



UNIVERSIDADE ESTADUAL DO NORTE DO PARANÁ
CAMPUS LUIZ MENEGHEL

GUSTAVO DE OLIVEIRA SOARES

**APLICAÇÃO DAS BOAS PRÁTICAS DO PROCESSO
UNIFICADO NO DESENVOLVIMENTO DE SOFTWARE
LABORATORIAL PARA A UENP-CLM**

Bandeirantes

2012

GUSTAVO DE OLIVEIRA SOARES

**APLICAÇÃO DAS BOAS PRÁTICAS DO PROCESSO
UNIFICADO NO DESENVOLVIMENTO DE SOFTWARE
LABORATORIAL PARA A UENP-CLM**

Trabalho de Conclusão de Curso submetido
à Universidade Estadual do Norte do Paraná
– *Campus* Luiz Meneghel, como requisito
parcial para a obtenção do grau de Bacharel
em Sistemas de Informação.

Orientador: Prof. Me. Carlos Eduardo Ribeiro

Bandeirantes

2012

GUSTAVO DE OLIVEIRA SOARES

**APLICAÇÃO DAS BOAS PRÁTICAS DO PROCESSO
UNIFICADO NO DESENVOLVIMENTO DE SOFTWARE
LABORATORIAL PARA A UENP-CLM**

Trabalho de Conclusão de Curso submetido
à Universidade Estadual do Norte do Paraná
– *Campus* Luiz Meneghel, como requisito
parcial para a obtenção do grau de Bacharel
em Sistemas de Informação.

COMISSÃO EXAMINADORA

Prof. Me. Carlos Eduardo Ribeiro
UENP – *Campus* Luiz Meneghel

Prof. Me. Bruno Miguel Nogueira dos Santos
UENP – *Campus* Luiz Meneghel

Prof. Me. Daniela de Freitas G. Trindade
UENP – *Campus* Luiz Meneghel

Bandeirantes, 26 de Junho de 2012.

RESUMO

Com o aumento no volume de dados produzidos pelo “Laboratório de Análises de Leite e Derivados”, presente na UENP-CLM, foi identificada a necessidade de uma ferramenta para gerenciar e organizar tais dados, que contribua para a melhoria e agilidade nas operações realizadas no laboratório. Diante disto, este trabalho apresenta todo o processo de desenvolvimento de um software para cumprir tal necessidade. Por meio da aplicação das boas práticas do Processo Unificado, foram executadas as principais atividades e técnicas que esta abordagem considera como elementos essenciais no desenvolvimento de um software nas fases de iniciação, elaboração e construção. Com isso, obteve como resultado a documentação do sistema e o software SISLAL (Sistema Laboratorial de Análise de Leite), na qual após passar por uma avaliação pelos membros do laboratório foi verificado que o mesmo atende as necessidades atuais. Os resultados indicam que o uso de boas práticas de desenvolvimento foi fundamental para obter o sucesso na produção do software, uma vez que mais importante quanto objetivar um produto de qualidade, é estabelecer um processo de desenvolvimento de qualidade.

Palavras-chave: Processo Unificado, Software Laboratorial, Software Livre.

LISTA DE FIGURAS

Figura 1 Quem está criando os dados?.....	14
Figura 2 Modelo em Cascata.....	20
Figura 3 Modelo Incremental.....	20
Figura 4 Desenvolvimento Evolucionário.....	21
Figura 5 Desenvolvimento orientado ao reuso.....	22
Figura 6 Estrutura do UP.....	30
Figura 7 Ciclo de vida do UP.....	34
Figura 8 Diagrama de Casos de Uso.....	40
Figura 9 Diagrama de Classe.....	50
Figura 10 Diagrama de Atividade - Novo Membro.....	51
Figura 11 Diagrama de Comunicação - Novo Membro.....	52
Figura 12 Diagrama de Sequência – Novo Membro.....	52
Figura 14 Modelo de Dados.....	53
Figura 16 Exemplo Protótipo de Interface.....	54
Figura 17 Diagrama de Pacote: Modelo.....	56
Figura 18 Diagrama de Pacote: Controle.....	57
Figura 19 Diagrama de Pacote: Visão.....	58
Figura 20 Tela de Login.....	58
Figura 21 Cadastrar Membro.....	59
Figura 22 Gerenciar Laudos.....	59
Figura 23 Gerenciar Produtor.....	60
Figura 24 Laudo Gerado pelo Sistema.....	60
Figura 25 Gráfico de Evolução das Análises.....	61

LISTA DE QUADROS

Quadro 1 Tipos de Diagramas Oficiais da UML.....	24
Quadro 2 Problemas detectados: controle de laudos	36
Quadro 3 Problemas detectados: controle de análises	36
Quadro 4 Problemas detectados: controle de evolução	37
Quadro 5 Requisitos Funcionais do SISLAL	38
Quadro 6 Requisitos não funcionais do SISLAL	39
Quadro 7 Lista de Riscos	42
Quadro 8 Tarefas realizadas na iteração I1 da Fase de Iniciação	43
Quadro 9 Produtos Liberados na iteração I1 da Fase de Iniciação	43
Quadro 10 Tarefas realizadas na iteração E1 da Fase de Elaboração.....	44
Quadro 11 Produtos Liberados na iteração E1 da Fase de Elaboração.....	44
Quadro 12 Tarefas realizadas na iteração C1 da Fase de Construção	45
Quadro 13 Produtos Liberados na iteração C1 da Fase de Construção	45
Quadro 14 Tarefas realizadas na iteração C2 da Fase de Construção	46
Quadro 15 Produtos Liberados na iteração C2 da Fase de Construção	46
Quadro 16 Tarefas realizadas na iteração C3 da Fase de Construção	46
Quadro 17 Produtos Liberados na iteração C3 da Fase de Construção	47
Quadro 18 Tarefas realizadas na iteração T1 da Fase de Transição.....	47
Quadro 19 Produtos Liberados na iteração T1 da Fase de Transição.....	47
Quadro 20 SubFluxo Cadastrar Membro	49

LISTA DE ABREVIATURAS E SIGLAS

CASE – *Computer-Aided Software Engineering*

CMETL – Centro Mesorregional de Excelência em Tecnologia do Leite

CNPq – Conselho Nacional de Desenvolvimento Científico e Tecnológico

IDC – *International Data Corporation*

IDE - *Integrated Development Environment*

SISLAL – Sistema Laboratorial de Análise de Leite

TI – Tecnologia da Informação

UC – *Use Case*

UENP-CLM – Universidade Estadual do Norte do Paraná – Campus Luiz Meneghel

UML – *Unified Modeling Language*

UP – *Unified Process*

SUMÁRIO

1.	Introdução	10
1.1	Objetivos	11
1.1.1	Objetivo Geral	11
1.2	Justificativa	12
1.3	Metodologia	13
1.4	Organização do Trabalho	13
2.	Fundamentação teórica	14
2.1	Automação	14
2.2	Laboratório de Análises de Leite e Derivados da UENP-CLM.....	16
2.3	Processos De Desenvolvimento De Software	17
2.3.1	Modelo em Cascata	19
2.3.2	Modelo Incremental.....	20
2.3.3	Desenvolvimento Evolucionário	21
2.3.4	Desenvolvimento orientado ao reuso	22
2.4	UML	23
2.5	Processo Unificado	25
2.5.1	Princípios Chave do UP.....	26
2.5.2	Elementos Essenciais do UP	28
2.5.3	Estrutura.....	30
3.	Desenvolvimento	35
3.1	Ferramentas Utilizadas	35
3.2	Fase de Iniciação	35
3.2.1	Visão do Sistema	36
3.2.2	Elicitação de requisitos	37
3.2.3	Casos de Uso	39
3.2.4	Identificar Riscos.....	40
3.2.5	Planejar Iterações	42
3.2.6	Cronograma.....	48
3.3	Fase de Elaboração	48

3.3.1	Detalhamento de Casos de Uso.....	48
3.3.2	Classes do Sistema	49
3.3.3	Modelo de Design.....	50
3.3.4	Modelo de Dados.....	53
3.3.5	Protótipo de Interface	54
3.4	Fase de Construção.....	54
3.4.1	Tecnologias Aplicadas.....	55
3.4.2	Arquitetura do Sistema	56
3.4.3	O sistema.....	58
4.	Considerações finais.....	62
	Referências	64
	APÊNDICE A – Plano de Visão.....	67
	APÊNDICE B – Plano do Projeto	72
	APÊNDICE C – Modelo de Casos De Uso.....	76
	APÊNDICE D – Modelo de Design.....	87
	ANEXO A – Entrevista: Elicitação de Requisitos.....	93

1. INTRODUÇÃO

A Tecnologia da Informação (TI) está presente no cotidiano de praticamente todas as pessoas. Cada vez mais produtos e serviços incorporam, de algum modo, tecnologias com o intuito de promover maior comodidade, agilidade e segurança para os usuários. A aplicação da TI está por toda parte: nos negócios, no transporte, na saúde, na educação, na comunicação, entre outros.

À medida que a utilização e por consequência a importância da TI cresceu, houve aumento também nos esforços para garantir a qualidade dos produtos de *software*. Nesse sentido, muitos estudos vêm sendo realizados e indicam cada vez mais que a qualidade do produto de *software* está fortemente relacionada com a qualidade do processo utilizado na sua construção (Perreira, 2005).

O desenvolvimento de um *software* é uma tarefa árdua, cujo sucesso é determinado por inúmeros fatores que ocorrem durante o processo e é afetado diretamente pelo planejamento elaborado, pelas pessoas envolvidas, pelas tecnologias utilizadas, pelo custo e tempo.

É comum encontrar *software* que possui baixa qualidade (não atende as necessidades do cliente), custo e prazo extrapolados. Isso ocorre, normalmente, devido à falhas no projeto do *software* ou até mesmo devido a inexistência de um projeto.

Com a utilização de boas práticas de desenvolvimento estes problemas podem ser evitados e/ou minimizados. Segundo Leite (2010), as empresas passaram a atentar para o fato de que, para sanar as falhas do projeto de *software* e alcançar competitividade pela qualidade deveriam investir na melhoria da qualidade, tanto dos produtos, quanto dos processos de produção e distribuição de *software*.

Deste modo, foi identificada a necessidade de implantar metodologias no processo de desenvolvimento de *software*. Atualmente um dos mais conhecidos no mercado é o Processo Unificado, este apresenta uma solução disciplinada de tarefas e técnicas que aumenta a produtividade no desenvolvimento. Destaca-se como características ser iterativo e incremental - consiste na divisão do projeto em miniprojetos; e ser personalizado a ponto de poder ser utilizado tanto em projetos de grande porte quanto de pequeno porte.

Além disso, o Processo Unificado propicia suporte para a criação de uma documentação rica que auxilia tanto no decorrer do desenvolvimento quanto nas futuras manutenções.

Neste contexto, o presente trabalho visa aplicar as boas práticas do Processo Unificado no desenvolvimento de um *software* laboratorial, que irá automatizar a rotina do Laboratório de Análises de Leite e Derivados da Universidade Estadual do Norte do Paraná – Campus Luiz Meneghel (UENP-CLM).

O Laboratório de Análises de Leite e Derivados da UENP-CLM, por meio de seus membros bolsistas e estagiários, tem gerado uma grande quantidade de documentos sobre laudos, amostras, análises e produtores associados. Contudo, há uma grande deficiência na organização, armazenamento e no acesso a esses documentos, uma vez que o controle é feito de forma tradicional - utilizando apenas anotações, para gerenciar os resultados das análises realizadas e laudos produzidos, o que ocasiona em retardo de trabalho.

Nesse sentido, foi identificada a necessidade de corrigir estes problemas que podem ser supridos por meio da implantação de um *software* que automatize o controle sobre os documentos gerados e auxilie nos processos de análise e tomada de decisões.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

O objetivo geral deste trabalho é aplicar as boas práticas do Processo Unificado no desenvolvimento de um *software* laboratorial, que irá automatizar a rotina do Laboratório de Análises de Leite e Derivados da UENP-CLM. Pretende-se com o *software* automatizar o controle sobre os documentos gerados, aperfeiçoar a emissão de resultados, monitorar as análises realizadas e auxiliar no processo de análise e tomada de decisões.

1.2 JUSTIFICATIVA

A aplicação da Tecnologia da Informação para a solução de problemas e automação de tarefas não é um processo simples, o *software* – um dos principais meios adotados na automação de serviços tem exigido cada vez mais dos desenvolvedores, que precisam aliar alto desempenho, qualidade do produto, baixo custo e cronograma reduzido.

É comum encontrar *softwares* que não atendam as reais necessidades do cliente, que apresentam erros, problemas de usabilidade e cujo custo e tempo de desenvolvimento foram extrapolados. Isso ocorre, normalmente, devido a falta de metodologias na elaboração e construção do *software*.

Segundo Máximo (2010), a adoção de metodologias é de vital importância para o desenvolvimento de sistemas, pois oferece subsídios para a produção de *software* de alta qualidade. Além da geração de uma documentação rica, proporciona em qualquer fase do projeto, uma visão do estado atual, sendo possível o acompanhamento produtivo de todo o desenvolvimento.

Neste contexto, este trabalho justifica-se pela importância da utilização de metodologias no processo de desenvolvimento de *software*, com o intuito de obter um produto de qualidade, que atenda às necessidades dos envolvidos no prazo determinado.

O *software* desenvolvido irá suprir os problemas encontrados no Laboratório de Análises de Leite e Derivados da UENP-CLM, no qual o controle de documentos gerados sobre laudos, análises e produtores é realizado na forma de anotações, o que gera ineficiência na organização, no armazenamento e no acesso aos documentos, além de retardar o processo de análise dos dados.

Desta forma, foi identificada a necessidade de implantar um *software* que tenha como funções automatizar o controle sobre os documentos citados acima, aperfeiçoar a emissão de relatórios (o produtor não necessitará locomover-se até o laboratório para obtenção de resultados), monitorar as análises realizadas e auxiliar no processo de tomada de decisões.

1.3 METODOLOGIA

O trabalho apresentado neste documento foi realizado por meio de várias etapas:

Etapa 1: estudo sobre o processo de negócio do Laboratório de Análise de Leite e Derivados da UENP-CLM, o que definiu a visão sobre as tarefas realizadas no laboratório e seus maiores problemas. Este processo foi concebido por meio de entrevistas que foram realizadas em conjunto com membros do laboratório.

Etapa 2: estudo sobre o processo de desenvolvimento de software e apresentação do Processo Unificado. Inicialmente foram expostas as principais características de algumas metodologias, como o modelo em cascata, o modelo incremental, o desenvolvimento evolucionário e o desenvolvimento orientado ao reuso. Logo após é detalhado o processo unificado, que foi a metodologia adotada para a construção do software.

Etapa 3: aplicação das metodologias do processo unificado para o desenvolvimento do projeto e implementação do software, explorando as atividades principais que foram realizadas em cada fase do ciclo de vida do software.

Etapa 4: Apresentação dos resultados por meio dos artefatos produzidos e telas referentes ao software desenvolvido. A avaliação dos resultados foi por meio de uma apresentação do software para os membros do laboratório, que na ocasião aprovaram o software conhecendo que o mesmo atende as necessidades atuais do laboratório.

1.4 ORGANIZAÇÃO DO TRABALHO

O restante do trabalho está organizado na seguinte forma. O capítulo 2 fornece uma visão sobre os processos de desenvolvimento de software e apresenta o Laboratório de Análise de Leite e Derivados presente na Universidade Estadual do Norte do Paraná – Campus Luiz Meneghel. O capítulo 3 apresenta a aplicação da metodologia do Processo Unificado no desenvolvimento do sistema. Finalmente, no capítulo 4 apresenta-se as considerações finais.

2. FUNDAMENTAÇÃO TEÓRICA

Este capítulo aborda o tema sobre a automação nas organizações, apresenta o Laboratório de Análises de Leite e Derivados da UENP-CLM e conceitua os processos de desenvolvimento de *software*, a UML (*Unified Modeling Language*) e o UP (Processo Unificado).

2.1 AUTOMAÇÃO

Uma das grandes dificuldades encontradas pelos gestores atualmente é gerenciar um grande volume de dados que são produzidos, o que dificulta o armazenamento, o acesso e a tomada de decisão sobre esses dados.

Segundo uma pesquisa encomendada pela Avanade – empresa provedora de serviços de TI – em novembro de 2010, 54% dos executivos afirmam estar sobrecarregados pela quantidade de dados que suas empresas gerenciam. Esses dados são provenientes de diversos ramos da empresa, como clientes, fornecedores, gestão, parceiros, entre outros, como pode ser visto na figura 1.



Figura 1 Quem está criando os dados?
Fonte: AVANADE (2010)

Ainda, segundo a pesquisa (AVANADE, 2010), 46% dos executivos relataram ter tomado alguma decisão equivocada devido a dados imprecisos ou desatualizados; a perspectiva para os próximos anos não é diferente, estima-se que o volume de dados nas empresas deve crescer 650% em cinco anos. Segundo a IDC (*International Data Corporation*) o volume mundial de dados dobra a cada 18 meses.

Deste modo, fica evidente que as organizações precisam estar preparadas para lidar com esse volume de dados, para isso necessitam começar a usufruir de tecnologias acessíveis para simplificar e aumentar a qualidade do serviço prestado.

No caso dos laboratórios de análises, o volume de dados é significativamente grande, aliado a isto está o enxuto corpo técnico de trabalho e o gerenciamento de processos e funções realizado de forma manual. Portanto, fica estabelecido um ambiente perigoso para a confiabilidade dos resultados gerados que carece de mecanismos de automação para diminuir ou minimizar o impacto causado por ocorrência de erros.

A utilização de sistemas pode se tornar um grande aliado, o uso destes sistemas na automação do controle e execução de tarefas, traz benefícios como melhora na produtividade e eficiência, a praticidade e a integridade do serviço oferecido.

Segundo Kishimoto e Moraes (2008), a utilização de sistemas de informação nos laboratórios de análises permite o aumento na quantidade de informação disponibilizada, mas, principalmente na velocidade em disponibilizar tais informações.

Um dos pontos críticos dos laboratórios de análise é em referência a medição da evolução e da qualidade das amostras que foram analisadas, para isto é crucial a importância da manipulação de dados consistentes para agilidade e rapidez na geração de resultado.

