estruturada em quatro blocos. O primeiro apresenta a questão de pesquisa e os objetivos deste trabalho, assim como a metodologia empregada para o desenvolvimento do trabalho.

O segundo apresenta a revisão bibliográfica realizada no intuito de esclarecer os fundamentos da disciplina de gerenciamento de projetos, assim com os fundamentos da gerência de requisitos, focada na volatilidade e na rastreabilidade dos requisitos.

No capítulo seguinte, Trabalhos Relacionados, é apresentado um estudo comparativo das funcionalidades encontradas no SARP com funcionalidades existentes de outras ferramentas propostas. Assim, é possível verificar de que forma outros trabalhos focaram as mesmas questões abordadas por este trabalho.

O capítulo intitulado Ferramenta Proposta apresenta o detalhamento das funcionalidades do SARP, assim como a documentação produzida para o desenvolvimento da ferramenta. Na seqüência é apresentado um breve estudo de caso de utilização da ferramenta e as conclusões.

2. DEFINIÇÃO DO PROBLEMA

De acordo com Vargas (2003), projeto é todo empreendimento não repetitivo, caracterizado por uma seqüência clara e lógica de atividades, que se destina a atingir um objetivo específico. Deve ser conduzido por pessoas, respeitando parâmetros pré-estabelecidos de custo, tempo e qualidade. Para que um projeto obtenha sucesso, é necessário que esses parâmetros sejam atingidos da forma como foram definidos.

Segundo Pressman (2001), os requisitos do *software*, os recursos necessários e as tarefas a serem realizadas, assim como outros fatores, são fundamentais para que um projeto de desenvolvimento de *software* seja bem sucedido. A análise de todos esses fatores corresponde às atividades típicas de um gerente de projeto, as quais começam muito antes da primeira linha de código ser escrita.

Dentro das atividades de um gerente de projeto, está a gerência de mudanças de requisitos. Essa disciplina é uma atividade significativa do projeto e deve ser executada sempre, independente da fase na qual o projeto se encontra. Segundo Ramzam & Jakram (2005) é através da gerência da mudança de requisitos que o gerente consegue atingir um dos principais objetivos do desenvolvimento de um *software*: satisfazer as necessidades dos clientes.

Os requisitos de um *software* muitas vezes estão interligados entre si. Em muitos casos, a modificação de um requisito pode impactar em outros requisitos. Inclusive naqueles que já estão construídos, testados e, até mesmo, implantados. Saber identificar o impacto de uma mudança, do ponto de vista de custo, qualidade e prazo, é tarefa do gerente do projeto.

Para Ramzam & Jakram (2005), a análise do impacto está diretamente ligada com a tomada de decisão, pois, quando mudanças ocorrem, decisões devem ser feitas. As organizações devem basear as suas decisões em dados quantitativos e mensuráveis e possuir processos bem definidos para decidir. Problemas de ordem econômica podem ser gerados devidos a essas mudanças. Alterações no orçamento e no cronograma são, muitas vezes, inevitáveis, dependendo do tamanho da modificação. Problemas de ordem técnica também podem ser originados nessas variações do escopo do projeto, tais como, defeitos e baixa qualidade do produto final. A mudança de um requisito pode afetar um projeto ao ponto de identificar que ele não é viável dentro das definições iniciais.

Determinar o tamanho do impacto que a mudança de um ou mais requisitos pode causar no desenvolvimento de *softwares* é uma tarefa crítica para que o gerente obtenha sucesso no seu empreendimento. A capacidade de controlar e administrar essas mudanças durante o desenvolvimento é de igual importância para completar lacunas do produto e para garantir que ele irá satisfazer as necessidades do cliente. De acordo com Brooks (1987), a volatilidade dos requisitos de *software* é uma das principais dificuldades que existem no desenvolvimento de produtos dessa natureza. Embora essa afirmação tenha sido dita há quase duas décadas, até hoje ela é pertinente.

Nos estágios iniciais dos projetos de *software*, durante as etapas de especificação e análise, um conjunto considerável de requisitos é identificado de acordo com as características que o sistema deverá possuir e, também, de acordo com as necessidades dos usuários desse sistema. Durante as etapas seguintes (construção, testes e implantação) alterações nesses conjuntos de requisitos podem ser necessárias. Essas mudanças, de acordo com O'Neal e

Carver (2001) podem ser modificações em requisitos já definidos ou a inclusão de novos requisitos que podem, ou não, impactar em outros requisitos e nas atividades do projeto.

É ingenuidade acreditar que todos os requisitos serão identificados nos estágios inicias de um projeto e que nenhuma mudança será necessária ao longo do tempo. É importante salientar que nem todas as alterações serão iguais e terão o mesmo impacto. Alguma delas podem ser extremamente críticas para o sucesso do produto final, enquanto que outras são opcionais e podem perfeitamente ser encaixadas em verões futuras da ferramenta se o cronograma do projeto o permitir. Existem, também, aquelas mudanças que causam um impacto quase que insignificante e acabam por ser absorvidas durante o projeto.

Se as mudanças dos requisitos em um projeto não são controladas, então as expectativas criadas com relação às estimativas iniciais jamais serão realistas. Porém, as mudanças, quando necessárias, devem ser admitidas para garantir que o projeto obtenha sucesso, pelo menos no que diz respeito em produzir algo que atendas as necessidades do usuário. De nada adianta produzir um sistema, respeitando as estimativas inicias de prazo e custo, se o produto final não for útil àquelas pessoas para as quais o sistema foi desenvolvido.

Para Lam *et al* (1999), a necessidade de adotar sistemáticas para trabalhar com a gerência de requisitos é justificada por, pelo menos, três motivos. E, por causa deles, as organizações devem aprender a gerenciar as mudanças de requisitos, como parte do processo de evolução dos seus processos. Os motivos dados por Lam são os seguintes:

a) em muitos casos, o desenvolvimento de *software* é feito de forma incremental e os produtos das alterações dos requisitos acabam sendo incorporados às entregas seguintes;
b) tipicamente, as mudanças de requisitos são os principais motivos pelas manutenções em sistemas.
c) muitas empresas possuem sistemas legados que são primordiais ao negócio da organização; substituir totalmente esses sistemas nem sempre é praticável, sendo assim, eles precisam evoluir para atender as novas necessidades da empresa de forma que ela possa continuar competitiva e sobreviva no mercado cada vez mais competitivo.

Reel (1999) afirma que existem cinco fatores críticos de sucesso no gerenciamento de projetos de sistemas e um deles corresponde à adoção de uma metodologia de monitoramento e controle dos processos que envolvem o projeto. O gerente deve exigir um posicionamento constante de sua equipe. Suas decisões devem ser baseadas em dados quantitativos e não somente no conhecimento empírico que possui, embora, em muitos casos, a experiência e o sentimento do gerente a respeito de determinado problema não podem ser ignorados. O autor aponta 10 sinais que podem indicar problemas futuros em projetos:

a) os gerentes de projeto não entendem as necessidades dos usuários;
b) o escopo do projeto é fracamente definido;
c) as mudanças no projeto são insuficientemente gerenciadas;
d) a tecnologia escolhida muda;
e) as necessidades do negócio mudam;
f) as entregas (*deadlines*) não são realistas;
g) os usuários são resistentes;
h) o patrocinador (*sponsor*) está “perdido”;
i) falta de pessoas com perfis (*skills*) adequados ao projeto;
j) gerentes ignoram as melhores práticas e as lições aprendidas em projetos anteriores.

Vargas (2003) também aponta causas de fracassos de projetos e que algumas delas derivam ou influenciam as más estimativas do projeto. São elas:

a) há pouca compreensão da complexidade do projeto;
b) o projeto inclui muitas atividades e pouco tempo para realizá-las;
c) as estimativas financeiras são pobres e incompletas;
d) o projeto foi estimado com base na experiência empírica, ou *feeling* dos envolvidos, deixando em segundo plano os dados históricos de projetos similares ou até mesmo análises estatísticas efetuadas previamente;
e) não foi destinado tempo para as estimativas e o planejamento.

2.1. Questão de Pesquisa

A questão de pesquisa que guiou o propósito desse trabalho foi: *como mensurar o impacto da modificação dos requisitos dentro do projeto e ao longo do desenvolvimento*?

2.2. Objetivos

O objetivo geral desse trabalho foi o desenvolvimento de uma ferramenta para auxiliar os gerentes de projeto na tomada de decisão, quando deparados com alterações de requisitos, considerando a utilização do paradigma orientado a objetos e modelos de estimativas de custo que utilizam esse padrão. Permitindo, assim, que ele visualize o tamanho do impacto causado pela alteração de requisitos, inclusive na forma de alocação de sua equipe.

2.2.1. Objetivos Específicos

Para atingir esse objetivo, foram definidos os seguintes objetivos específicos:

- Realizar um estudo sobre gerenciamento de projetos.
- Realizar um estudo sobre os métodos e técnicas de estimativas de *softwares* que utilizam conceitos de orientação a objeto.
- Realizar um estudo sobre a volatilidade dos requisitos e o impacto dessas mudanças nos projetos.
- Identificar as características que a ferramenta deverá possuir.
- Desenvolver a ferramenta de acordo com o modelo proposto.
- Realizar experimentos em cenários reais para validar a ferramenta proposta.

2.3. Metodologia

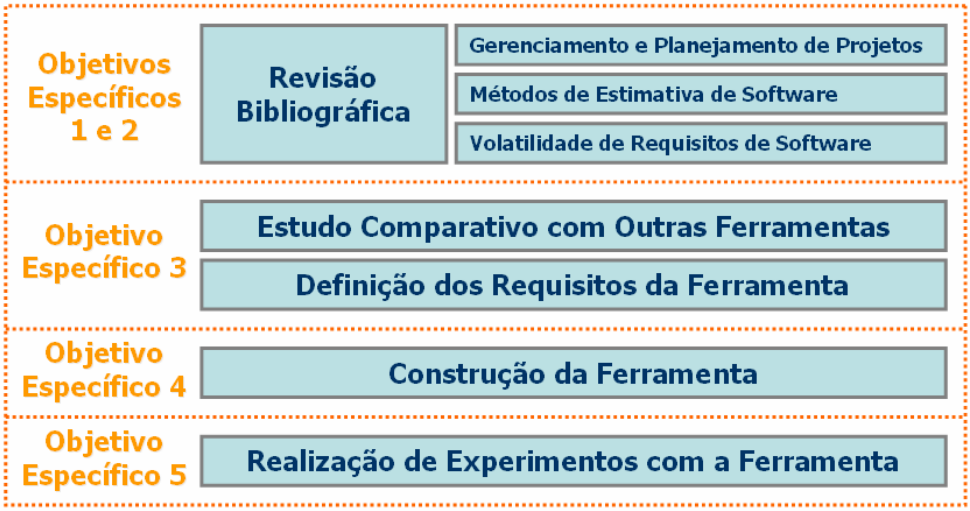


Figura 1: Esquematização da metodologia adotada.

Para atingir o objetivo principal desse trabalho, conforme descrito na Figura 1, foram executadas as seguintes atividades:

- **Estudo sobre Gerenciamento e Planejamento de Projetos:** identificação dos principais conceitos sobre gerenciamento e planejamento de projetos, visando esclarecer as funções que um gerente de projeto assume.
- **Estudo sobre Métodos de Estimativa de Software:** revisão dos métodos de estimativa de *software* que possibilitam mensurar sistemas que adotam o paradigma de orientação a objetos.
- **Estudo sobre a Volatilidade dos Requisitos de Software:** revisão de estudos sobre a volatilidade dos requisitos de *software*.
- **Estudo Comparativo de outras Ferramentas:** identificação de ferramentas desenvolvidas com os mesmo objetivos, no intuito de identificar características e funcionalidades que podem agregar valor ao presente trabalho.
- **Definição dos Requisitos da Ferramenta:** definição dos modelos que servirão como base para a construção da ferramenta de acordo com os estudos realizados. Utilização do conhecimento adquirido para determinar os requisitos da ferramenta proposta.
- **Projeto e Modelagem da Ferramenta:** especificação e documentação das funcionalidades da ferramenta. Utilizando a UML (Unified Modeling Language) para gerar a documentação do sistema.
- **Construção da Ferramenta:** a construção, propriamente dita, da ferramenta na linguagem Java.
- **Realização de experimentos com a ferramenta:** utilização de *cases* reais de modificação de requisitos para testar a ferramenta.

3. FUNDAMENTAÇÃO TEÓRICA

3.1. Princípios da Gestão de Projetos

Segundo Humphrey (2005), quando um novo projeto de desenvolvimento de *software* é iniciado, os requisitos identificados são, na grande maioria dos casos, vagos e incompletos. Dessa forma, deve-se manter o foco em determinar quais são os pontos que necessitam de maiores elucidações e onde conseguir essas informações.

Humphrey (2005) afirma que modelos conceituais devem ser desenvolvidos e refinados na medida em que esse detalhamento é realizado, entretanto, esse refinamento deve refletir no planejamento do projeto, refinando estimativas de tamanho e cronogramas. A Figura 2 mostra o modelo gráfico proposto por Humphrey (2005) para descrever como deve ocorrer esse processo iterativo de planejamento de um projeto.

Podemos observar que no ciclo de planejamento proposto por Humphrey, após a definição do cronograma, existe uma etapa de verificação, na qual são comparadas as necessidades mapeadas inicialmente com os dados do plano do projeto. Não havendo coerência entre os dois, volta-se a etapa inicial de estabelecimento do compromisso. Além disso, segundo o autor, em alguns projetos o custo estimado acaba sendo muito alto ou o cronograma muito longo e, nesses casos, novas negociações sobre os requisitos devem ocorrer levando a um replanejamento do projeto.

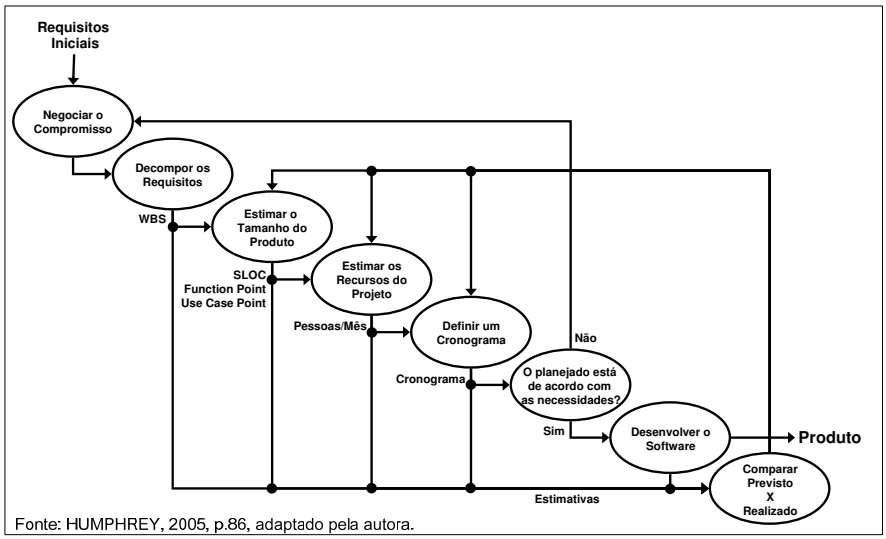


Figura 2: Ciclos de Planejamento de Projeto de acordo com Humphrey (2005).

3.2. A Disciplina de Gestão de Projetos Segundo o PMBOK

O *Project Management Institute* (PMI) é o órgão pioneiro na regulamentação e na distribuição do conhecimento sobre gerenciamento de projetos. Através do modelo chamado PMBOK (*Project Management Body Of Knowledge*), desenvolvido pelo instituto, pode-se ter acesso a orientações (boas práticas) sobre gerenciamento de projeto, sendo uma bibliografia

de referência cujo propósito é estabelecer padrões às terminologias e processos adotados. O PMBOK está atualmente em sua terceira edição, publicada em 2004, e propõem 44 processos. Dos 39 processos da edição 2000, dois foram excluídos, 13 renomeados e foram adicionados novos sete processos.

Segundo o PMI (2004), projeto é "*um esforço temporário empreendido para criar um produto, serviço ou resultado exclusivo*". O termo *temporário* ressalta o aspecto de que um projeto possui início e fim delimitado. O final de um projeto é alcançado quando os objetivos do projeto são atendidos, quando fica claro que os objetivos não podem ser atingidos conforme o planejado, ou quando as necessidades que foram identificadas inicialmente não mais existem, invalidando o projeto.

Gerência de projetos, de acordo com o PMI (2004), é "*a aplicação de conhecimentos, habilidades, ferramentas e técnicas em projetos com o objetivo de atingir ou até mesmo exceder às expectativas dos clientes e demais partes interessadas do projeto"*.

O PMBOK organiza os processos por áreas de conhecimento e não por fases de projeto, de maneira que os processos de cada área de conhecimento estão inseridos dentro de um dos cinco grupos de processos de gerenciamento de projetos. São eles:

a) **Grupo de processos de iniciação**: estudos de viabilidade e autorização para início.
b) **Grupo de processos de planejamento**: definição dos objetivos e da estratégia de implementação, assim como programação das atividades, prazos, custos, riscos e formação de equipe.
c) **Grupo de processos de execução**: coordenação das pessoas e recursos para execução do plano do projeto.
d) **Grupo de processos de monitoramento e controle**: medição do progresso do projeto visando identificar desvios para implementar ações corretivas.
e) **Grupo de processos de encerramento**: entrega do produto e formalização da aceitação do trabalho executado.

Conforme as orientações do PMI (2004), as organizações, ou os gerentes, podem dividir os projetos em fases no intuito de prover melhor controle sobre o processo de desenvolvimento do produto final. Essa divisão em fases é popularmente chamada de *ciclo de vida do projeto* e está intrinsecamente relacionada com a natureza do projeto (construção civil, construção aeroespacial, desenvolvimento de *software* e muitos outros).

A transição de uma fase para outra dentro do clico de vida do projeto normalmente envolve algum tipo de revisão ou validação dos produtos gerados antes que esses produtos sejam entregues à fase seguinte. Para Vargas (2003) esses produtos podem ser especificações, estudos de viabilidade, desenhos detalhados de um processo, protótipos, etc. Alguns desses produtos podem fazer parte do processo, mas não necessariamente fazem parte do produto final que está sendo desenvolvido e pelo qual o projeto foi idealizado. Entretanto, esses artefatos gerados em função do processo podem ser tão importantes quando o produto final, pois são eles que garantem que o produto final será desenvolvido conforme o estipulado nas fases iniciais do projeto.

Os cinco grupos de processos de gerenciamento de projetos reconhecidos pelo PMBOK seguem o modelo PDCA (*Plan-Do-Check-Act* ou Planejar-Fazer-Verificar-Agir) e estão relacionados entre si pelos resultados que produzem, ou seja, a saída de um torna-se entrada para outro, como mostra a Figura 3.

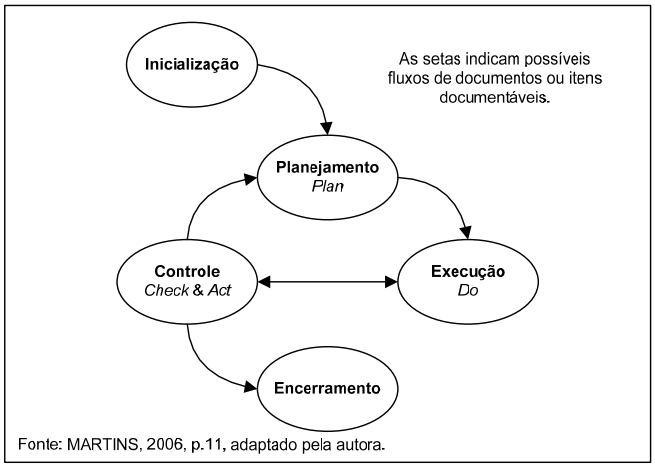


Figura 3: Processos de gerenciamento que compõem um projeto e o modelo PDCA.

Cada grupo de processo é formado por atividades que podem ser sobrepostas, ocorrendo em intensidades variáveis ao longo de cada fase do projeto (PMI, 2004). A Figura 4 ilustra a sobreposição dos grupos do processo e a intensidade de cada um deles ao longo de cada fase do projeto.

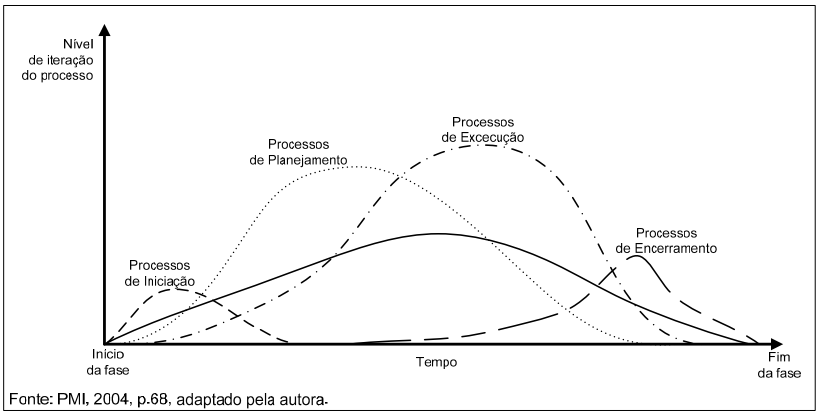


Figura 4: Nível de iteração dos grupos de processos nas fases de um projeto.

Por fim, as iterações dos grupos de processos também ocorrem entre as fases do projeto, de acordo com o ciclo de vida estabelecido. A Figura 5 exemplifica essas iterações. No exemplo da figura, os produtos da fase de projeto são entradas para a fase seguinte de desenvolvimento.

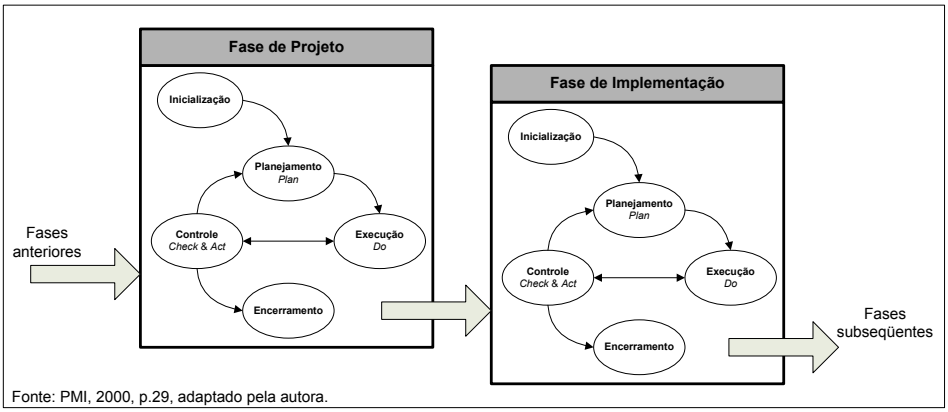


Figura 5: Iteração dos grupos de processos entre as fases do projeto.

Tabela 1: Mapeamento dos processos dentro dos grupos e das áreas de conhecimento.

Áreas de Conhecimento	Grupo de Processos de Gerenciamento de Projetos				
	Inicialização	**Planejamento**	**Execução**	**Monitoramento e Controle**	**Encerramento**
4. Gerenciamento de Integração do Projeto	4.1. Desenvolver o termo de abertura do projeto 4.2. Desenvolver a declaração de escopo preliminar do projeto	4.3. Desenvolver o plano de gerenciamento do projeto	4.4. Orientar e gerenciar a execução do projeto	4.5. Monitorar e controlar o trabalho do projeto 4.6. Controle integrado de mudanças	4.7. Encerrar o projeto
5. Gerenciamento do Escopo do Produto		5.1. Planejamento do escopo 5.2. Definição do escopo 5.3. Criar EAP		5.4. Verificação do escopo 5.5. Controle do escopo	
6. Gerenciamento de Tempo do Projeto		6.1. Definição da atividade 6.2. Seqüenciamento de atividades 6.3. Estimativa de recursos da atividade 6.4. Estimativa de duração da atividade 6.5. Desenvolvimento do cronograma		6.6. Controle do cronograma	
7. Gerenciamento de Custos do Projeto		7.1. Estimativa de custos 7.2. Orçamentação		7.3. Controle de custos	
8. Gerenciamento da Qualidade do Projeto		8.1. Planejamento da qualidade	8.2. Realizar a garantia da qualidade	8.3. Realizar o controle da qualidade	
9. Gerenciamento de Recursos Humanos do Projeto		9.1. Planejamento de recursos humanos	9.2. Contratar ou mobilizar a equipe do projeto 9.3. Desenvolver a equipe do projeto	9.4. Gerenciar a equipe do projeto	
10. Gerenciamento das Comunicações do Projeto		10.1. Planejamento das comunicações	10.2. Distribuição das Informações	10.3. Relatório de desempenho 10.4 Gerenciar as partes Interessadas	
11. Gerenciamento de Riscos do Projeto		11.1. Planejamento do gerenciamento de riscos 11.2. Identificação de riscos 11.3. Análise qualitativa de riscos 11.4. Análise quantitativa de riscos 11.5. Planejamento de respostas a riscos		11.6. Monitoramento e Controle de Riscos	
12. Gerenciamento de Aquisições do Projeto		12.1. Planejar compras e aquisições 12.2. Planejar contratações	12.3. Solicitar respostas de fornecedores 12.4. Selecionar fornecedores	12.5. Administração de contrato	12.6. Encerramento do contrato

Fonte: PMI, 2004, p.70, adaptado pela autora.

A Tabela 1 reflete o mapeamento dos 44 processos de gerenciamento de projetos que compõem a terceira edição do PMBOK dentro dos cinco grupos de processos e dentro das nove áreas de conhecimento descritas a seguir.

a) **Gerenciamento de Integração do Projeto**: descreve os processos necessários para assegurar a perfeita coordenação entre todos os processos envolvidos.
b) **Gerenciamento do Escopo do Produto**: descreve os processos necessários para assegurar que o projeto contemple todo e somente o trabalho requerido para completar o projeto com sucesso.
c) **Gerenciamento de Tempo do Projeto**: descreve os processos necessários para assegurar o encerramento do projeto no tempo definido.
d) **Gerenciamento de Custos do Projeto**: descreve os processos necessários para assegurar que o projeto se encerrará dentro do orçamento.

e) **Gerenciamento da Qualidade do Projeto**: descreve os processos necessários para assegurar que o projeto satisfará as necessidades contratadas.
f) **Gerenciamento de Recursos Humanos do Projeto**: descreve os processos necessários para assegurar o melhor desempenho das pessoas envolvidas.
g) **Gerenciamento das Comunicações do Projeto**: descreve os processos necessários assegurar, no tempo certo, a geração, disseminação e armazenamento das informações do projeto.
h) **Gerenciamento de Riscos do Projeto**: descreve os processos necessários na identificação, análise e controle dos riscos inerentes ao projeto.
i) **Gerenciamento de Aquisições do Projeto**: descreve os processos necessários para aquisição de bens e serviços fora da organização.