De acordo com Mugnol e Ferraz (2006), a automação nos laboratórios de análise pode ser visualizada como um conjunto que engloba as atividades de gerenciamento de processos envolvidos no controle de equipamento e instrumentos laboratoriais, controle de amostras e processos analíticos. Além disso, o controle de laudos, de clientes e aperfeiçoamento na emissão de resultados são outros fatores que pertencem a automação nestes laboratórios.

No próximo tópico é apresentado o Laboratório de Análises de Leite e Derivados da UENP-CLM, que teve seus processos de negócio automatizado com a realização deste trabalho.

2.2 LABORATÓRIO DE ANÁLISES DE LEITE E DERIVADOS DA UENP-CLM

O laboratório de Análises de Leite e Derivados da Universidade Estadual do Norte do Paraná – Campus Luiz Meneghel, iniciou sua atividade em Outubro de 2008 por meio de um projeto criado pelo Prof. Dr. Eder Paulo Fagan.

Segundo o coordenador do projeto, o objetivo do laboratório é difundir o desenvolvimento de inovações tecnológicas para a cadeia produtiva do leite da mesorregião do Norte pioneiro do Paraná.

Para isso, o laboratório visa fornecer assistência técnica aos produtores, associações e micro usinas de leite visando aumentar a capacitação dos técnicos e produtores de leite, como também melhorar a qualidade de vida das propriedades em regime de agricultura familiar.

Em uma pesquisa realizada pela EMBRAPA (Empresa Brasileira de Pesquisa Agropecuária) em 2010, conclui-se que o Brasil é o quinto maior produtor de leite do mundo, sendo que o Paraná está entre os três maiores estados produtores de leite do Brasil. Com esses dados pode-se observar que o agronegócio do leite é uma atividade de grande importância para o desenvolvimento de diversas regiões brasileiras e para o continuísmo de seu crescimento produtivo o atestado de sua qualidade é de vital importância neste processo.

A avaliação da qualidade do leite que é realizada no laboratório, permite aos produtores melhor capacitação dos processos de produção a fim de melhorar a qualidade do produto, isso graças a assistência prestada aos produtores.

Segundo Auad *et al.*, (2010) o leite pode ser avaliado de acordo com sua qualidade composicional (gordura, proteína, lactose, extrato seco desengordurado, sólidos totais) e por sua qualidade higiênico sanitária (contagem bacteriana total, contagem de células somáticas e resíduos químicos contaminantes).

Atualmente, o projeto conta com um corpo técnico de dezessete estagiários, um bolsista CNPq e um recém-formado com aperfeiçoamento profissional. Atende a 24 propriedades, duas associações e três laticínios em 11 municípios da região: Andirá, Abatiá, Bandeirantes, Barra do Jacaré, Cornélio Procópio, Jaboti, Jacarezinho, Leópolis, Ribeirão do Pinhal, Santa Amélia e Santo Antônio da Platina.

Recentemente, em maio deste presente ano, foi inaugurado na UENP-CLM o Centro Mesorregional de Excelência em Tecnologia do Leite (CMETL), na qual a universidade foi contemplada com um dos oito CMETL que serão instalados no Paraná.

Agora, contando com uma excelente estrutura, esse centro consolidará ainda mais o laboratório, que atenderá os municípios da Amunop (Associação dos Municípios do Norte do Paraná) e Amunorpi (Associação dos Municípios do Norte Pioneiro) com mais desenvoltura e qualidade na prestação de serviço. A tendência é aumentar o número de propriedades, associações e laticínios associados.

No próximo tópico são abordadas as metodologias adotadas para o desenvolvimento do *software* laboratorial que atenderá as necessidades do Laboratório de Análises de Leite e Derivados da UENP-CLM.

2.3 PROCESSOS DE DESENVOLVIMENTO DE SOFTWARE

Segundo Pressman (2006), um processo de desenvolvimento de *software*, é uma coleção de padrões que definem um conjunto de atividades, ações, tarefas de trabalho, produtos de trabalho e/ou comportamentos relacionados necessários ao desenvolvimento de *softwares*. Ainda, segundo Máximo (2010):

[...] Processo de Desenvolvimento de Software (PDS) é um dos principais mecanismos para se obter software de qualidade e cumprir corretamente os contratos de desenvolvimento. Consiste na adoção de um ciclo de vida de sistemas, padronizando procedimentos, atividades e técnicas, utilizando ferramentas e gerando produtos. A adoção de um PDS é de vital importância para o desenvolvimento de sistemas, pois trata-se da organização do trabalho que está sendo desenvolvido. Além da geração de uma documentação rica, que auxiliará tanto no decorrer do desenvolvimento quanto nas futuras manutenções, o PDS proporciona, em qualquer fase do projeto, uma visão do estado atual; oferece subsídios para o acompanhamento produtivo de todo o desenvolvimento [...].

Como pode ser observado é crucial a utilização de padrões de desenvolvimento em projetos de *softwares*, estes padrões oferecem subsídios para obter um produto de qualidade, controle da situação do projeto por parte do gerente de projeto (auxiliando em situações como estimativa de custo e tempo de entrega no prazo) e apoio aos desenvolvedores no fornecimento de documentação rica.

Além disso, a utilização de processos de desenvolvimento de *software* minimiza o impacto causado por erros no projeto e deve facilitar a mudança e acréscimo de requisitos, como destaca Souza (2009), indicando que os eventos mais comuns que acontece em um projeto de *software* é o crescimento de requisitos e a contínua mudança no conteúdo da informação.

Além dos benefícios já mencionados, outra importante característica que os padrões de desenvolvimento possuem é sua capacidade de customização, sendo que, podem ser utilizado em projeto de pequeno, média ou larga escala. Evidentemente, nos projetos pequenos não há necessidade de realizar todas as atividades executadas em grandes projetos, e sim apenas atividades que satisfaçam suas necessidades atuais.

Como existem diversos processos de desenvolvimento disponíveis, tem-se a necessidade de escolher qual processo se encaixa melhor na realidade da organização. Tyrrel (2000, apud PERREIRA, 2005), destaca um conjunto de objetivos que os processos de desenvolvimento devem atender:

- Efetividade: os processos devem ajudar a determinar as necessidades do cliente e verificar se o que foi produzido satisfaz o cliente;
- Manutenibilidade: o processo deve ser capaz de expor a maneira de pensar de projetistas e programadores de forma que suas intenções sejam claras. Assim, torna-se fácil encontrar e reparar falhas no produto;
- Previsibilidade: o processo deve ajudar a prever quanto tempo será necessário para o desenvolvimento de cada parte do produto;
- Repetível: é importante que o processo seja criado de forma a poder ser reutilizado em vários projetos;

- Qualidade: o processo deve permitir engenheiros de *software* assegurar um produto de qualidade elevada. Para isso, o processo deve fornecer uma ligação entre os desejos do cliente e o produto de um desenvolvedor;
- Melhoria: o processo deve ser constantemente melhorado. Dessa forma, o próprio processo deve permitir a identificação de pontos de melhoria; e
- Rastreabilidade: o processo deve permitir que a gerência, os desenvolvedores e o cliente sigam o status do projeto.

Além dos objetivos citados acima, Sommerville (2003) destaca um conjunto de atividades que um processo de desenvolvimento deve apresentar: especificação de *software* (elicitação de requisitos e restrições do *software*), projeto e implementação (codificação que atenda a especificação), validação (verificação se atende as necessidades do cliente) e evolução (evolução para atender as necessidades mutáveis do cliente).

Conforme os autores Sommerville e Pressman, os principais processos de desenvolvimento de *software*, são:

- Modelo em Cascata;
- Modelo Incremental;
- Desenvolvimento Evolucionário;
- Desenvolvimento orientado ao reuso; e
- Processo Unificado;

Uma breve descrição de cada modelo é apresentada a seguir.

2.3.1 Modelo em Cascata

O modelo em cascata ou ciclo de vida de *software* é o paradigma mais antigo da engenharia de *software*, foi idealizado em 1970 por Royce. Este modelo, conforme ilustrado na figura 2, oferece uma abordagem sistemática e sequencial para o desenvolvimento de *software* (PRESSMAN, 2006).

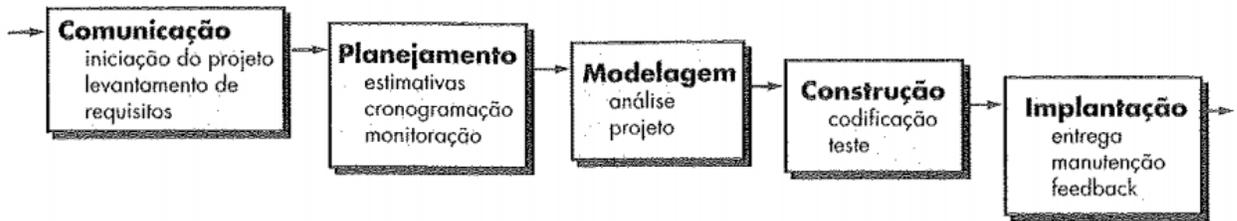


Figura 2 Modelo em Cascata
Fonte: Pressman (2006)

Neste modelo, ao final de cada fase é realizada uma validação, e só é permitido o avanço para a próxima fase se a precedente foi concluída (aprovada na validação).

Segundo Sommerville (2003), o problema do modelo em cascata é ser inflexível quanto a divisão do projeto em estágios distintos. Os requisitos são levantados no estágio inicial do processo, o que dificulta fazer alterações nos requisitos com o projeto já em andamento. Portanto, o modelo em cascata deve ser usado somente quando os requisitos forem bem compreendidos.

Ao completar todas as fases tem como resultado o *software*.

2.3.2 Modelo Incremental

Segundo Mills (1980, *apud* SOMMERVILLE, 2003), a abordagem do desenvolvimento incremental (figura 3) é um meio de reduzir o 'retrabalho' no processo de desenvolvimento e de proporcionar aos clientes algumas oportunidades de adiar decisões sobre seus requisitos detalhados, até que eles tenham alguma experiência com o sistema.

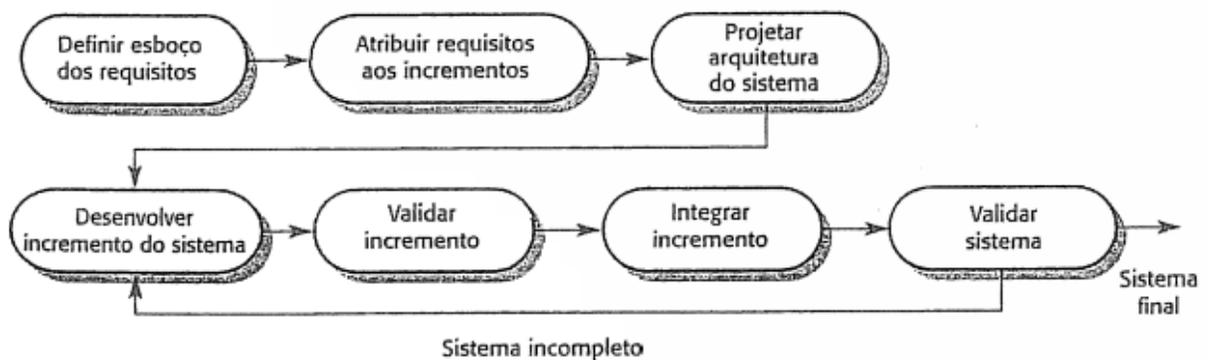


Figura 3 Modelo Incremental
Fonte: Sommerville (2003)

No processo de desenvolvimento incremental, os clientes esboçam as funções a serem fornecidas pelo sistema, identificando quais são mais importantes. Em seguida é definida uma série de estágios de entrega, sendo que cada estágio fornece um subconjunto de funcionalidades do sistema (SOMMERVILLE, 2003).

Ao contrário do modelo em cascata na qual o ciclo é realizado apenas uma vez, neste modelo incremental são repetidos várias vezes o ciclo, sendo que cada ciclo equivale a uma versão do sistema.

2.3.3 Desenvolvimento Evolucionário

De acordo com Sommerville (2003), o desenvolvimento evolucionário, ilustrado na figura 4, tem como base desenvolver e implementar um produto inicial e expor o resultado para cliente. A partir do *feedback* do cliente o produto é aprimorado por meio de inúmeras versões até que o produto esperado tenha sido desenvolvido. Neste modelo, as atividades de especificação, desenvolvimento e validação são realizadas concorrentemente.

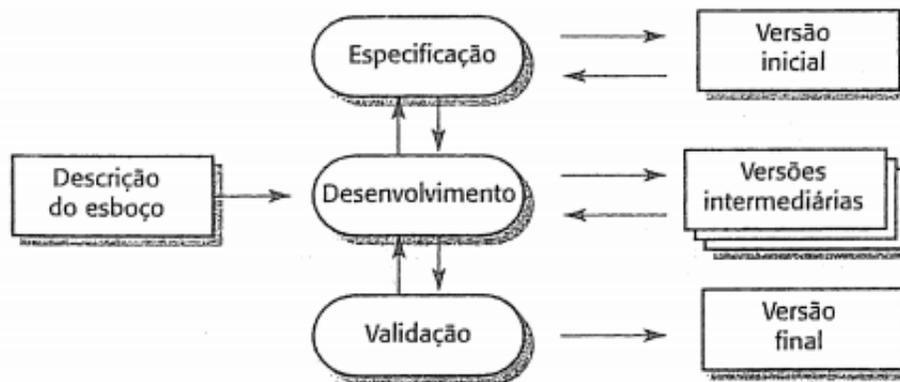


Figura 4 Desenvolvimento Evolucionário
Fonte: Sommerville (2003)

Segundo Pedri (2008), o problema deste modelo é que o processo não é visível, pois o desenvolvimento ocorre de maneira rápida, não sendo viável produzir documentos que retratem cada versão do *software*. Com isso, mudanças constantes durante o desenvolvimento tendem a abalar a estrutura do *software*.

2.3.4 Desenvolvimento orientado ao reuso

O desenvolvimento orientado ao reuso, ilustrado na figura 5, dispõe de uma larga base de componentes de *software* reutilizáveis, que podem ser acessados para a integração de componentes (SOMMERVILLE, 2003). Isso ocorre normalmente quando o desenvolvedor conhece projetos ou códigos similares e os utiliza para solução do problema atual.

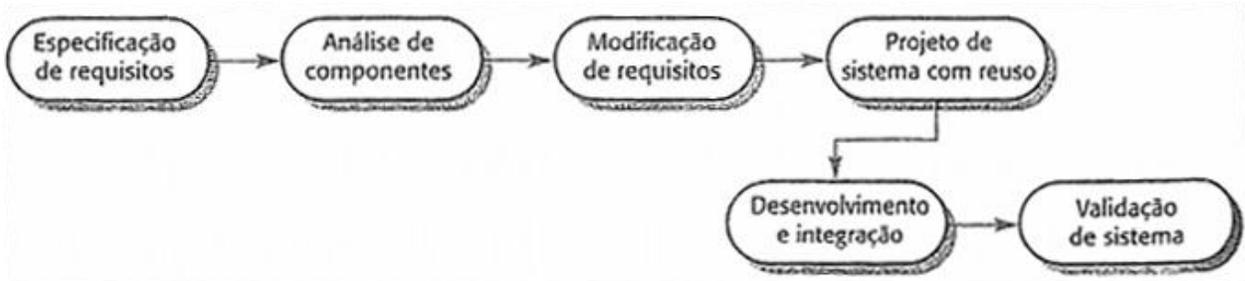


Figura 5 Desenvolvimento orientado ao reuso
Fonte: Sommerville (2003)

Após a atividade de especificação de requisitos, é realizada uma busca por componentes implementados em outros projetos, caso obtiver sucesso na busca, é realizado adaptações no código e no componente para a integração entre ambos (Cavalheiro, 2007).

O modelo orientado ao reuso tem a vantagem óbvia de reduzir a quantidade de código a ser desenvolvido, contudo, as adequações nos requisitos são inevitáveis, e isso pode resultar em um sistema que não atenda as necessidades do usuário (SOMMERVILLE, 2007).

Para a realização deste trabalho, o modelo escolhido foi o processo unificado e por isso receberá maior atenção no decorrer do trabalho. Assim como os outros modelos, o processo unificado está associado a notação UML (*Unified Modeling Language*), deste modo uma breve apresentação é mostrada a seguir.

2.4 UML

A UML (*Unified Modeling Language*) é uma linguagem de modelagem utilizada para especificação, documentação, visualização e desenvolvimento de sistemas orientado a objeto (BOOCH et. al, 2005).

Segundo Pressman (2002), a UML é o resultado da combinação das melhores características de métodos individuais de análise e projeto orientados a objetos num método unificado.

A UML tem como objetivo modelar sistemas de forma correta e consistente com o auxílio dos conceitos da orientação a objeto, tornando real a possibilidade de métodos conceituais também serem executáveis (MAKINO, 2009).

Furlan (1998, apud Makino, 2009) destaca os três principais benefícios da UML:

- Visualização: os relacionamentos existentes entre os diversos componentes da aplicação podem se visualizados de forma a antever o produto final;
- Gerenciamento de complexidade: cada aspecto do sistema é desenhado à parte em um modelo específico para que se possa estudar e compreender a estrutura e o comportamento, bem como identificar oportunidades de reutilização de componentes;
- Comunicação: por meio da utilização de símbolos e padrões, torna-se possível uma comunicação direta e não ambígua entre os participantes do projeto em relação aos detalhes de comportamento do sistema.

Vale destacar que a UML é apenas uma linguagem, deste modo, é somente uma parte de um método para desenvolvimento de *software*. Segundo Fowler (2005), a UML diz como expressar um projeto orientado a objeto, enquanto o modelo de desenvolvimento foca os resultados do processo.

Portanto, para alcançar um projeto de *software* eficiente, deve-se utilizar a UML em conjunto com um modelo de desenvolvimento (BOOCH et. al, 2005), e seu funcionamento independe do modelo adotado.

A UML é utilizada nas fases de iniciação e elaboração em um processo de desenvolvimento de *software* e oferece a elas diagramas que permitem descrever todas as visões do sistema, esses diagramas são listados no quadro 1.

DIAGRAMA	OBJETIVO
Atividade	Comportamento procedimental e paralelo.
Casos de uso	Como os usuários interagem com o sistema.
Classe	Classe, característica e relacionamentos.
Componentes	Estrutura e conexão de componentes.
Comunicação	Interação entre objetos; ênfase nas ligações.
Distribuição	Distribuição de artefatos nos nós.
Estado	Como os eventos alteram um objeto no decorrer de sua vida.
Estruturas compostas	Decomposição de uma classe em tempo de execução.
Objeto	Exemplo de configurações de instâncias.
Pacote	Estrutura hierárquica em tempo de compilação.
Sequência	Interação entre objetos; ênfase na sequência.
Sincronismo	Interação entre objetos; ênfase no sincronismo.
Visão Geral da Interação	Mistura de diagrama de sequência e de atividade.

Quadro 1 Tipos de Diagramas Oficiais da UML
 Fonte Adaptado: Fowler, 2005.

Como pode ser observado no quadro 1, a UML possui treze diagramas, entretanto, não é necessário a utilização de todos esses diagramas ao esboçar um *software*, e sim apenas os necessários para atender a demanda atual.

A UML oferece um poderoso suporte para esboçar o *software*, mas para isso é crucial saber quais componentes devem ser utilizados em determinada atividade. Desta forma, Fowler (2005) destaca que a UML pode ser aplicada nos seguintes aspectos:

- Análise de requisitos: durante esta atividade procura descobrir o que os usuários e clientes de um produto de *software* querem que o sistema faça. Para isso, pode-se utilizar diagramas de casos de uso, classes, atividades e estados.
- Projeto: durante o desenvolvimento de um projeto com a utilização de diagramas mais técnicos, como sequência, comunicação, classe, pacote, distribuição e estado.

- Documentação: neste quesito aplica-se a UML na atividade de engenharia reversa, isto é, documentar um *software* já construído.

Como já foi dito, a UML precisa ser aplicada em conjunto com um processo de desenvolvimento de *software*, portanto, o próximo tópico aborda o Processo Unificado, processo de desenvolvimento utilizado neste trabalho.

2.5 PROCESSO UNIFICADO

O Processo Unificado (*Unified Process* - UP) é um processo de engenharia de *software* que apresenta uma abordagem disciplinada para a atribuição de tarefas e de responsabilidades dentro de uma organização de desenvolvimento. (RUP, 2006).

O UP captura muitas das melhores práticas do desenvolvimento de *software* moderno, de forma que possam ser adaptadas para uma grande variedade de projetos e de organizações (RUP, 2006).

A partir da versão sete do UP lançada em 2006, o termo “melhores práticas” foi substituído por “princípios chaves para o desenvolvimento orientados a negócios”, entretanto há autores que continuam utilizando a terminologia antiga.

De acordo com Martins (2007, apud Makino 2009), a utilização do processo unificado possibilita:

- Atacar os riscos antecipadamente e continuamente;
- Certificar-se de entregar algo de valor ao cliente;
- Focar no *software* executável;
- Acomodar mudanças;
- Liberar um executável da arquitetura antecipadamente;
- Construir o sistema com componentes; e
- Trabalhar junto como uma equipe;

Desta forma, observa-se que o UP absorve as melhores características de cada modelo convencional, além de incorporar princípios de desenvolvimento ágil.

De acordo com Souza (2002), o RUP é baseado em componentes, na qual a aplicação a ser construída será formada por componentes interconectados por meio de

interfaces bem definidas. Este mesmo autor destaca as principais características do RUP:

- Dirigido a casos de uso: o processo de desenvolvimento segue um fluxo de ações para a realização de casos de uso. Desta forma, os casos de uso são especificados, projetados e no fim, é a base para a construção de casos de teste.
- Centrado na arquitetura: fornece *templates* (modelos) para projetar, desenvolver e validar a arquitetura. Conhecido por proporcionar uma visão do projeto como um todo, que torna visível as suas características mais importantes.
- Iterativo e incremental: o projeto pode ser dividido em pequenos ciclos ou miniprojetos. Cada miniprojeto é uma iteração que resulta em um incremento. Iterações referem-se a passos no fluxo de desenvolvimento, e incrementos a evoluções do produto.

Nos próximos tópicos são apresentados os conceitos sobre os princípios chave, os elementos essenciais e a estrutura do UP.

2.5.1 Princípios Chave do UP

O UP possui seis princípios chave, de acordo com o RUP (2006) que caracterizam as boas práticas de indústria na criação, implementação e evolução de sistemas. A seguir é exposto sucintamente cada princípio:

Adaptar o processo

Neste princípio o foco é dimensionar corretamente o processo para necessidades do projeto de acordo com o tamanho e distribuição da equipe, complexidade do sistema e a fase atual do projeto. O uso adicional de processo, como por exemplo, uso de mais artefatos do que o necessário pode ser prejudicial ao projeto, pois aumenta sua complexidade (RUP, 2006).

Equilibrar as prioridades do investidor

Este princípio expõe a importância de estabelecer o equilíbrio das necessidades conflitantes do cliente e também da reutilização de recursos com essas necessidades. Para isto, é necessário capturar e gerenciar os requisitos de forma efetiva, desta forma o cliente tem papel fundamental neste processo (RUP, 2006).

Por exemplo, o cliente solicita um *software* que atenda a todas suas necessidades, mas que possua custo e prazo reduzidos. A solução para isto é o uso da reutilização de componente que permite a redução de custo e tempo.

Trabalhar em conjunto com equipes

Este princípio destaca a importância da comunicação entre os membros do projeto com o intuito de propiciar um ambiente de colaboração, e por consequência melhora na produtividade da equipe (RUP, 2006).

Demonstrar o valor iterativamente

De acordo com o RUP (2006), este princípio exalta os benefícios adquiridos com o uso do desenvolvimento iterativo. A adoção do desenvolvimento iterativo, que é a divisão do projeto em vários ciclos, torna possível acomodar com maior facilidade as alterações, fornece *feedback* ao cliente e até mesmo entre os membros do projeto – ao final de cada ciclo é obtido uma versão do *software* e reduz riscos iniciais (PRESSMAN, 2006).

Elevar o nível de abstração

Este princípio destaca o uso de ferramentas, estruturas e linguagens com o objetivo de elevar o nível de abstração para diminuir o nível de complexidade, implicando em melhor compreensão entre os membros do projeto e aumento na produtividade (RUP, 2006).

A modelagem visual é um exemplo que pode ser aplicado neste caso, mais precisamente a UML (mais utilizada) responsável por esboçar o *software* por meio de diagramas que possibilita melhor compreensão do problema.

Focalizar continuamente na qualidade

Este princípio enfatiza a obtenção da qualidade, destaca que a utilização do desenvolvimento iterativo auxilia neste processo, que deve ser medido não somente o *software*, mas sim todo o ciclo de desenvolvimento (RUP, 2006).

2.5.2 Elementos Essenciais do UP

O Processo Unificado oferece dez elementos essenciais que devem ser seguidos para alcançar o equilíbrio entre a entrega de um *software* de qualidade e a rapidez nessa entrega (RUP, 2006). São eles:

Desenvolver uma Visão

Segundo Anunciação (2009), o objetivo deste documento é desenvolver uma visão compartilhada que capture as necessidades reais das partes interessadas.

O documento visão fornece informações para a aprovação do projeto e está, portanto, relacionado ao caso de negócio, além de funcionar como base na validação de decisões futuras (RUP, 2006).

Gerenciar o Plano

O objetivo deste documento é gerenciar o escopo, monitorar e controlar o projeto. Constam informações sobre organização do projeto, planejamento, orçamento, recursos, avaliação, teste, entre outros (RUP, 2006).

Mitigar Riscos

É crucial para a saúde do projeto a identificação de possíveis erros que podem ocorrer durante o projeto. Para isto, é projetada uma lista com possíveis riscos acompanhados de um plano de contingência - caso ocorra o problema, este plano é executado para solução ou minimizar o impacto causado (RUP, 2006).

Casos de Negócio

De acordo com Anúnciação (2009), este documento é responsável por examinar as justificativas econômicas, que determinam se o investimento no projeto é compensatório ou não.

Projetar a Arquitetura

O objetivo deste documento é refinar a arquitetura, analisar o comportamento e projetar os componentes do sistema e suas interfaces (RUP, 2006).

Protótipo

Utilizar a concepção de iterativo para a criação, teste e avaliação do produto, isto é, após desenvolver um conjunto de funções, já é realizado o teste e avaliação dessas novas funções, a fim de afastar problemas e detectar risco o mais cedo possível (RUP, 2006).

Avaliação

De acordo com Anúnciação (2009), este documento tem objetivo avaliar resultados (status e progresso) do projeto regularmente a cada iteração.

Controle de Mudanças

Este documento tem por objetivo documentar e controlar defeitos, solicitações e melhorias no produto a pedido do usuário. A vantagem do uso deste documento é que eles fornecem um registro de decisões, por isso permite que o impacto da possível mudança se compreendido por todos os membros da equipe do projeto (RUP, 2006).

Suporte ao usuário

Além do desenvolvimento do *software*, deve atentar-se para a produção de documentos de uso e instalação como guia do usuário, guia de instalação e notas sobre o release. Quando necessário, deve aplicar também o treinamento para o usuário (ANUNCIACÃO, 2009).

Processo

De acordo com o RUP (I2006) é essencial a adoção de um processo que se ajuste ao projeto e produto que está sendo desenvolvido. Na ausência deste processo a equipe por ter comunicações e compreensões errôneas sobre quem vai fazer o quê e quando.

2.5.3 Estrutura

A seguir será apresentada a estrutura do UP, nos seus dois aspectos principais: a estrutura estática, que representa as disciplinas que agrupam as atividades logicamente; e a estrutura dinâmica, que representa as fases e mostra os aspectos do ciclo de vida à medida que se desenvolve. A figura 7 ilustra as duas visões da estrutura.

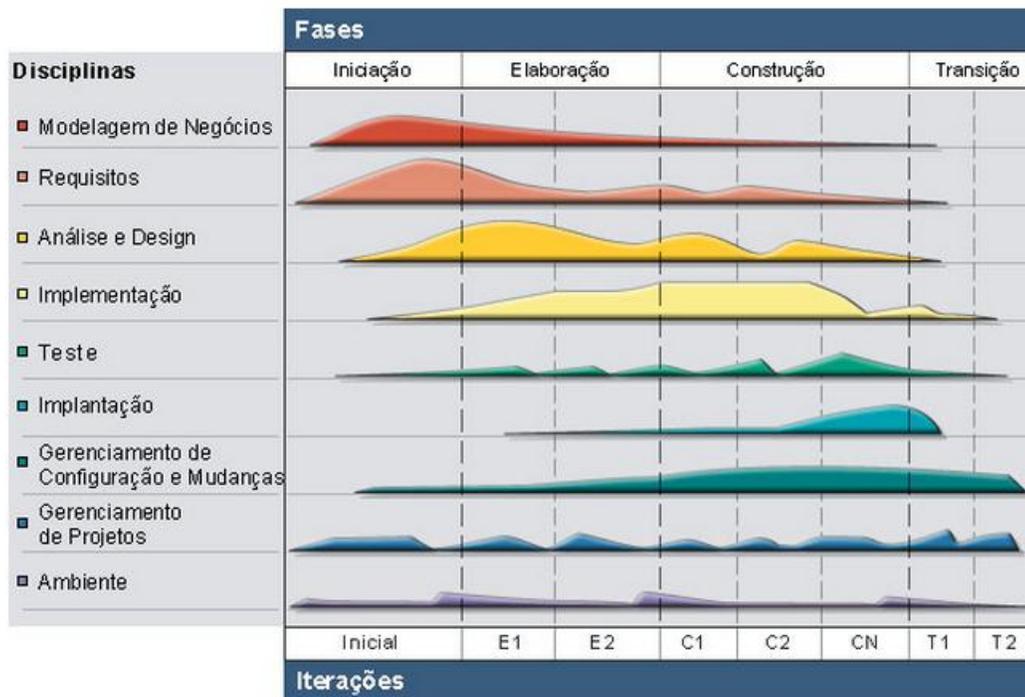


Figura 6 Estrutura do UP
Fonte: UP (2006)

Estrutura Estática – Disciplinas

As disciplinas apresentam as atividades que devem ser realizadas para produzir um determinado conjunto de artefatos (documentos).

De acordo com Kruchten (2003, apud ANUNCIÇÃO, 2009) o UP possui nove disciplinas, no qual seis são referentes a processos de engenharia e três são referentes a processos de suporte, sendo:

- Processos de Engenharia: modelagem de negócio, requisitos, análise e design, implementação, teste e implantação.
- Processos de Suporte: gerenciamento e configuração de mudança, gerenciamento de projetos e ambiente.

A seguir serão descritos as disciplinas de processos de engenharia com suas respectivas funções e responsabilidades:

- **Modelagem de Negócio**

Segundo Albuquerque (2003, apud Anunciação, 2009), a modelagem de negócio tem o objetivo de entender a estrutura e a dinâmica da organização a qual o sistema será implantado, identificar problemas correntes na organização e possibilidades de melhoria.

Os principais artefatos produzidos nesta disciplina são: plano de negócio, especificação suplementar e glossário (RUP, 2006).

- **Requisitos**

Segundo Anunciação (2009), o objetivo desta disciplina é definir o que o sistema deve fazer, especificando as características técnicas e funcionais. Além, disso deve fornecer uma base para estimar o custo e o tempo do projeto.

Os principais artefatos produzidos nesta disciplina são: plano de negócio, especificação suplementar e glossário. As principais atividades executadas são: detalhar os requisitos, identificar atores e casos de uso e detalhar um caso de uso (RUP, 2006).

- **Análise e Design**

De acordo com Anuniação (2009), o objetivo desta disciplina é traduzir os requisitos em uma especificação que guiará a implementação.

Para Toledo (2004) esta disciplina é uma das principais na construção de um *software*, além da modelagem visual, é responsável por definir a arquitetura, o ambiente operacional, a escalabilidade e a performance.

Os principais artefatos produzidos nesta disciplina são: documento de arquitetura de *software*, modelo de design, modelo de dados e protótipo de interface (RUP, 2006).

- **Implementação**

O objetivo desta disciplina é a codificação (gerar código-fonte), realizar testes do novo componente desenvolvido e implantar o componente ao sistema executável. Os principais artefatos produzidos nesta disciplina são: modelo de implementação e plano de integração do build (RUP, 2006).

- **Teste**

Segundo Anuniação (2009), objetivo desta disciplina é a avaliação do *software*, abrangendo testes na detecção de falhas que comprometem a qualidade do produto e validação das funcionalidades de acordo com os requisitos e com o design.

Toledo (2004) afirma que o principal objetivo desta fase é garantir que o sistema irá funcionar conforme o esperado, permitindo assim, a implantação do *software*.

Os principais artefatos produzidos são: plano de teste, procedimento de teste, caso de teste, modelo de teste e sumário de avaliação de teste (RUP, 2006).

- **Implantação**

Esta disciplina descreve as atividades que garantem que o *software* será disponibilizado a seus usuários finais. Segundo Toledo (2004), as atividades são: teste do sistema no ambiente do negócio, distribuição do *software*, instalação do *software*,

treinamento dos usuários, migração de dados para o novo sistema e criação do manual do usuário.

Os principais artefatos produzidos são: plano de implantação, material de suporte ao usuário, materiais de treinamento (RUP, 2006).

Estrutura Dinâmica – Fases

De acordo com o RUP (2006), o Processo Unificado possui quatro fases sequenciais: Iniciação, Elaboração, Construção e Transição, formando assim o ciclo de vida do UP. A seguir são descritas as fases com seus respectivos objetivos.

- **Iniciação**

A primeira fase do UP tem como finalidade principal a compreensão de todos os envolvidos quanto a visão do sistema e definir estimativas para as próximas fases (RUP, 2006). Segundo Toledo (2004), seus objetivos específicos são: estabelecer o escopo do projeto, os casos de uso, a arquitetura, elaborar o cronograma, estimar o custo, prazo e riscos.

- **Elaboração**

A segunda fase do UP tem como finalidade principal fornecer uma base estável para a codificação do sistema na fase de construção (RUP, 2006). De acordo com Anúnciação (2009), nesta fase a visão do produto e a arquitetura são amadurecidas, os riscos são amenizados e um protótipo do sistema é criado para demonstração aos clientes.

- **Construção**

Na terceira fase do UP, é criado o código-fonte a partir da base criada nas fases anteriores, esse processo se repetirá até o ponto no qual o *software* estará apto para seguir para a próxima fase. O objetivo principal desta fase é construir o produto, e paralelamente evoluir os documentos de visão e arquitetura até que o produto esteja pronto para ser entregue aos usuários (ANUNCIAÇÃO, 2009).

- **Transição**

Na quarta e última fase do ciclo de vida do UP, o objetivo é disponibilizar o produto para o usuário final, o que inclui testes no ambiente de execução, a instalação, o treinamento e material de suporte.

Segundo Anunciação (2009) as fases são delimitadas no tempo com marcos de ciclo de vida, isto é, no final de cada fase é possível avaliar se os objetivos traçados foram alcançados e a partir dessa avaliação é definido se o projeto passará para a próxima fase.

De acordo com o RUP (2006), um projeto pode ser subdividido em iterações, que são um subconjunto de funções. Após a iteração passar por todas as fases, é concluído um ciclo de desenvolvimento, o que gera uma versão do produto, como mostra a figura 8. Assim como as fases, as iterações devem possuir seu próprio marco para avaliar se atendeu as expectativas.