3.2.1. Área de Conhecimento *Gerenciamento de Integração do Projeto*

A área de conhecimento em Gerenciamento de Integração do Projeto inclui os processos e as atividades necessárias para identificar, definir, combinar, unificar e coordenar os diversos processos e atividades de gerenciamento de projetos dentro dos cinco grupos de processos. A integração consiste em fazer escolhas sobre em que pontos devem ser concentrados os recursos e os esforços, antecipando possíveis problemas e tratando-os antes de se tornarem críticos.

A necessidade de integração se torna evidente em situações nas quais os processos individuais interagem, havendo necessidade de um controle integrado para realizar os objetivos do projeto dentro dos procedimentos definidos pela organização. A área de gerenciamento de integração de projeto inclui os seguintes processos (PMI, 2004):

a) **Desenvolver o termo de abertura do projeto (4.1)**: desenvolver o termo que autoriza formalmente um projeto ou uma fase do projeto.
b) **Desenvolver a declaração do escopo preliminar do projeto (4.2)**: desenvolvimento da declaração do escopo preliminar do projeto que fornece uma descrição em alto nível do escopo a ser trabalhado.
c) **Descrever o plano de gerenciamento do projeto (4.3)**: documentação das ações necessárias para definir, preparar, integrar e coordenar todos os planos auxiliares em um plano de gerenciamento do projeto.
d) **Orientar e gerenciar a execução do projeto (4.4)**: execução do trabalho definido no plano de gerenciamento do projeto para atingir os requisitos definidos para o projeto.
e) **Monitorar e controlar o trabalho do projeto (4.5)**: monitoramento e controle dos processos usados para iniciar, planejar, executar e encerrar um projeto.
f) **Controle integrado de mudanças (4.6)**: realizado desde o início do projeto até o seu término. É necessário, pois raramente um projeto é executado seguindo com exatidão o plano inicial. Os planos do projeto precisam ser mantidos rigorosamente atualizados em função das mudanças, de forma que elas, quando aprovadas, sejam incorporadas a uma nova revisão dos planos do projeto. Todas as mudanças solicitadas e documentadas precisam ser aceitas ou rejeitadas por uma autoridade dentro da equipe de gerenciamento de projetos ou por uma organização externa que tenha representatividade.
g) **Encerrar o projeto (4.7):** finalização de todas as atividades em todos os grupos de processos de gerenciamento de projetos para encerrar formalmente o projeto ou uma de suas fases.

3.2.2. Área de Conhecimento *Gerenciamento do Escopo do Projeto*

O gerenciamento do escopo do projeto inclui os processos necessários para garantir que o projeto inclua todo trabalho necessário para terminar o projeto. Essa área de conhecimento trata especificamente da definição e controle do que está e do que não está incluído no projeto. Em geral, o resultado de um projeto é um único produto que pode incluir componentes auxiliares, sendo que cada um deles terá seu próprio escopo delimitado e inter-relacionado aos demais componentes do produto principal.

No contexto de projetos, o termo *escopo* pode se referir a:

a) **Escopo do produto**: as características os funções que descrevem um produto, serviço ou resultado.
b) **Escopo do projeto**: o trabalho que precisa ser realizado para entregar um produto, serviço ou resultado com as características e funções especificadas.

A área de gerenciamento de escopo inclui os seguintes processos (PMI, 2004):

a) **Planejamento do Escopo (5.1)**: criação de um plano de gerenciamento de escopo do projeto que irá documentar como o escopo do projeto será definido, verificado e controlado. Nesse processo também é definido com a estrutura analítica do projeto (EAP) será criada e definida.
b) **Definição do Escopo (5.2)**: desenvolvimento de uma declaração detalhada do escopo do projeto como base para futuras decisões do projeto.
c) **Criar EAP (5.3)**: subdivisão das principais entregas do projeto e do trabalho do projeto em componentes menores e mais facilmente gerenciáveis.
d) **Verificação do Escopo (5.4)**: processo de obtenção da aceitação formal das partes interessadas do escopo do projeto e das entregas associadas. Esse processo inclui uma revisão das entregas para garantir que cada uma delas foi terminada de forma satisfatória. Esse processo difere do controle da qualidade. Enquanto o controle da qualidade trata do atendimento dos requisitos de qualidade especificados para as entregas, a verificação de escopo trata especificamente da aceitação das entregas. Em geral o controle da qualidade ocorre antes da verificação do escopo, mas esses dois processos podem ser realizados em paralelo.
e) **Controle do Escopo (5.5)**: esse processo trata de influenciar os fatores que criam mudanças no escopo do projeto e de controlar o impacto dessas mudanças. Esse processo também é utilizado para gerenciar as mudanças no momento em que elas efetivamente ocorrem e é interligado com outros processos de controle. As mudanças que não são controladas frequentemente são chamadas de aumento do escopo do projeto. As mudanças são inevitáveis e, portanto, existe a necessidade de algum tipo de processo sistemático para gerir essas mudanças.

Para o presente trabalho destacamos alguns artefatos de entrada ou produzidos pelos processos dessa área, assim como algumas técnicas e ferramentas sugeridas (PMI, 2004):

a) **EAP - Estrutura analítica do projeto:** também conhecida como WBS (*Work Breakdown Structure*), é uma decomposição hierárquica orientada às entregas do trabalho a ser executado pela equipe do projeto, que organiza e define o escopo total do projeto. A EAP subdivide o trabalho em partes menores que são mais facilmente gerenciáveis. Entretanto, ela não pode ser confundida com outros tipos de estruturas analíticas usadas para representar informações do projeto, tais como organograma, estrutura analítica de riscos ou estrutura analítica de recursos, por exemplo.

b) **Solicitação de mudança aprovada**: podem ocasionar mudanças no escopo e na qualidade do projeto, alterações nos custos estimados ou no cronograma definido. Essas mudanças são frequentemente identificadas e aprovadas enquanto o trabalho do projeto está em andamento e, em alguns casos, ocorre o retrabalho. Ou seja, atividades que já estavam concluídas precisam novamente ser trabalhadas.
c) **Inspeção**: essa técnica inclui atividades como medição, exame e verificação para determinar se o trabalho e as entregas atendem aos requisitos e aos critérios de aceitação do produto. As inspeções podem receber diversos nomes, como revisões do produto, auditorias, homologações, etc.
d) **Sistema de controle de mudanças**: define os procedimentos necessários para efetuar mudanças no escopo do projeto e no escopo do produto. O sistema deve contemplar a documentação necessária sobre as mudanças e os níveis de aprovação necessários para autorizar mudanças. O sistema de controle de mudanças de escopo pode estar integrado a qualquer sistema de informação utilizado para o gerenciamento de projetos de forma a auxiliar no controle do escopo do projeto.
e) **Análise da variação**: as medições de desempenho do projeto são utilizadas para avaliar a extensão da variação. Determinar a causa da variação em relação ao planejado e decidir se são necessárias ações corretivas são aspectos importantes do controle de escopo do projeto.
f) **Replanejamento**: as solicitações de mudanças aprovadas que afetam o escopo do projeto pode exigir modificações na EAP, na declaração do escopo do projeto e no plano de gerenciamento do escopo do projeto. Essas solicitações de mudanças aprovadas podem resultar em atualizações nos componentes do plano de gerenciamento de projeto.
g) **Sistema de gerenciamento de configuração**: fornece procedimentos para obtenção da situação das entregas e garante que as mudanças solicitadas no escopo do projeto e no escopo do produto serão cuidadosamente consideradas e documentadas, antes de serem processadas pelo processo de controle integrado de mudanças.

3.2.3. Área de Conhecimento *Gerenciamento de Tempo do Projeto*

O gerenciamento de tempo do projeto inclui os processos necessários para realizar a conclusão do projeto dentro dos tempos estabelecidos. Em alguns projetos, especificamente nos de escopo reduzido, o seqüenciamento das atividades, a estimativa de recursos da atividade, a estimativa de duração da atividade e o desenvolvimento do cronograma estão tão estreitamente ligados que são considerados um único processo que pode vir a ser realizado por uma única pessoa. A área de gerenciamento de escopo do projeto inclui os seguintes processos (PMI, 2004):

a) **Definição das Atividades (6.1)**: identificação das atividades específicas que devem ser realizadas para produzir os diversos subprodutos do projeto;
b) **Seqüenciamento das Atividades (6.2)**: identificação e documentação das dependências entre as atividades;
c) **Estimativa de Recursos das Atividades (6.3)**: envolve determinar os recursos necessários para executar cada atividade, assim como as quantidades necessárias e quando cada recurso estará disponível para realizar as atividades do projeto.
d) **Estimativa de Duração das Atividades (6.4)**: estimação do número de períodos de trabalhos (prazos) que serão necessários para completar as atividades individuais;

e) **Desenvolvimento do Cronograma (6.5)**: criação do cronograma do projeto a partir da análise da seqüência das atividades, suas durações e as necessidades de recursos;
f) **Controle do Cronograma (6.6)**: criação do cronograma do projeto a partir da análise da seqüência das atividades, suas durações, e as necessidades de recursos;

Para o presente trabalho destacamos alguns artefatos de entrada ou produzidos pelos processos dessa área, assim como algumas técnicas e ferramentas sugeridas, segundo o PMBOK (PMI, 2004):

a) **Lista de Atividades:** lista contendo todas as atividades de cronograma que estão planejadas para serem executadas durante o projeto. Essa lista não deve conter atividades relacionadas a itens que estão fora do escopo do projeto. Cada atividade deve conter um identificador da atividade e uma descrição do escopo de trabalho.
b) **Atributos das Atividades:** os atributos das atividades são informações adicionais a lista de atividades. A lista de atributo de cada atividade pode incluir informações sobre atividades predecessoras, atividades sucessoras, relações lógicas, folgas, origens dos requisitos, datas obrigatórias, restrições e suposições. Os atributos também podem conter uma pessoa responsável pela execução da atividade, um local onde será executada, assim como o esforço necessário para executar a tarefa. Esses atributos são utilizados para selecionar, ordenar e planejar o desenvolvimento do cronograma do projeto.
c) **Método do diagrama de precedência (MDP)**: é um método que utiliza caixas e setas para construir o diagrama de rede do cronograma do projeto. Com esse diagrama pode-se representar as dependências existentes entre as atividades. A Figura 6 apresenta o exemplo de um diagrama de precedência das atividades. Essa técnica inclui quatro tipos de relações de dependências entre as atividades. São elas:
    - **Término-para-Início (Finish-to-Start)**: o início do trabalho de uma atividade sucessora depende do término da atividade predecessora. Este é o método de análise mais comummente utilizado, segundo o PMBOK (PMI, 2004).
    - **Término-para-Término (Finish-to-Finish):** o término dos trabalhos de uma atividade sucessora depende do término dos trabalhos da atividade predecessora.
    - **Início-para-Início (Start-to-Start)**: início dos trabalhos de uma atividade sucessora depende do início dos trabalhos da atividade predecessora.
    - **Início-para-Término (Start-to-Finish)**: o término de uma atividade sucessora é dependente do início da atividade predecessora.
d) **Opinião especializada:** normalmente necessária para avaliar as relações entre as entradas e os processos. Um grupo de especialistas em planejamento de recursos e estimativas pode contribuir no processo de estimativa de recursos das atividades.
e) **Método do caminho crítico:** técnica de análise de rede utilizada pra identificar possíveis gargalos no projeto. Essa técnica é detalhada no capítulo 3.3 (Análise do Caminho Crítico).
f) **Análise de cenário do tipo "e se?":** análises do tipo "*o que aconteceria se*" podem ser feitas usando a rede de atividades para simular diferentes cenários, como atrasos na entrega de um componente importante, aumento no tempo de uma atividade de engenharia específica ou na introdução de fatores externos, por exemplo. Os resultados de simulações desses tipos podem ser usados para avaliar a

viabilidade do cronograma e no preparo de planos de contingência para superar ou mitigar o impacto de situações inesperadas.

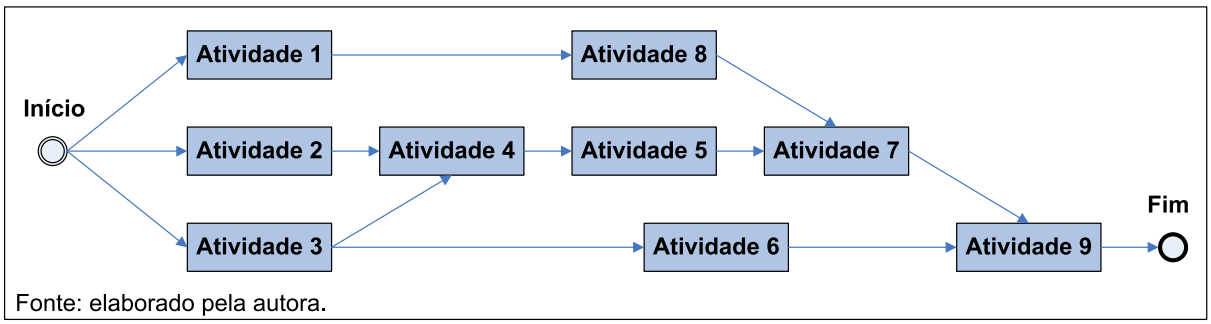


Figura 6: Exemplo de diagrama precedência ou diagrama de rede.

3.2.4. CMMI e a Gestão de Projetos

O CMMI (*Capability Maturity Model Integration*) objetiva o estabelecimento de controles e métricas para o desenvolvimento de *software* de mais qualidade. Esse modelo, publicado pelo SEI (*Software Engineering Institute*) e pela *Carnegie Mellon University*, possui um conjunto de "melhores práticas" que devem ser utilizadas para um fim específico. Ele incorpora necessidades de melhorias de processo de desenvolvimento de *software* e está alinhado com o PMBOK. Dentre as áreas de processos existentes no CMMI, existem duas que mais se destacam quanto ao gerenciamento de projetos: a área “Planejamento de Projeto” e a área “Monitoramento e Controle de projeto”.

A área de processo “Planejamento de Projeto”, segundo o CMMI, envolve: desenvolver o plano do projeto; interagir com os *stakeholders* de forma apropriada; obter compromissos com o plano e manter o plano.

De acordo com o modelo, o planejamento inicia com os requisitos que definem o produto, envolve realizar estimativas sobre os produtos de trabalho, assim como determinar os recursos necessários, negociar compromissos, produzir cronograma e identificar e analisar os riscos do projeto. O modelo indica que iterações nessas atividades podem ser necessárias para estabelecer o plano final do projeto, da mesma forma como Humphrey (2005) afirma. Além disso, o CMMI exige que o plano do projeto deva ser revisado conforme a evolução do projeto, para suportar mudanças nos requisitos e compromissos, estimativas imprecisas, ações corretivas e mudanças no processo.

A área de processo “Monitoramento e Controle do Projeto”, segundo o CMMI, tem como objetivo oferecer um entendimento do progresso do projeto, de maneira que as ações corretivas apropriadas possam ser tomadas quando o desempenho do projeto se desviar significativamente do plano original.

Um plano de projeto documentado é a base para monitorar as atividades, comunicar status e tomar as ações corretivas necessárias. O progresso é basicamente dado pela comparação dos atributos reais do projeto (produtos de trabalho e tarefas, esforço, custo e cronograma) com o que está definido no plano. O modelo exige que quando a situação real do projeto se desviar significativamente dos valores esperados, as ações corretivas deve ser executadas. Estas ações podem exigir o replanejamento do projeto, o que pode levar a revisão do plano original, estabelecimento de novos acordos ou inclusão de novas atividades.

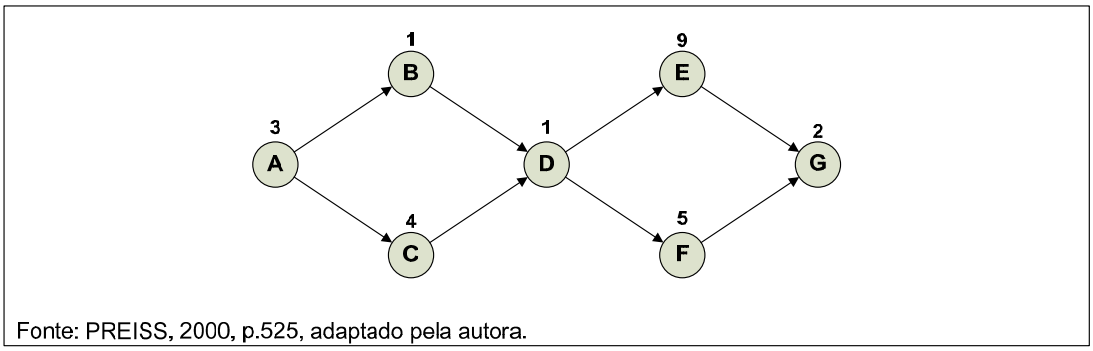
3.3. Análise do Caminho Crítico

De acordo com Preiss (2000) a análise do caminho crítico se aplica a inúmeros contextos, desde o planejamento de projetos à análise combinatória de circuitos lógicos. Goldbarg & Luna (2000) afirmam que esse modelo é extremamente útil para a solução de problemas que possuem um número muito grande de atividades que ocorrem em paralelo, com duração variada e que apresentam, além disso, pontos de concorrência e interdependência.

Podemos representar as atividades, e as precedências existentes entre elas, através de um grafo direcionado acíclico ponderado em vértices. Ou seja, cada vértice do grafo representa uma atividade e o peso do vértice representa o custo (em tempo) pra a execução da atividade. O direcionamento das arestas entre os vértices representam as restrições de escalonamento (precedências). Por exemplo, uma aresta do vértice *v* para o vértice *w* indica que a atividade *v* deve ser completada antes da atividade *w* ser iniciada. Fica claro, então, que esse grafo é acíclico, ou seja, é um grafo direcionado e que não contem ciclos, segundo Preiss (2000).

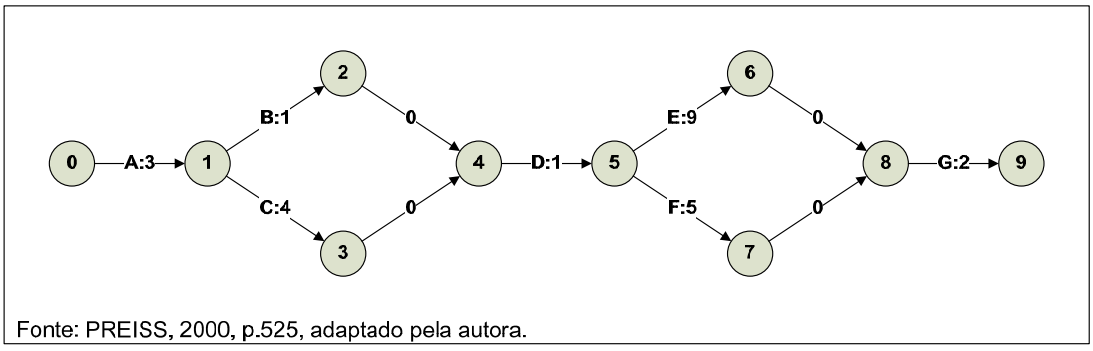
A análise do caminho crítico responde a duas perguntas, conforme Preiss (2000):

a) Qual o menor tempo necessário para completar todas as tarefas?
b) Data uma atividade, é possível atrasar o tempo de realização da atividade sem comprometer o tempo total? Se sim, de quanto tempo pode ser esse atraso?



Fonte: PREISS, 2000, p.525, adaptado pela autora.

Figura 7: Um grafo nó-atividade.



Fonte: PREISS, 2000, p.525, adaptado pela autora.

Figura 8: Um grafo nó-evento correspondendo à Figura 7.

O grafo do tipo *nó-atividade*, exemplificado pela Figura 7, é um grafo ponderado em vértices. Entretanto, de acordo com Preiss (2000), os algoritmos utilizados para análise dos grafos, tais como Dijkstra, Floyd, Prim e Kruskal, requerem grafos ponderados em arestas. Sendo assim, é necessário transformar o grafo ponderado em vértice no seu *dual* ponderado

em arestas. Esse novo grafo é chamado grafo *nó-evento*. No grafo dual as arestas representam as atividades e os vértices representam o início e o término de cada atividade. A Figura 8 exibe o dual do grafo da Figura 7. Quando uma atividade depende de mais de um predecessor, é necessário inserir arestas fictícias. Como essas arestas não representam atividades, mas sim restrições, seus pesos são iguais a zero.

No intuito de identificar o caminho crítico de um grafo, para cada vértice no grafo nó-evento, definimos dois tempos. O primeiro (E_w) é o *menor tempo de início* para o evento *v*. Esse tempo indica o momento mais cedo no qual o evento *v* pode começar. O menor tempo de início (*earliest time*) é definido pela Eq. 1.

$$E_w = \begin{cases} 0 & w = v_i, \\ \min_{(v,w)\in\mathscr{P}(w)} E_v + C(v,w) & \text{caso contrário.} \end{cases} \qquad \text{Eq. 1}$$

Onde:

v_i é o evento inicial,
$\mathscr{P}(w)$ é o conjunto de arestas incidentes à *w*,
$C(v,w)$ é o peso do vértice (v,w).

O maior tempo de início (*latest time*) define o maior tempo no qual o evento *v* pode ocorrer e pode ser obtido da mesma forma que o menor tempo de início. O maior tempo de início é definido pela Eq. 2:

$$L_v = \begin{cases} E_{v_f} & w = v_f, \\ \max_{(v,w)\in\mathscr{S}(w)} E_w - C(v,w) & \text{caso contrário.} \end{cases} \qquad \text{Eq. 2}$$

Onde:

v_f é o evento final.

Identificados os menores e maiores tempos de início de cada atividade podemos, então, computar o tempo disponível em cada uma delas (folgas). Definimos as folgas de tempo de uma atividade como a quantidade de tempo que uma atividade pode ser atrasada sem afetar o tempo total de realização do projeto. A folga de tempo de uma atividade representada pela aresta (v,w) é dada pela Eq. 3.

$$S(v,w) = L_w - E_v - C(v,w) \qquad \text{Eq. 3}$$

As atividades com folga zero são *críticas*, ou seja, essas atividades precisam ser realizadas no seu tempo, pois qualquer atraso na sua execução afetará o tempo total do projeto. Um *caminho crítico* no grafo nó-evento é um caminho composto apenas por atividades críticas do início ao fim do grafo.

Tabela 2: Resultado da análise do caminho crítico do grafo apresentado na Figura 7.

Atividade	$C(v,w)$	E_v	L_w	$S(v,w)$
A	3	0	3	0
B	1	3	7	3
C	4	3	7	0
D	1	7	8	0
E	9	8	7	0
F	5	8	17	4
G	2	17	18	0

Fonte: PREISS, 2000, p.527.

A Tabela 2 mostra o resultado da análise do caminho crítico do gráfico nó-atividade mostrado na Figura 7. Os resultados indicam que o caminho crítico para esse grafo é:

{A, C, D, E, G}

3.4. O Modelo de Estimativa de Esforço *Pontos por Caso de Uso*

Os Pontos por Caso de Uso (UCP - *Use Case Points*) foram criados por Gustav Karner em 1993 com a finalidade de ser um modelo de estimativa para projetos desenvolvidos no paradigma de orientação a objeto. Para a criação desse modelo, Karner baseou-se na Análise de Ponto de Função, Mark II e no Modelo de Casos de Uso. Posteriormente o Modelo de Casos de Uso foi incorporado na UML (*Unified Modeling Language*) e vem sendo utilizado em larga escala pelas empresas atualmente.

No modelo de estimativa UCP, algumas características da Análise por Ponto de Função formam substituídas, foram criados fatores ambientais e foi proposta uma estimativa com relação à produtividade dos indivíduos envolvidos no projeto. Os fatores ambientais e técnicos utilizados por esse modelo para ajustar a estimativa final levam em consideração dados sobre a experiência da equipe com projetos do mesmo porte. Karner (1993) propôs que a produtividade considerada nas estimativas deveria ser de 20 homens/hora por UCP. No seu estudo, tento encontrar uma relação entre a UCP e a quantidade de recursos necessários, mas os dados que ele obteve não foram suficientes para determinar esse relacionamento.

Assim como os outros modelos de estimativa de *softwares* que existem atualmente, o UCP possui um processo de contagem que será descrito a seguir.

3.4.1. Contagem dos UCPs

O cálculo do UCP é feito mediante a avaliação e classificação de cada caso de uso encontrado no sistema. Todos os atores encontrados nos Casos de Uso também devem ser classificados. Segundo Karner (1993), o processo de contagem de está dividido em seis etapas:

a) contar os atores e atribuir o grau de complexidade;
b) contar os casos de uso e atribuir o grau de complexidade;
c) calcular os UCPs não ajustados;
d) determinar o fator de complexidade técnica;
e) determinar a eficiência do fator ambiental;
f) calcular os UCPs ajustados.

Para contar os atores e atribuir o grau de complexidade deve-se identificar e multiplicar o número total de atores de acordo com o tipo de complexidade deles, conforme mostra a Tabela 3. Em seguida deve-se realizar o cálculo dos pelos dos atores não ajustados (UAW - *Unajusted Actor Weight*) que é realizado somando-se os produtos do número de atores de cada tipo pelo seu respectivo peso.

Para contar os casos de uso e atribuir o grau de complexidade (segunda etapa) deve-se identificar a complexidade através do número de transações que o caso de uso envolve. Segundo Bittner & Spencer (2003), transação é definida como um conjunto de atividades atômicas, as quais são executadas completamente ou não. Quando existir uma mesma

transação em todos os diagramas de seqüência como, por exemplo, o *login* ou outros procedimentos de segurança, a transação deve ser contada apenas uma vez, pois a funcionalidade é implementada apenas uma vez e reutilizada em outros casos de uso.