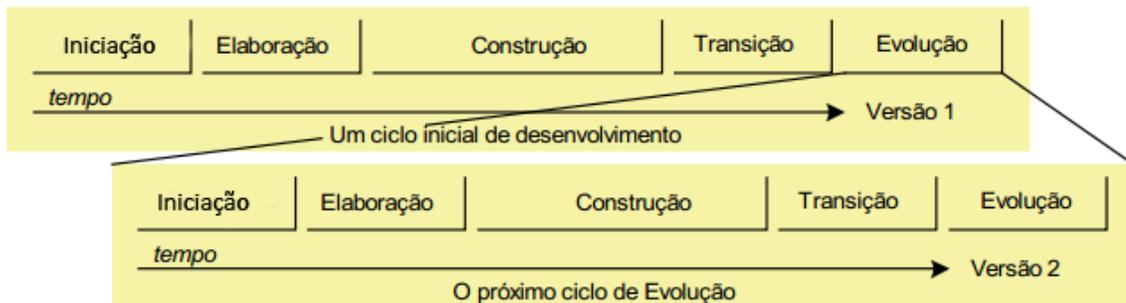


Figura 7 Ciclo de vida do UP
Fonte: UP (2006)

3. DESENVOLVIMENTO

Este capítulo abordará o processo para o desenvolvimento do software “SISLAL – Sistema Laboratorial de Análises de Leite e Derivado” a partir da aplicação da metodologia do processo unificado.

Além disso, apresentará as principais atividades executadas em cada fase do ciclo de vida do Processo Unificado e descreverá as tecnologias e ferramentas utilizadas no desenvolvimento do sistema. Ressalta-se que no desenvolvimento iterativo ocorre a passagem, diversas vezes, por uma determinada fase.

Inicialmente, será apresentada uma breve descrição das ferramentas utilizadas para a construção das atividades executadas neste capítulo.

3.1 Ferramentas Utilizadas

- Astah: é uma ferramenta CASE (*Computer-Aided Software Engineering*) de modelagem UML que permite a criação de diagramas de classe, casos de uso, sequência, atividade, pacote, entre outros.
- DBDesigner: é uma ferramenta CASE voltada para a modelagem de dados do sistema – modelo lógico do banco de dados.
- Eclipse: é uma IDE de desenvolvimento de aplicações voltada para criação do código-fonte do sistema.
- Pencil: é uma ferramenta voltada para a criação de protótipos de interface.

3.2 Fase de Iniciação

A seguir são descritas a execução das principais atividades executadas nesta fase, de acordo com o referencial teórico.

3.2.1 Visão do Sistema

Nesta atividade foi definida a visão que os envolvidos têm do produto, em termos de suas necessidades e das funcionalidades para atendê-las, demonstrando os problemas e as motivações para a criação do software.

O SISLAL tem por objetivo automatizar a rotina do Laboratório de Análise de Leite e Derivados da UENP-CLM, com a pretensão de automatizar o controle sobre os documentos gerados, aperfeiçoar a emissão de resultados, monitorar as análises realizadas e auxiliar no processo de tomada de decisões.

Com o SISLAL implantado, o laboratório ganhará velocidade, organização e precisão dos resultados em seus processos de negócio, além de poder usufruir de toda informação gerada em um período 24/7 (24 horas por dia, 7 dias por semana).

Abaixo é mostrado os principais problemas encontrados no processo de negócio do laboratório. Para mais informações sobre a visão do sistema consulte o APÊNDICE A.

O problema	<ul style="list-style-type: none"> • Laudos controlados manualmente
Afeta	<ul style="list-style-type: none"> • Membros do laboratório • Produtores
Impacto	<ul style="list-style-type: none"> • Retardo do processo de trabalho • Armazenamento ineficaz de laudos realizados • Alto risco de perdas de documentos • Ineficiência na consulta de resultados • Ineficiência na consulta de laudos armazenados
Solução	<ul style="list-style-type: none"> • Controle de laudo, no qual será possível cadastro e consulta de amostras e laudos submetidos pelo laboratório.

Quadro 2 Problemas detectados: controle de laudos

O problema	<ul style="list-style-type: none"> • Inexistência de monitoramento de análise
Afeta	<ul style="list-style-type: none"> • Membros do laboratório
Impacto	<ul style="list-style-type: none"> • Descontrole de análise a serem submetidas • Descontrole sobre o status atual de cada análise
Solução	<ul style="list-style-type: none"> • Monitoramento das análises, no qual será possível consultar a quantidade de análises a serem submetidas e o status de cada uma delas.

Quadro 3 Problemas detectados: controle de análises

O problema	<ul style="list-style-type: none"> • Inexistência de controle de evolução da qualidade do leite e derivados
Afeta	<ul style="list-style-type: none"> • Membros do laboratório • Produtores
Impacto	<ul style="list-style-type: none"> • Ineficiência da qualidade do serviço prestado • Retardo da emissão do resultado caso o serviço seja requisitado pelo produtor
Solução	<ul style="list-style-type: none"> • Controle de evolução, permitindo através de gráficos/relatórios, análises do laudo atual com laudos anteriores de acordo com seu produtor.

Quadro 4 Problemas detectados: controle de evolução

Após a compreensão do processo de negócio do laboratório e a detecção dos maiores problemas, foi realizada a elicitación de requisitos, com o objetivo de elencar as funcionalidades que o sistema deverá possuir.

3.2.2 Elicitación de requisitos

De acordo com Braga (2008) um dos grandes problemas enfrentados no ambiente de desenvolvimento de software é a elicitación de requisitos. Para Pressman (2006) uma compreensão completa dos requisitos é fundamental para um projeto bem sucedido.

Diante disto, para evitar problemas durante esta importante etapa, é adequado o uso de técnicas para auxiliar a coleta, a organização e o detalhamento de requisitos. Braga (2008) destaca algumas técnicas, são elas: entrevista, questionário, brainstorming e prototipação.

Neste trabalho, a técnica adotada foi a entrevista, que o RUP (2006) destaca como uma técnica direta e eficaz, que requer a preparação de uma lista de perguntas projetadas para se obter a compreensão dos problemas reais e das soluções possíveis.

Desta forma, foi criada uma lista de perguntas, baseado em um modelo fornecido pelo RUP (2006) para a obtenção dos requisitos do sistema. Esta lista pode ser encontrada no Anexo A – Entrevista: Elicitación de Requisitos do presente trabalho.

Com a utilização da técnica citada acima, chegou-se aos seguintes requisitos:

Requisitos Funcionais

RF01	Apresentar tela de acordo com o perfil do usuário
RF02	Bloquear/Desbloquear acesso ao laudo
RF03	Bloquear/Desbloquear o acesso do usuário ao sistema
RF04	Bloquear/Desbloquear produtor
RF05	Cadastrar laudos
RF06	Cadastrar membros do laboratório
RF07	Cadastrar o material de amostra
RF08	Cadastrar observação do laudo
RF09	Cadastrar os itens das análises
RF10	Cadastrar os tipos de amostra
RF11	Cadastrar produtores
RF12	Cadastrar resultado de análise
RF13	Cadastrar usuário
RF14	Configurar perfil
RF15	Controlar Acesso ao sistema
RF16	Editar/Remover laudo
RF17	Editar/Remover material de amostra
RF18	Editar/Remover membro
RF19	Editar/Remover produtor
RF20	Editar/Remover resultado de análise
RF21	Editar/Remover tipo de amostra
RF22	Gerar gráfico de evolução de qualidade de amostra
RF23	Gerar relatório de cadastrados
RF24	Gerar relatório de membros cadastrados
RF25	Gerar relatório de produtores cadastrados
RF26	Visualizar informações do laudo
RF27	Visualizar o laudo

Quadro 5 Requisitos Funcionais do SISLAL

Requisitos não funcionais

RNF01	Documentar o código-fonte
RNF02	Executar ações em no máximo 5 clicks
RNF03	Expirar sessão do usuário após 10 minutos de inatividade
RNF04	Fornecer manual do usuário
RNF05	Permitir acesso simultâneo de vários usuários
RNF06	Permitir exportação dos relatórios em PDF
RNF07	Possuir diferentes perfis de usuário
RNF08	Utilizar gerenciador de versões
RNF09	Utilizar padrões de codificação

Quadro 6 Requisitos não funcionais do SISLAL

Após conhecer os requisitos do sistema, o próximo passo foi organizá-los em grupos (casos de uso), tema do próximo tópico.

3.2.3 Casos de Uso

Nesta atividade foram detectados os casos de uso e os *stakeholders* do sistema. O SISLAL possui como casos de uso manter membro, manter produtor, manter análise, manter laudo, medir evolução, emitir relatório e visualizar laudo; e como *stakeholders* o administrador, o estagiário e o produtor.

A figura 9 ilustra o diagrama de casos de uso do sistema; o detalhamento de cada caso de uso é realizado na próxima fase do ciclo de desenvolvimento.

Como pode ser visto na figura 9, o sistema contém restrições para cada tipo de usuário, representado pelos *stakeholders*, e em determinados casos, para a execução de um caso de uso é necessário a execução prévia de outro caso de uso.

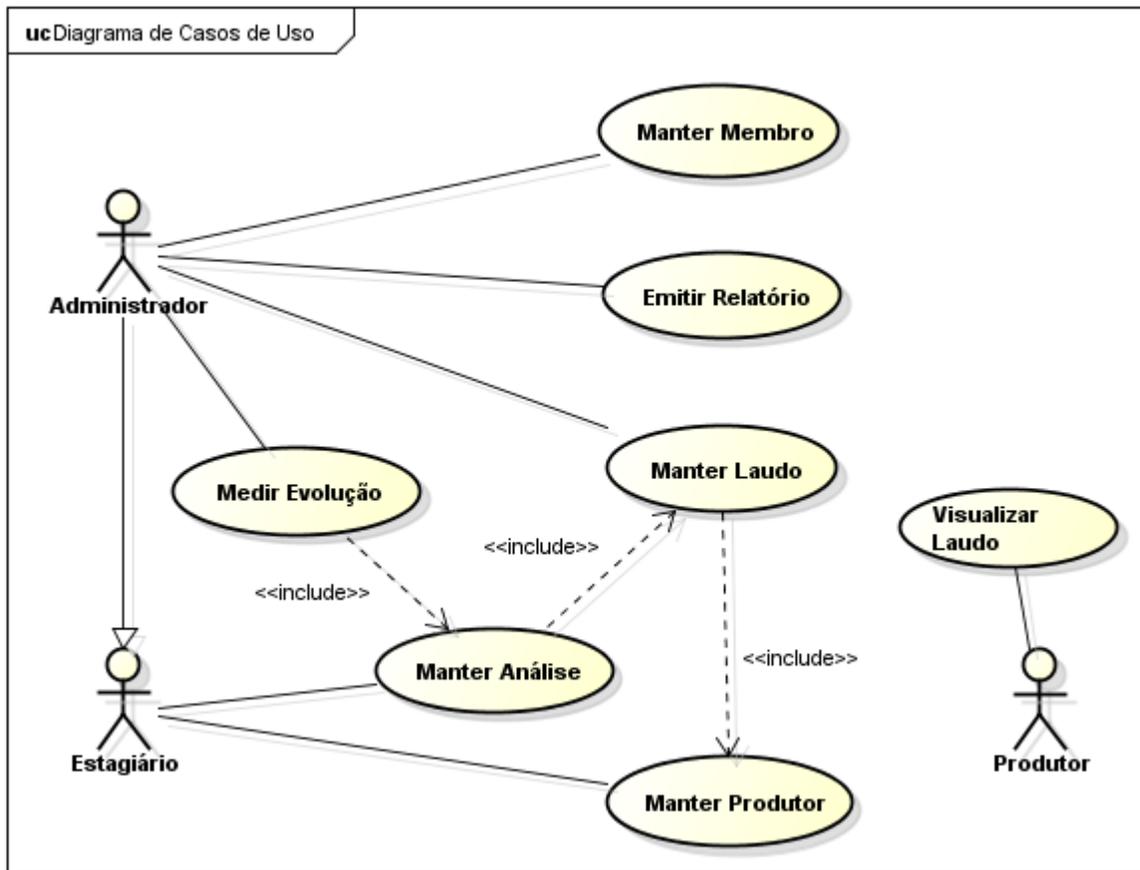


Figura 8 Diagrama de Casos de Uso

3.2.4 Identificar Riscos

Segundo Pressman (2006), uma importante tarefa em um projeto é prever os riscos que podem afetar a programação do projeto ou a qualidade do software e tomar as medidas necessárias para evitar esses riscos.

Para isso, deve-se elaborar um documento e elencar os possíveis riscos que podem afetar o projeto seguido de plano para minimizar seus efeitos.

O quadro 7 expõe os riscos previstos para este projeto, apresentando a data em que foi identificado o risco, o nome, a descrição, o impacto que causará ao projeto, a probabilidade de ocorrer o risco e medidas que deverão ser tomadas se caso ocorrer o risco.

Por exemplo, o primeiro risco identificado foi “Instabilidade no Ambiente de Testes” no dia 30/03/2012, esse risco poderá ocorrer devido a erros no ambiente de

teste que eram inexistentes no ambiente de desenvolvimento. Para este risco o impacto causado será igual a 3 com probabilidade de ocorrer igual a 2. Caso este risco realmente ocorrer, a medida a ser tomada será estudar a possibilidade de reconfigurar ou trocar o ambiente de teste.

Risco ID	Data Identificada	Nome	Descrição	Impacto (1 a 3)	Magnitude		Plano de Gestão
					Probabilidade (1 a 3)	(1 a 3)	
1	30/03/2012	Instabilidade no Ambiente de Testes	Sistema gera erros no ambiente de teste inexistentes no ambiente de desenvolvimento	3	2	6	Identificar a possibilidade de reconfiguração ou troca do ambiente de teste.
2	30/03/2012	Especificação de Requisitos Falha	Os requisitos levantados não representa fielmente as necessidades do usuário	3	2	6	Realizar nova abordagem de elicitação de requisitos seguindo uma metodologia; calcular o impacto que estas mudanças terá no prazo do projeto.
3	01/04/2012	Alteração nos requisitos	São propostas mudanças nos requisitos que exigem significativo retrabalho	2	3	6	Extraia informações que podem ser rastreadas, para avaliar o impacto das mudanças nos requisitos, maximize a inclusão de informação no projeto
4	01/04/2012	Tempo subestimado	O tempo necessário para desenvolver o software é subestimado	3	2	6	Investigue a possibilidade do uso de componentes reutilizáveis

Risco ID	Data Identificada	Nome	Descrição	Impacto (1 a 3)	Magnitude (1 a 3)	Plano de Gestão	
5	01/04/2012	Tamanho subestimado	O tamanho do sistema foi subestimado	2	2	4	Investigar possibilidade de aumento da carga horária dos membros do projeto; negociar uma extensão do prazo de entrega do
6	01/04/2012	Mudança na tecnologia	A tecnologia sobre a qual o sistema esta sendo desenvolvido foi superada por nova tecnologia.	1	1	1	Investigar a possibilidade a substituir a tecnologia, e avaliar o impacto causado
7	03/04/2012	Treinamento indisponível	O treinamento necessário para os usuários não está disponível	1	1	1	Identificar a possibilidade de expansão do prazo para aplicar o treinamento

Quadro 7 Lista de Riscos

3.2.5 Planejar Iterações

De acordo com o RUP (2006) o objetivo do planejamento de iteração é apresentar minuciosamente as atividades e tarefas a serem desenvolvidas por sequências de tempo, produtos liberados e marcos (critério de avaliação).

Normalmente, as iterações são determinadas para cada fase do ciclo de desenvolvimento, na qual cada uma deverá possuir ao menos uma iteração.

Deste modo, a seguir são apresentadas as iterações projetadas para as fases de iniciação, elaboração, construção e transição, durante a produção do SISLAL. Ressalta-se que a fase de transição não chegou a ser executada, porém o plano de iteração para esta fase foi concebido.

- **Plano da Fase de Iniciação**

Nesta fase é definido o Plano de Iteração Preliminar com enfoque em definir o sistema e identificar os casos de uso e atores.

I1 - Tarefas de Iniciação

O quadro 8 expõe as tarefas a serem executadas durante a fase de iniciação, estimando a data de início e término para cada tarefa.

Tarefa	Início	Fim
Iniciação	12/03/2012	02/04/2012
Elicitação de Requisitos	12/03/2012	13/04/2012
Início do Plano de Visão	14/03/2012	20/03/2012
Identificar Riscos	21/03/2012	22/03/2012
Identificar Casos de Uso	23/03/2012	23/03/2012
Planejar Iterações	24/03/2012	27/03/2012
Início do Plano do Projeto	28/03/2012	39/03/2012
Revisar Documentos	29/03/2012	02/04/2012

Quadro 8 Tarefas realizadas na iteração I1 da Fase de Iniciação

I1 - Produtos Liberados

O quadro 9 expõe os produtos que são liberados após a passagem pela fase de iniciação.

Conjunto de Artefato	Distribuível
Conjunto de Requisitos	Documento de Visão
Conjunto de Gerenciamento	Plano do Projeto Lista de Riscos do Projeto, Plano de Iteração Preliminar

Quadro 9 Produtos Liberados na iteração I1 da Fase de Iniciação

I1 - Critérios de Avaliação

Ter compreendido a visão do sistema, as necessidades dos envolvidos e identificados os possíveis riscos.

- **Plano da Fase de Elaboração**

Nesta fase é definido o Plano de Iteração com enfoque em definir a arquitetura do sistema e fornecer uma base estável para o esforço de construção.

E1 - Tarefas de Elaboração

O quadro 10 expõe as tarefas a serem executadas durante a fase de elaboração, estimando a data de início e término para cada tarefa.

Tarefa	Início	Fim
Elaboração	03/04/2012	24/04/2012
Detalhar Casos de Uso	03/04/2012	05/04/2012
Definir classes do sistema	05/04/2012	06/04/2012
Definir o modelo de design	07/04/2012	12/04/2012
Revisar plano de visão	13/04/2012	13/04/2012
Incrementar Riscos	14/04/2012	14/04/2012
Atualizar Plano de Projeto	16/04/2012	17/04/2012
Definir o modelo de dados	18/04/2012	19/04/2012
Criar protótipo de tela	19/04/2012	20/04/2012
Planejar Iterações	21/04/2012	22/04/2012
Revisar Documentos	23/04/2012	24/04/2012

Quadro 10 Tarefas realizadas na iteração E1 da Fase de Elaboração

E1 - Produtos Liberados

O quadro 11 expõe os produtos que são liberados após a passagem pela fase de elaboração.

Conjunto de Artefato	Distribuível
Conjunto de gerenciamento	Lista de risco, Plano de Iteração, Plano do projeto, Visão
Conjunto de modelos	Protótipo de Interface, modelo de design, modelo de dados

Quadro 11 Produtos Liberados na iteração E1 da Fase de Elaboração

E1 - Critérios de Avaliação

A visão e os requisitos do produto são estáveis, a arquitetura é estável, os principais riscos foram detectados, os documentos a serem utilizados na próxima fase foram definidos.

- **Plano da Fase de Construção**

Nesta fase é definido o Plano de Iteração com foco no desenvolvimento do sistema com base na arquitetura definida na fase anterior.

C1 - Tarefas de Construção

O quadro 12 expõe as tarefas a serem executadas durante a primeira iteração da fase de construção, estimando a data de início e término para cada tarefa.

Tarefa	Início	Fim
Construção 1	26/04/2012	12/05/2012
Implementar e testar camada de persistência	26/04/2012	26/04/2012
Implementar classes e atributos da camada de negócios	27/04/2012	29/04/2012
Implementar camada de controle	30/04/2012	30/04/2012
Implementar e testar Manter Pessoa	01/05/2012	03/05/2012
Implementar e testar Manter Usuário	04/05/2012	04/05/2012
Implementar e testar Manter Produtor	05/05/2012	07/05/2012
Implementar e testar Manter Membro	07/05/2012	09/05/2012
Atualizar o Modelo de Dados	10/05/2012	10/05/2012
Atualizar Especificações suplementares	11/05/2012	11/05/2012
Atualizar o Modelo de Design	12/05/2012	12/05/2012

Quadro 12 Tarefas realizadas na iteração C1 da Fase de Construção

C1 - Produtos Liberados

O quadro 13 expõe os produtos que são liberados após a primeira iteração da fase de construção.

Conjunto	Distribuível
Camada de persistência, camada de negócio, camada de controle, manter pessoa, manter produtor e manter membro	Módulo produtor e módulo membro
Manter Usuário	Módulo Login
Artefatos atualizados	Modelo de Design, Modelo de Dados, Especificação Suplementar
Todas	Verão beta

Quadro 13 Produtos Liberados na iteração C1 da Fase de Construção

C1 - Critérios de Avaliação

A versão do produto é estável para ser implantado para o usuário.

C2 - Tarefas de Construção

O quadro 14 expõe as tarefas a serem executadas durante a segunda iteração da fase de construção, estimando a data de início e término para cada tarefa.

Tarefa	Início	Fim
Construção	12/05/2012	25/05/2012
Implementar e testar Manter Laudo	12/05/2012	18/05/2012
Implementar e testar Manter Material	14/05/2012	15/05/2012
Implementar e testar Manter Tipo	16/05/2012	17/05/2012
Implementar e testar Manter Análise	18/05/2012	21/05/2012
Atualizar o Modelo de Dados	22/05/2012	22/05/2012
Atualizar o Modelo de Design	23/05/2012	23/05/2012
Apresentação	25/05/2012	25/05/2012

Quadro 14 Tarefas realizadas na iteração C2 da Fase de Construção

C2 - Produtos Liberados

O quadro 15 expõe os produtos que são liberados após a segunda iteração da fase de construção.

Conjunto	Distribuível
manter laudo, manter material, manter tipo e manter análise	Módulo Laudo, Módulo Material, Módulo Tipo e Módulo Análise
Artefatos atualizados	Modelo de Dados
Todas	Verão beta

Quadro 15 Produtos Liberados na iteração C2 da Fase de Construção

C2 - Critérios de Avaliação

A versão do produto é estável para ser implantado para o usuário.

C3 - Tarefas de Construção

O quadro 16 expõe as tarefas a serem executadas durante a segunda iteração da fase de construção, estimando a data de início e término para cada tarefa.

Tarefa	Início	Fim
Construção	25/05/2012	25/05/2012
Implementar e testar Gerar Relatórios	25/05/2012	27/05/2012
Implementar e testar Gerar Laudo	28/05/2012	29/05/2012
Implementar e testar Controle de Evolução	30/05/2012	06/06/2012
Atualizar o Modelo de Dados	07/06/2012	07/06/2012
Apresentação	11/06/2012	11/06/2012

Quadro 16 Tarefas realizadas na iteração C3 da Fase de Construção

C3 - Produtos Liberados

O quadro 17 expõe os produtos que são liberados após a terceira iteração da fase de construção.

Conjunto	Distribuível
manter laudo, manter material, manter tipo e manter análise	Módulo Laudo, Módulo Relatório, Módulo Evolução
Artefatos atualizados	Modelo de Dados
Sistema	O sistema executável

Quadro 17 Produtos Liberados na iteração C3 da Fase de Construção

C4 - Critérios de Avaliação

A versão do produto é estável para ser implantado para o usuário.

- **Plano de Fase de Transição**

Nesta fase é definido o Plano de Iteração com foco em assegurar que o software esteja disponível para o usuário final.

T1 - Tarefas de Transição

O quadro 18 expõe as tarefas a serem executadas durante a segunda iteração da fase de construção, estimando a data de início e término para cada tarefa.

Tarefa	Início	Fim
Transição	20/06/2012	29/06/2012
Implantação no ambiente do usuário	20/05/2012	26/05/2012
Criar manual do usuário	20/05/2012	23/05/2012
Realizar Treinamento	27/05/2012	29/06/2012

Quadro 18 Tarefas realizadas na iteração T1 da Fase de Transição

T1 - Produtos Liberados

O quadro 19 expõe os produtos que são liberados após a passagem pela fase de transição.

Conjunto	Distribuível
Manual do usuário	Material de Suporte
Sistema	Versão Final

Quadro 19 Produtos Liberados na iteração T1 da Fase de Transição

3.2.6 Cronograma

O objetivo do cronograma é estimar o projeto em um período de tempo (dias ou horas). Este projeto foi estimado em 109 dias. Para mais detalhes consulte o artefato Plano do Projeto – APÊNDICE B.

3.3 Fase de Elaboração

A seguir são descritas a execução das principais atividades executadas nesta fase, que de acordo com o referencial teórico, possui o objetivo de fornecer uma base estável para a codificação do sistema.

Nesta fase são expandidos os casos de uso; definidos os diagramas de classe, atividade, comunicação e sequência; modelado o banco de dados e é criado o protótipo do sistema.

3.3.1 Detalhamento de Casos de Uso

Nesta atividade foi definido um fluxo de ações para cada caso de uso, gerando assim um documento que servirá de base para a codificação do sistema e criação de casos de teste.

Para isto, é definida uma estrutura que abordará as informações principais de um caso de uso, como: o ator que irá interagir com este caso de uso, o objetivo, o fluxo principal (sequência de passos a serem executados) e o fluxo alternativo (caso o fluxo principal não possa ser executado normalmente). Normalmente, para apresentar o maior número de detalhes possível, um caso de uso é dividido em vários subfluxos, que juntos formam o caso de uso em si.

No quadro 20 é exposto o subfluxo “Cadastrar Membro” do caso de uso “Manter Membro”. Consulte o artefato Modelo de Casos de Uso – APÊNDICE C – para visualizar os demais casos de uso e seus respectivos subfluxos.

SubFluxo 1.1	Novo Membro
Objetivo	Cadastrar um novo membro do laboratório
Fluxo Principal	<ol style="list-style-type: none"> 1. O sistema solicita os dados para o cadastro do novo membro. 2. O ator informa o nome, CPF, e-mail, telefone, nome de usuário, senha e perfil.

	<ol style="list-style-type: none"> 3. O ator seleciona a operação “Salvar”. 4. O sistema valida os dados informados. 5. O sistema grava as informações na base de dados. 6. O sistema emite mensagem “Cadastrado com sucesso”. 7. O sistema retorna ao caso de uso UC1.
Fluxo Alternativo	<p>Dados inválidos</p> <ol style="list-style-type: none"> 2.a. O sistema emite alerta para o usuário 2.b. O sistema retorna ao passo 2 do fluxo principal <p>Operação Cancelada</p> <ol style="list-style-type: none"> 3.a O sistema retorna ao caso de uso UC1. <p>Duplicidade de dados</p> <ol style="list-style-type: none"> 5.a. O sistema informa ao usuário problema de duplicidade de dados em relação a nome de usuário e/ou CPF. 5.b. O sistema retorna ao passo 2 do fluxo principal

Quadro 20 SubFluxo Cadastrar Membro

Como pode ser visto, o quadro 20 apresenta a sequência de passos necessários para o cadastro de um membro do laboratório, abordando os possíveis problemas que podem ocorrer durante este processo e o que deve ser feito caso isto ocorra. Por exemplo, caso o usuário entre com informações inválidas, o sistema deverá emitir um alerta ao mesmo relando o problema encontrado e deverá aguardar a entrada de dados válidos.

3.3.2 Classes do Sistema

Nesta atividade foi definido o diagrama de classe do sistema. De acordo com Pedri (2008), este diagrama representa uma visão de como as classes estão estruturadas e relacionadas. A figura 10 ilustra o diagrama de classe do sistema SISLAL, apresentando as classes utilizadas, os atributos pertencentes a cada uma e os relacionamentos entre elas. Esta estrutura é responsável pela regra de negócio do sistema.

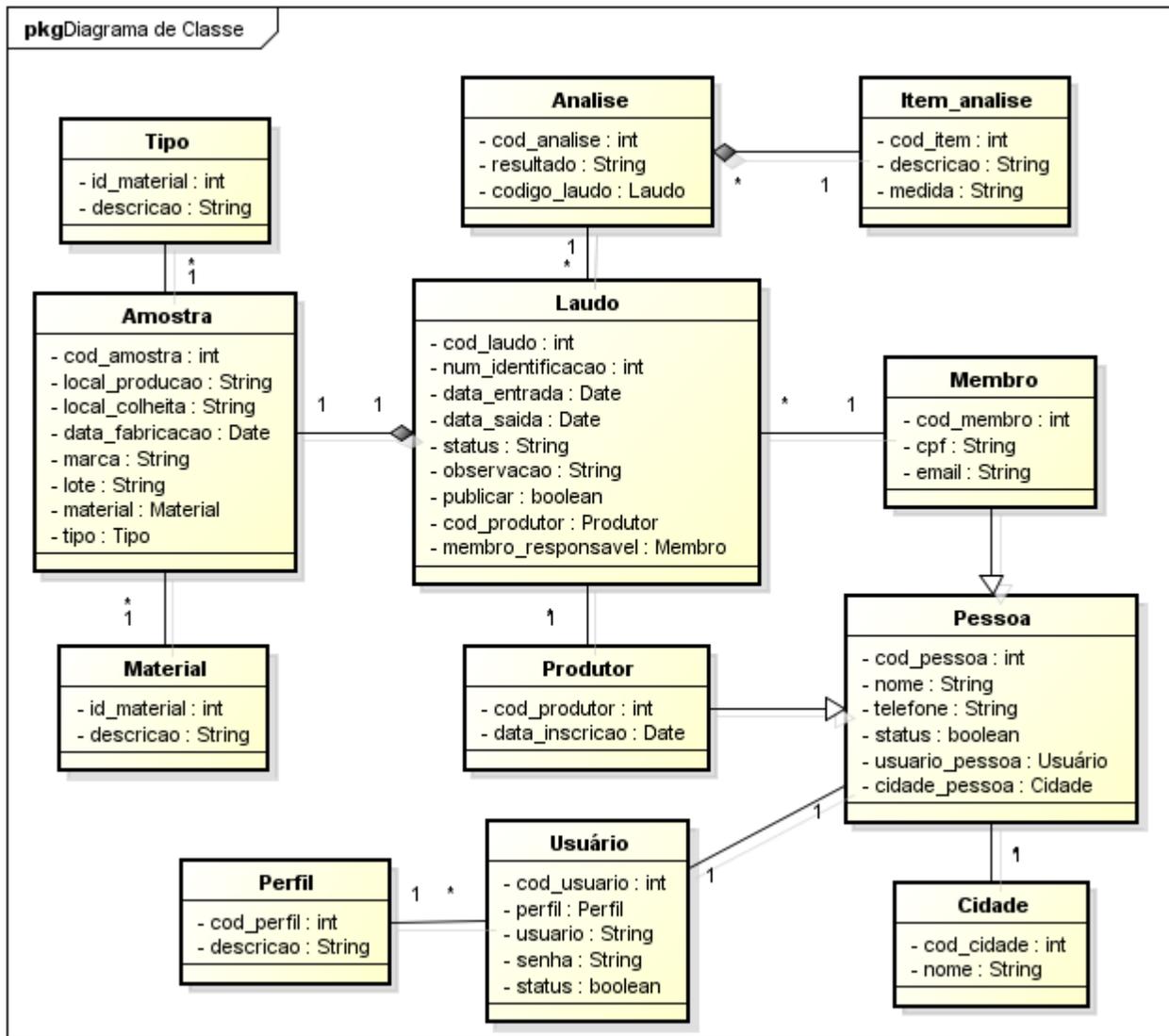


Figura 9 Diagrama de Classe
Fonte: Autor

3.3.3 Modelo de Design

Nesta atividade o objetivo é demonstrar os relacionamentos, a estrutura e o comportamento entre os diversos componentes da aplicação, por isso é uma abstração da implementação do sistema (RUP, 2006).

Deste modo, foram construídos diagramas de atividade, comunicação e sequência para cada caso de uso e seus subfluxos com o objetivo apresentar visualmente uma abstração do sistema.

A figura 10 ilustra o diagrama de atividade do subfluxo novo membro, pertencente ao caso de uso manter membro. Este diagrama visa mostrar o fluxo de ações que serão realizadas, definindo seu comportamento.

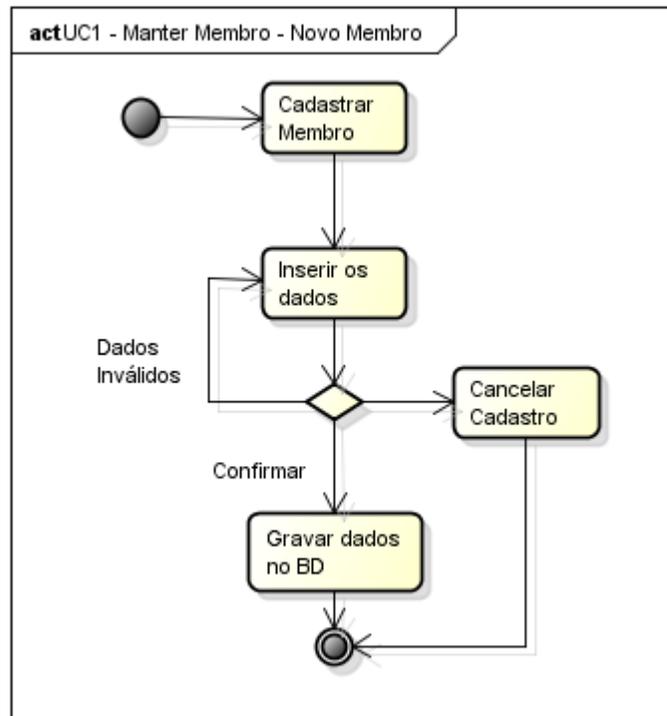


Figura 10 Diagrama de Atividade - Novo Membro

A figura 11 ilustra o diagrama de comunicação do subfluxo novo membro, pertencente ao caso de uso manter membro. Este diagrama tem como objetivo mostrar os relacionamentos entre as classes e objetos, apresentando a troca de mensagem que ocorre entre eles, por meio de chamadas de métodos.

Neste caso, é possível observar as classes e objetos que são utilizados para cadastrar um novo membro, como a classe espelho "Membro", a classe controladora "ControleMembro" e a classe responsável pela persistência "Dao"; e a constante troca de mensagens (métodos) que há entre elas.

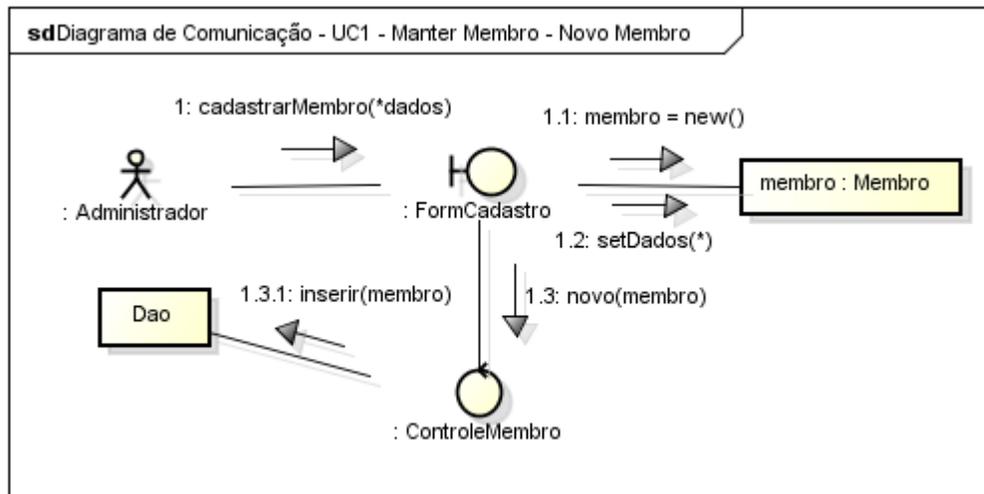


Figura 11 Diagrama de Comunicação - Novo Membro

A figura 12 ilustra o diagrama de seqüência do subfluxo novo membro, pertencente ao caso de uso manter membro. Este diagrama tem como objetivo mostrar a seqüência da troca de mensagens que ocorre para executar uma operação.

Neste caso, observa-se a seqüência das ações realizadas para cadastrar um novo membro. Inicialmente o ator (*stakeholder*) faz uma requisição a interface e fornece como entrada os dados do membro, a seguir é criado um objeto do tipo membro e é atribuído a esse objeto os valores fornecidos pelo ator, o próximo passo é passar este objeto a classe controladora e finalmente este objeto é passado a classe responsável pela persistência que gravará os dados na base de dados.