Tabela 3: Peso dos atores.

Complexidade do Ator	Peso	Descrição
Simples	1	Outro sistema acessado através de uma API de programação.
Médio	2	Outro sistema interagindo através de um protocolo de comunicação, como TCP/IP ou FTP
Complexo	3	Um usuário interagindo através de uma interface gráfica (*stand-alone* ou *web*).

Para se chegar ao valor relativo aos casos de uso não ajustados (UUCW - *Unajusted Use Case Weight*), deve-se multiplicar cada caso de uso pelo seu respectivo peso, conforme a Tabela 4, e realizar o somatório desses valores, da mesma forma como é feito para se obter o valor dos atores não ajustados (UAW).

Tabela 4: Peso dos casos de uso.

Complexidade do Caso de Uso	Peso	Número de Transações	Número de Classes	Descrição
Simples	5	Até 3.	Até 5.	Possui até três transações incluindo os passos alternativos e, para ser realizado, envolve menos de cinco classes de análise.
Médio	10	Entre 4 e 7.	Entre 5 e10.	Possui de quatro a sete transações incluindo os passos alternativos e, para ser realizado, envolve de cinco a dez classes de análise.
Complexo	15	Mais que 7.	Mais que 10.	Possui mais de sete transações incluindo os passos alternativos e, para ser realizado, envolve mais de dez classes de análise.

Para calcular o valor dos casos de uso não ajustados (UUCP - *Unajusted Use Case Points*), deve-se somar o somatório dos pesos dos atores (UAW) como o somatório dos pesos dos casos de usos (UUCW), como mostra a Eq. 4.

$$\mathrm{UUCP} = \sum \mathrm{UAW} + \sum \mathrm{UUCW} \qquad \text{Eq. 4}$$

Identificados os pontos de caso de uso não ajustados, é necessário definir o fator de complexidade técnica (FCT). Esses fatores podem variar em uma escala de zero a cinco, de acordo com o grau de dificuldade do sistema a ser construído. O valor zero indica pouca criticidade e baixa complexidade (irrelevante para o projeto) e o valor cinco indica alta criticidade e alta complexidade (essencial para o projeto). Após determinar os valores de cada um dos fatores e de multiplicar esses valores pelos respectivos pesos, conforme a Tabela 5, é necessário somar o total encontrado e aplicar a fórmula definida na Eq. 5, para obter o fator de complexidade técnica (FCT):

$$\mathrm{FCT} = 0{,}6 + \left(0{,}01 \times \sum \mathrm{Fatores\ Técnicos}\right) \qquad \text{Eq. 5}$$

Definido o fator de complexidade técnica, é necessário determinar a eficiência do fator ambiental, identificados na Tabela 6. Para isso, é atribuído a cada fator um valor, que pode variar de zero a cinco, conforme descrito a seguir:

a) zero indica baixa habilidade (pouca experiência no domínio da aplicação);
b) três indica média experiência;
c) cinco indica alta experiência.

Deve-se, então, multiplicar o valor atribuído ao fator pelo respectivo peso e aplicar a fórmula descria na Eq. 6 para se obter o valor do fator ambiental.

$$FCA = 1{,}4 + \left(-0{,}03 \times \sum \text{Fator Ambiental}\right) \quad \text{Eq. 6}$$

Para chegar ao valor dos pontos por caso de uso (UCP), devem ser multiplicados os pontos por caso de uso não ajustados, o fator de complexidade técnica e o fator de complexidade ambiental, como demonstra a Eq. 7.

$$UCP = UUCP \times FCT \times FCA \quad \text{Eq. 7}$$

Tabela 5: Fatores de complexidade técnica.

	Descrição	**Peso**
T1	Sistemas Distribuídos	2,0
T2	Desempenho da aplicação	1,0
T3	Eficiência do usuário final (on-line)	1,0
T4	Processamento interno complexo	1,0
T5	Reusabilidade do código em outras aplicações	1,0
T6	Facilidade de instalação	0,5
T7	Usabilidade (facilidade operacional)	0,5
T8	Portabilidade	2,0
T9	Facilidade de manutenção	1,0
T10	Concorrência	1,0
T11	Características especiais de segurança	1,0
T12	Acesso direto para terceiros	1,0
T13	Facilidades especiais de treinamento	1,0

Tabela 6: Fatores de complexidade ambiental.

	Descrição	**Peso**
A1	Familiaridade com o Processo de Desenvolvimento de Software	1,5
A2	Experiência na Aplicação	0,5
A3	Experiência com OO, na Linguagem e na Técnica de Desenvolvimento	1,0
A4	Capacidade do Líder de Projeto	0,5
A5	Motivação	1,0
A6	Requisitos estáveis	2,0
A7	Trabalhadores com dedicação parcial	-1,0
A8	Dificuldade da Linguagem de Programação	-1,0

3.4.2. Calculando o Esforço de Desenvolvimento

Uma vez identificados os pontos por caso de uso conforme a Eq. 7, podemos chegar ao esforço de desenvolvimento multiplicando o valor encontrado pela produtividade de desenvolvimento. Karner propôs, inicialmente, uma produtividade de 20 horas/homem por UCP. Schneider & Winters (2001) recomendam que os fatores técnicos e ambientais sejam considerados na hora de determinar a produtividade a ser utilizada. A proposta desses autores consiste em:

a) Identificar, dentre os primeiros seis fatores técnicos (de T1 a T6), quantos receberam nível de influência maior que três.
b) Somar a esse valor à quantidade de fatores ambientais entre A7 e A8 que receberam valor de influência menor que três.

O somatório identificado determina a quantidade sugerida de horas por ponto de caso de uso a ser adotada no projeto, sendo a média sugerida de:

a) 20 horas por ponto para um resultado de dois ou menos;
b) 28 horas por ponto caso o somatório resulte em três ou quatro;
c) 36 horas por ponto para valores maiores que quatro.

3.5. Volatilidade dos Requisitos e Impactos nos Projetos de Software

Segundo O'Neal & Carver (2001), é ingenuidade acreditar que todos os requisitos serão identificados nos estágios inicias de um projeto e que nenhuma mudança será necessária ao longo do tempo. Os autores salientam que nem todas as alterações serão iguais e terão o mesmo impacto. Alguma delas podem ser extremamente críticas para o sucesso do produto final, enquanto que outras são opcionais, podendo ser perfeitamente encaixadas em verões futuras do sistema se o cronograma do projeto o permitir. Existem, também, aquelas mudanças que causam um impacto quase que insignificante e acabam sendo absorvidas durante o projeto.

Segundo Kotonya e Sommerville (1998), alterações em requisitos ocorrem enquanto os eles estão sendo elucidados, analisados, validados e, até mesmo, após os sistemas ser colocado em operação. Essas alterações são inevitáveis e não são sinônimos de más práticas de engenharia de requisitos, mas sim resultados da combinação de diversos fatores como os exibidos na Tabela 7.

Tabela 7: Fatores que geram alterações em requisitos

Fator	Descrição
Erros, conflitos e insistências entre os requisitos	Depois que os requisitos estão analisados e implementados, erros ou inconsistências podem ser detectados.
Conhecimentos do usuário final sobre o sistema	Depois de implementado, o usuário final começa a ter um maior entendimento do requisito. Com isso o usuário passa ter mais certeza sobre o que ele realmente quer ou precisa.
Problemas técnicos, de cronograma ou de custos	Problemas podem ser detectados durante a implementação. Sendo que esses problemas podem fazer com que o requisito tenha um tamanho maior ou ser mais caro do que se estimou inicialmente.
Mudanças nas prioridades do cliente	As prioridades do cliente podem ser modificadas ao longo do tempo devido a mudanças na estratégia de negócios da empresa, concorrência, alterações de equipes, etc.
Mudanças de ambiente	O ambiente no qual o sistema operaria sofreu alterações, então devem ser feitas adaptações no sistema para que fique compatível com o ambiente.
Mudanças organizacionais	A organização que utilizaria o sistema pode ter passado por mudanças na sua estrutura ou nos seus processos, resultando em novos requisitos.

Fonte: Kotonya e Sommerville (1998), p. 116

Mesmo que as alterações em requisitos sejam uma realidade em projetos de desenvolvimento de sistemas, para Kotonya e Sommerville (1998) alguns requisitos são mais voláteis que outros. Os autores asseguraram que os requisitos mais estáveis são aqueles que estão intrinsecamente relacionados com a essência da organização, sendo que esses requisitos sofrem alterações de uma forma muito mais lenta que os requisitos voláteis. Os atores citam quatro diferentes tipos de volatilidade de requisitos.

a) **Requisitos mutáveis**: são os requisitos que sofrem alterações em função do ambiente no qual o sistema está operando. Por exemplo, requisitos que trabalham com arrecadação de impostos estão sujeitos a modificações quando leis tributárias são modificadas.

b) **Requisitos emergentes**: são aqueles que não são possíveis de determinar por completo nas etapas inicias, mas que surgem após serem desenvolvidos. Por exemplo, quando o usuário entra em contato com um requisito que foi desenvolvido ele passa a ter maior visibilidade da solução proposta e com isso cria novas formas de representar a informação que está sendo apresentada.
c) **Requisitos conseqüentes**: são requisitos baseados em suposições de como o sistema será usado. Porém, quando o sistema é colocado em uso, algumas dessas suposições estão erradas e isso acaba gerando demandas de modificações por parte dos usuários.
d) **Requisitos compatíveis**: são aqueles que dependem de outros equipamentos ou processos. Quando alterações ocorrem nesses equipamentos, esses requisitos são impactados.

3.5.1. Gerência de Mudanças

A gerência de mudanças está focada na definição de processos e padrões a serem dotados quando alterações são necessárias nos requisitos de um sistema. Essa gerência deve garantir que informações similares sejam coletadas para cada modificação proposta e que as decisões tomadas devem levar em consideração os custos e benefícios da alteração proposta. Kotonya e Sommerville (1998) afirmam que sem uma gerência de mudanças formal não é possível garantir que as decisões tomadas estejam alinhadas com os objetivos de negócio da organização.

O processo necessário para dar suporte à gerência de mudanças, segundo Kotonya e Sommerville (1998), pode ser pensado como um processo de três estágios como mostra a Figura 9.

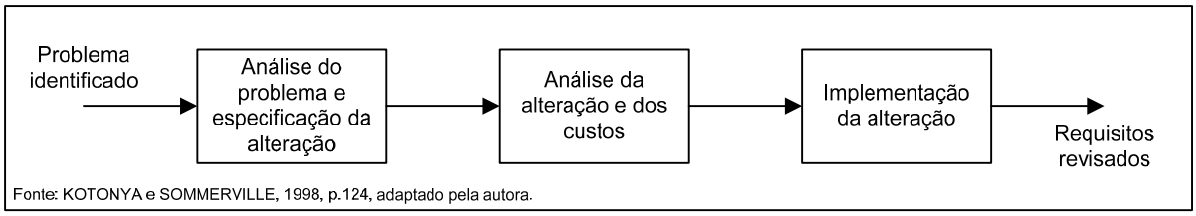


Figura 9: Estágios do processo de gerência de mudanças.

Segundo os autores a segunda etapa (análise da alteração e dos custos) é um processo mais genérico que pode ser dividido em seis atividades básicas.

a) Verificar se solicitação de mudança é válida, pois em muitos casos, o não entendimento de um usuário sugere alterações em requisitos que não são necessárias efetivamente.
b) Identificar quais requisitos serão diretamente afetados pela alteração.
c) Utilizar a rastreabilidade dos requisitos para verificar quais os demais requisitos que também podem ser impactados pela mudança.
d) Mostrar para o usuário as alterações que são necessárias para atender a solicitação de mudança.
e) Estimar os custos de realizar as mudanças, levando em consideração o esforço de desenvolvimento e o tempo necessário para implementar a mudança. A disponibilidade de recursos para viabilizar a implementação também deve ser pensada.

f) Negociar com os usuários os custos identificados no intuito de verificar se são aceitáveis para eles.

Conforme os autores, uma solicitação de mudança pode ser rejeitada em três momentos desse processo.

a) Quando a solicitação não é válida. Isso normalmente ocorre quando o usuário não possui um completo entendimento sobre o requisito e propõem uma mudança que não é necessária.
b) A mudança resulta em inúmeras outras alterações que não são aceitáveis para o usuário.
c) O custo de implementar tal solicitação torna a mudança inviável (muito caro ou muito demorado).

3.5.2. Rastreabilidade

Para Kotonya e Sommerville (1998) uma parte crítica do processo de gerência de mudanças é calcular o impacto que uma alteração pode causar no resto do sistema. Se a alteração é proposta enquanto os requisitos estão sendo elucidados, é necessário verificar o impacto dessa alteração nos demais requisitos. Se a alteração é proposta quando os requisitos já estão em desenvolvimento, ou até mesmo em operação, é necessário verificar o impacto tanto nos demais requisitos, quanto nos produtos gerados a partir dos requisitos impactados. Ou seja, é necessário avaliar o impacto causado em todos os produtos gerados ao longo da implementação do requisito.

Para calcular os impactos de uma modificação em um dado requisito, de forma consistente, é necessário (a) obter informações a respeito das dependências dele com outros requisitos e (b) obter informações sobre o relacionamento dele com produtos gerados ao longo do seu desenvolvimento. Esse conjunto de informações é usualmente chamado de rastreabilidade.

Segundo Tvete (1999), a rastreabilidade dos requisitos é importante para assegurar que o produto seja desenvolvido de acordo com as expectativas do usuário e que o cliente esteja recebendo aquilo que ele realmente solicitou. Segundo o autor, para garantir que tal rastreabilidade seja válida, todos os requisitos identificados devem ser coletados, mesmo aqueles que não serão implementados.

Kotonya e Sommerville (1998) propõem uma arquitetura para armazenamento de requisitos muito semelhante à arquitetura encontrada no artigo de Tvete (1999). Embora essas publicações tenham sido realizadas há quase uma década, a forma com a gestão de requisitos é abordada está adequadas às necessidades atuais.

Os modelos propostos pelos autores sugerem que sejam armazenadas as seguintes informações sobre todos os requisitos:

a) **Identificador único**: chave que irá identificar o requisito dentro da base de dados.
b) **Nome**: nome do requisito.
c) **Descrição ou Declaração**: descrição textual do requisito; pode ser em linguagem natural ou uma representação gráfica.
d) **Data de criação**: a data de quando o requisito foi originalmente inserido na base de dados.
e) **Data de alteração**: a data da última alteração que o requisito sofreu.

f) **Fontes**: referência a uma ou mais fonte que tenha originado o requisito. Essa informação pode contribuir na análise quando modificações no requisito são solicitadas.
g) **Situação**: variável que representa a situação do requisito, podendo assumir os valores *proposto*, *sob revisão*, *aceito* ou *rejeitado*. Requisitos rejeitados devem ser mantidos na base de dados. Dessa forma, se o requisito for proposto novamente, a análise da nova proposta pode ser simplificada por existir um histórico na base de dados.
h) **Dependentes**: é uma lista de requisitos que são dependentes deste requisito.
i) **Dependo de**: é uma lista de requisitos dos quais este requisito depende.

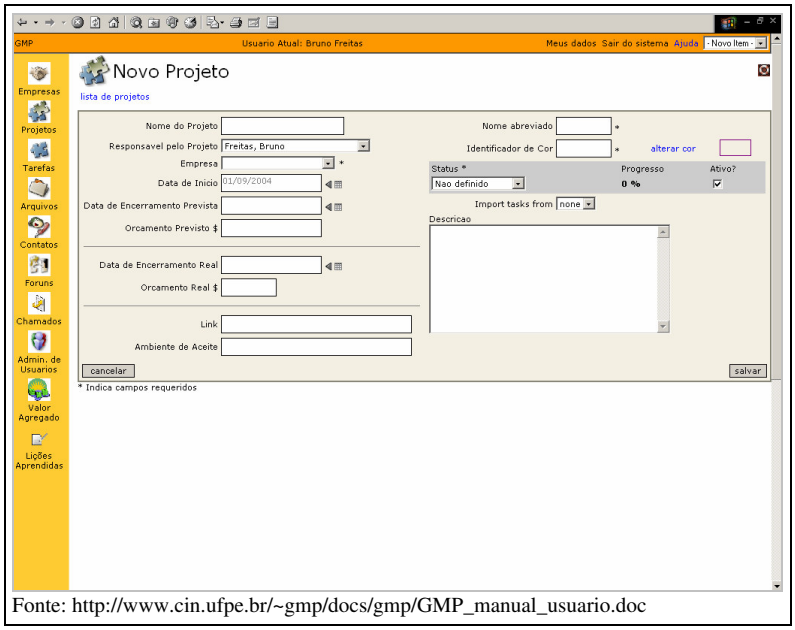
Além dessas informações, para garantir a rastreabilidade de forma completa, segundo Kotonya e Sommerville (1998), também é necessário inserir nessa arquitetura dados sobre os artefatos gerados ao longo do desenvolvimento e um requisito, como especificações, modelos de classe, diagramas, códigos fontes, etc. Entretanto, os autores afirmam que é trabalhoso manter essa rastreabilidade e que o esforço é apenas justificável para projetos grandes.

4. TRABALHOS RELACIONADOS

O objetivo deste capítulo é traçar um paralelo entre a ferramenta desenvolvida e algumas ferramentas similares. Foram selecionadas, através de pesquisa na *Internet*, ferramentas que propõe auxiliar o gerenciamento de projetos e/ou a gerência de requisitos, tendo em vista a identificação de características que essas ferramentas possuem e, assim, fazer uma comparação entre elas e o SARP

4.1. GMP

O Gerenciador de MultiProjetos (GPM) é uma ferramenta de gerenciamento de projetos de desenvolvimento de *software* que está sendo desenvolvido por um grupo de trabalho da Universidade Federal de Pernambuco desde 2003 (http://www.cin.ufpe.br/~gmp). O foco da ferramenta está em disponibilizar mecanismos de acompanhamento e comparação entre projetos, observando aspectos como análise da evolução dos gastos dos projetos ao longo do tempo.

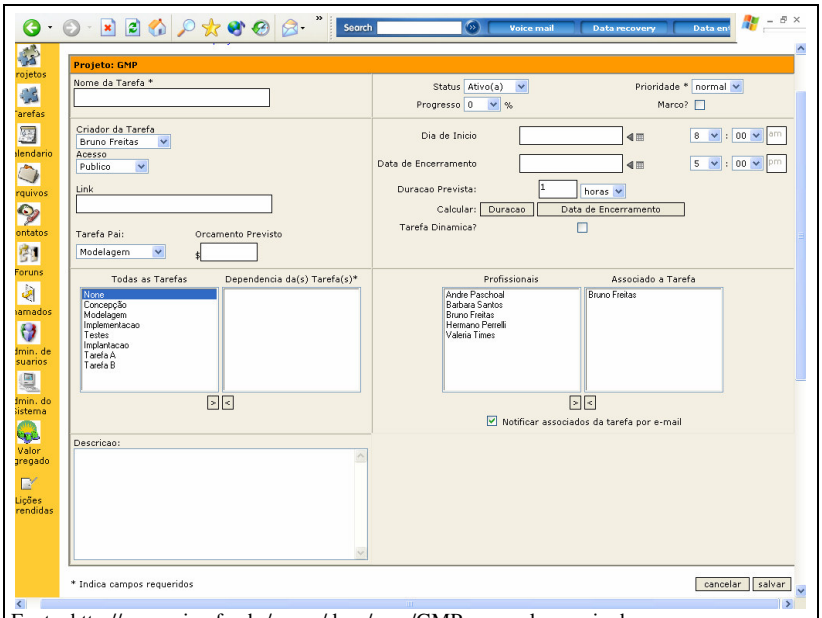


Fonte: http://www.cin.ufpe.br/~gmp/docs/gmp/GMP_manual_usuario.doc

Figura 10: Interface de visualização de Projetos do GMP.

a) **Controle de custo dos projetos**: permite o controle dos gastos dos projetos através da análise de índices, tais como, valor planejado, custo real, valor agregado, variância de custos, variância de cronograma, índice de performance de custos, estimativa de conclusão do projeto e índice de performance do cronograma.
b) **Controle do andamento dos projetos**: permite o acompanhamentos das atividades do projeto baseando-se nos percentuais de conclusão de cada requisito, quantidade de classes desenvolvidas, quantidade de linhas de código implementadas, etc. A ferramenta disponibilizar a visualização do andamento do projeto através de Gráficos de Gantt.

c) **Base de lições aprendidas**: permite que as lições aprendidas durante os projetos sejam armazenadas na base de dados da ferramenta e que possam ser consultados por outros usuários a qualquer tempo.



Fonte: http://www.cin.ufpe.br/~gmp/docs/gmp/GMP_manual_usuario.doc

Figura 11: Interface de manutenção de atividades do GPM.

4.2. Gemetrics*Web*

Gemetrics é uma ferramenta voltada ao gerenciamento de projetos que é produto de uma dissertação de mestrado (Vavassori, 2002). Em 2005 uma nova versão da ferramenta (Gemetrics*Web*) foi apresentada no XIX Simpósio Brasileiro de Engenharia de Software (Vavassori, 2005).

Fonte: http://www.sbbd-sbes2005.ufu.br/arquivos/Gemetrics*Web*.pdf

Figura 12: Interface de visualização de projetos do Gemetrics.

O foco dessa ferramenta está em auxiliar as atividades do gerente de projeto de *software* para cada etapa do processo de gerenciamento:

a) **Etapa de Planejamento**: permite o planejamento de recursos humanos, tempo e tarefas, sendo que estas últimas constituem atividades vinculadas às etapas de um processo de desenvolvimento. Durante a execução do projeto as alterações relevantes nessas atividades são gerenciadas.
b) **Etapa de Monitoramento e Controle**: a ferramenta permite o acompanhamento do projeto através de gráficos de Gantt.
c) **Etapa de Encerramento**: os autores da ferramenta afirmam que através da base histórica que se forma a partir do uso da ferramenta é possível avaliar quão adequado está o processo de desenvolvimento.

4.3. SIGERAR

Ferramenta voltada para o gerenciamento de requisitos que foi apresentada no WER06 - Workshop em Engenharia de Requisitos 2006 (Grande, 2006). A aplicação desenvolvida possui características de ferramentas comerciais existentes, como, por exemplo, controle de acesso e permissões, controle de versões, glossários, notificações, fórum de discussões, entre outros.

O diferencial da ferramenta, segundo os autores, está no tratamento que a ferramenta possui para controlar a rastreabilidade dos requisitos no processo de alteração. O SIGERAR não está focado apenas na elaboração da matriz de rastreabilidade, mas também no processo que envolve a gerência dos requisitos. A ferramenta exige que os responsáveis pelos requisitos impactados por alguma alteração analisem e atribuam valores de risco, importância, impacto, prioridade e custo de cada um dos requisitos envolvidos.

As principais funcionalidades dos SIGERAR são:

a) **Alocação dos usuários**: permite a alocação dos usuários do sistema nos projetos cadastrados na ferramenta.
b) **Cadastro requisitos e dependências**: permite criar uma base de dados de requisitos, informando as dependências existentes entre os requisitos, no intuito e obter a matriz de rastreabilidade dos requisitos;
c) **Controle de alteração dos requisitos**: o sistema utiliza a matriz de rastreabilidade gerada a partir do cadastro dos requisitos para identificar todos os requisitos dependentes (de forma recursiva) que podem sofrer algum tipo de impacto quando uma alteração se faz necessária. A partir daí os analistas responsáveis são notificados sobre as modificações e devem atribuir valores quantitativos aos impactos no intuito de estimar os custos das modificações.

4.4. XP3

O XP3 é uma ferramenta voltada ao gerenciamento de requisitos de *software* em projetos baseados na metodologia *Extreme Programming* (Dall'Oglio, 2006). Segundo essa metodologia os projetos devem ser divididos em ciclos de desenvolvimento e a cada iteração desses ciclos deve ser realizado um planejamento. A característica principal desta técnica é a

aplicação prática de um conjunto de princípios que garantem agilidade no processo de desenvolvimento. O foco do XP3 está em disponibilizar mecanismos de gerenciar os requisitos em projetos que seguem essa metodologia de desenvolvimento, promovendo o desenvolvimento evolutivo dos requisitos a cada ciclo de iteração do processo.

Dentre as funcionalidades disponibilizadas pela ferramenta, podemos destacar:

a) **Cadastro de colaboradores**: permite vincular todos os stakeholders do projeto, podendo ser o gerente, cliente ou desenvolvedores.
b) **Documentação suplementar**: permite a vinculação de diversos documentos que possam complementar um projeto, como modelos, gráficos, etc.
c) **Gerência de *Releases* e Iterações**: permite o controle sobre as diferentes *Releases* do projeto, definindo datas e início e término para cada uma delas, assim como a subdivisão em destes em Iterações.
d) **Visão Geral do Processo**: disponibiliza uma visão geral do fluxo de atividades do projeto na forma de fluxograma.
e) **Rastreabilidade**: oferece rastreabilidade entre os requisitos, entre requisitos e outros artefatos produzidos ao longo do projeto e, também, rastreabilidade "para trás" que permite identificar as origens de um requisito.
f) **Progresso do Projeto**: permite o acompanhamento das atividades do projeto.

4.5. Comparação entre as Ferramentas Pesquisadas e o SARP

No intuito de traçar um paralelo entre as ferramentas pesquisadas e a ferramenta proposta pelo presente trabalho, foram elencadas as características descritas na seqüência. A Tabela 8 apresenta uma matriz de comparação de acordo com as características identificadas.

a) **Base de dados de requisitos**: permite manter em uma base de dados centralizada os requisitos, de forma a controlar históricos de modificação ao longo do tempo.
b) **Matriz de rastreabilidade**: permite a criação de uma matriz de rastreabilidade dos requisitos de cada projeto. De forma a identificar os relacionamentos existentes entre os requisitos e os subprodutos.
c) **Base de dados dos casos de uso**: permite manter uma base de dados centralizada dos casos de uso do sistema que está sendo desenvolvido, assim como vinculá-los aos requisitos dos quais são originados.
d) **Controle de atividades**: permite o controle das atividades do projeto, vinculando essas atividades a requisitos ou ao desenvolvimento de casos de uso.
e) **Seqüenciamento das atividades**: permite que as atividades sejam seqüenciadas dentro do projeto de forma a estabelecer o cronograma.
f) **Impacto sobre as atividades quando alterações em requisitos ocorrem**: visualização de quais atividades do projeto precisam ser revistas e até mesmo replanejadas quando ocorre uma alteração em requisito ou em um subproduto.
g) **Controle de horas trabalhadas**: controle das horas trabalhadas em cada uma das atividades no intuito de gerar uma base de dados para extrair métricas e estimativas mais precisas.
h) **Definição de equipe**: permite a elaboração da equipe do projeto de acordo com os profissionais existentes.
i) **Controle de acesso por usuário**: disponibilizar perfis de acesso ao sistema de acordo com a responsabilidade de cada indivíduo dentro do projeto.

j) **Controle de custos dos projetos**: prover mecanismos de acompanhar a evolução dos custos dos projetos.
k) **Geração de Cenários**: permitir a manipulação na forma como os recursos humanos estão alocados dentro do projeto no intuito de identificar o melhor cenário de alocação de recursos.
l) **Estimativa de Esforço**: basear a definição da duração das atividades de desenvolvimento em algum modelo de estimativa de esforço.