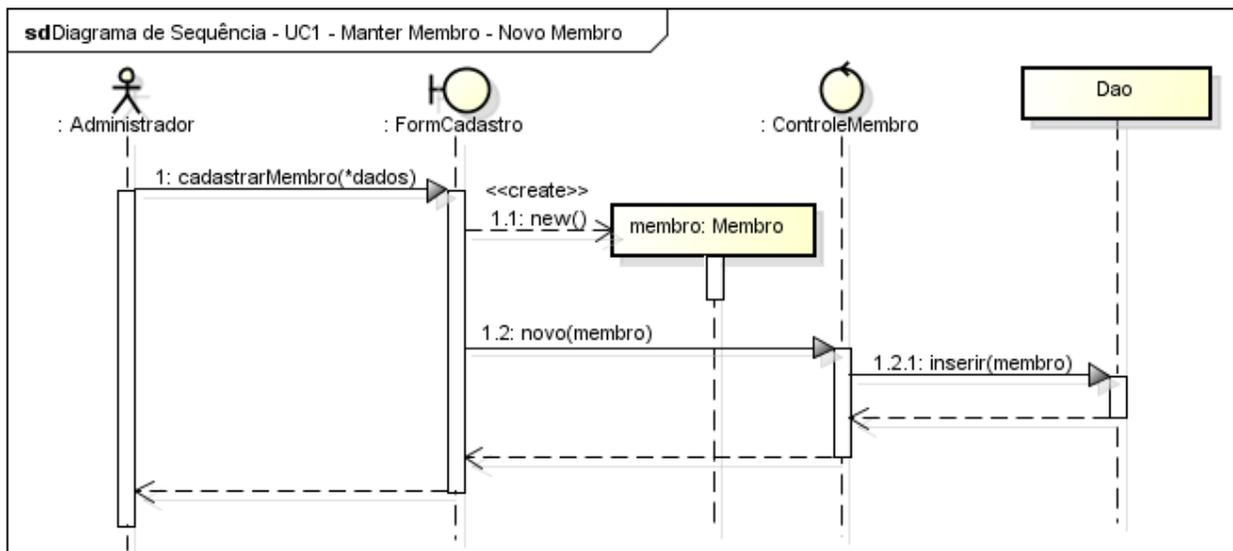


Figura 12 Diagrama de Sequência – Novo Membro

3.3.5 Protótipo de Interface

Nesta atividade foi desenvolvido o protótipo de interface do sistema, que de acordo com Aguiar (2007), é a representação, não funcional, da ideia de um produto em concepção por meio de representação gráfica.

Segundo o RUP (2006), a principal finalidade de criar um protótipo de interface é validar o design, incluindo sua capacidade de uso antes do início real do desenvolvimento. Dessa forma, é possível garantir que o sistema correto esteja sendo criado antes de desperdiçar tempo e recursos no desenvolvimento.

A figura 12 ilustra o protótipo de interface para o componente “Gerenciar Resultados da Análise” - caso de uso “Manter Análise”.

Gerenciar Resultados da Análise

Identificação do Laudo: text
 Produtor: text
 Membro Responsável: text
 Data de Entrada: dd/mm/aaaa

Análises	Medidas	Resultados
Gordura	%	
Densidade	g/mL	

Confirmar

Cancelar

Figura 14 Exemplo Protótipo de Interface
 Fonte: Autor

Este módulo é um dos mais importantes do sistema, pois armazena os resultados obtidos pela análise, que servirá de feedback aos produtores sobre a qualidade da amostra e utilizados em futuras análises de evolução do produto.

3.4 Fase de Construção

A seguir são descritas a execução das principais atividades executadas nesta fase, de acordo com o referencial teórico. Primeiramente serão abordadas as tecnologias utilizadas nesta fase.

3.4.1 Tecnologias Aplicadas

Nesta seção são apresentadas as tecnologias utilizadas o desenvolvimento do software, são elas:

Padrão MVC

O MVC é um padrão de arquitetura que propõe a separação das classes de um sistema em três grupos: Modelo, Visão e Controle (BUSCHMANN, 1996 apud OLIVEIRA, 2007).

De acordo com Barros (2007), o modelo é formado por classes que representam os dados do sistema e responsável pela regra de negócio; a visão tem por objetivo apresentar os dados ao usuário e capturar eventos; e o controle é responsável pela conexão entre o modelo e a visão.

Desta forma, o objetivo do MVC é separar os dados da lógica de negócio, os dados de apresentação e o fluxo dos dados, o que contribui para o controle e gerenciamento do código-fonte e aumento de produtividade no desenvolvimento e em futuras manutenções.

Java

De acordo com seu site oficial, Java é uma linguagem de programação e uma plataforma lançada pela Sun Microsystems em 1995. É a tecnologia que capacita muitos programas da mais alta qualidade, com utilitários, jogos e aplicativos corporativos. A aplicação do Java neste trabalho deu-se nas camadas Modelo e Visão.

Flex

Segundo Schmitz (2008), o Flex é um framework de desenvolvimento de aplicações RIA (Aplicações Ricas para Internet), contendo diversos componentes visuais e uma poderosa linguagem de programação chamada Action Script. O Flex foi aplicado na camada de visão no desenvolvimento do sistema.

Hibernate

De acordo com o site oficial, o Hibernate é um framework para o mapeamento objeto-relacional utilizado na persistência dos dados do sistema mediante o uso de arquivos XML e anotações nativas.

Neste sentido, para o desenvolvimento do software SISLAL, foi utilizada a linguagem de programação Java para criar as regras de negócio, o framework Flex na criação das interfaces visuais, o framework Hibernate para a persistência dos dados, e fornecendo de base para a execução de todos os processos citados acima, o padrão MVC.

3.4.2 Arquitetura do Sistema

Por meio do padrão MVC o sistema foi dividido em agrupamentos lógicos, obtendo um conjunto de pacotes e classes para cada divisão. São eles:

Modelo

A figura 17 representa os pacotes e classes deste agrupamento.

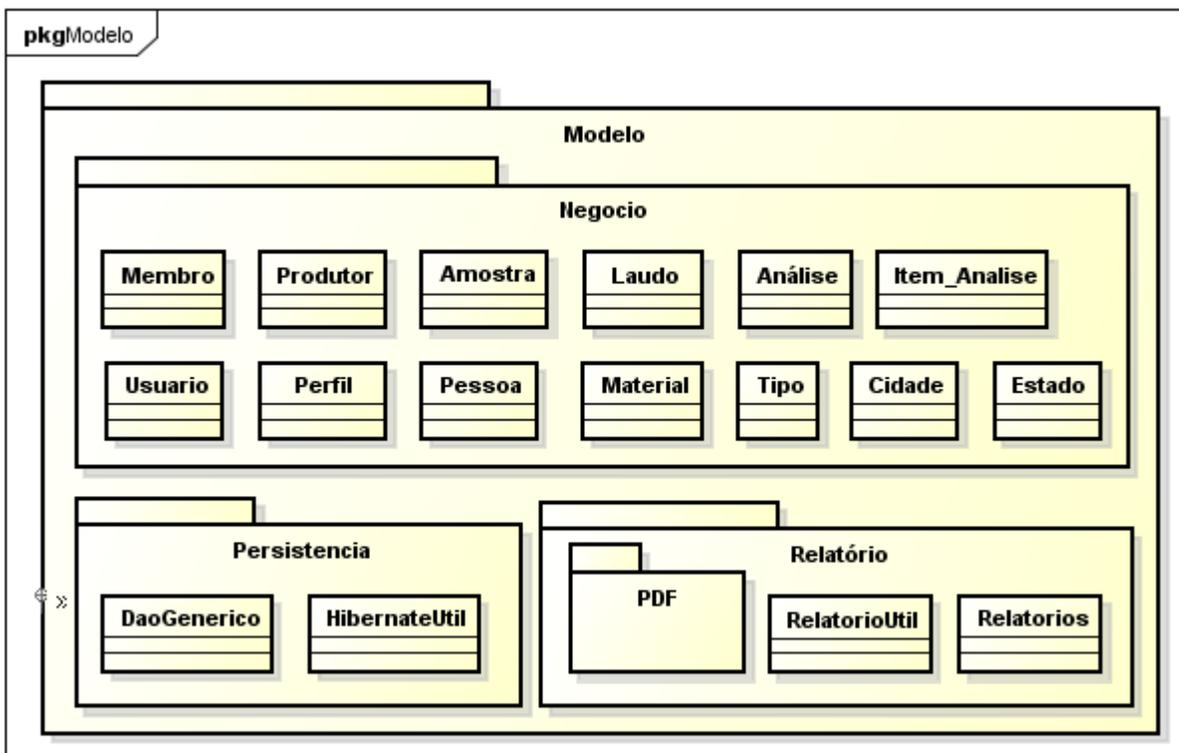


Figura 15 Diagrama de Pacote: Modelo

Este agrupamento é responsável pela regra de negócios do sistema. No pacote “negócio” estão as classes principais com seus respectivos atributos; no pacote “persistência” estão as classes responsáveis pela persistência dos dados na base de dados; e no pacote “relatório” estão contidas as classes responsáveis por gerar os relatórios do sistema.

Controle

A figura 18 representa os pacotes e classes deste agrupamento, contendo as classes controladoras, que são responsáveis por realizar a comunicação entre a interface gráfica (pacote visão) com a regra de negócio (pacote modelo).

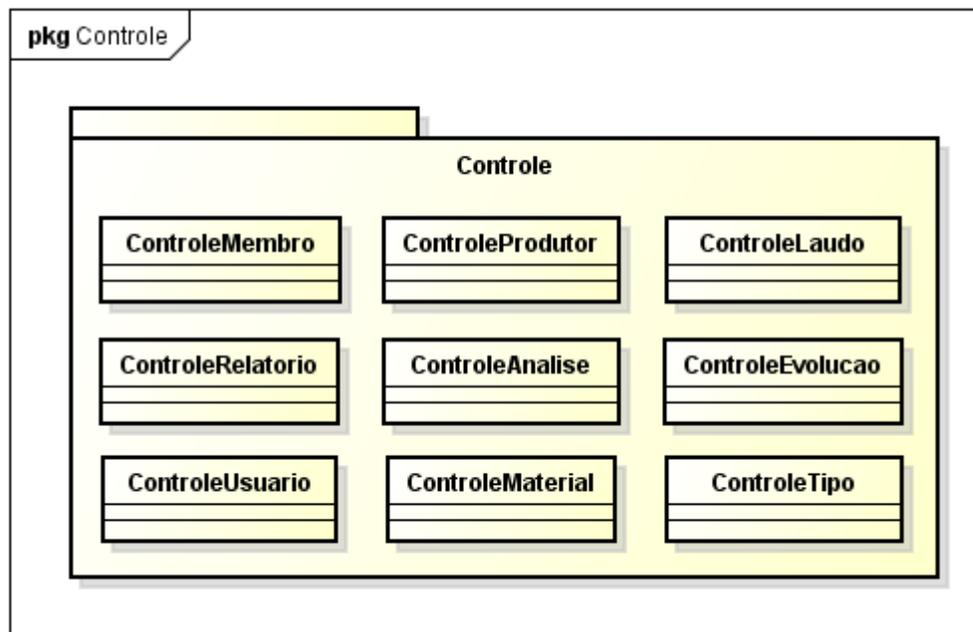


Figura 16 Diagrama de Pacote: Controle
Fonte: Autor

Visão

A figura 19 representa os pacotes e classes deste agrupamento que é responsável pela interface gráfica do sistema. Neste agrupamento estão contidos as classes responsáveis pelas telas do sistema, representadas pela página inicial (index), módulos, componentes, entre outros.

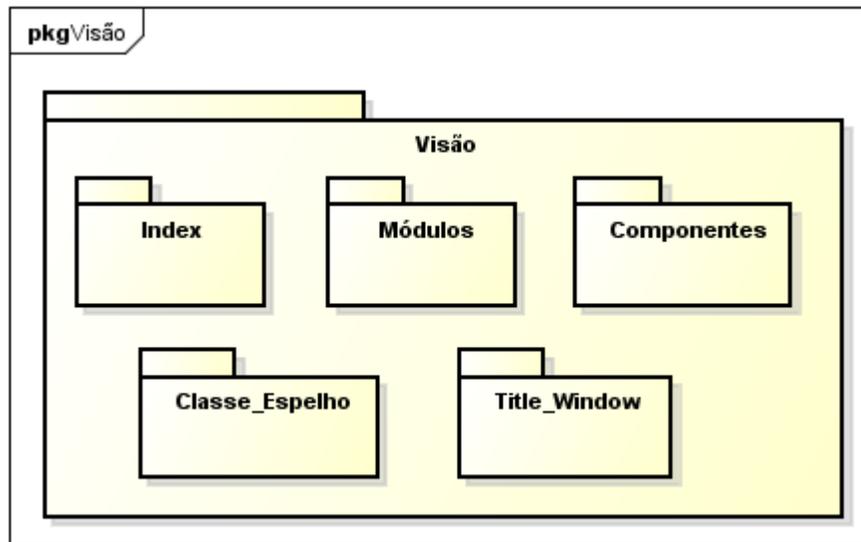


Figura 17 Diagrama de Pacote: Visão
Fonte: Autor

3.4.3 O sistema

Após a execução das principais atividades e a produção dos artefatos, definiu-se a base necessária para o desenvolvimento do software SISLAL.

Neste sentido, esta seção apresenta o software desenvolvido, contudo, como o foco do presente trabalho não é a implementação e sim o processo de desenvolvimento, serão apresentadas apenas as telas principais do sistema.

Inicialmente o usuário deve autenticar-se para acessar o sistema (Figura 20).



A tela de login, intitulada 'Acesso', contém os seguintes elementos:

- Usuário:
- Senha:
- [Esqueceu sua senha?](#)
- Botão Entrar

Figura 18 Tela de Login

Logo após, é apresentado a tela principal do sistema de acordo com o perfil do usuário (administrador, estagiário ou produtor). A seguir são apresentadas algumas telas do sistema SISLAL.

Cadastrar Membro

Tela responsável por cadastrar os membros do laboratório (Figura 21).

Nome: *

E-mail:

CPF:

Telefone: () -

Nome Usuário: *

Perfil: * ▼

Figura 19 Cadastrar Membro

Gerenciar Laudos

Nesta tela (Figura 22), é possível o gerenciamento do laudo, tem com funções principais: editar os dados do laudo, remover o laudo do sistema, visualizar o laudo em conjunto com seus respectivos resultados e alterar o status do laudo.

	Ident.	Produtor	Entrada	Saida	Status	Publicado
<input type="radio"/>	5	Produtor-04	27/05/2012		Aberto	
<input type="radio"/>	6	Produtor-02	30/05/2012		Aberto	
<input type="radio"/>	3	Produtor-03	27/05/2012		Aberto	
<input type="radio"/>	1	Produtor-01	18/05/2012	01/06/2012	Finalizado	
<input type="radio"/>	4	Produtor-01	18/05/2012		Aberto	
<input type="radio"/>	2	Produtor-02	25/05/2012	06/06/2012	Finalizado	

<< >> 6 Registros encontrados Registros por página: 10 ▼

Figura 20 Gerenciar Laudos

Controle de Evolução

Gráfico Gerado para medir a evolução de amostras de um determinado produtor (Figura 25).

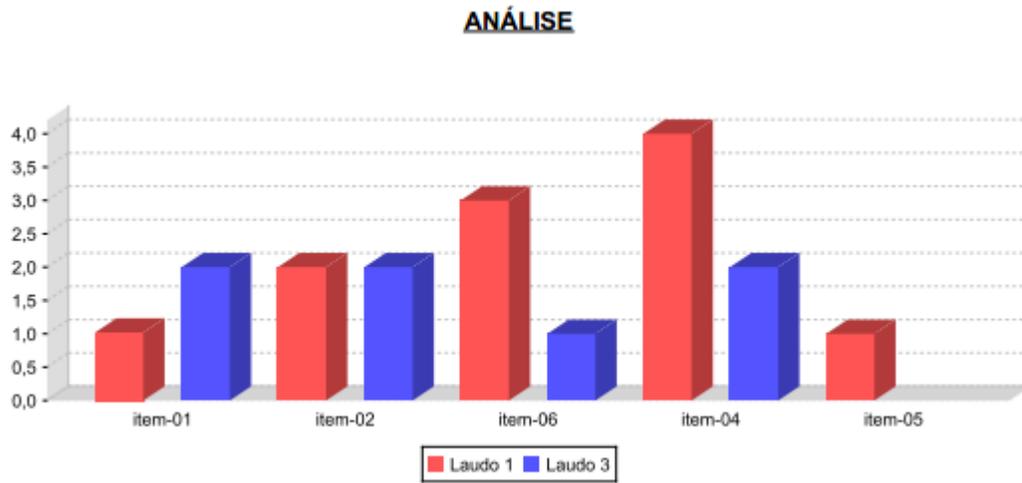


Figura 23 Gráfico de Evolução das Análises

4. CONSIDERAÇÕES FINAIS

A aplicação de sistemas de informação têm se tornado uma prática bastante comum nas organizações, cada vez mais essa tecnologia é utilizada na automatização de tarefas com o intuito de agilizar o processo de negócio.

Do mesmo modo que cresce seu uso, cresce também a necessidade de oferecer a essas organizações um produto confiável, seguro, de qualidade e que atendas as suas necessidades.

Nesse sentido, dificuldades são encontradas para estabelecer um produto de qualidade. Contudo, mais importante quanto objetivar um produto de qualidade, é estabelecer um processo de desenvolvimento de qualidade. Deste modo, a aplicação de boas práticas de desenvolvimento é fundamental para obter sucesso na produção de um software.

Partindo deste contexto, este trabalho utilizou boas práticas de desenvolvimento aderindo a um conjunto disciplinado de tarefas e atividades executadas em cada fase do ciclo de desenvolvimento, na produção de um software laboratorial de análise de leite e derivados para a UENP-CLM.

Oferecendo um conjunto de atividades a serem realizadas em um projeto de software, o Processo Unificado, guiou este trabalho para a obtenção de um projeto de desenvolvimento de qualidade, por meio de artefatos e suporte para a modelagem visual do sistema - UML.

Diante disto, a realização deste trabalho pôde ter como resultados uma documentação rica do software, que foi construída e utilizada durante todo o processo de desenvolvimento, e poderá ainda ser utilizada em futuras manutenções e atualizações do software; e o software SISLAL (Sistema Laboratorial de Análise de Leite) que foi construído tendo como base a documentação gerada nas fases de iniciação e elaboração. Ressalta-se que, as etapas iniciais de um desenvolvimento de software (iniciação e elaboração) são as mais importantes, pois são elas que dão base e condução ao projeto.

Após a construção do software, o próximo passo realizado foi a avaliação do mesmo, que se deu por meio de uma apresentação aos membros do laboratório,

obtendo uma plena aceitação e satisfação com o software desenvolvido, no qual pode-se concluir que o software tem capacidade para atender as necessidades atuais do laboratório, atendendo assim, ao objetivo deste trabalho.

Acredita-se que com o uso do SISLAL, o laboratório ganhará agilidade, maior produtividade e eficiência no controle sobre documentos gerados e na emissão de resultados e relatórios.

Para trabalhos futuros, este trabalho poderia ser estendido por meio da execução da próxima fase do ciclo de desenvolvimento do software – a transição, na qual poderá ser possível analisar com mais clareza e consistência as reais vantagens obtidas com a implantação do SISLAL no Laboratório de Análise de Leite e Derivados da UENP-CLM.

Referências

AGUIAR, Yusca; LULA, Bernardo; LIMA, Carlos; LIMA, Giuseppe; GOUVEIA, Rodrigo. **Uso de Protótipos no Processo de Concepção de Interfaces do Usuário**. II Congresso de Pesquisa e Inovação da Rede Norte Nordeste de Educação Tecnológica João Pessoa - PB – 2007.

ANUNCIACÃO, Gustavo Tibério. **Uma extensão do Rational Unified Process baseada na criação de valor**. Dissertação (Mestrado em Ciência da Computação). Universidade Federal de Pernambuco. Recife, 2009.

AUAD, A. M.; SANTOS, A. M. B.; CARNEIRO, A. V. et. al. **Manual de Bovinocultura Leiteira**. 1.ed. Brasília: LK Editora; Belo Horizonte: SENAR-AR/MG; Juiz de Fora: Embrapa Gado de Leite, 2010.

AVANADE. **Global Survey: The Business Impact of Big Data**. 2010. Disponível em: <<http://www.avanade.com>> Acesso em: 01/06/2012.

BANZATO, Eduardo. **Análise de viabilidade para projetos de Automação**. 2010. Disponível em: <<http://www.guialog.com.br/artigo/Y681.htm>>. Acesso em: 02/02/2012.

BARROS, Tiago; SILVA, Mauro; ESPÍNOLA, Emerson. **State MVC: Estendendo o padrão MVC para uso no desenvolvimento de aplicações para dispositivos móveis**. 2007.

BOOCH, Grady. **UML: guia do usuário / Grady Booch, James Rumbaugh, Ivar Jacobson**. 6ª reimpressão. Rio de Janeiro: Elsevir, 2005.

BRAGA, Fabrício Portes. **Técnicas de Levantamento de requisitos**. 2008.

CAVALHEIRO, Lucas Diego Zwirtes. 2007. **Modelagem de um Processo de Desenvolvimento de Software Baseado nas Técnicas e Metodologias de Projetos Open Source**. Monografia – Universidade Federal de Pelotas. Pelotas.

EMBRAPA. **Produção, Industrialização e Comercialização**. 2011. Disponível em: <<http://www.cnpqgl.embrapa.br/nova/informacoes/estatisticas/producao/producao.php>> Acesso em: 04/06/2012.

FOWLER, Martin. **UML Essencial: Um breve guia para a linguagem padrão de modelagem de objetos**. 3. ed. Porto Alegre: Bookman, 2005.

GINIGE, Athula. MURUGESAN, San. **The essence of web engineering**. IEEE Multimedia, vol.8, nº. 1, pp.22-25, April-June 2001. Disponível em: <www.computer.org/multimedia/mu2001/u2022abs.htm>. Acesso em: 07/02/2012.

GONÇALVES, Rodrigo Franco; GAVA, Vagner Luiz; PESSÔA, Marcelo Schneck de Paula; SPINOLA, Mauro de Mesquita. **Uma proposta de processo de produção de aplicações Web.** Revista Produção, v.15, n.3, p.376-389. 2005.

JAVA. O que é a tecnologia Java e por que é necessária? Acesso em: 11/06/2012. Disponível em: <http://www.java.com/pt_BR/download/faq/whatis_java.xml>

KISHIMOTO, Edilson Tsutomu; MORAES, José Carlos. **Qual a contribuição da automação para um laboratório de ensaios?.** ENQUALAB-2008–Congresso da Qualidade em Metrologia. São Paulo, 2008.

LEITE, Cassia Rodrigues de Carvalho Ferreira. **ProMeProS: um processo de melhoria de processos de software.** 2010. 234 p. Dissertação (Mestrado) - Universidade Federal de Minas Gerais. Belo Horizonte.

MAKINO, Amanda. **Abordagem da Metodologia Rup no desenvolvimento de um Sistema de Gestão Comercial.** Monografia - Faculdade de Tecnologia de Taquaritinga. 2009.

MUGNOL, Katia Cristina Ugolini; FERRAZ, Marcos Bosi. **Sistema de Informação como ferramenta de cálculo e gestão de custos em laboratórios de análises clínicas.** Jornal Brasileiro de Patologia e Medicina Laboratorial, vol.42, n.2. Rio de Janeiro, 2006.

OLIVEIRA, Rafael Braga de. **Framework FUNCTEST: Aplicando Padrões de Software na Automação de Testes Funcionais.** 2007. 109 p. Dissertação (Mestrado) Universidade de Fortaleza, UNIFOR. Fortaleza.

PAULA FILHO, Wilson de Pádua. **Engenharia de Software.** Rio de Janeiro: LTC, 2003.

PEDRI, João Paulo. **FURBUP: Um Processo de Software para uso Acadêmico Baseado no Openup.** 2008.144 p. Monografia – Universidade Regional de Blumenau. Blumenau.

PERREIRA, Eliana Beatriz. **Uma proposta para adaptação de processos de desenvolvimento de software baseado no *Rational Unified Process*.** 2005. 125 p. Dissertação (Mestrado) – Fac. de Informática, PUCRS. Porto Alegre.

PRESSMAN, Roger. **Engenharia de Software.** 6. ed. São Paulo: McGraw-Hill, 2006.

RUP. Rational Unified Process – IBM Corp. 2006.

SCHIMTZ, Daniel Pace. Adobe Flex Builder 3.0: Conceitos e Exemplos / Daniel Pace Schimitz. Rio de Janeiro: Brasport, 2008.

SOUZA, Osnete Ribeiro de. **Processos de Apoio ao Desenvolvimento de aplicações Web.** Dissertação: (Mestrado em Ciências de Computação e Matemática Computacional). Instituto de Ciências Matemáticas e de Computação – ICMC USP. São Paulo, 2005.

SOUZA, Ricardo André Cavalcante de. **Uma Extensão do Fluxo de Análise e Projeto do RUP para o Desenvolvimento de Aplicações Web.** 2002. 178 p. Dissertação (Mestrado em Ciência da Computação) - Universidade Federal de Pernambuco. Recife.

SOMMERVILLE, Ian. **Engenharia de Software.** 6. ed. São Paulo: Addison Wesley, 2003.

TOLEDO, Ricardo Cortez. **Integração das metodologias RUP e PMI no desenvolvimento de projetos de software: A Experiência do Escritório de Projetos da Academia Nacional da Polícia Federal.** Monografia – Fundação Getúlio Vargas. Brasília-DF, 2004;

UENP. **UENP inaugura Centro Mesorregional de Excelência em Tecnologia do Leite do Norte Pioneiro.** Disponível em: <<http://www.uenp.edu.br>> Acesso em: 11/06/2012

APÊNDICE A – PLANO DE VISÃO

1 Introdução

1.1 Finalidade

A finalidade deste documento é definir a visão que os envolvidos têm do produto, em termos de suas necessidades e das funcionalidades para atendê-las, demonstrando problemas e motivações para a criação do software.

1.2 Escopo

Este documento de visão se aplica ao Software Laboratorial para a UENP-CLM – SISLAL. O projeto SISLAL tem por objetivo a automação do Laboratório de Análise de Leite e Derivados da UENP-CLM, atuando no controle de entradas e saídas de laudos, monitoramento de análises e emissão de resultados.

2 Posicionamento

2.1 Oportunidade de Negócio

Com o SISLAL implantado, o projeto “Centro Mesorregional de Excelência em Tecnologia do Leite” ganhará velocidade, organização e precisão dos resultados em seus processos de negócio, além de poder usufruir de toda informação gerada em um período 24/7 (24 horas por dia, 7 dias por semana).

2.2 Instrução do Problema

O problema	Laudos controlados manualmente
Afeta	Membros do laboratório Produtores
Impacto	Retardo do processo de trabalho Armazenamento ineficaz de laudos realizados Alto risco de perdas de documentos Ineficiência na consulta de resultados Ineficiência na consulta de laudos

	armazenados
Solução	Controle de laudo, no qual será possível cadastro e consulta de amostras e laudos submetidos pelo laboratório.
O problema	Inexistência de monitoramento de análise
Afeta	Membros do laboratório
Impacto	Descontrole de análise a serem submetidas Descontrole sobre o status atual de cada análise
Solução	Monitoramento das análises, no qual será possível consultar a quantidade de análises a serem submetidas e o status de cada uma delas.

O problema	Inexistência de controle de evolução da qualidade do leite e derivados
Afeta	Membros do laboratório Produtores
Impacto	Ineficiência da qualidade do serviço prestado Retardo da emissão do resultado caso o serviço seja requisitado pelo produtor
Solução	Controle de evolução, permitindo através de gráficos/relatórios, análises do laudo atual com laudos anteriores de acordo com seu produtor.

O problema	Ausência de registro do responsável pela análise
Afeta	Membros do laboratório
Impacto	Atendente não tem informação suficiente sobre a análise, para atender os produtores Desconhecido o número de análises realizadas

	pelo membro do projeto
Solução	Armazenar informações do responsável pela análise junto ao laudo.

2.3 Instrução sobre a Posição do Produto

Para	Laboratórios de Análise de Leite e Derivados
Que	Desejam monitorar análises submetidas, armazenar laudos e melhorar o sistema de emissão de resultado
O SISLAL	é um produto de software
Que	Automatiza a rotina de um laboratório de análise, atuando no controle de entrada e saída de laudos, monitoramento de análises e emissão de resultados.
Ao contrário de	Do controle manual dos laudos, utilizando anotações, ocasionando em um grande volume de documentos impressos, dificultando seu manuseio e análise sobre ele.
Nosso produto	Oferece comodidade, facilidade e agilidade no processo de armazenamento e análise dos laudos.

3 Descrições do Envolvido

3.1 Resumo do Envolvido

Nome	Descrição	Responsabilidades
Bolsista DTI-B CNPq	Membro do projeto "Centro Mesorregional de Excelência em	- Estabelecer os requisitos - Verificação e aceitação do escopo e do design da interface

Nome	Descrição	Responsabilidades
	Tecnologia do Leite”	com o usuário
Usuário	Usuário final do sistema	- Manipular o sistema

3.2 Ambiente do Usuário

O sistema proposto executará em ambiente web, sendo assim, será esperado que os usuários tenham conexão a Internet, na qual será possível acessar o sistema em qualquer horário do dia com browser e sistema operacional de sua preferência.

4 Visão Geral do Produto

4.1 Perspectiva do Produto

Espera-se com este sistema a informatização dos processos do laboratório, visando um rigoroso controle de suas atividades, bem como prover todas as informações necessárias para os coordenadores do projeto. Destaca-se que o sistema será autossuficiente, não necessitando a integração com outros.

4.2 Premissas e Dependências

- O sistema será hospedado em um servidor da UENP-CLM.
- Estará disponível nos principais sistemas operacionais.
- Será desenvolvido utilizando a linguagem de programação Java.

4.3 Necessidades

Necessidade	Prioridade	Liberação Planejada
Controle de laudo	Alta	Organização e agilidade no armazenamento e consulta
Monitoramento de análise	Alta	Controle do ciclo de vida, monitorando desde sua entrada até a saída, visualização do status atual.
Controle de evolução da amostra	Alta	Auxílio no monitoramento da qualidade do produto analisado em períodos distintos
Manter produtores	Alta	Controle do serviço prestado para cada

		produtor
Manter de membros do laboratório	Alta	Monitoramento do serviço prestado pelo indivíduo ao projeto
Relatórios	Média	Obtenção de informações crucias para a continuidade do projeto e qualidade do serviço prestado

4.4 Alternativas e Competição

A concorrência oferece este tipo de serviço, porém oferece gerenciamento de laboratórios em geral e, ao contrário deste projeto, com licenças pagas.

5 Outros Requisitos do Produto

Consulte o documento Especificação Suplementar.

APÊNDICE B – PLANO DO PROJETO

1 Introdução

1.1 Finalidade

A finalidade deste documento é reunir todas as informações necessárias ao controle do projeto. Ele descreve a abordagem dada ao desenvolvimento do software e é o plano de nível mais alto gerado e usado pelos gerentes para direcionar o esforço de desenvolvimento.

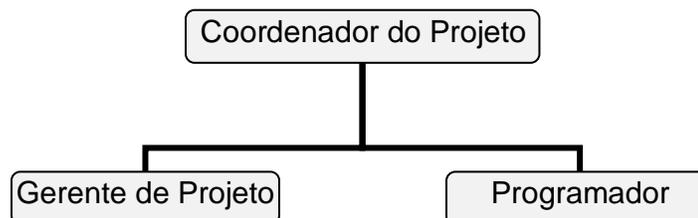
1.2 Escopo

Este plano descreve o plano geral a ser utilizado pela equipe para desenvolver o sistema. Os detalhes das iterações individuais serão descritos no Plano de Iteração.

2 Organização do Projeto

2.1 Estrutura Organizacional

A equipe do projeto será organizada da seguinte forma:



2.2 Interfaces Externas

A equipe se relacionará com membros do projeto “Centro Mesorregional de Excelência em Tecnologia do Leite” para reunir requisitos e revisar protótipos.

2.3 Funções e Responsabilidades

A tabela a seguir mostra as funções representadas no diagrama do projeto acima e suas responsabilidades principais.

Função	Responsabilidade
Coordenador do Projeto	Acompanhamento das atividades e interação com terceiros.
Gerente de Projeto	Planeja as atividades, aloca os recursos, molda prioridades, interação com cliente e usuário e gera os artefatos do projeto.

3 Planejamento do Projeto

O desenvolvimento do sistema será conduzido utilizando estratégia incremental, ocorrendo em diversas iterações dentro de uma fase. Esta estratégia determina a visão do negócio e requisitos do sistema e, em seguida, realiza o desenvolvimento em uma sequência de builds incorporados até o sistema estar completo.

3.1 Estimativa

A estimativa do projeto foi definida de acordo com o Plano de Iteração e utilizando a técnica *Planning Poker*. Deste modo, conclui-se que o tempo estimado para o desenvolvimento deste projeto é de 109 dias.

3.2 Iterações

Fase	Iteração	Objetivo
Iniciação	I1	Definir o sistema e criar suporte necessário as próximas fases.
Elaboração	E1	Concluir análise e design para todos os casos de uso, criar arquitetura estável.
Construção	C1	Implementar e testar a camada de persistência, de negócio, de controle e alguns casos de uso
	C2	Implementar e testar alguns casos de uso para fornecer a versão beta.
	C3	Implementar e testar o restante de casos de uso, corrigir defeitos da versão beta de acordo com o feedback dos testes; e definir a versão final
Transição	T1	Implantar versão final do software para o usuário, oferecer o manual de usuário e treinamento

3.3 Marcos

Fase	Marco
Iniciação	Objetivos do ciclo de vida – análise dos objetivos do ciclo de vida do projeto e decide se deseja prosseguir com o projeto ou cancelá-lo.
Elaboração	Arquitetura de Ciclo de Vida – examinar os objetivos e o escopo detalhados do sistema, a opção de arquitetura e a resolução dos principais riscos.
Construção	Recurso Operacional Inicial – o produto estará pronto para ser passado para a próxima fase, todas funcionalidades e testes foram concluídos.
Transição	Liberação do Produto – decide se os objetivos foram atendidos, e se outro ciclo de desenvolvimento deve ser iniciado de acordo com a revisão e aceitação do cliente. Além do software, foi desenvolvido um manual do usuário.

3.3.1 Critérios de Avaliação e Artefatos do Marco

Fase	Critérios de Avaliação	Artefatos
Iniciação	<ul style="list-style-type: none"> • Consentimento dos envolvidos sobre a definição do escopo • Consenso de que o conjunto correto de requisitos foi capturado e de que existe uma compreensão compartilhada desses requisitos. • Consenso de que as estimativas de programação, as prioridades, os riscos e o processo de desenvolvimento são adequados. • Todos os riscos foram identificados e existe uma estratégia atenuante para cada um. 	<ul style="list-style-type: none"> - Visão - Lista de Riscos - Plano do Projeto - Plano de Iteração - Glossário - Modelo de Caso de Uso
Elaboração	<ul style="list-style-type: none"> • A visão e os requisitos do produto são estáveis • A arquitetura é estável • As abordagens principais a serem usadas no teste e na avaliação foram comprovadas • O teste e a avaliação de protótipos executáveis demonstraram que os principais elementos de risco foram tratados e resolvidos com credibilidade. • O plano de iteração para a fase de construção tem detalhes e fidelidade suficientes para 	<ul style="list-style-type: none"> - Protótipos - Lista de Riscos - Documento de Arquitetura de Software - Modelo de Design - Modelo de Dados

	permitir o avanço do trabalho	<ul style="list-style-type: none"> - Visão - Plano de Projeto - Plano de Iteração - Modelo de Caso de Uso
Construção	<ul style="list-style-type: none"> • Release do produto é estável e desenvolvido o suficiente para ser implantado • Todos os envolvidos estão prontos para a transição 	<ul style="list-style-type: none"> - “O Sistema” - Plano de Implementação - Modelo de Implementação - Conjunto de Teste - Material de Suporte do Usuário - Plano de Iteração - Infraestrutura de Desenvolvimento - Modelo de Dados
Transição	<ul style="list-style-type: none"> • O usuário está satisfeito 	<ul style="list-style-type: none"> - Build do Produto - Material de Suporte do Usuário - Elementos de Implementação

4 Recursos do Projeto

4.1 Plano de Equipe

Os indivíduos desse projeto são nomeados na seção 2.1.