Tabela 8: Matriz das principais funcionalidades das ferramentas estudadas.

	GPM	**Gemetrics**	**SIGERAR**	**XP3**	**SARP**
Base de dados de requisitos	Não	Não	Sim	Sim	Sim
Matriz de rastreabilidade	Não	Não	Sim	Sim	Sim
Base de dados dos casos de uso	Não	Não	Não	Não	Sim
Controle de atividades	Sim	Sim	Não	Sim	Sim
Seqüenciamento das atividades	Sim	Sim	Não	Sim	Sim
Impacto sobre as atividades quando alterações em requisitos ocorrem	Não	Não	Não	Não	Sim
Controle de horas trabalhadas	Sim	Sim	Sim	Não	Sim
Definição de equipe	Sim	Sim	Sim	Sim	Sim
Controle de acesso por usuário	Sim	Sim	Sim	Sim	Sim
Controle de custos dos projetos	Sim	Não	Não	Não	Não
Geração de Cenários	Não	Não	Não	Não	Sim
Estimativa de Esforço	Não	Não	Não	Não	Sim

Fonte: elaborado pela autora.

5. FERRAMENTA PROPOSTA

Com base na análise de ferramentas semelhantes e na revisão bibliográfica realizada sobre gerência e planejamento de projetos, estimativa de *software* e volatilidade de requisitos, foram identificados os requisitos necessários para a construção da ferramenta que visa auxiliar os gerentes de projetos quando deparados com alterações de requisitos ao longo de um projeto.

Com base nos requisitos identificados foi desenvolvida a ferramenta SARP: Sistema de Alocação de Recursos em Projetos. Os principais requisitos identificados foram:

a) Manter uma base de dados de requisitos para cada sistema desenvolvido. Permitindo o controle das modificações, o relacionamento entre os requisitos e entre os artefatos produzidos a partir deles ao longo do projeto.
b) Permitir que o gerente de projeto tenha controle e visibilidade das atividades executadas dentro do projeto.
c) Permitir que o gerente monte a equipe de projeto com base nos recursos humanos existente na empresa e de acordo com seus perfis técnicos.
d) Dispor interfaces para que os diversos participantes do projeto possam visualizar o andamento do projeto e suas responsabilidades, assim como informar o andamento das suas atividades.
e) Disponibilizar interfaces de cadastros básicos, tais como, tipos de atividades, papéis, usuários, etc.

5.1. Desenvolvimento

A ferramenta SARP foi desenvolvida utilizando exclusivamente *software* livre, sendo que a linguagem de programação adotada foi Java e, como banco de dados, o MySQL. A interface *desktop*, apesar de não apresentar as facilidades de uma aplicação *web*, foi necessária devido à utilização de componentes gráficos que foram adotados no intuito de prover uma interface mais amigável ao usuário.

Tabela 9: Tecnologias utilizadas no desenvolvimento do SARP.

Tecnologia	Descrição	Site
J2SE 1.5	Java 2 Standard Edition	http://java.sun.com/j2se/1.5.0/
Forms & Looks	API para desenvolvimento da camada de apresentação da ferramenta, focada na usabilidade da interface com o usuário. Facilita o desenvolvimento de aplicações Java/Desktop.	http://www.jgoodies.com/
jGraph	Componente que permite a criação e a manipulação de grafos.	http://www.jgraph.com/
MySql	Base de dados gratuita.	http://www.mysql.com/ ou http://www.mysqlbrasil.com.br/

Fonte: elaborado pela autora.

5.2. Arquitetura

Para o desenvolvimento da aplicação foi utilizada uma arquitetura MVC (*Model-View-Controller*), como mostra a Figura 13. Essa arquitetura, segundo Deitel (2002) é baseada em princípios de modularização de aplicações que consiste em dividir a aplicação em camadas, resultando em maior flexibilidade e reutilização. O padrão MVC define três camadas para a aplicação, são elas (GRIFFANTE, 2006):

a) ***Model***: contém o modelo que representa a estrutura de baixo nível do projeto, podendo ser, por exemplo, o modelo que implementa a camada de dados.
b) ***View***: contém todas as classes de interface com usuário de modo que esta somente requisite o processamento de eventos pelo *Controller*.
c) ***Controller***: contém as classes utilizadas para validar regras de negócio, processar os dados e a lógica de negócio.

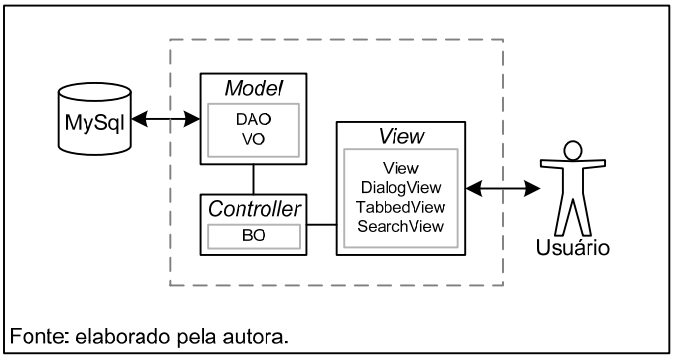


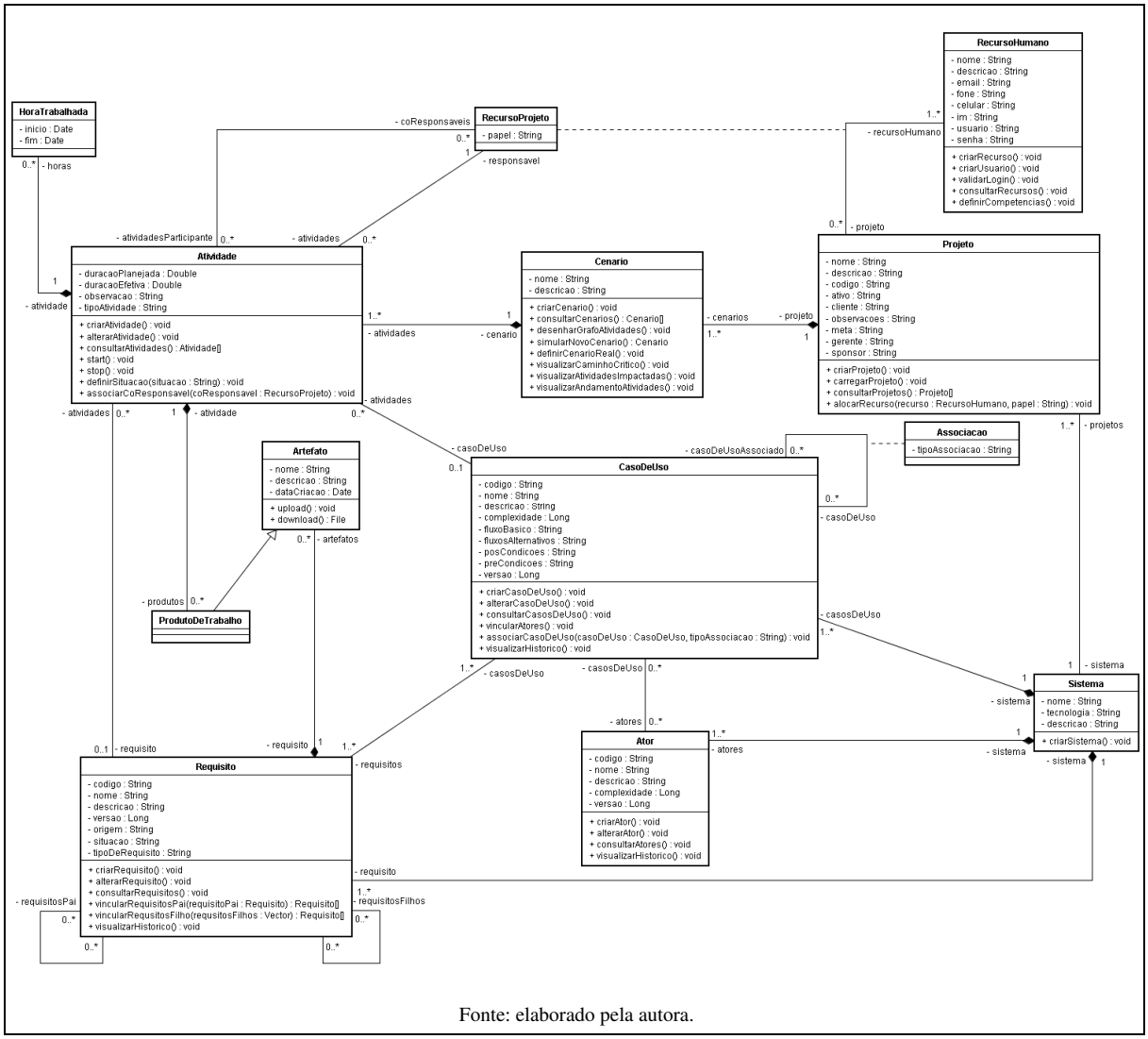
Figura 13: Arquitetura do SARP.

A seguir uma breve descrição dos padrões de classes utilizadas no desenvolvimento do SARP:

a) **Classes *Value Object* (VO)**: *design pattern* aplicados a objetos que trafegam entre as camadas da aplicação. São compostos exclusivamente de atributos privados e métodos *gets* e *sets* de acesso aos valores dos atributos.
b) **Classes *Data Access Object* (DAO)**: classes de persistência. São acessadas exclusivamente por classes da camada de controle.
c) **Classes *Business Object* (BO)**: classes responsáveis pelo encapsulamento das regras de negócio que atuam sobre um determinado tipo de objeto.
d) **Classes *View*, *DialogView* e *TabbedView***: utilizadas para criação dos formulários e edição de dados.
e) **Classes *SearchView***: utilizadas para telas de pesquisa de dados.

5.3. Diagrama de Classes

Segundo Martins (2006), o diagrama de classes permite identificar as classes, seus atributos e os relacionamentos existentes entre as classes. A Figura 14 apresenta o modelo de classes da ferramenta. Nesse modelo estão contidas apenas as classes referentes às entidades existentes no sistema. Não estão sendo exibidas as classes auxiliares da arquitetura.



Fonte: elaborado pela autora.

Figura 14: Diagrama de classes do SARP.

5.4. Diagrama de Caso de Uso

Segundo Martins (2006), o diagrama de caso de uso permite organizar o sistema em nichos de funcionalidades (pacotes) e visualizar a interações dos atores com o sistema em uma visão alto nível.

A Figura 15 exibe, de uma forma geral, o diagrama de casos de uso da ferramenta desenvolvida. Alguns casos de uso exibidos na figura possuem relacionamentos com outros casos de uso mais específicos que serão detalhados mais adiante. A partir desse caso de uso podemos identificar a interação de dois atores com o sistema, conforme descritos a seguir:

a) **Gerente**: é o gerente do projeto que utiliza a ferramenta para controlar e gerenciar um ou mais projetos.
b) **Desenvolvedor**: demais envolvidos com o projeto. O termo *desenvolvedor* é utilizado para centralizar todos os demais papéis existentes em projetos de desenvolvimento de *software* que não o próprio gerente de projetos.

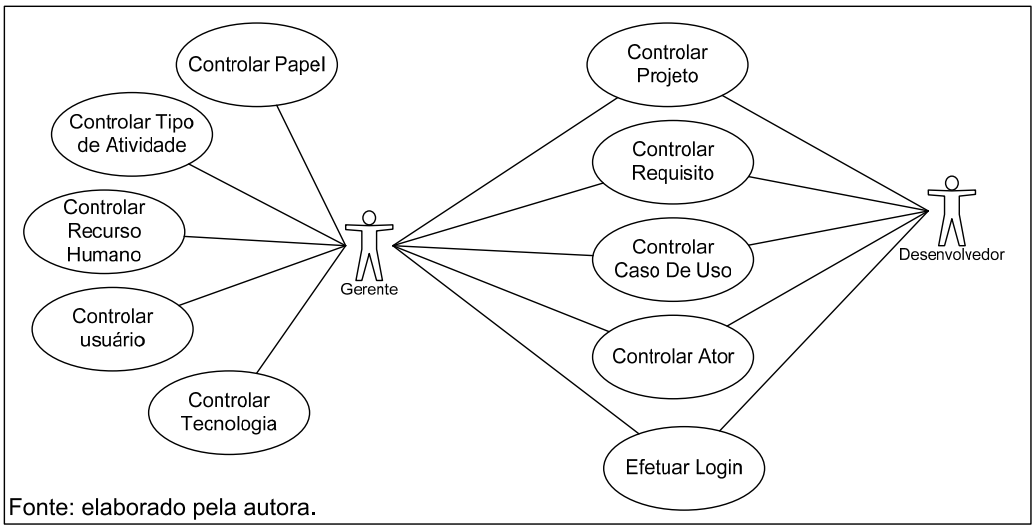


Figura 15: Diagrama geral dos casos de uso do SARP.

5.5. Módulos do SARP

No intuito de deixar a interface gráfica do SARP mais amigável, a ferramenta foi dividia em três módulos. Cada módulo pode ser a acessado através do menu principal da ferramenta, conforme descritos a seguir.

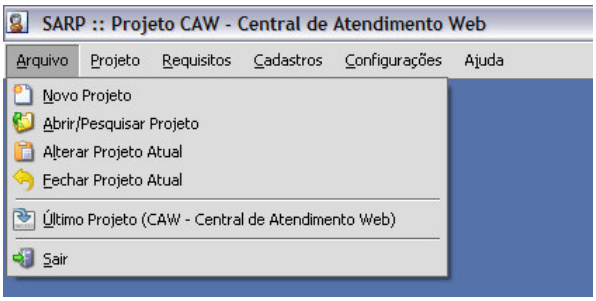


Figura 16: Menu *Arquivo*.

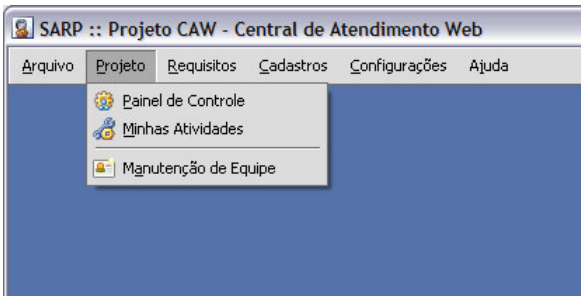


Figura 17: Menu de acesso ao *Módulo Projeto*.

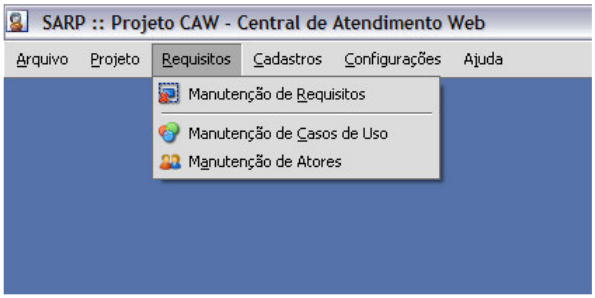


Figura 18: Menu de acesso ao *Módulo Requisitos*.

Figura 19: Menu de acesso ao *Módulo Cadastros*.

a) **Módulo Projeto**: esse módulo é acessado através de dois menus. Através das funcionalidades disponibilizadas pelo menu *Arquivo* (Figura 16) é possível manipular as informações cadastrais de um projeto. Através do menu *Projeto* (Figura 17) é possível ter acesso às interfaces *Painel de Controle*, *Minhas Atividades* e *Manutenção de Equipe*.

b) **Módulo Requisitos**: através do menu *Requisitos* (Figura 18) é possível acessar as interfaces de manutenção de *Requisitos*, *Casos de Uso* e *Atores*.
c) **Módulo Cadastros**: através do menu Cadastros (Figura 19) é possível acessar as interfaces de manutenção de *Recursos Humanos*, *Usuários*, *Tecnologias*, *Tipos de Atividades* e *Papéis*.

5.6. *Worflow* SARP

Para que o SARP possa servir como uma ferramenta de apoio aos gerentes de projeto e aos demais participantes é necessário seguir um *workflow*, conforme descrito na Figura 20.

Em um primeiro momento o gerente de projetos deve configurar o projeto através da interface de inclusão de projeto. Uma vez incluído o projeto, ele pode ser carregado para a sessão e com isso ficam disponíveis os menus *Projeto* e *Requisitos*. Uma vez configurado o projeto deverá ser feita a configuração da equipe que irá participar do projeto. Essa configuração pode ser modificada ao longo do projeto sempre que houver um replanejamento.

Em muitos projetos de desenvolvimento de *software* não se tem por completo, nas etapas iniciais, a definição de todos os requisitos que o sistema deverá possuir. Nesses casos são necessárias etapas (atividades) de elucidação desses requisitos. Durante essas etapas, requisitos são detalhados e novos requisitos podem emergir. Cabe ao gerente do projeto identificar as ações que serão tomadas sobre essas modificações de escopo do produto. Recapitulando o que foi visto por Kotonya e Sommerville (1998), algumas alterações podem ser absorvidas pelo projeto, mas outras, mais complexas, podem inviabilizar a execução do projeto dentro dos critérios estabelecidos.

Entretanto, mesmo não tendo domínio sobre todos os requisitos do sistema que será construído, o gerente do projeto já deve ter em mente as atividades que serão necessárias para dar continuidade ao projeto. Dessa forma, é possível criar cenários de alocação de equipe para essas atividades iniciais, afinal recursos estarão alocados para esclarecer os requisitos. É muito interessante, para a própria organização executora do projeto, manter dados históricos sobre tempos despendidos em todas as etapas de desenvolvimento. Esses dados podem servir para aprimorar modelos de estimativa e servir de base para novos projetos.

Na medida em que o projeto vai sendo detalhados (requisitos especificados), e de acordo com o modelo do ciclo de vida do projeto, as atividades de desenvolvimento podem ser inseridas no cenário principal de atividades do projeto.

Tendo essas atividades inseridas, o gerente do projeto (e toda a equipe) começa a ter visibilidade do andamento do projeto. Isso se deve ao fato de que é feito um controle das horas planejadas *versus* horas realizadas de todas as atividades controladas pelo SARP. Por exemplo, digamos que dentro de um determinado projeto exista um requisito chamado *Controle de Estoque* e que a atividades de elucidação desse requisito foi atribuída ao analista João com um tempo estimado de 40 horas. A partir do momento em que essas 40 horas forem ultrapassadas, será possível, através do *Painel de Controle*, visualizar que essa atividade está durando mais tempo que o planejado e que ela poderá impactar em outras atividades planejadas. Cabe ao gerente de projeto tomar uma ação corretiva.

Com o intuito de auxiliar o gerente a tomar essa ação corretiva é que existe a possibilidade de geração de cenários no SARP. Continuando com o exemplo anterior, digamos que a atividade seguinte a elucidação do requisito *Controle de Estoque* (do analista

João) não possua relação direta com requisito em questão. Dessa forma, essa segunda atividade poderia muito bem ser executada por outro analista. Com isso, o gerente estaria sendo pró-ativo frente a um problema dentro do seu projeto, buscando minimizar os impactos no cronograma causados por atrasos ou outras complicações. Para ajudar o gerente a definir qual o melhor cenário de alocação dos recursos, o SARP disponibiliza o cálculo do caminho crítico sobre as atividades configuradas em um cenário.

A Figura 20 deixa claro que as atividades de controle do projeto são cíclicas podendo ser repetidas quantas vezes forem necessárias, até que o projeto seja encerrado.

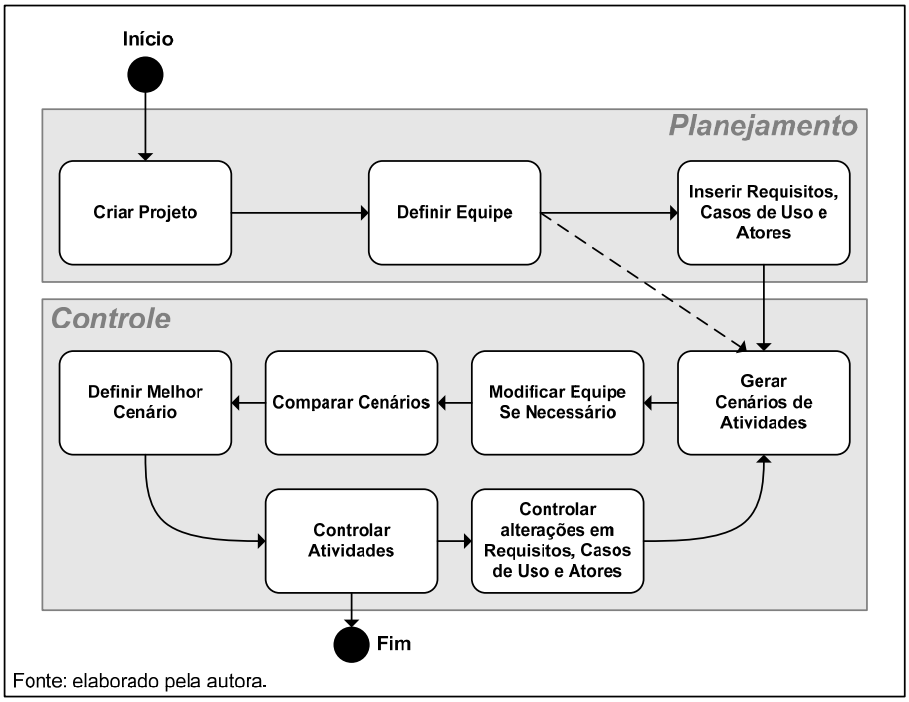


Figura 20: *Workflow* das atividades que devem ser executadas através do SARP.

5.7. Módulo *Projeto*

Através do módulo *Projeto* a equipe envolvida pode controlar as atividades realizadas. O gerente, através da interface *Painel de Controle*, pode gerenciar todas as atividades do projeto. Já os demais participantes podem controlar as suas respectivas atividades através da interface *Minhas Atividades*. Essas duas interfaces são muito semelhantes, porém a interface *Minhas Atividades* é mais restrita, permitindo apenas o controle do andamento da execução das atividades do usuário que está *logado* no SARP, enquanto que a interface *Painel de Controle* fornece um número maior de funcionalidades ao gerente.

As principais funcionalidades oferecidas através da interface de *Painel de Controle* são:

a) **Geração de cenários de alocação de recursos nas atividades**: permite a geração de diferentes cenários de alocação de recursos às atividades do projeto no intuito de auxiliar o gerente de projetos na tomada de decisão em relação à melhor forma de conduzir o projeto.
b) **Inclusão, alteração, exclusão de atividades**: permite que o gerente defina as atividades que serão executadas ao longo do projeto.

c) **Seqüenciamento de atividades**: permite que o gerente defina a ordem de prioridade em que as atividades devem ocorrer ao longo do projeto.
d) **Definição de cenário real do projeto**: permite que o gerente escolha, dentre os cenários de alocação de recursos que ele criou, qual será o cenário real do projeto.
e) **Visualização do andamento das atividades**: permite que o gerente acompanhe o andamento das atividades do projeto (no cenário real).
f) **Visualização do impacto de modificações em requisitos e casos de uso sobre as atividades do projeto**: permite o gerente verificar quais atividades do projeto são impactadas quando um requisito e/ou um caso de uso é modificado.
g) **Visualização do caminho crítico das atividades**: permite que o gerente visualize o caminho crítico das atividades do projeto.

A Figura 21 exibe todos os casos de uso que podem ser executados pelo gerente no intuito de controlar um determinado projeto, enquanto que a Figura 22 exibe as operações que um desenvolvedor pode executar. Os principais casos de usos existentes nessas figuras são detalhados na seqüência.

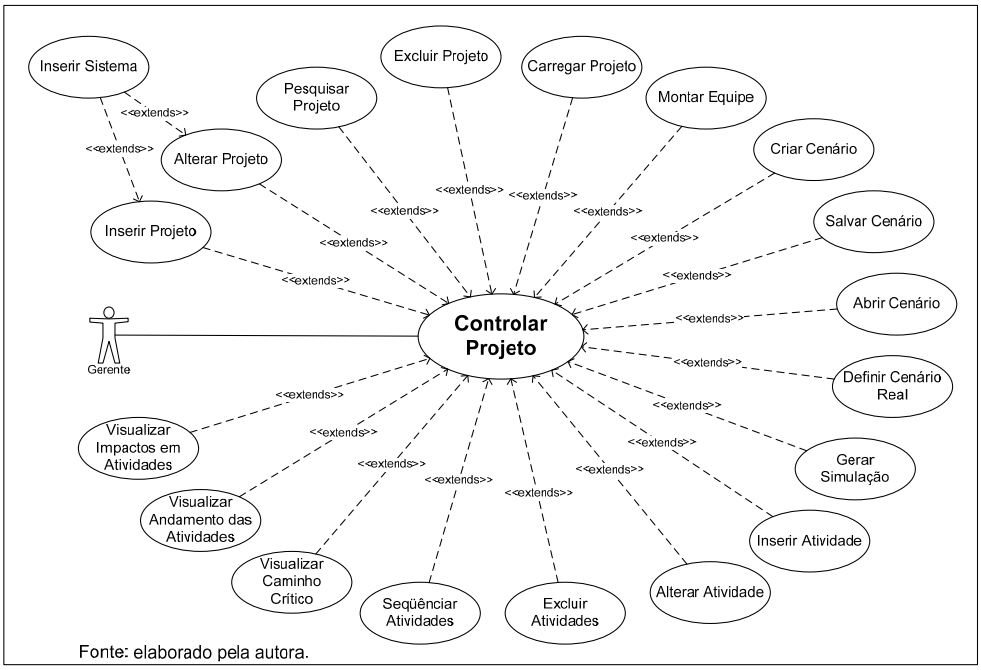


Figura 21: Caso de uso *Controlar Projeto* do ponto de vista do gerente.

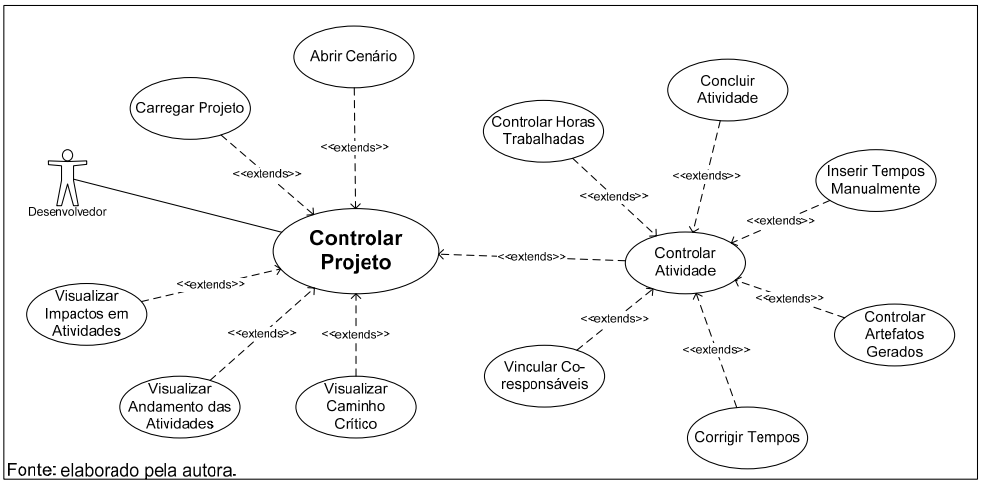


Figura 22: Caso de uso *Controlar Projeto* do ponto de vista do desenvolvedor.

5.7.1. Caso de Uso: *Inserir Projeto*

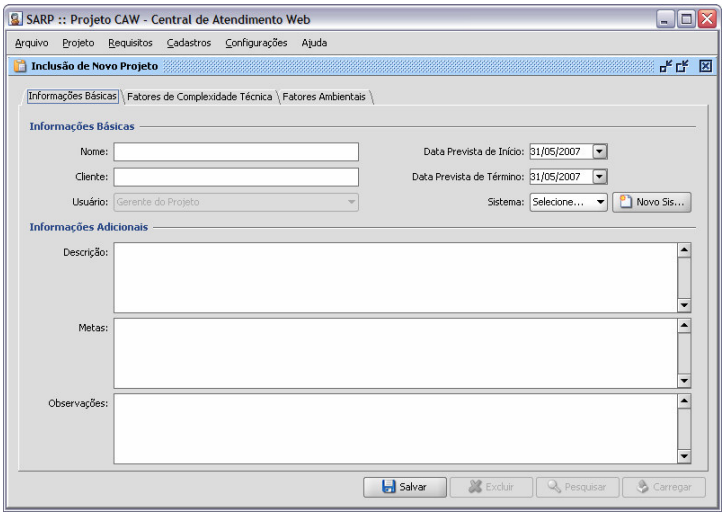


Figura 23: Interface de *Manutenção de Projeto*, aba *Informações Básicas*.

Atores

- Gerente

Objetivos

- Permitir a inclusão de um novo projeto na base de dados do SARP.

Pré-Condições

- O usuário deverá estar *logado* no sistema.

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Acessa o item *Novo Projeto* no menu *Arquivo*.	
	2) Exibe interface de *Inclusão de Novo Projeto* (Figura 23).
3) Informa os dados do projeto. 4) Informa o sistema vinculado ao projeto. 5) Informa os fatores de complexidade técnica e fatores ambientais. 6) Clica no botão *Salvar*.	
	7) Sistema avalia se os campos obrigatórios foram preenchidos. 8) Sistema persiste os dados do projeto na base de dados do SARP.

Fluxos Alternativos

- Em (4), caso o projeto esteja vinculado a um sistema novo (não cadastrado no SARP), o usuário pode inserir um novo sistema (caso de uso *Inserir Sistema*).
- Em (7), caso algum campo obrigatório não tenha sido preenchido, deve ser exibida mensagem de alerta informando qual campo não foi preenchido.

Pós-Condições

- O projeto deve estar inserido na base de dados do SARP.

Observações

- As informações *Fatores de Complexidade Técnica* e *Fatores Ambientais* não são obrigatórias. Porém, elas são necessárias para o cálculo do esforço de desenvolvimento dos casos de uso. Se esses dados não forem totalmente preenchidos,

o SARP será incapaz de calcular o esforço de desenvolvimento com base na produtividade dos desenvolvedores durante a inclusão de uma atividade.

5.7.2. Caso de Uso: *Abrir/Pesquisar Projeto*

Atores

- Gerente
- Desenvolvedor

Objetivos

- Permitir a busca por um determinado projeto na base de dados do SARP.

Pré-Condições

- O usuário deverá estar *logado* no sistema.

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Acessa o item *Abrir/Pesquisar Projeto* no menu *Arquivo*.	
	2) Exibe interface de *Pesquisa de Projeto* (Figura 24).
3) Informa o nome do projeto ou cliente (ambas opcionais). 4) Clica no botão *Pesquisar*.	
	5) Busca na base de dados os projetos que coincidam com os filtros de pesquisa aplicados. 6) Exibe os projetos encontrados.
7) Seleciona um dos projetos exibidos pelo sistema.	
	8) Carrega a interface *Manutenção de Projeto* com os dados do projeto selecionado.

Fluxos Alternativos

- Depois de (8), o caso de uso *Alterar Projeto*, *Excluir Projeto* e *Carregar Projeto* podem ser executados se o usuário assim quiser.

Pós-Condições

- Não se aplicam.

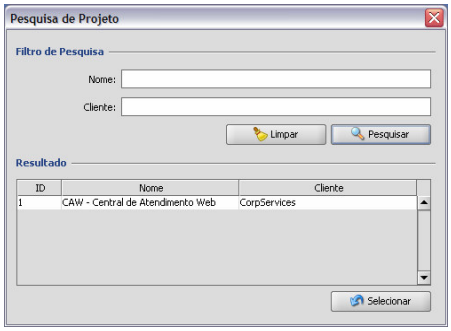


Figura 24: Interface de *Pesquisa de Projeto*.

5.7.3. Caso de Uso: *Carregar Projeto*

Atores
- Gerente
- Desenvolvedor

Objetivos
- Permitir que um projeto seja carregado para a sessão da aplicação para que possa ser manipulado.

Pré-Condições
- O usuário deverá estar *logado* no sistema.

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Acessa o item *Último Projeto* existente menu *Arquivo* ou clica no botão *Carregar* existente na interface de *Manutenção de Projeto*.	
	2) Carrega dados do projeto para a sessão da aplicação. 3) Habilita os itens *Alterar Projeto Atual* e *Fechar Projeto Atual* do menu *Arquivo*. 4) Habilita os menus *Projeto* e *Requisitos*. 5) Se existirem cenários reais configurados habilita o item *Minhas Atividades* do menu *Projeto*.

Fluxos Alternativos
- Não possui fluxo alternativo.

Pós-Condições
- O projeto deve estar carregado na sessão da aplicação.

5.7.4. Caso de Uso: *Alterar Projeto*

Atores
- Gerente

Objetivos
- Permitir que os dados cadastrais de um projeto sejam modificados e persistidos.

Pré-Condições
- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*) ou o usuário deve ter efetuado uma pesquisa de projeto (Caso de Uso *Abrir/Pesquisar Projeto*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Se um projeto está carregado na sessão da aplicação, acessa o item *Alterar Projeto Atual* no menu *Arquivo*; ou se foi executado o Caso de Uso A*brir/Pesquisar Projeto.*	
	2) Exibe interface de *Manutenção de Projeto* (Figura 23) com os dados populados de acordo com o projeto que está carregado na sessão da aplicação.

Ação do Ator	Comportamento do Sistema
3) Faz as modificações necessárias nos dados cadastrais do projeto. 4) Clica no botão *Salvar*.	
	5) Sistema avalia se os campos obrigatórios foram preenchidos. 6) Sistema persiste os dados do projeto na base de dados do SARP.

Fluxos Alternativos

- Em (5), caso algum campo obrigatório não tenha sido preenchido, deve ser exibida mensagem de alerta informando qual campo não foi preenchido.

Pós-Condições

- O projeto deve estar alterado na base de dados do SARP.

5.7.5. Caso de Uso: *Montar Equipe*

Atores

- Gerente

Objetivos

- Definir, dentre os recursos humanos disponíveis, quais que estarão alocados ao projeto carregado na sessão da aplicação, assim como seus respectivos papéis dentro do projeto.

Pré-Condições

- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Acessa o item *Manutenção de Equipe* no menu *Projeto*.	
	2) Carrega a interface *Manutenção de Equipe* (Figura 25) com os dados de equipe já cadastrados para o projeto que está carregado na sessão.
3) Informa *Nome*, *Tecnologia* e ou *Experiência* e clica no botão *Pesquisar*.	
	4) Carrega *grid* de recursos disponíveis de acordo com os critérios de pesquisa.
5) Seleciona um ou mais recursos disponíveis consultados. 6) Informa o papel que esses recursos irão ter no projeto. 7) Clica no botão *Vincular*.	
	8) Verifica se os recursos já não estão associados ao projeto com o papel informado. 9) Vincula ao projeto os recursos humanos informados e associa a eles o papel definido. 10) Persiste esse vínculo na base de dados do SARP.

Fluxos Alternativos

- Em (8), se for constatado que um dos recursos selecionados já estiver vinculado ao projeto que está carregado, não permite criar novamente o vínculo.

Pós-Condições

- A equipe do projeto estará configurada e persistida na base de dados, estando disponível para os outros módulos do SARP.

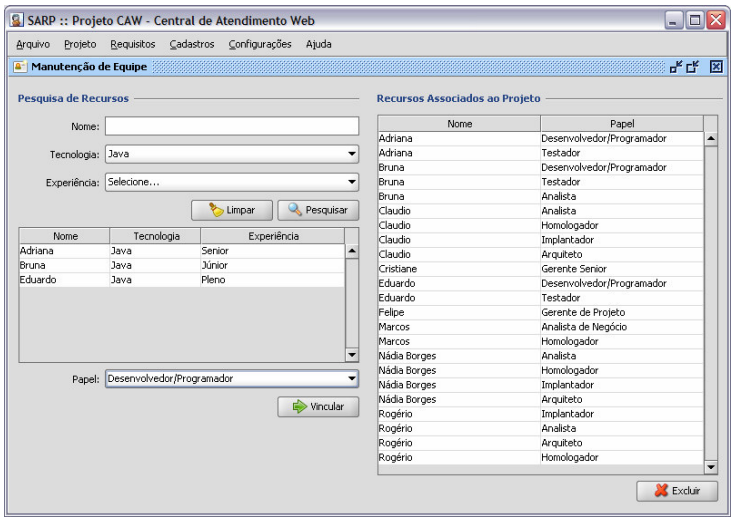


Figura 25: Interface de *Manutenção de Equipe*.

5.7.6. Interface *Painel de Controle*

Através da interface *Painel de Controle,* que pode ser acessada pelo menu *Projeto*, é possível ter acesso a um conjunto de funcionalidade que, utilizadas de forma conjunta, permite ao gerente planejar e controlar o projeto que está carregado na sessão da aplicação.

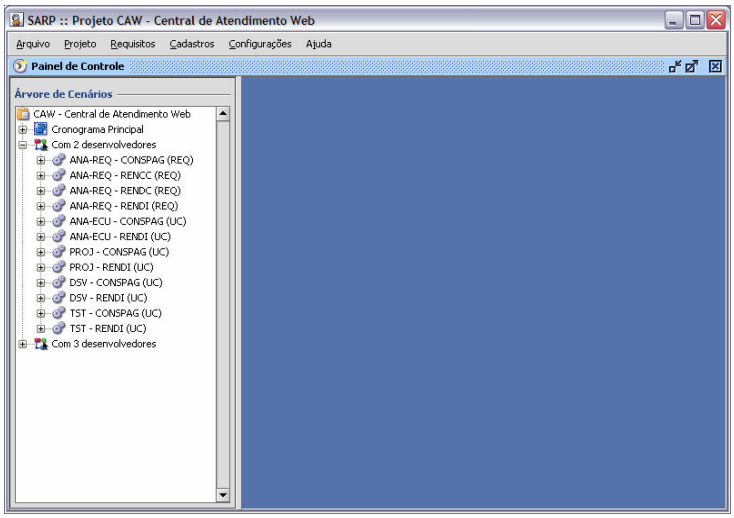


Figura 26: Interface *Painel de Controle*.

A Figura 26 exibe a tela principal da interface de *Painel de Controle*, a Figura 27 descreve os itens exibidos na *Árvore de Cenários* e a Figura 28 descreve as opções existentes na *pop-up* que abre sobre a árvore quando clicamos com o botão direito do mouse sobre ela.

A *Árvore de Cenários*, detalhada na Figura 27, exibe todos os cenários existentes para o projeto.

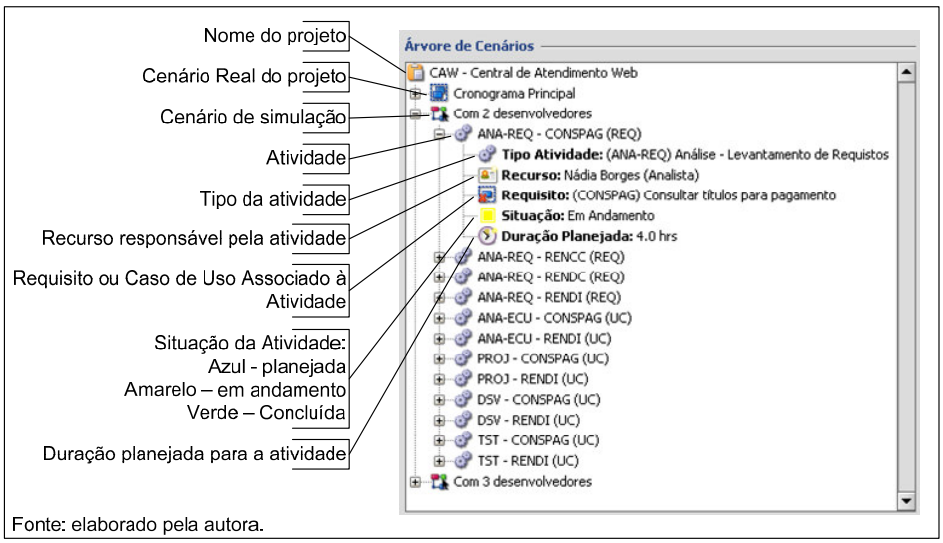


Figura 27: Painel que exibe a *Árvore de Cenários* na interface *Painel de Controle.*

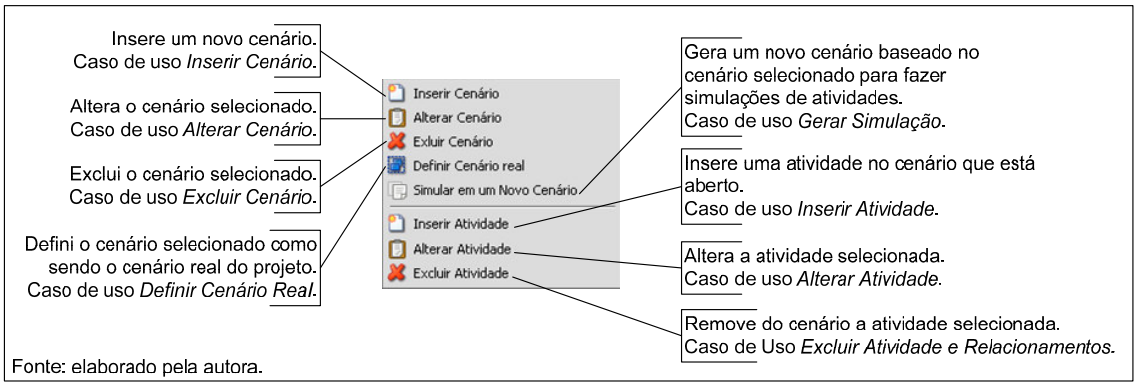


Figura 28: Descrição das opções da *pop-up* da *Árvore de Cenários*.

5.7.6.1. Caso de Uso: *Criar Cenários*

Atores

- Gerente

Objetivos

- Permitir que o gerente de projeto possa criar e configurar cenários de alocação de equipe dentro do projeto que está carregado na sessão da aplicação.

Pré-Condições

- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).
- A interface *Painel de Controle* deve estar aberta.

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Clica com o botão direito do mouse sobre o nome do projeto exibido no painel *Árvore de Cenários.*	
	2) Abre *pop-up* de opções (Figura 28).
3) Clica na opção *Novo Cenário*.	
	4) Exibe interface *Manutenção de Cenário* (Figura 29).
5) Preenche os dados do cenário. 6) Clica no botão *Salvar*.	
	7) Sistema avalia se os campos obrigatórios foram preenchidos. 8) Sistema persiste os dados do cenário na base de dados do SARP. 9) Abre janela do novo cenário inserido, com zero atividades.

Fluxos Alternativos

- Em (7), caso algum campo obrigatório não tenha sido preenchido, deve ser exibida mensagem de alerta informando qual campo não foi preenchido.

Pós-Condições

- O novo cenário criado estará inserido na base de dados com nenhuma atividade.

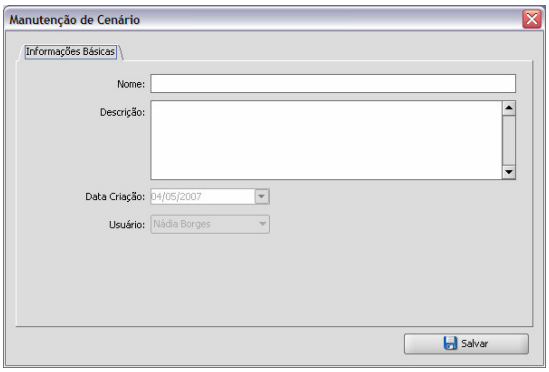


Figura 29: Interface de *Manutenção de Cenário*.

5.7.6.2. Caso de Uso: *Abrir Cenários*

Atores

- Gerente

Objetivos

- Permitir que o gerente de projeto possa abrir um cenário de atividades armazenado.

Pré-Condições

- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).
- A interface *Painel de Controle* deve estar aberta.

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Duplo clique sobre o nome do cenário na *Árvore de Cenários* (Figura 27).	
	2) Abre a janela do cenário selecionado. 3) Exibe as atividades, caso existam atividades para o cenário selecionado (Figura 30). 4) Deixa disponível a barra de ferramentas (Figura 31).

Fluxos Alternativos

- Não existe.

Pós-Condições

- A janela do cenário e a barra de ferramentas deverão estar visíveis.
- As atividades do cenário deverão estar visíveis. As cores de cada respeitam as seguintes regras:
  - **Azul**: atividade planejada.
  - **Amarelo**: atividade em andamento.
  - **Verde**: atividade concluída.

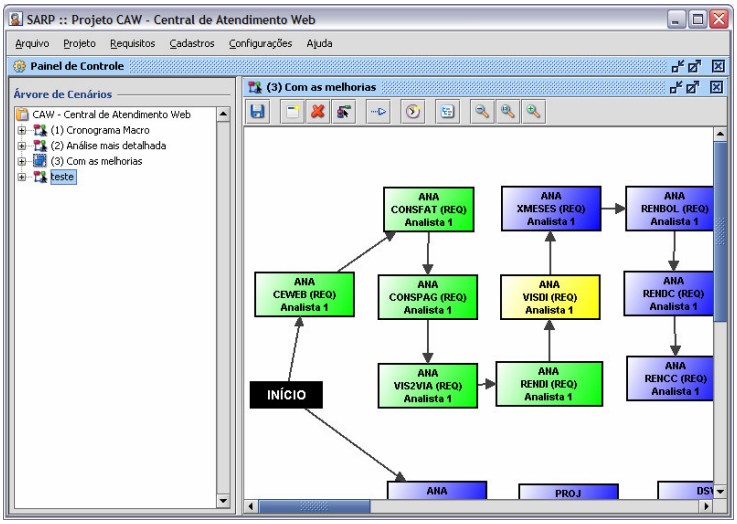
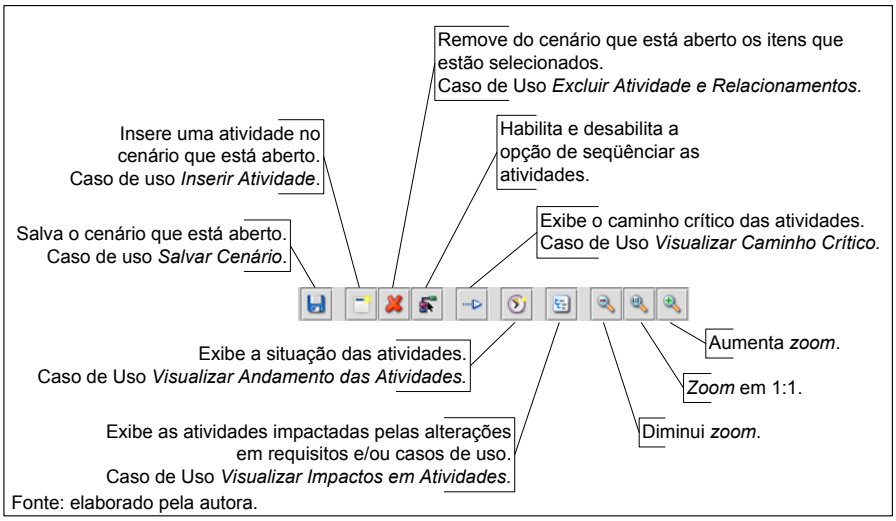


Figura 30: Interface *Painel de Controle* com um cenário aberto.



Fonte: elaborado pela autora.

Figura 31: Descrição dos botões da barra de ferramentas da janela de cenários.

5.7.6.3. Caso de Uso: *Inserir Atividade*

Atores

- Gerente

Objetivos

- Permitir e a inclusão de uma atividade dentro de um cenário de alocação de recursos.

Pré-Condições

- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).
- A interface *Painel de Controle* deve estar aberta.
- O cenário no qual se deseja inserir a atividade deve estar aberto (Caso de Uso *Abrir Cenários*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Clica no ícone *Nova Atividade* existente na barra de ferramentas (Figura 31) da janela do cenário ou no mesmo ícone existente na *pop-up* da *Árvore de Cenários* (Figura 28).	
	2) Exibe a interface de *Manutenção e Atividades* (Figura 32). 3) Popula combo de *Tipo de Atividades*. 4) Popula combo de todos os requisitos cadastrados para o projeto. 5) Popula combo de todos os casos de cadastrados para o projeto.
6) Informa o *Tipo de Atividade.*	
	7) Popula combo de *Recursos* conforme o *Tipo de Atividade* selecionado. Somente os recursos que possuem os papéis executores da atividade selecionada serão exibidos na combo.
8) Seleciona o recurso responsável pela atividade. 9) Informa requisito ou caso de uso associado à atividade.	
	10) Popula as combos de versões dos requisitos ou casos de uso, dependendo de qual item for selecionado.
11) Informa a duração da atividade. 12) Clica no botão *Salvar.*	
	13) A atividade é salva em memória e só será efetivamente persistida no banco de dados quando o cenário no qual ela está inserida for salvo. 14) Interface de *Manutenção de Atividade* é fechada. 15) Atividade é exibida no grafo de atividades.

Fluxos Alternativos

- O passo (1) pode ser substituído por:
  - (1.A) **Ação do Ator**: Clica com o botão direito do mouse sobre o diagrama de atividades do cenário selecionado.
  - (1.B) **Comportamento do Sistema**: Exibe *pop-up* de opções de ações sobre o cenário.
  - (1.C) **Ação do Ator:** Clica na opção *Inserir Atividade*.

- Em (8), se o tipo de atividade informado é *Desenvolvimento*, ao selecionar um recurso, popula o campo *Produtividade* com a produtividade do recurso selecionado dentro da tecnologia do projeto.
- Em (9), se o tipo de atividade informado é *Desenvolvimento*, ao selecionar um caso de uso o sistema irá calcular os Pontos por Caso de Uso (PCU) de acordo com a complexidade do caso de uso selecionado. Identificado o valor do PCU é estimado o tempo total da atividade de acordo com a produtividade do desenvolvedor selecionado. Essa informação é exibida no campo *Duração Calculada*.

Pós-Condições

- Atividade será exibida no grafo das atividades do cenário.

Observações

- Em (11), o gerente que está configurando a atividade tem a liberdade de definir a duração da atividade, independente do valor estimado pelo sistema. O SARP apenas auxilia o gerente na definição do tempo da atividade, não obriga que esse tempo estimado seja realmente utilizado.
- Em (15), não existe a persistência dos dados da atividade do banco de dados do SARP. As informações de todas as atividades de um cenário só são efetivamente gravadas no banco quando o cenário é salvo (caso de uso *Salvar Cenário*).
- A Figura 33 mostra a interface que permite associar co-responsáveis às atividades. São pessoas que de alguma forma têm participação na atividade mais que não são os responsáveis diretos por ela (caso de uso *Vincular Co-responsáveis*).
- Ao inserir uma atividade ela fica automaticamente com a situação igual à *Planejada* e é exibida na cor Azul.

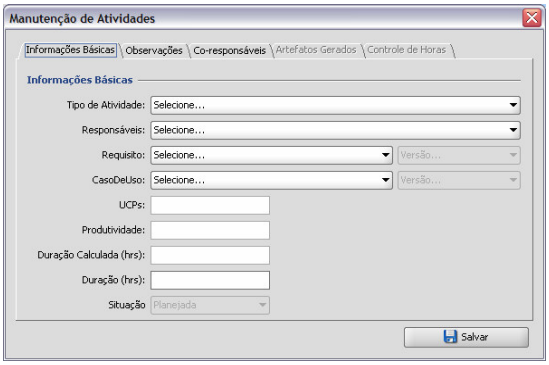


Figura 32: Interface de *Manutenção de Atividades*, aba *Informações Básicas*.

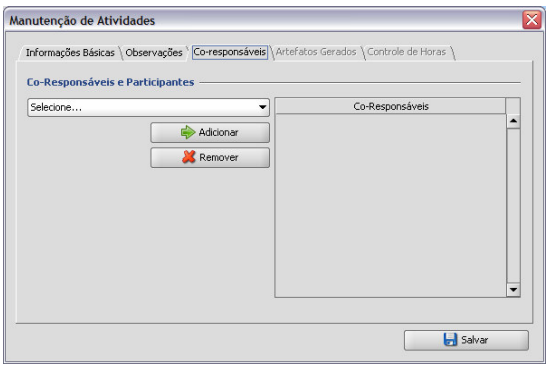


Figura 33: Interface de *Manutenção de Atividades*, aba *Co-Responsáveis*.

5.7.6.4. Caso de Uso: *Alterar Atividade*

Atores

- Gerente
- Desenvolvedor

Objetivos

- Permitir a alteração de uma atividade dentro de um cenário de alocação de recursos.

Pré-Condições

- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).
- A interface *Painel de Controle* deve estar aberta.
- O cenário no qual se deseja inserir a atividade deve estar aberto (Caso de Uso *Abrir Cenários*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Clica com o botão direito do mouse sobre a atividade que deseja alterar.	
	2) Exibe *pop-up* de opções de ações sobre a atividade (Figura 34).
3) Clica na opção *Alterar Atividade*.	
	4) Exibe a interface *Manutenção de Atividades* com os dados da atividade selecionada populados.

Fluxos Alternativos

- A partir de (4) o fluxo segue os mesmos passos do caso de uso *Inserir Atividade*.

Pós-Condições

- Atividade será exibida no grafo das atividades do cenário com os dados atualizados.

Observações

- A alteração da atividade não é persistida do banco de dados do SARP após o encerramento do caso de uso. As informações de todas as atividades de um cenário só são efetivamente gravadas no banco quando o cenário é salvo (caso de uso *Salvar Cenário*).

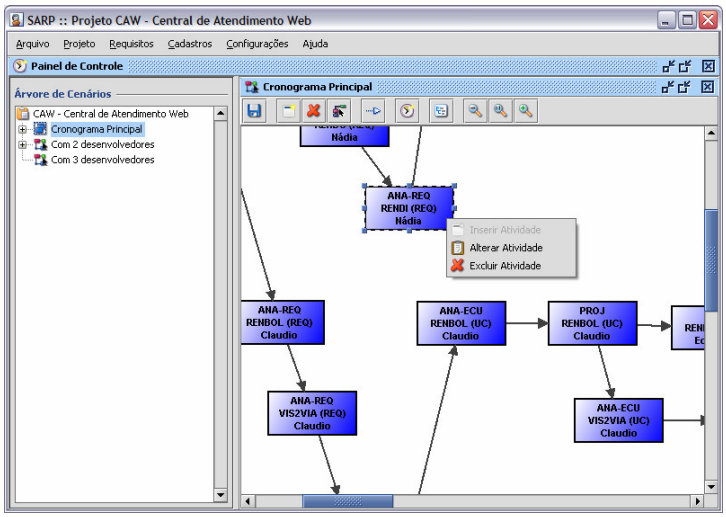


Figura 34: Interface *Painel de Controle*, *pop-up* de opções sobre uma atividade.

5.7.6.5. Caso de Uso: *Excluir Atividade*

Atores
- Gerente

Objetivos
- Permitir que as atividades e os relacionamentos existentes entre elas possam ser removidos.

Pré-Condições
- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).
- A interface *Painel de Controle* deve estar aberta.
- O cenário do qual se deseja remover atividades e/ou relacionamento deve estar aberto (Caso de Uso *Abrir Cenários*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Seleciona as atividades e os relacionamentos que deseja remover. 2) Clica no ícone *Excluir Atividade* existente na barra de ferramentas da janela do cenário (Figura 31).	
	3) Remove do cenário de atividades as atividades e relacionamentos selecionados.

Fluxos Alternativos
- O fluxo (2) pode ser substituído por clicar sobre o botão *Excluir Atividade* existente na *pop-up* de opções sobre uma atividade (Figura 34).

Pós-Condições
- As atividades e os relacionamentos selecionados devem ser removidos do cenário.

Observações
- Em (3) não existe a remoção dos dados da atividade e dos relacionamentos do banco de dados do SARP. As informações de todas as atividades de um cenário só são efetivamente gravadas no banco quando o cenário é salvo (caso de uso *Salvar Cenário*).

5.7.6.6. Caso de Uso: *Seqüênciar Atividades*

Atores
- Gerente

Objetivos
- Permite definir as prioridades (seqüenciamento) das atividades de acordo com a ordem que elas devem ser executadas.

Pré-Condições
- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).
- A interface *Painel de Controle* deve estar aberta.

- O cenário no qual se deseja definir o seqüenciamento de atividades deve estar aberto (Caso de Uso *Abrir Cenários*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Clica no botão *Conexão entre as atividades* na barra de ferramentas na janela do cenário (Figura 31).	
	2) Permite criar ligações entre as atividades através de setas.
3) Conecta as atividades.	
	4) Insere seta definindo o seqüenciamento.

Fluxos Alternativos

- Não tem.

Pós-Condições

- O seqüenciamento estará inserido no cenário das atividades.

Observações

- Em (4) não existe a persistência no banco de dados do SARP das informações a respeito dos seqüenciamentos definidos. Essas informações só são efetivamente gravadas no banco quando o cenário é salvo (caso de uso *Salvar Cenário*).

5.7.6.7. Caso de Uso: *Salvar Cenário*

Atores

- Gerente

Objetivos

- Permite salvar as atividades inserias ou modificas em um cenário, assim como o seqüenciamento dessas atividades

Pré-Condições

- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).
- A interface *Painel de Controle* deve estar aberta.
- O cenário que se deseja salvar deve estar aberto (Caso de Uso *Abrir Cenários*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Clica no ícone *Salvar Cenário* na barra de ferramentas na janela do cenário (Figura 31).	
	2) Percorre todas as atividades existentes no cenário e persiste no banco de dados as informações. 3) Percorre todos os relacionamentos existentes e persiste no banco de dados essas informações.

Fluxos Alternativos

- Não tem.

Pós-Condições

- Todas as atividades e relacionamento entre elas deverão estar persistidas no banco de dados do SARP.

5.7.6.8. Caso de Uso: *Definir Cenário Real*

Atores
- Gerente

Objetivos
- Permite definir que um cenário de alocação de recursos seja definido com o cenário real do projeto.

Pré-Condições
- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).
- A interface *Painel de Controle* deve estar aberta.

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Clica com o botão direito do mouse sobre o nome do projeto exibido no painel *Árvore de Cenários.*	
	2) Abre *pop-up* de opções (Figura 28).
3) Clica na opção *Definir Cenário Real*.	
	4) Persiste no banco de dados a informação de que aquele é o cenário real do projeto.

Fluxos Alternativos
- Não tem.

Pós-Condições
- O cenário real estará persistido no banco de dados como cenário real do projeto.

5.7.6.9. Caso de Uso: *Visualizar Andamento das Atividades*

Atores
- Gerente
- Desenvolvedor

Objetivos
- Permite aos membros do projeto visualizar as atividades que estão atrasadas dentro do cenário real do projeto, ou seja, quais atividades estão demorando mais tempo para serem executadas do que o planejado inicialmente. Com isso o gerente pode tomar ações corretivas no intuito de minimizar problemas futuros no projeto.

Pré-Condições
- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).
- A interface *Painel de Controle* deve estar aberta.
- O cenário real do projeto deve estar aberto (Caso de Uso *Abrir Cenários*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Clica no ícone *Visualizar Atividades Atrasadas* na barra de ferramentas na janela do cenário (Figura 31).	
	2) Exibe em vermelho as atividades que estão atrasadas (realizado maior que o planejado) e em cinza as atividades que estão em dia ou que ainda não foram iniciadas (Figura 35).

Fluxos Alternativos

- Não tem.

Pós-Condições

- Atividades atrasadas devem estar em vermelho e atividades em dia em cinza.

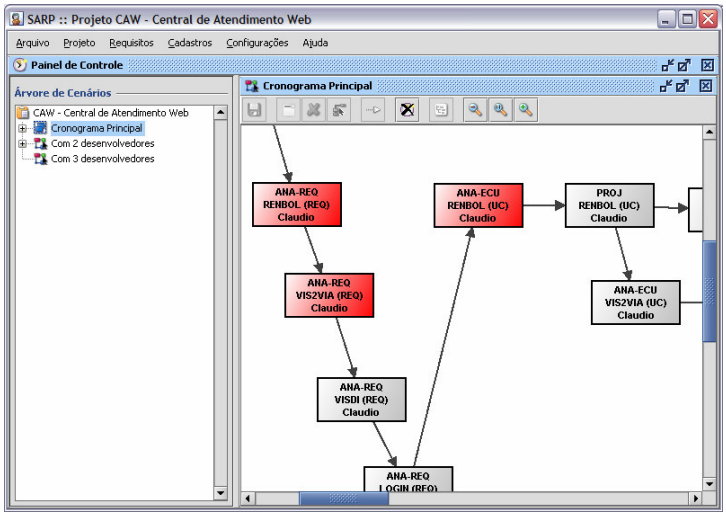


Figura 35: Visualização das atividades atrasadas.

5.7.6.10. Caso de Uso: *Visualizar Impactos em Atividades*

Atores

- Gerente
- Desenvolvedor

Objetivos

- Permite aos membros do projeto identificar quais atividades que podem ser impactadas por liberações de novas versões de requisitos e/ou casos de uso. Possibilitando assim que o gerente de projetos visualize o impacto das modificações de requisitos dentro das atividades do projeto.

Pré-Condições

- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).
- A interface *Painel de Controle* deve estar aberta.
- O cenário real do projeto deve estar aberto (Caso de Uso *Abrir Cenários*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Clica no ícone *Visualizar Impactos em Atividades* na barra de ferramentas na janela do cenário (Figura 31).	
	2) Exibe em vermelho as atividades que foram configuradas para determinadas versões de requisitos, sendo que estes tiveram liberações de novas versões. 3) Exibe alerta indicando quais requisitos tiveram liberações de novas versões.

Fluxos Alternativos

- Não tem.

Pós-Condições

- Atividades impactadas por modificações em requisitos e/ou casos de uso devem ser exibidas em vermelho e as demais em cinza.

Observações

- Essa funcionalidade compara a versão do requisito vinculado à atividade para determinar se houve alguma alteração na versão do requisito desde o momento em que a atividade foi inserida. Caso seja identificada essa diferença, o SARP sinaliza em vermelho as atividades que devem ser observadas com mais detalhes.
- Quando uma atividade está vinculada a um caso de uso, ela também está associada a uma determinada versão desse caso de uso. O SARP verifica se os requisitos que deram origem ao caso de uso não sofreram alguma alteração na versão, ou se o próprio caso de uso não foi modificado após a definição da atividade.

5.7.6.11. Caso de Uso: *Visualizar Caminho Crítico*

Atores

- Gerente
- Desenvolvedor

Objetivos

- Permite aos membros do projeto visualizar quais atividades do projeto são críticas, ou seja, que não possuem folgas e que, por isso, precisam ser executadas conforme o planejado.

Pré-Condições

- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).
- A interface *Painel de Controle* deve estar aberta.
- Um cenário de alocação de recursos deve estar aberto (Caso de Uso *Abrir Cenários*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Clica no ícone *Visualizar Caminho Crítico* na barra de ferramentas na janela do cenário (Figura 31).	

Ação do Ator	Comportamento do Sistema
	2) Exibe em vermelho as atividades que fazem parte do caminho crítico do projeto e em cinza as que não fazem (Figura 36).

Fluxos Alternativos

- Não tem.

Pós-Condições

- Atividades que fazem parte do caminho crítico devem estar em vermelho e atividades que não fazem em cinza.

Observações

- Para encontrar o caminho crítico, o SARP possui algumas regras para determinar o tempo de cada atividade.
  - Se a atividade está planejada então o SARP utiliza o tempo planejado.
  - Se a atividade está em andamento e até o momento não foram consumidas mais horas do que o planejado para a atividade em questão, o SARP também utiliza o tempo planejado. Entretanto, se a atividade está em andamento, mas já ultrapassou o tempo planejado, o SARP passa a utilizar o tempo consumido no cálculo do caminho crítico.
  - Se a atividade está concluída o SARP utiliza o tempo consumido.

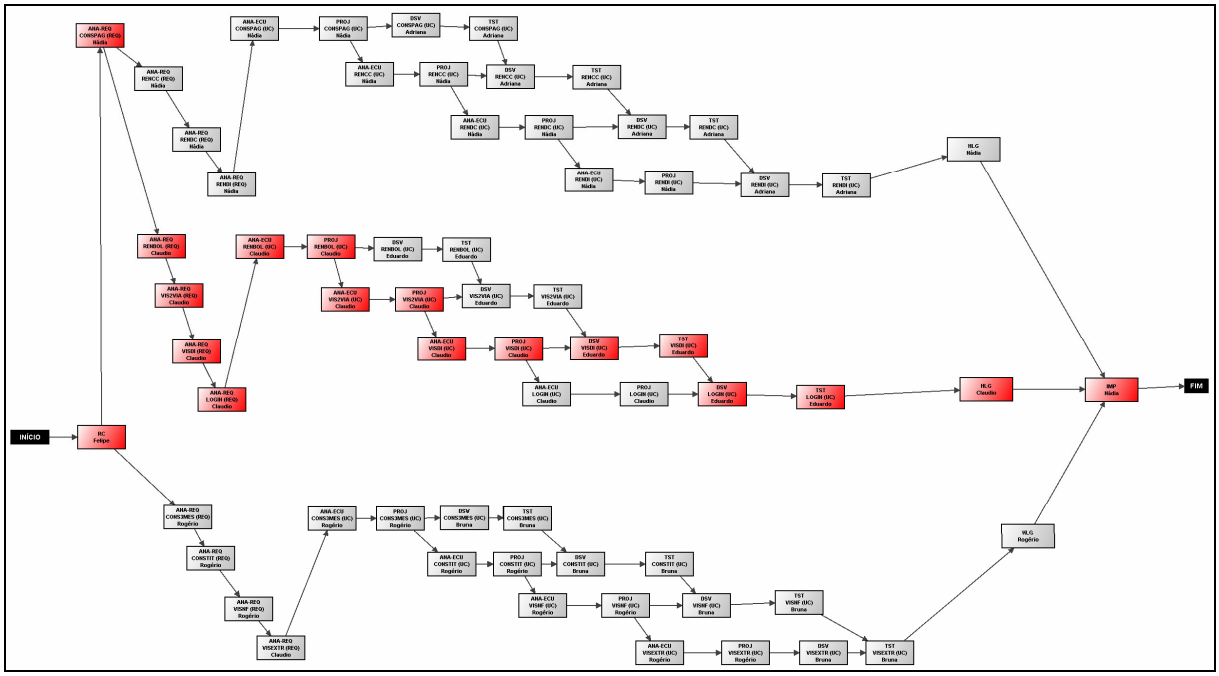


Figura 36: Exibição do caminho crítico.

5.7.6.12. Caso de Uso: *Gerar Simulação*

Atores

- Gerente

Objetivos

- Permite ao gerente do projeto criar "cópias" de cenários de alocação de recursos e modificar esses cenários no intuito de gerar novas combinações de recursos e

atividades. Com essas simulações o gerente pode chegar a um *cenário ótimo* dentro da realidade do projeto.

Pré-Condições

- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).
- A interface *Painel de Controle* deve estar aberta.

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Clica com o botão direito do mouse sobre o nome do projeto, no painel *Árvore de Cenários*, que deseja ser copiado para gerar uma simulação.	
	2) Abre *pop-up* de opções (Figura 28).
3) Clica na opção *Simular em um Novo Cenário.*	
	4) Exibe interface *Manutenção de Cenário* (Figura 29) para informar o nome do cenário da simulação.
5) Preenche os dados do cenário. 6) Clica no botão *Salvar*.	
	7) Sistema persiste os dados do cenário na base de dados do SARP. 8) Abre janela do novo cenário inserido, com todas as atividades existentes no cenário original que gerou a simulação.

Fluxos Alternativos

- Não tem.

Pós-Condições

- Cenário da simulação deverá estar aberto.

Observações

- A partir desse novo cenário gerado para simulação o gerente de projeto pode modificar as atividades, inserir novas, remover existentes e definir precedências no intuito e realizar combinações de alocações. Todas as funcionalidades existentes na interface *Painel de Controle* ficam disponíveis para esse novo cenário.

5.7.7. Interface *Minhas Atividades*

A interface *Minhas Atividades* possibilita o acesso ao conjunto de caso de uso *Controlar Minhas Atividades* que, utilizados de forma conjunta, permitem aos participantes do projeto inserir no sistema informações a respeito do andamento das suas atividades e dos artefatos gerados durante a execução das atividades. Esse conjunto de funcionalidades está disponível na interface *Minhas Atividades* que pode ser acessada através do menu *Projeto*.

A Figura 22 provê o detalhamento do caso de uso *Controlar Minhas Atividades.* Os principais casos de uso que podem ser executados a partir da interface de *Minhas Atividades* estão detalhados nas subseções seguintes, os demais podem ser encontrados no CD anexo.

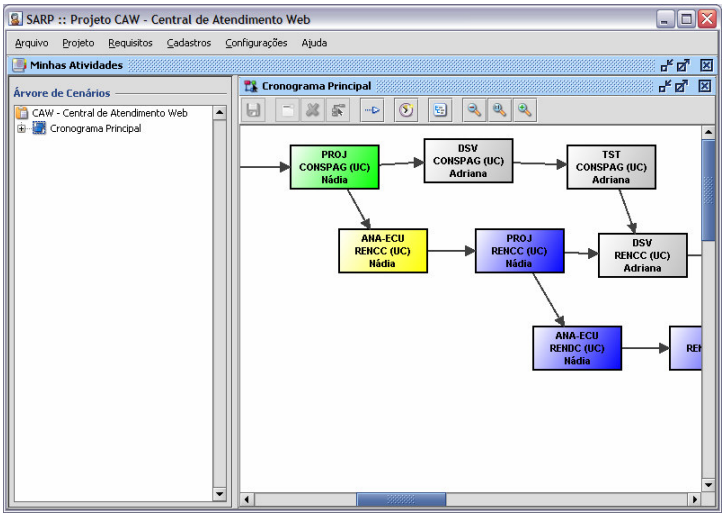


Figura 37: Interface *Minhas Atividades*.

O *layout* dessa interface (Figura 37) é idêntico ao *layout* da interface *Painel de Controle*, porém o comportamento é diferente. No painel *Árvore de Cenários* é exibido apenas o cenário real do projeto, sendo que esse cenário é automaticamente aberto ao acessar a interface. A forma de exibição das atividades também difere do painel de controle: apenas as atividades associadas ao usuário *logado* ficam disponíveis para edição. As atividades vinculadas aos demais participantes do projeto são exibidas em cinza. Algumas opções da barra de ferramentas da janela do cenário também ficam restringidas, uma vez que essas opções correspondem a funcionalidades que apenas o gerente de projeto pode executar. O participante do projeto pode apenas executar os casos de uso *Visualizar Caminho Crítico*, *Visualizar Impactos nas Atividades* e *Visualizar Atividades Atrasadas*.

5.7.7.1. Caso de Uso: *Controlar Horas Trabalhadas*

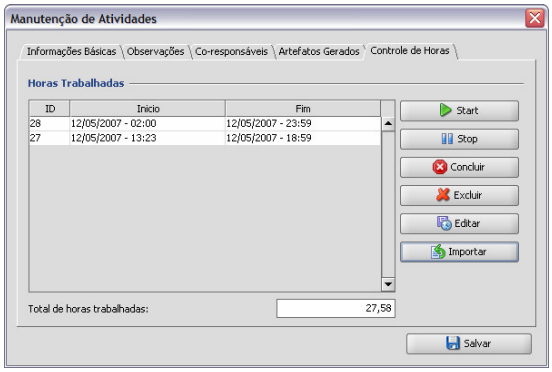


Figura 38: Interface de *Manutenção de Atividades*, aba *Controle de Horas*.

Atores

- Gerente
- Desenvolvedor

Objetivos

- Permite aos participantes do projeto informar as horas despendidas na execução de cada uma das atividades. Assim, é possível manter no SARP uma base de dados de horas trabalhadas e esses dados podem ser cruzados com outros dados como, por exemplo, tipos de atividades, no intuito de melhorar as estimativas de planejamento.

Pré-Condições

- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).
- A interface *Minhas Atividades* deve estar aberta.

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Clica com o botão direito do mouse sobre a atividade que deseja trabalhar.	
	2) Exibe *pop-up* de opções de ações sobre a atividade (Figura 34).
3) Clica na opção *Alterar Atividade*.	
	4) Exibe a interface *Manutenção de Atividades* com os dados da atividade selecionada populados.
5) Acessa a aba *Controle de Horas*.	
	6) Exibe as horas já trabalhadas detalhadamente e o somatório de horas trabalhadas no campo *Total* (Figura 38).
7) Inicia ou pára o controle de horas através dos botões *Start* e *Stop*, respectivamente.	
	8) Insere na base de dados a hora e o minuto em que o botão foi pressionado.

Fluxos Alternativos

- Em (8), se for a primeira vez em que um controle de horas está sendo inserido o sistema modifica a situação da atividade de *Planejada* para *Em Andamento*.
- Em (7), ao invés de controlar os tempos através dos botões *Start* e *Stop*, é possível importar as horas trabalhadas a partir de um arquivo CSV. Para isso o usuário deve clicar no botão *Importar*, selecionar o arquivo que contém o controle de horas e o SARP irá associar essas horas à atividade que está sendo editada.

Pós-Condições

- Horas de início e fim devem estar inseridas na base de dados.

Observações

- Essa interface também está disponível no *Painel de Controle* para o gerente de projeto. Porém, não é possível iniciar e parar o controle de horas, mas somente visualizar as horas já inseridas pelo responsável da atividade.
- Para concluir uma atividade deve ser executado o caso de uso *Concluir Atividade* detalhado no CD anexo.

5.7.7.2. Caso de Uso: *Controlar Artefatos Gerados*

Atores

- Gerente
- Desenvolvedor

Objetivos

- Permite aos participantes do projeto vincular às suas atividades artefatos produzidos durante a execução da atividade. Esses artefatos podem ser fluxos, atas de reunião, imagens, especificações, ou seja, qualquer arquivo que possa ser interessante centralizar na base de dados do SARP.

Pré-Condições

- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).
- A interface *Minhas Atividades* deve estar aberta.

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Clica com o botão direito do mouse sobre a atividade que deseja trabalhar.	
	2) Exibe *pop-up* de opções de ações sobre a atividade (Figura 34).
3) Clica na opção *Alterar Atividade*.	
	4) Exibe a interface *Manutenção de Atividades* com os dados da atividade selecionada populados.
5) Acessa a aba *Artefatos Gerados.*	
	6) Exibe os artefatos já inseridos na atividade (Figura 39).
7) Clica no botão *Upload* para inserir um artefato.	
	8) Insere o artefato na base de dados do SARP.

Fluxos Alternativos

- Em (7), ao invés de fazer o *upload*, pode ser feito o *download* ou a exclusão de um artefato já inserido na base de dados.

Pós-Condições

- O artefato estará inserido na base dados do SARP.

Observações

- Essa interface também está disponível no *Painel de Controle* para o gerente de projeto. Porém, não é possível fazer *uploads*, mas somente *downloads* de artefatos já inseridos.

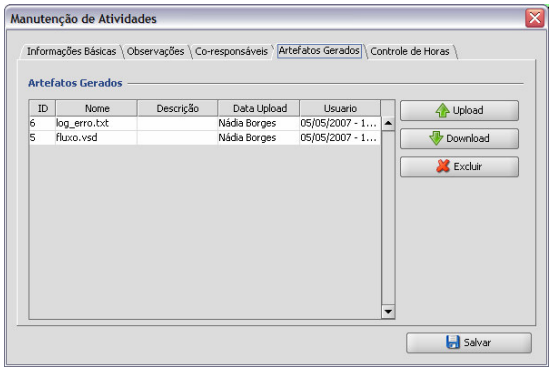


Figura 39: Interface de *Manutenção de Atividades*, aba *Artefatos Gerados.*

5.8. Módulo *Requisitos*

Através do módulo *Requisitos* toda a equipe envolvida com o projeto pode ter acesso aos requisitos que estão detalhados para o projeto. Os requisitos podem estar descritos de forma textual e podem ser relacionados a casos de uso, através das interfaces *Manutenção de Requisitos* e *Manutenção de Casos De Uso*.

A interface de *Manutenção de Requisitos* provê funcionalidades que permite o usuário controlar as modificações dos requisitos, os relacionamentos existentes entre eles, os relacionamentos existentes entre os requisitos e os casos de uso e, também, o controle de artefatos vinculados. Por artefatos vinculados entende-se qualquer tipo de arquivo (documento, fluxo, imagem, etc.) que pode estar diretamente relacionado ao requisito e cujo armazenamento seja conveniente ao projeto

Disponibilizando essas funcionalidades para controlar os requisitos, estamos de acordo com o modelo propostos por Kotonya e Sommerville (1998).

A interface de *Manutenção de Casos de Uso* não deixa de ser uma especialização de um artefato gerado a partir de um requisito. Os casos de uso, de acordo com Martins (2006), servem de notação para capturar os requisitos com base em cenários de utilização. Eles podem ser refinados em outros casos de uso ou em seqüências detalhadas de interações entre objetos. Esses artefatos utilizam o vocabulário do domínio do problema e, por isso, servem para detalhar as funcionalidades do sistema na linguagem dos usuários e/ou clientes. Podem ainda ser usados para prover uma definição mais rigorosa dos requisitos.

Junto com a interface de *Manutenção de Casos de Uso* está a *Manutenção de Atores*. Os atores estão diretamente relacionados aos casos de uso. Eles podem representar um papel exercido por uma pessoa ou por um sistema externo que interage com o sistema desenvolvido. Por esses motivos é interessante manter suas descrições detalhadas e armazenadas juntos com os casos de uso no intuito de prover um entendimento maior sobre o contexto da aplicação.

Cada uma das funcionalidades existentes nesse módulo é detalhada na seqüência na forma de casos de uso.

5.8.1. Caso de Uso: *Controlar Requisitos*

O caso de uso *Controlar Requisitos* é um conjunto de outros casos de uso que, utilizados de forma conjunta, permitem aos participantes do projeto manter uma base de dados de requisitos. Esse conjunto de funcionalidade está disponível na interface *Manutenção de Requisitos* que pode ser acessada através do menu *Requisitos*. A Figura 40 exibe os casos de usos detalhados, sendo que os mais importantes são detalhados nas subseções a seguir e os demais podem ser consultados na documentação contida no CD anexo.

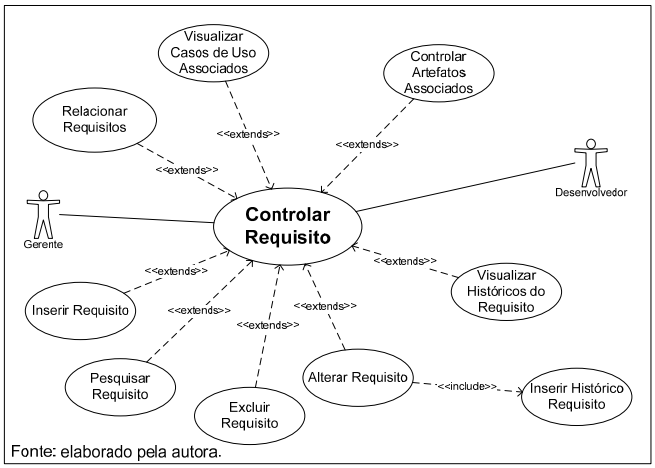


Figura 40: Detalhamento do caso de uso *Controlar Requisito*.

5.8.1.1. Caso de Uso: *Inserir Requisito*

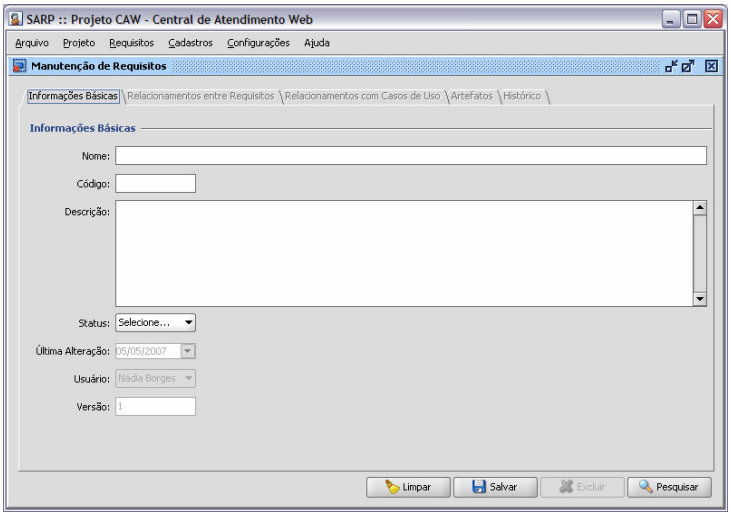


Figura 41: Interface de *Manutenção de Requisitos*, aba *Informações Básicas*.

Atores

- Gerente
- Desenvolvedor

Objetivos

- Permitir a inclusão de um novo requisito na base de dados do SARP.

Pré-Condições

- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
9) Acessa o item *Manutenção de Requisitos* no menu *Requisito*.	
	10) Exibe interface de *Manutenção de Requisitos* (Figura 41).
11) Informa os dados do requisito. 12) Clica no botão *Salvar*.	
	13) Sistema avalia se os campos obrigatórios foram preenchidos. 14) Sistema persiste os dados do requisito na base de dados do SARP.

Fluxos Alternativos

- Em (5), caso algum campo obrigatório não tenha sido preenchido, deve ser exibida mensagem de alerta informando qual campo não foi preenchido.

Pós-Condições

- O requisito deve estar inserido na base de dados do SARP.

5.8.1.2. Caso de Uso: *Pesquisar Requisito*

Atores
- Gerente
- Desenvolvedor

Objetivos
- Permitir a busca por um determinado requisito do sistema na base de dados do SARP.

Pré-Condições
- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Acessa o item *Manutenção de Requisitos* no menu *Requisitos*.	
	2) Exibe interface de *Manutenção de Requisitos* (Figura 41).
3) Clica no botão *Pesquisar*.	
	4) Exibe interface de *Pesquisa de Requisitos*.
5) Informa o nome do requisito (opcional). 6) Clica no botão *Pesquisar*.	
	7) Busca na base de dados os requisitos que coincidam com os filtros de pesquisa aplicados. 8) Exibe os requisitos encontrados.
9) Seleciona um dos requisitos exibidos pelo sistema.	
	10) Carrega a interface *Manutenção de Requisitos* com os dados do requisito selecionado.

Fluxos Alternativos
- Depois de (10), o caso de uso *Alterar Requisito* pode ser executado.

Pós-Condições
- Requisito selecionado estará detalhado na interface *Manutenção de Requisitos*.

5.8.1.3. Caso de Uso: *Alterar Requisito*

Atores
- Gerente
- Desenvolvedor

Objetivos
- Permitir que os dados cadastrais de um requisito sejam modificados e persistidos.

Pré-Condições
- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Acessa o item *Manutenção de Requisitos* no menu *Requisitos*.	
	2) Exibe interface de *Manutenção de Requisitos* (Figura 41).
3) Efetua uma pesquisa (caso de uso *Pesquisar Requisito*). 4) Faz as modificações necessárias nos dados do requisito. 5) Clica no botão *Salvar*.	
	6) Sistema avalia se os campos obrigatórios foram preenchidos. 7) Exibe interface para informar a justificativa da alteração.
8) Informa a justificativa e clica em *Salvar.*	
	9) Insere histórico da modificação do requisito (caso de uso *Inserir Histórico Requisito*). 10) Sistema persiste os dados do requisito na base de dados do SARP.

Fluxos Alternativo

- Em (6), caso algum campo obrigatório não tenha sido preenchido, deve ser exibida mensagem de alerta informando qual campo não foi preenchido.

Pós-Condições

- O requisito deve estar alterado na base de dados do SARP.

5.8.1.4. Caso de Uso: *Relacionar Requisitos*

Atores

- Gerente
- Desenvolvedor

Objetivos

- Permitir o controle da rastreabilidade do requisito com outros requisitos. Essa dependência pode ser de dois tipos:
  - *Requisitos dos quais eu dependo*: são aqueles requisitos que, se sofrerem modificações, podem impactar no requisito que está sendo editado.
  - *Requisitos que dependem de mim*: são aqueles requisitos que podem ser impactos se o requisito que está sendo editado sofrer modificações.

Pré-Condições

- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Acessa o item *Manutenção de Requisitos* no menu *Requisitos*.	
	2) Exibe interface de *Manutenção de Requisitos* (Figura 41).

Ação do Ator	Comportamento do Sistema
3) Efetua uma pesquisa (caso de uso *Pesquisar Requisito*). 4) Acessa a aba *Relacionamentos entre Requisitos.*	
	5) Sistema exibe os relacionamentos já inseridos para o requisito que está sendo trabalhado (Figura 42).
6) Modifica os relacionamentos através dos botões *Adicionar* e *Remover*.	
	7) Atualiza as *grids* de requisitos relacionados de acordo com as ações do usuário.
8) Clica no botão *Salvar*.	
	9) Persiste os relacionamentos entre requisitos na base de dados do SARP.

Fluxos Alternativos

- Em (6), não permite a duplicação de relacionamento entre requisitos.

Pós-Condições

- As associações entre os requisitos devem ter sito persistidas na base de dados do SARP.

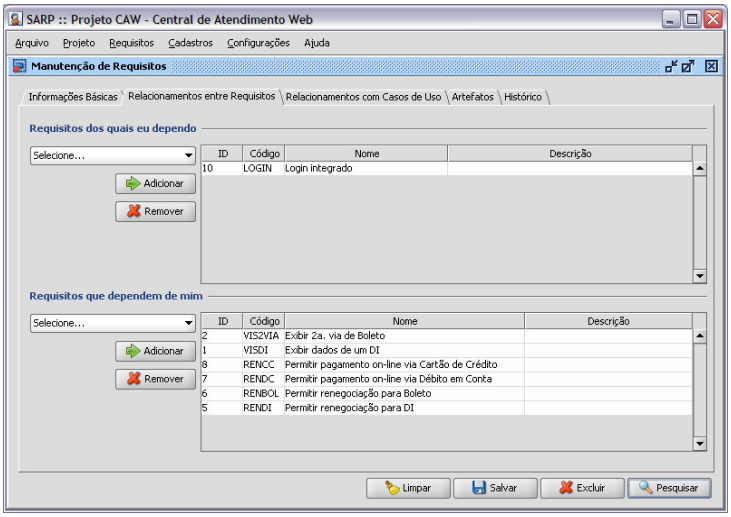


Figura 42: Interface de *Manutenção de Requisitos*, aba *Relacionamento entre Requisitos.*

5.8.1.5. Caso de Uso: *Visualizar Casos de Uso Associados*

Atores

- Gerente
- Desenvolvedor

Objetivos

- Permitir o controle da rastreabilidade do requisito com os casos de uso gerados a partir do requisito.

Pré-Condições

- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Acessa o item *Manutenção de Requisitos* no menu *Requisitos*.	
	2) Exibe interface de *Manutenção de Requisitos* (Figura 41).
3) Efetua uma pesquisa (caso de uso *Pesquisar Requisito*). 4) Acessa a aba *Relacionamentos com Casos de Uso*.	
	5) Sistema exibe os relacionamentos já inseridos para o requisito que está sendo detalhado.

Fluxos Alternativos

- Não tem.

Pós-Condições

- As associações entre o requisito e os casos de uso devem ficar disponíveis.

5.8.1.6. Caso de Uso: *Controlar Artefatos Associados*

Atores

- Gerente
- Desenvolvedor

Objetivos

- Permite aos participantes do projeto vincular artefatos aos requisitos. Esses artefatos podem ser fluxos, diagramas, imagens, ou seja, qualquer arquivo que possa ser interessante centralizar na base de dados do SARP.

Pré-Condições

- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Acessa o item *Manutenção de Requisitos* no menu *Requisitos*.	
	2) Exibe interface de *Manutenção de Requisitos* (Figura 41).
3) Efetua uma pesquisa (caso de uso *Pesquisar Requisito*). 4) Acessa a aba *Artefatos.*	
	5) Exibe os artefatos já associados ao requisito (Figura 43).
6) Clica no botão *Upload* para inserir um artefato.	
	7) Exibe interface para selecionar o arquivo que se deseja vincular ao requisito.
8) Seleciona o arquivo que se deseja relacionar ao requisito.	
	9) Insere o artefato na base de dados do SARP, relacionando ele ao requisito.

Fluxos Alternativos

- Em (6), ao invés de fazer o *upload*, pode ser feito o *download* ou a exclusão de um artefato já inserido na base de dados.

Pós-Condições

- O artefato estará inserido na base dados do SARP.

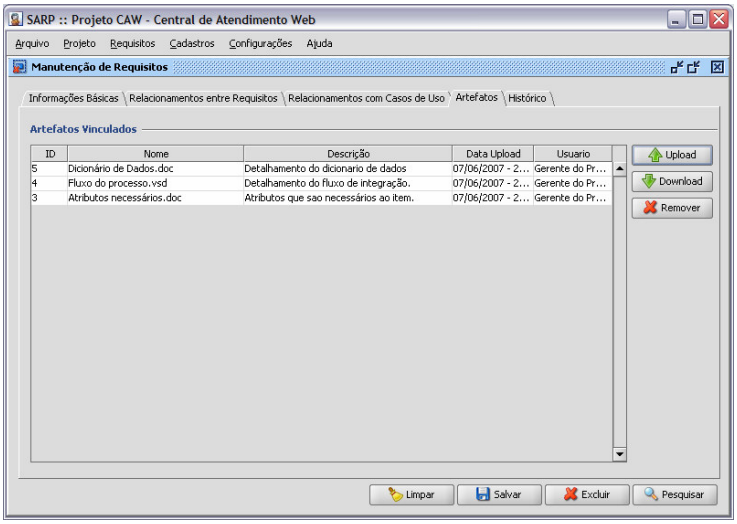


Figura 43: Interface de *Manutenção de Requisitos*, aba *Artefatos.*

5.8.1.7. Caso de Uso: *Visualizar Históricos do Requisito*

Atores

- Gerente
- Desenvolvedor

Objetivos

- Permite aos participantes do projeto visualizar os históricos de modificações de um requisito.

Pré-Condições

- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Acessa o item *Manutenção de Requisitos* no menu *Requisitos*.	
	2) Exibe interface de *Manutenção de Requisitos* (Figura 41).
3) Efetua uma pesquisa (caso de uso *Pesquisar Requisito*). 4) Acessa a aba *Históricos.*	
	5) Exibe todos os históricos de modificação do requisito (Figura 44).

Fluxos Alternativos

- Não tem.

Pós-Condições

- Os históricos são exibidos na *grid*.

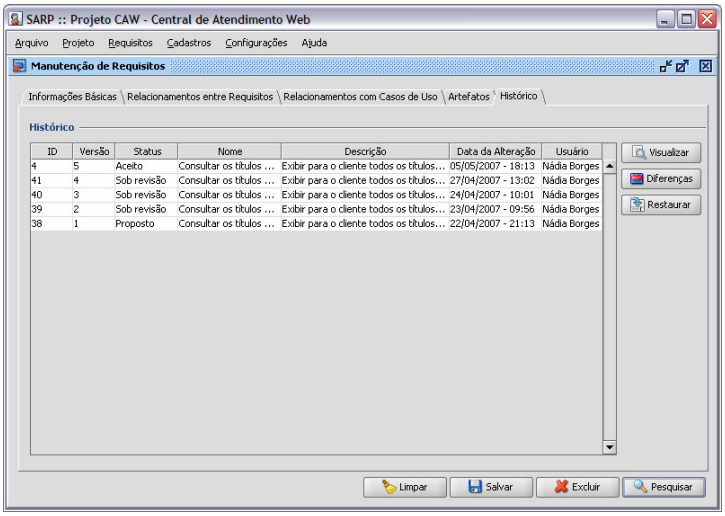


Figura 44: Interface de *Manutenção de Requisitos*, aba *Histórico.*

5.8.2. Caso de Uso: *Controlar Casos de Uso*

O caso de uso *Controlar Casos de Uso* é um conjunto de outros casos de uso que, utilizados de forma conjunta, permitem aos participantes do projeto manter uma base de dados dos casos de uso do sistema. Esse conjunto de funcionalidade está disponível na interface *Manutenção de Casos de Uso* que pode ser acessada através do menu *Requisitos*. A Figura 45 exibe os casos de usos detalhados, sendo que os mais importantes são detalhados nas subseções a seguir e os demais podem ser consultados na documentação contida no CD anexo.

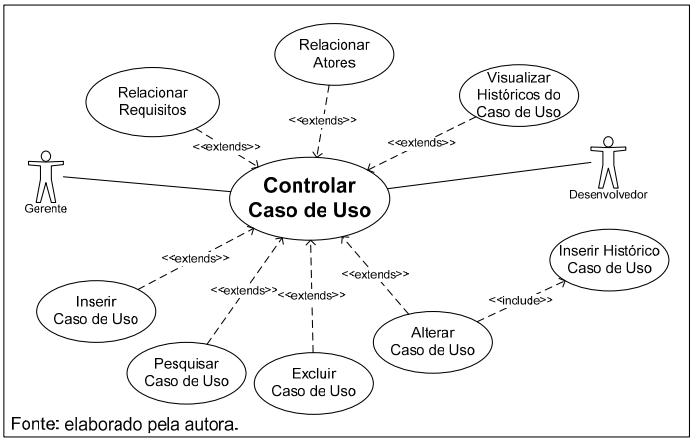


Figura 45: Detalhamento do caso de uso *Controlar Caso de Uso*.

5.8.2.1. Caso de Uso: *Inserir Caso de Uso*

Atores

- Gerente
- Desenvolvedor

Objetivos

- Permitir a inclusão de um novo caso de uso na base de dados do SARP.

Pré-Condições

- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Acessa o item *Manutenção de Casos de Uso* no menu *Requisito*.	
	2) Exibe interface de *Manutenção de Casos de Uso* (Figura 46).
3) Informa as informações básicas do caso de uso. 4) Informa os requisitos com os quais caso de uso se relaciona (caso de uso *Relacionar Requisitos*). 5) Informa os atores com os quais caso de uso se relaciona (caso de uso *Relacionar Atores*) 6) Informa o conteúdo do caso de uso (fluxo básico, fluxos alternativos, pré e pós-condições). 7) Clica no botão *Salvar*.	
	8) Sistema avalia se os campos obrigatórios foram preenchidos. 9) Sistema persiste os dados do caso de uso na base de dados do SARP.

Fluxos Alternativos

- Em (8), caso algum campo obrigatório não tenha sido preenchido, deve ser exibida mensagem de alerta informando qual campo não foi preenchido.

Pós-Condições

- O caso de uso deve estar inserido na base de dados do SARP.

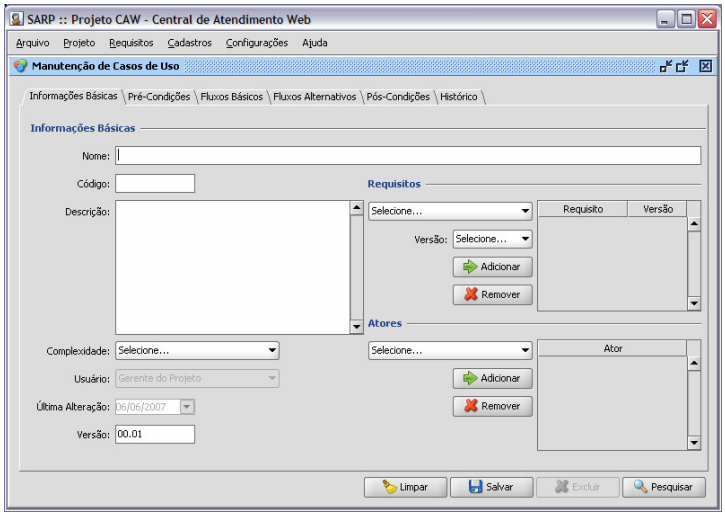


Figura 46: Interface de *Manutenção de Casos de Uso*, aba *Informações Básicas*.

5.8.2.2. Caso de Uso: *Alterar Caso de Uso*

Atores
- Gerente
- Desenvolvedor

Objetivos
- Permitir que os dados cadastrais de um caso de uso sejam modificados e persistidos.

Pré-Condições
- O usuário deverá estar *logado* no sistema.
- O projeto deve estar carregado na sessão da aplicação (Caso de Uso *Carregar Projeto*).

Fluxo Principal

Ação do Ator	Comportamento do Sistema
1) Acessa o item *Manutenção de Casos de Uso* no menu *Requisitos*.	
	2) Exibe interface de *Manutenção de Casos de Uso* (Figura 46).
3) Efetua uma pesquisa (caso de uso *Pesquisar Casos de Uso*). 4) Faz as modificações necessárias nos dados do caso de uso. 5) Clica no botão *Salvar*.	
	6) Sistema avalia se os campos obrigatórios foram preenchidos. 7) Exibe interface para informar a justificativa da alteração.
8) Informa a justificativa e clica em *Salvar*.	
	9) Insere histórico da modificação do caso de uso. 10) Sistema persiste os dados do caso de uso na base de dados do SARP.

Fluxos Alternativos
- Em (6), caso algum campo obrigatório não tenha sido preenchido, deve ser exibida mensagem de alerta informando qual campo não foi preenchido.

Pós-Condições
- O caso de uso deve estar alterado na base de dados do SARP.

5.8.3. Caso de Uso: *Controlar Atores*

O caso de uso *Controlar Atores* é um conjunto de outros casos de possibilita manter uma base de dados dos atores que interagem com o sistema que está sendo desenvolvido. Esse conjunto de funcionalidade está disponível na interface *Manutenção de Atores* que pode ser acessada através do menu *Requisitos*. Mais detalhes sobre esses casos de uso podem ser encontrados no CD anexo.

5.9. Casos de Uso Módulo *Cadastros*

As funcionalidades disponíveis através do módulo *Cadastro* permitem que os dados auxiliares sejam inseridos na base de dados do SARP. Esses dados são independentes do projeto, pois são utilizados em todos os projetos inseridos no SARP. Esse módulo, como o nome já diz, é um conjunto de interfaces que permitem a inserção, alteração, pesquisa e exclusão de registros. As funcionalidades oferecidas através do módulo *Cadastros* são:

a) Manutenção de Papéis
b) Manutenção de Tipos de Atividades
c) Manutenção de Tecnologias
d) Manutenção de Recursos Humanos
e) Manutenção de Usuários

Mais detalhes sobre os casos de uso das funcionalidades desse módulo podem ser encontrados no CD anexo.

6. ESTUDO DE CASO

O presente capítulo apresenta um breve estudo de caso da utilização do sistema proposto, SARP, no intuito de exemplificar a utilização da ferramenta no auxílio do gerenciamento de um projeto específico de desenvolvimento de *software*.

As informações a respeito do projeto apresentado, tais como empresa, itens do projeto, assim como os indivíduos participantes desse empreendimento estão sendo ocultados ou modificados devido a cláusulas contratuais de sigilo existente entre as partes envolvidas.

6.1. Contextualização

O objeto deste estudo de caso é um projeto de manutenção evolutiva de um sistema existente em uma empresa que presta serviços e comercializa produtos através da *Internet*. O sistema, cujo desenvolvimento iniciou no ano de 2001 e que foi implantado em 2002, necessita dessas manutenções evolutivas para atender as necessidades de negócio da organização. Ele é responsável pelo processamento mensal de aproximadamente 2,5 milhões de registros, sendo que esse processamento está voltado aos processos de faturamento e de cobrança bancária da organização.

O projeto estudado iniciou em março de 2007 e tem como previsão de término de novembro de 2007. Por ser um projeto de manutenção o objetivo dele não é construir um produto inteiramente novo, mas sim acrescentar novas funcionalidades ao sistema existente de forma a impulsionar os processos da organização.

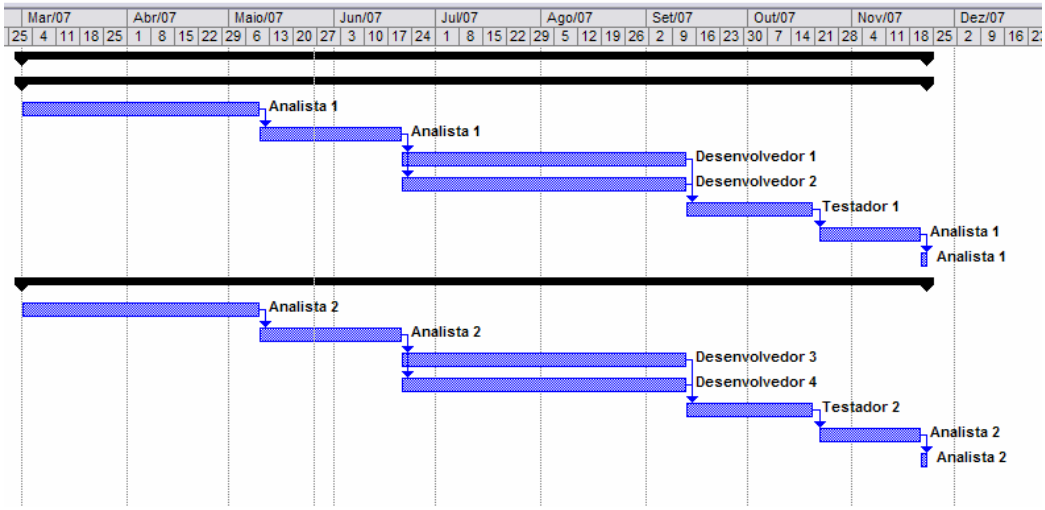
Como em outros projetos de manutenção já concluídos sobre essa ferramenta, foi estipulado um escopo macro de funcionalidades a serem desenvolvidas ao longo do período determinado. São elas:

a) **Central de Atendimento *Web***: disponibilizar uma interface *web* através da qual os próprios clientes da organização poderiam renegociar e efetuar pagamentos de faturas em aberto, desonerando, assim, o *call center*.
b) **Nova Modalidade Pré-Paga**: adaptar o sistema de faturamento, que hoje opera apenas com a modalidade de pagamento *pós-paga,* para faturar e cobrar por produtos e serviços também na modalidade *pré-paga.* Ou seja, só haveria a liberação de determinado produto ou serviço após a confirmação do pagamento.

O cronograma macro do projeto estabelecido inicialmente previa a existência de duas equipes trabalhando separadamente, cada uma responsável pelo desenvolvimento de um dos módulos descritos acima. Como não existe dependência direta entre os módulos o desenvolvimento pode ocorrer de forma paralela, sendo que cada uma das frentes de trabalho segue o ciclo de vida cascata, modelo clássico para o desenvolvimento de sistemas (Pressman, 2001).

A Figura 47 exibe o cronograma macro do projeto criado utilizando a ferramenta MS Project da Microsoft, enquanto que a Figura 48 exibe o mesmo cronograma de atividades no SARP. As atividades exibidas em amarelo estavam em andamento e as atividades azuis estavam planejadas no momento em que o *screenshot* foi realizado.

Nome da tarefa	Duração	Início	Término	Predecessoras	Nomes de recursos
– Projeto	**196 dias**	**Qui 1/3/07**	**Qui 22/11/07**		**Gerente**
– Central de Atendimento Web	**196 dias**	**Qui 1/3/07**	**Qui 22/11/07**		
Análise	50 dias	Qui 1/3/07	Qua 9/5/07		Analista 1
Projeto	30 dias	Qui 10/5/07	Qua 20/6/07	3	Analista 1
Desenvolvimento	60 dias	Qui 21/6/07	Qua 12/9/07	4	Desenvolvedor 1
Desenvolvimento	60 dias	Qui 21/6/07	Qua 12/9/07	4	Desenvolvedor 2
Testes	30 dias	Qui 13/9/07	Sex 19/10/07	5;6	Testador 1
Homologacao	24 dias	Seg 22/10/07	Ter 20/11/07	7	Analista 1
Implantação	2 dias	Qua 21/11/07	Qui 22/11/07	8	Analista 1
– Nova Modalidade Pré-Paga	**196 dias**	**Qui 1/3/07**	**Qui 22/11/07**		
Análise	50 dias	Qui 1/3/07	Qua 9/5/07		Analista 2
Projeto	30 dias	Qui 10/5/07	Qua 20/6/07	11	Analista 2
Desenvolvimento	60 dias	Qui 21/6/07	Qua 12/9/07	12	Desenvolvedor 3
Desenvolvimento	60 dias	Qui 21/6/07	Qua 12/9/07	12	Desenvolvedor 4
Testes	30 dias	Qui 13/9/07	Sex 19/10/07	13;14	Testador 2
Homologacao	24 dias	Seg 22/10/07	Ter 20/11/07	15	Analista 2
Implantação	2 dias	Qua 21/11/07	Qui 22/11/07	16	Analista 2



Fonte: elaborado pela autora.

Figura 47: Cronograma macro do projeto.

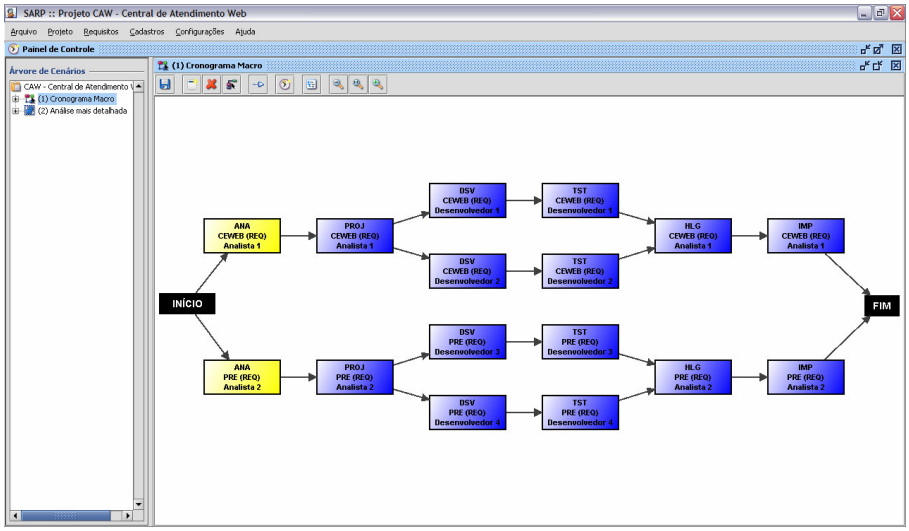


Figura 48: Cronograma macro do projeto inserido no SARP.

As atividades exibidas em amarelo são atividades da etapa de análise vinculadas aos requisitos *Central de Atendimento Web* (CEWEB) e *Nova Modalidade Pré-Paga* (PRE). Como no início do projeto não existia conhecimento detalhado a respeito das funcionalidades

específicas de cada módulo, o cronograma acaba sendo mais amplo. No decorrer do projeto essas atividades macro tendem a ser substituídas por atividades menores correlacionadas a requisitos mais específicos.

6.1.1. Equipe do Projeto

A equipe destinada ao projeto foi composta por dois analistas de sistemas, quatro desenvolvedores e por um gerente de projeto. A Figura 49 exibe a configuração da equipe do projeto no SARP, sendo que um mesmo recurso assumiu diferentes papéis dentro do projeto. Essa definição dos papéis é muito importante, pois, dependendo do tipo de uma atividade que será executada, ela só poderá ser executada por membros da equipe que possuem o papel necessário. Por exemplo, o recurso *Desenvolvedor 1* está associado aos papéis *Desenvolvedor/Programador* e *Testador*. Logo, ao criar uma atividade do tipo *Análise*, esse recurso não estará disponível para ser utilizado.

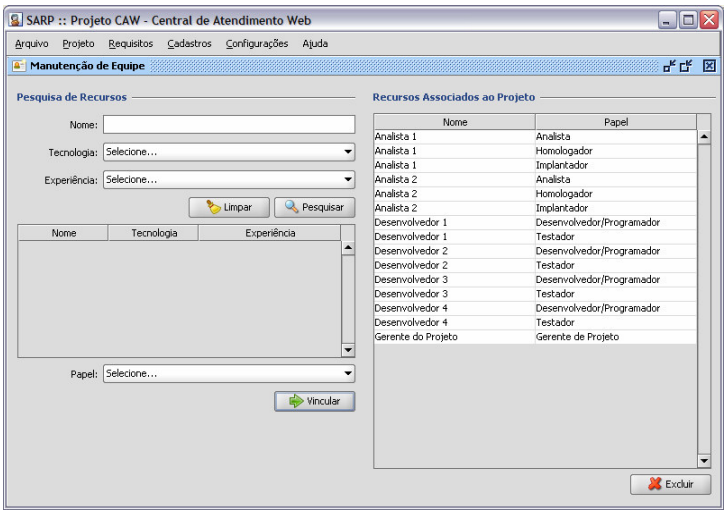


Figura 49: Configuração da equipe do projeto no SARP.

6.2. Andamento do Projeto

A equipe responsável pelo desenvolvimento da *Central de Atendimento Web* conseguiu concluir a primeira etapa do processo (Análise) praticamente dentro do cronograma inicial estipulado. Nessa etapa foram envolvidos alguns usuários chaves que fazem parte das equipes do *call center*. Esses usuários conhecem mais profundamente as necessidades de renegociações de dívidas dos clientes da empresa e por isso se tornaram os provedores de requisitos da *Central de Atendimento Web*. Os requisitos funcionais foram identificados através de reuniões com esses usuários e atas foram elaboradas no intuito de garantir o aceite formal dos itens discutidos.

Nas primeiras entrevistas realizadas foram identificas as principais funcionalidades que o *site* deveria apresentar. Ao todo foram levantados 12 requisitos funcionais, sendo que estes foram detalhados ao longo da etapa de análise. Uma vez identificados requisitos consistentes, o gerente do projeto pôde ajustar o cronograma de forma a detalhar as atividades da etapa de análise, vinculando-as aos seus respectivos requisitos. O que antes era uma atividade única de análise sobre o requisito *Central de Atendimento Web* passou a ser um conjunto de atividades menores, associadas diretamente aos seus requisitos. Para isso, um

novo cenário de alocação dos recursos foi gerado pelo gerente, conforme vemos na Figura 50. Nessa figura é exibida a interface *Minhas Atividades* e nela apenas as atividades vinculadas ao recurso que está *logado* no SARP aparecem habilitadas. Todas as demais atividades são exibidas, mas não ficam disponíveis para modificações. As atividades que aparecem em verde estão concluídas.

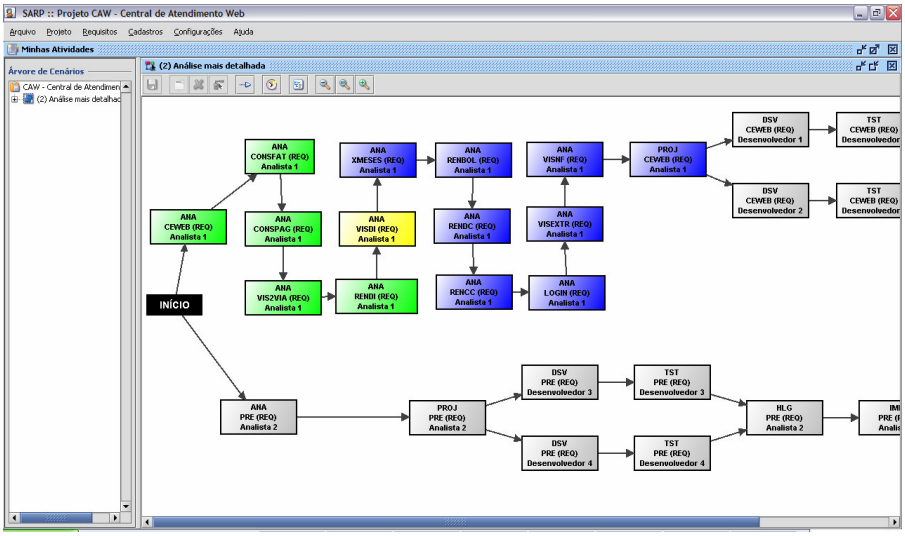


Figura 50: Cronograma detalhado das atividades de análise – Visão do recurso *Analista 1*.

Durante a etapa de análise os requisitos são detalhados pelos analistas responsáveis. Nessa etapa os requisitos são aprofundados pelos analistas e pelos usuários chaves no intuito de se alcançar um ponto ótimo sobre os requisitos. O SARP, através da interface *Manutenção de Requisitos* permite que seja acompanhada a evolução dos requisitos de duas formas:

a) controlando o histórico das modificações dos requisitos e
b) controlando as associações entre os requisitos.

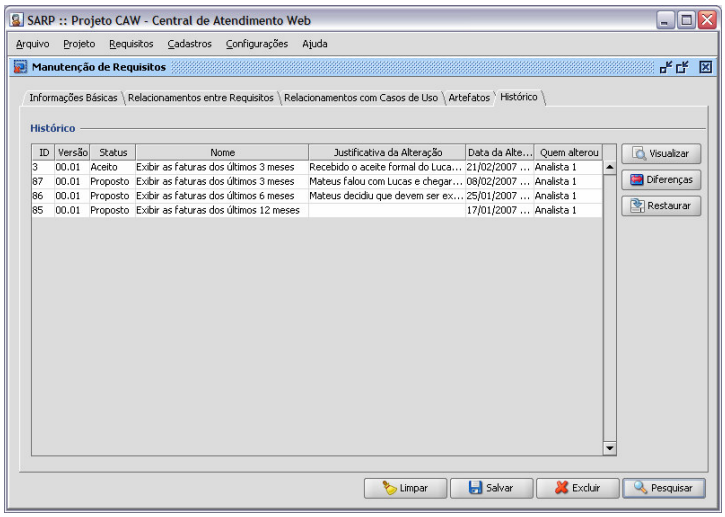


Figura 51: Histórico de modificações do requisito *Exibir faturas dos últimos 3 meses*.

A Figura 51 exibe o histórico de modificações do requisito *Exibir faturas dos últimos 3 meses*. Podemos verificar que o requisito passou por três modificações até ter seu *Status* alterado para *Aceito*, sendo que para cada modificação houve a inserção de uma justificativa. Já a Figura 52 exibe os relacionamentos existentes entre o requisito *Consultar faturas para pagamento* com outros requisitos dos quais ele depende e que dependem dele. Isso indica que

se houver uma modificação no requisito de *Consultar faturas para pagamento* os requisitos ali relacionados podem ser impactados de alguma forma. Fica estabelecida, então, a matriz de rastreabilidade entre os requisitos do sistema.

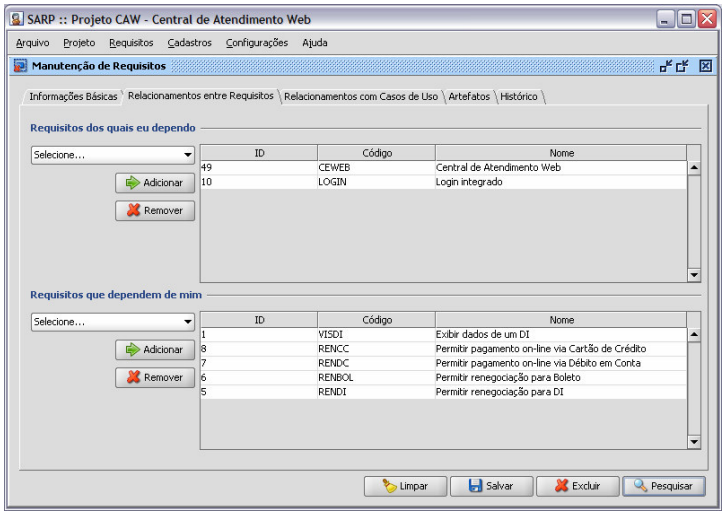


Figura 52: Requisitos relacionados ao requisito *Consultar faturas para pagamento*.

A equipe responsável pelo desenvolvimento do segundo módulo do projeto, *Nova Modalidade Pré-Paga*, não obteve o mesmo desempenho do primeiro módulo. Ao contrário da *Central de Atendimento Web*, as necessidades de negócio sobre a modalidade de pagamento pré-paga não estavam claras, tanto para a área de produtos da empresa, quanto pra área contábil.

A etapa de análise desse módulo se estendeu mais que o planejado inicialmente, sem que os requisitos fossem esclarecidos. Houve apenas um entendimento macro de quais módulos do sistema de faturamento e cobrança poderiam ser impactados, mas a forma de como seria esse impacto não pode ser detalhada. A Figura 53 mostra como o SARP permite a o acompanhamento do andamento do projeto, destacando em vermelho aquelas atividades que levaram (ou estão levando) mais tempo para serem concluídas do que o planejado inicialmente.

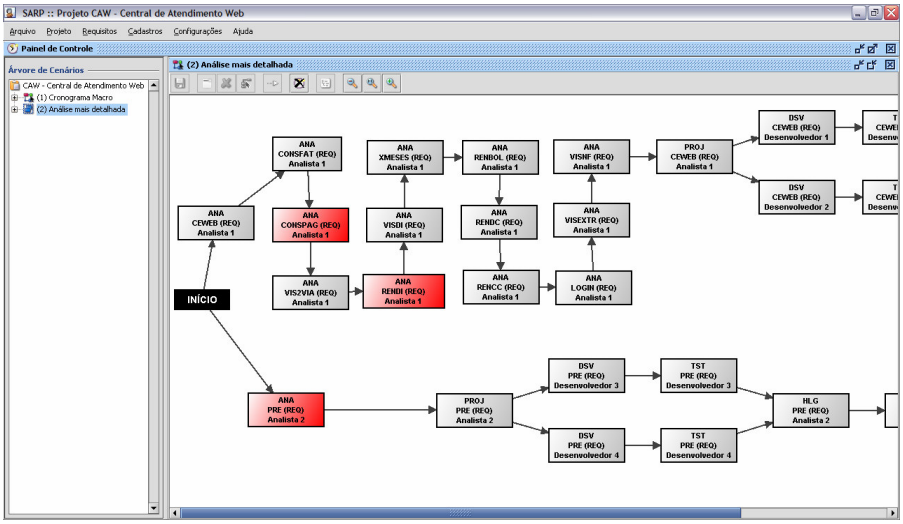


Figura 53: Destaque das atividades que despenderam mais tempo do que planejado.

Em virtude dessa carência de definições a respeito das necessidades de negócio da empresa sobre o escopo da nova modalidade pré-paga, o gerente do projeto resolveu, então, que as áreas clientes precisariam primeiro definir internamente as suas necessidades antes de envolver novamente os analistas do projeto.

Como os analistas já estavam alocados para o projeto e não poderiam ficar sem atividades, aguardando por definições das áreas clientes, o gerente sugeriu ao *sponsor* (patrocinador) que fossem incluídos no escopo do projeto o desenvolvimento de uma melhoria em um relatório existente no sistema de faturamento, assim como a criação de uma nova interface que permitisse a identificação de certos pagamentos efetuados através de cartão de crédito. O *sponsor* aprovou essa alteração de escopo do projeto e o gerente adaptou o cronograma para que elas fossem contempladas. A Figura 54 mostra o cronograma ajustado com as novas atividades.

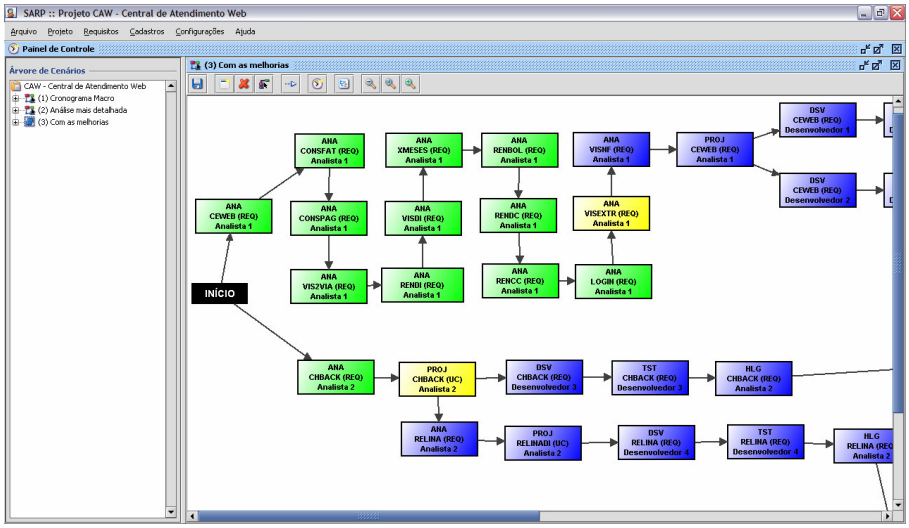


Figura 54: Inclusão dos novos itens no cronograma.

No momento em que este estudo de caso foi elaborado, a frente de trabalho *Central de Atendimento Web* encerrou a etapa de análise e iniciou a etapa de projeto. As necessidades a serem analisadas para contemplar a nova modalidade de cobrança ainda não foram definidas pelos usuários chaves. A modificação do relatório existente e a criação da nova interface de consulta de pagamentos também já foram analisadas e encontram-se agora na fase de projeto.

7. CONSIDERAÇÕES FINAIS

O desenvolvimento deste trabalho permitiu o aprofundamento do conhecimento a respeito da disciplina de gerenciamento de projeto segundo PMI, através das boas práticas propostas pelo instituto através do PMBOK. O estudo focou as áreas de conhecimento de gerenciamento de escopo e de tempo do projeto alvos das funcionalidades do SARP. Foi possível verificar, também, que modelos conceituais sobre a teoria dos grafos podem ser plenamente aplicáveis ao gerenciamento de projetos.

Em um segundo momento foi possível verificar estudos a respeito da volatilidade dos requisitos de *software* e os impactos que essas mudanças podem causar em projetos de desenvolvimento, caso não sejam devidamente controlados. Vimos que inúmero autores reforçam a idéia de que é necessário manter uma matriz de rastreabilidade entre os requisitos no intuito de garantir que as mudanças possam ser controladas e gerenciadas pelas equipes de desenvolvimento. A comparação realizada entre os trabalhos relacionados permitiu uma maior visibilidade de quais funcionalidades realmente eram significantes para o desenvolvimento deste trabalho.

O desenvolvimento da ferramenta permitiu a criação de uma base de dados que pode ser utilizada para centralizar os requisitos de um *software*, independente do projeto que está ou não em andamento. Vimos que o ciclo de vida de um sistema não corresponde, necessariamente, ao ciclo de vida do projeto que o concebeu. O ciclo de vida dos sistemas estende-se além dos limites dos projetos. Muitas vezes, novos projetos são elaborados sobre sistemas existentes e nesses casos uma documentação detalhada das funcionalidades da ferramenta pode ser muito útil e necessária.

A funcionalidade que o SARP disponibiliza para gerar diferentes cenários de alocação de recursos pode ser utilizada, tanto para determinar a forma de alocação das atividades do projeto, como servir de controle histórico de como o projeto estava planejado.

O SARP foi apresentado informalmente para um pequeno grupo de gerentes de projetos e esses profissionais identificaram que a interface gráfica de alocação das atividades e montagem do cronograma ficou simples e atraente aos olhos de quem lida com esses tipos de controles diariamente. Outra característica destacada por esses profissionais está ligada ao modelo de estimativa utilizado pela ferramenta para estimar o tempo necessário para o desenvolvimento de um caso de uso. Apesar de não terem sido realizados estudos aprofundados sobre esse aspecto, foi constado através dessas entrevistas informais que os tempos calculados pelo método não correspondem aos tempos de desenvolvimento aplicados no mercado. Mas como a ferramenta não obriga que os tempos calculados sejam efetivamente utilizados na elaboração do cronograma das atividades, esse aspecto acabou não gerando maiores discussões entre os profissionais.

7.1. Trabalhos Futuros

Com este trabalho foi possível abordar uma pequena porção do universo gerenciamento de projetos. Os temas relativos a essa área e a área de gerência de requisitos vão muito além dos assuntos abordados por esta pesquisa. Dessa forma, espera-se que o

SARP possa servir de inspiração para trabalhos futuros. Como o tempo de elaboração da monografia é limitado, inúmeras funcionalidades que foram pensadas para a ferramenta, não puderam ser desenvolvidas. Com isso, ficam algumas sugestões de melhorias que podem ser exploradas no SARP:

- Automatizar um padrão de nomenclatura para versionamento de requisitos e casos de uso, adotando práticas sugeridas pelo CMMI com a definição de *baselines*.
- Disponibilizar uma forma gráfica de visualizar a matriz de rastreabilidade, deixando-a mais amigável.
- Os dados que são inseridos no SARP e que permitem o acompanhamento do projeto podem ser cruzados no intuito de estabelecer uma série de métricas que podem ser utilizadas de entradas para novos projetos.
- Disponibilizar uma interface *web* para a ferramenta, visto que hoje ela é *desktop*.

REFERÊNCIAS

BITTNER, Kurt; SPENCER, Ian. **Use case modeling**. Boston: Addison-Wesley, 2003.

BROOKS, F.P. "No silver bullet", **Software, IEEE.** Vol. 20. N. 4, p. 10--19.

CARNEGIE MELLON UNIVERSITY. **Capability Maturity Model Integration** (CMMI), Version 1.1: CMMI for Software Engineering (CMMI-SW, V1.1) – Staged Representation. Pittsburg, 2002.

CHANG, C. & CHRISTENSEN, M. "A net practice for *software* project management", **Software, IEEE.** Vol. 16. N. 6, p. 80--89.

DALL'OGLIO, Pablo. **Uma Ferramenta para Gerenciamento de Requisitos em Projetos baseados em Extreme Programming**. Monografia de Graduação. Unisinos, 2006.

DEITEL, Harvey M. **Advanced java 2 platform**: how to program. Upper Sadler River: Prentice Hall, 2002.

GOLDBARG, Marco Cesar; LUNA, Henrique Pacca. **Otimização combinatória e programação linear**: modelos e algoritmos. Rio de Janeiro: Campus, 2000.

GRANDE, José Inácio; MARTINS, Luiz Eduardo G.. **SIGERAR: Uma Ferramenta para Gerenciamento de Requisitos**. Anais do WER06 - Workshop em Engenharia de Requisitos, Rio de Janeiro, RJ, Brasil, Julho 13-14, 2006, p. 75-83. Disponível em: http://wer.inf.puc-rio.br/WERpapers/artigos/artigos_WER06/degrande.pdf. Acesso em: abr. 2007.

GRIFFANTE, Giuliano B. **Uma Proposta MVC Para Aplicações Desktop**. Portal Java. Disponível em: http://www.portaljava.com/home/modules.php?name=Content&pa=showpage&pid=138. Acesso em: dez. 2006.

HUMPHREY, Watts S. **A discipline for software engineering**. Boston: Addison-Wesley, 2005.

KARNER, Gustav. **Resource Estimation for Objectory Projects**. 1993. Disponível em: http://www.bfpug.com.br/Artigos/UCP/Karner - Resource Estimation for Objectory Projects.doc. Acesso em: nov. 2006.

KERZNER, Harold. **Gestão de projetos: as melhores práticas.** 2. ed. Porto Alegre: Bookman, 2006.

LAM, W.; et al, "Managing requirements change using metrics and action planning", In: "**Proceedings of the Third European Conference on Software Maintenance and Reengineering, 1999**". p. 122--128.

KOTONYA, Gerald; SOMMERVILLE, Ian. **Requirements engineering**: processes and techniques. Chichester: John Wiley & Sons, 1998.

MARTINS, José C. C.; **Gerenciando Projetos de Desenvolvimento de Software com PMI, RUP e UML**. Rio de Janeiro: Brasport, 2006.

O'NEAL, J. & CARVER, D., "Analyzing the impact of changing requirements". In: "**IEEE International Conference on Software Maintenance, 2001. Proceedings.**" p. 190--195.

PMBOK Guide: **A Guide to the Project Management Body of Knowledge**; Project Management Institute; 2000. Disponível em http://eg*web*.mines.edu/eggn491. Acesso em: dez. 2006.

PMI. **A Guide to the Project Management Body of Knowledge**. Project Management Institute; 2000. Disponível em: http://eg*web*.mines.edu/eggn491. Acesso em: nov. 2006.

PMI. **Project management body of knowledge**. Project Management Institute; 2004. Disponível em: http://www.pmi.org. Acesso em: nov. 2006.

PREISS, Bruno R. **Estruturas de dados e algoritmos**: padrões de projetos orientados a objetos com java. Rio de Janeiro: Campus, 2000.

PRESSMAN, R. S. **Software engineering**: a practitioner's approach. 5. ed. Boston: McGraw-Hill, 2001.

SCHNEIDER, Geri; WINTERS, Jason P. **Applying use cases**: a practical guide. 2. ed. Reading: Addison-Wesley, 2001. 188 p.

VARGAS, Ricardo Viana. **Manual prático de plano de projeto**. 3. ed. Rio de Janeiro: Brasport, 2003.

VAVASSORI, Fabiane B.; **Metodologia para o Gerenciamento Distribuído de Projetos e Métrica de Software**. Dissertação de Mestrado. Universidade Federal De Santa Catarina, 2002. Disponível em: http://teses.eps.ufsc.br/defesa/pdf/4193.pdf. Acesso em: jan. 2007.

VAVASSORI, Fabiane B.; Silva, Júlia Marques Carvalho. Gemetrics *Web*: Ferramenta para Gerenciamento de Projetos Abordando Aspectos Temporais. In: **XIX Simpósio Brasileiro De Engenharia De Software (SBES'2005)**, 2005. Uberlândia. São Paulo. Disponível em: http://www.sbbd-sbes2005.ufu.br/arquivos/Gemetrics*Web*.pdf. Acesso em: jan. 2007.

REEL, J. (1999), "Critical success factors in *software* projects", **Software, IEEE**. Vol. 16. N. 3, p. 10--19.

TVETE, Bjarne (1999). "Introducing Efficient Requirements Management", In: **10th International Workshop on Database & Expert Systems Applications**, Florence, Italy: IEEE Computer Society, p. 370--375.