APÊNDICE C – MODELO DE CASOS DE USO

1 Introdução

1.1 Finalidade

Esse documento é um modelo das funções pretendidas do sistema, utilizado como fonte de informações essencial para atividades de análise, design e teste.

2 Catálogo de Atores

Administrador	Membro responsável pelo gerenciamento de usuários e operação das funções primordiais do sistema.
Estagiário	Membro encarregado de gerenciar funções básicas do sistema.
Produtor	Possui acesso a seus laudos no sistema

3 Casos de Uso

3.1 Manter Membro do Laboratório

Atores	Administrador
Objetivo	Manter os dados dos membros do laboratório
Pré-requisito	Usuário logado no sistema com perfil de acesso Administrador
Fluxo Principal	<ol style="list-style-type: none"> 1. O ator inicia caso de uso selecionando opção “Gerenciar” e depois “Membros”. 2. O sistema exibe os membros cadastrados, contendo os seguintes dados: nome, e-mail, telefone e status. 3. O ator pode selecionar um membro para a operação. 4. O ator seleciona uma operação: Novo (Subfluxo 1.1), Editar (Subfluxo 1.2) ou Excluir (Subfluxo 1.3) 5. O ator seleciona a opção “Sair” e o caso de uso é encerrado.
Fluxo Alternativo	

SubFluxo 1.1	Novo Membro
Objetivo	Cadastrar um novo membro do laboratório
Fluxo Principal	<ol style="list-style-type: none"> 8. O sistema solicita os dados para o cadastro do novo membro. 9. O ator informa o nome, CPF, e-mail, telefone, nome de usuário, senha e perfil.

	<p>10. O ator seleciona a operação “Salvar”.</p> <p>11. O sistema valida os dados informados.</p> <p>12. O sistema grava as informações na base de dados.</p> <p>13. O sistema emite mensagem “Cadastrado com sucesso”.</p> <p>14. O sistema retorna ao caso de uso UC1.</p>
Fluxo Alternativo	<p>Dados inválidos</p> <p>2.a. O sistema emite alerta para o usuário</p> <p>2.b. O sistema retorna ao passo 2 do fluxo principal</p> <p>Operação Cancelada</p> <p>3.a O sistema retorna ao caso de uso UC1.</p> <p>Duplicidade de dados</p> <p>5.a. O sistema informa ao usuário problema de duplicidade de dados em relação a nome de usuário e/ou CPF.</p> <p>5.b. O sistema retorna ao passo 2 do fluxo principal</p>

SubFluxo 1.2	Editar Membro
Objetivo	Atualizar os dados do membro cadastrado
Pré-requisito	Possuir membro cadastrado
Fluxo Principal	<p>1. O ator seleciona o membro que deseja editar os dados</p> <p>2. O sistema exibe as informações (nome, CPF, e-mail, telefone, nome de usuário, senha e perfil) do membro.</p> <p>3. O ator modifica os dados.</p> <p>4. O ator seleciona a operação “Salvar”.</p> <p>5. O sistema valida os dados informados.</p> <p>6. O sistema grava as informações na base de dados.</p> <p>7. O sistema emite mensagem “Dados atualizados com sucesso”.</p> <p>8. O sistema retorna ao caso de uso UC1.</p>
Fluxo Alternativo	<p>Dados inválidos</p> <p>3.a. O sistema emite alerta para o usuário</p> <p>3.b. O sistema retorna ao passo 3 do fluxo principal</p> <p>Operação Cancelada</p> <p>4.a O sistema retorna ao caso de uso UC1.</p> <p>Duplicidade de dados</p> <p>6.a. O sistema informa ao usuário problema de duplicidade de dados em relação a nome de usuário e/ou CPF.</p>

	6.b. O sistema retorna ao passo 3 do fluxo principal
SubFluxo 1.3	Excluir Membro
Objetivo	Exclui membro que está cadastrado
Pré-requisito	Possuir membro cadastrado
Fluxo Principal	<ol style="list-style-type: none"> 1. O ator seleciona o membro que deseja excluir 2. O sistema emite mensagem de confirmação de exclusão 3. O ator seleciona a opção "Confirmar". 4. O sistema exclui o membro da base de dados. 5. O sistema emite mensagem "Excluído com sucesso". 6. O sistema retorna ao caso de uso UC1.
Fluxo Alternativo	<p>Operação Cancelada</p> <p>3.a. O sistema retorna ao caso de uso UC1.</p>

3.2 Manter Produtor

UC2	Manter Produtor
Atores	Administrador, Estagiário
Objetivo	Manter os dados dos produtores das amostras
Pré-requisito	Usuário logado no sistema.
Fluxo Principal	<ol style="list-style-type: none"> 1. O ator inicia caso de uso selecionando opção "Gerenciar" e depois "Produtor". 2. O sistema exibe os produtores cadastrados, contendo os seguintes dados: nome, telefone e cidade. 3. O ator pode selecionar um produtor para a operação. 4. O ator seleciona uma operação: Novo (Subfluxo 2.1), Editar (Subfluxo 2.2) ou Excluir (Subfluxo 2.3). 5. O ator seleciona a opção "Sair" e o caso de uso é encerrado.
Fluxo Alternativo	

SubFluxo 2.1	Novo Produtor
Objetivo	Cadastrar um novo produtor
Fluxo Principal	<ol style="list-style-type: none"> 1. O sistema solicita os dados para o cadastro do novo produtor. 2. O ator informa o nome, telefone e cidade. 3. O ator seleciona a operação "Salvar". 4. O sistema valida os dados informados. 5. O sistema grava as informações na base de dados. 6. O sistema emite mensagem "Cadastrado com sucesso". 7. O sistema retorna ao caso de uso UC2.

Fluxo Alternativo	<p>Dados inválidos</p> <p>2.a. O sistema emite alerta para o usuário</p> <p>2.b. O sistema retorna ao passo 2 do fluxo principal</p> <p>Operação Cancelada</p> <p>3.a O sistema retorna ao caso de uso UC2.</p>
--------------------------	---

SubFluxo 2.2	Editar Produtor
Objetivo	Atualizar os dados do produtor cadastrado
Pré-requisito	Possuir produtor cadastrado
Fluxo Principal	<ol style="list-style-type: none"> 1. O ator seleciona o produtor que deseja editar os dados 2. O sistema exibe as informações (nome, telefone e cidade) do produtor. 3. O ator modifica os dados. 4. O ator seleciona a operação "Salvar". 5. O sistema valida os dados informados. 6. O sistema grava as informações na base de dados. 7. O sistema emite mensagem "Dados atualizados com sucesso". 8. O sistema retorna ao caso de uso UC2.
Fluxo Alternativo	<p>Dados inválidos</p> <p>3.a. O sistema emite alerta para o usuário</p> <p>3.b. O sistema retorna ao passo 3 do fluxo principal</p> <p>Operação Cancelada</p> <p>4.a O sistema retorna ao caso de uso UC2.</p>

SubFluxo 2.3	Excluir Produtor
Objetivo	Excluir produtor cadastrado
Pré-requisito	Possuir produtor cadastrado
Fluxo Principal	<ol style="list-style-type: none"> 7. O ator seleciona o produtor que deseja excluir 8. O sistema emite mensagem de confirmação de exclusão 9. O ator seleciona a opção "Confirmar". 10. O sistema exclui o produtor da base de dados. 11. O sistema emite mensagem "Excluído com sucesso". 12. O sistema retorna ao caso de uso UC2.
Fluxo Alternativo	<p>Operação Cancelada</p> <p>3.a. O sistema retorna ao caso de uso UC2.</p>

3.3 Manter Laudos

UC3	Manter Laudos
Atores	Administrador
Objetivo	Manter laudos produzidos no laboratório
Pré-requisito	Usuário logado no sistema.
Fluxo Principal	<ol style="list-style-type: none"> 1. O ator inicia caso de uso selecionando opção “Gerenciar” e depois “Laudo”. 2. O sistema exibe os laudos cadastrados, contendo os seguintes dados: número de identificação, produtor, data de entrada, data de saída, status e publicado. 3. O sistema apresenta ao usuário diversas opções de busca de laudo: por status, por produtor, pelo número de identificação e buscar todos. 4. O ator pode selecionar um laudo para a operação. 5. O ator seleciona uma operação: Novo (Subfluxo 3.1), Editar (Subfluxo 3.2) ou Excluir (Subfluxo 3.3). 6. O ator seleciona a opção “Sair” e o caso de uso é encerrado 7. O ator seleciona a opção “Sair” e o caso de uso é encerrado.
Fluxo Alternativo	

SubFluxo 3.1	Novo Laudo
Objetivo	Cadastrar um novo laudo
Pré-requisito	Possuir produtor cadastrado
Fluxo Principal	<ol style="list-style-type: none"> 1. O sistema fornece uma lista de produtores cadastrados e solicita os demais dados para o cadastro do novo laudo. 2. O ator informa o produtor 3. O ator informa identificação do material, número de amostras, local de produção, local de colheita, marca, tipo, identificação do lote, data de fabricação, termo de colheita e data de entrada. 4. O ator seleciona a operação “Salvar”. 5. O sistema valida os dados informados. 6. O sistema grava as informações na base de dados. 7. O sistema emite mensagem “Cadastrado com sucesso”. 8. O sistema retorna ao caso de uso UC3.
Fluxo Alternativo	<p>Produtor não cadastrado</p> <p>2.a. O sistema direciona ao subfluxo 2.1 do caso de uso UC2.</p> <p>Dados inválidos</p> <p>2.a. O sistema emite alerta para o usuário</p> <p>2.b. O sistema retorna ao passo 2 do fluxo principal</p>

	Operação Cancelada 3.a O sistema retorna ao caso de uso UC2.
--	---

SubFluxo 3.2	Editar Laudo
Objetivo	Atualizar os dados do laudo cadastrado
Pré-requisito	Possuir laudo cadastrado
Fluxo Principal	<ol style="list-style-type: none"> 1. O ator seleciona o laudo que deseja editar os dados 2. O sistema exibe todas as informações do laudo. 3. O ator modifica os dados. 4. O ator seleciona a operação "Salvar". 5. O sistema valida os dados informados. 6. O sistema grava as informações na base de dados. 7. O sistema emite mensagem "Dados atualizados com sucesso". 8. O sistema retorna ao caso de uso UC3.
Fluxo Alternativo	<p>Dados inválidos</p> <p>3.a. O sistema emite alerta para o usuário</p> <p>3.b. O sistema retorna ao passo 3 do fluxo principal</p> <p>Operação Cancelada</p> <p>4.a O sistema retorna ao caso de uso UC3.</p>

SubFluxo 3.3	Excluir Laudo
Objetivo	Exclui laudo cadastrado
Pré-requisito	Possuir laudo cadastrado
Fluxo Principal	<ol style="list-style-type: none"> 1. O ator seleciona o laudo que deseja excluir 2. O sistema emite mensagem de confirmação de exclusão 3. O ator seleciona a opção "Confirmar". 4. O sistema exclui o produtor da base de dados. 5. O sistema emite mensagem "Excluído com sucesso". 6. O sistema retorna ao caso de uso UC3.
Fluxo Alternativo	<p>Operação Cancelada</p> <p>3.a. O sistema retorna ao caso de uso UC3.</p>

3.4 Manter Análises

UC4	Manter Análises
------------	------------------------

Atores	Administrador, Estagiário
Objetivo	Manter os resultados das análises da amostra
Pré-requisito	Usuário logado no sistema Laudo cadastrado com status “Em Andamento”
Fluxo Principal	<ol style="list-style-type: none"> 1. O ator inicia caso de uso selecionando opção “Análises”. 2. O sistema oferece duas opções ao usuário: “Gerenciar Itens de Análise” e “Gerenciar Análise”. 3. O ator seleciona uma opção. 4. Para a opção “Gerenciar Itens de Análise” o sistema direciona ao subfluxo 4.1. 5. Para a opção “Gerenciar Análises” o sistema exibe os laudos cadastrados cujo status esteja “Em Andamento”, contendo os seguintes dados: data de entrada, produtor e membro responsável. <ol style="list-style-type: none"> a. O ator seleciona um laudo para manipular. b. O ator seleciona uma operação: Inserir (Subfluxo 4.2), Editar (Subfluxo 4.3) ou Excluir (Subfluxo 4.4). 6. O ator seleciona a opção “Sair” e o caso de uso é encerrado.
Fluxo Alternativo	<p>Não existe laudo cadastrado</p> <p>5.a O sistema direciona ao caso de uso UC3.</p>

SubFluxo 4.1	Gerenciar Itens de Análise
Objetivo	Manter os itens de uma análise
Fluxo Principal	<ol style="list-style-type: none"> 1. O sistema apresenta uma lista de itens de análise cadastrados e fornece ao usuário as opções: Novo, Editar e Excluir. 2. O ator seleciona um item para manipular. 3. O ator seleciona uma opção: <ol style="list-style-type: none"> a. Para a opção “Novo”: <ol style="list-style-type: none"> i. O sistema solicita os dados para o cadastro ii. O ator informa os dados (descrição e medidas) iii. O ator seleciona a opção “Salvar”. iv. O sistema grava os dados na base de dados. v. O sistema emite mensagem de sucesso. vi. O sistema retorna ao passo 1 do fluxo principal. b. Para a opção “Editar”: <ol style="list-style-type: none"> i. O sistema informa os dados do item selecionado. ii. O ator modifica os dados. iii. O ator seleciona a opção “Salvar”. iv. O sistema emite mensagem de sucesso v. O sistema retorna ao passo 1 do fluxo principal. c. Para a opção “Excluir”: <ol style="list-style-type: none"> i. O ator seleciona o item que deseja excluir ii. O sistema emite mensagem de confirmação de exclusão iii. O ator seleciona a opção “Confirmar”. iv. O sistema exclui o item da base de dados. v. O sistema emite mensagem de sucesso. vi. O sistema retorna ao passo 1 do fluxo principal. 4. O ator seleciona a opção “Sair” e o caso de uso é

	encerrado.
Fluxo Alternativo	Operação Cancelada 3 O sistema retorna ao SubFluxo 4.1.

SubFluxo 4.2	Inserir Resultado
Objetivo	Inserir um novo resultado de uma análise
Pré-requisito	Possuir itens de análise cadastrado
Fluxo Principal	<ol style="list-style-type: none"> 1. O sistema apresenta ao usuário uma lista de itens de análise e solicita ao usuário preencher o resultado da análise de acordo com o item. 2. O sistema solicita o dado para a inserção do resultado. 3. O ator informa o resultado e o responsável pela análise. 4. O ator seleciona a operação “Salvar”. 5. O sistema valida os dados informados. 6. O sistema grava as informações na base de dados. 7. O sistema emite mensagem “Cadastrado com sucesso”. 8. O sistema retorna ao passo 1 do fluxo principal. 9. O sistema retorna ao passo 5 do fluxo principal do caso de uso UC4.
Fluxo Alternativo	<p>Não há itens de análise cadastrados</p> <p>O sistema direciona ao subfluxo 4.1</p> <p>Operação Cancelada</p> <p>4.a O sistema retorna ao caso de uso UC4.</p>

SubFluxo 4.3	Editar Resultado
Objetivo	Atualizar o resultado de uma análise
Pré-requisito	Possuir resultado cadastrado
Fluxo Principal	<ol style="list-style-type: none"> 1. O sistema apresenta ao usuário uma lista de itens de análise com os resultados gravados. 2. O ator seleciona o item e modifica os dados do mesmo. 3. O ator seleciona a operação “Salvar”. 4. O sistema valida os dados informados. 5. O sistema grava as informações na base de dados. 6. O sistema emite mensagem “Dados atualizados com sucesso”. 7. O sistema retorna ao passo 1 do fluxo principal. 8. O sistema retorna ao passo 5 do fluxo principal do caso de uso UC4.
Fluxo Alternativo	<p>Operação Cancelada</p> <p>3.a O sistema retorna ao caso de uso UC4.</p>

SubFluxo 4.3	Excluir Resultado
Objetivo	Excluir resultado da análise cadastrado
Pré-requisito	Possuir resultado cadastrado
Fluxo Principal	<ol style="list-style-type: none"> 1. O sistema apresenta ao usuário uma lista de itens de análise com os resultados gravados. 2. O ator seleciona o item na qual terá seu resultado excluído. 3. O sistema emite mensagem de confirmação de exclusão 4. O ator seleciona a opção “Confirmar”. 5. O sistema exclui o resultado da base de dados. 6. O sistema emite mensagem “Excluído com sucesso”. 7. O sistema retorna ao passo 1 do fluxo principal. 8. O sistema retorna ao passo 5 do fluxo principal do caso de uso UC4
Fluxo Alternativo	Operação Cancelada 4.a. O sistema retorna ao caso de uso UC4.

3.5 Gerar Relatório

UC5	Gerar Relatório
Atores	Administrador
Objetivo	Organizar dados, informações e resultados e transmitir ao usuário de maneira satisfatória.
Pré-requisito	Usuário logado no sistema com perfil de acesso Administrador Laudos Cadastrados
Fluxo Principal	<ol style="list-style-type: none"> 1. O ator inicia caso de uso selecionando opção “Relatórios”. 2. O sistema apresenta ao usuário os relatórios disponíveis. 3. O ator seleciona o relatório desejado: <ol style="list-style-type: none"> a. Laudos realizados por período (subfluxo 5.1) b. Laudos realizados por produtor (subfluxo 5.2) 4. O ator seleciona a opção “Sair” e o caso de uso é encerrado.
Fluxo Alternativo	

SubFluxo 5.1	Emitir Laudos realizados por período
Objetivo	Emitir laudos produzidos em um determinado período
Fluxo Principal	<ol style="list-style-type: none"> 1. O sistema solicita os dados para gerar o relatório. 2. O ator informa o período desejado (data inicial e data final) 3. O ator seleciona a operação “Emitir relatório”. 4. O sistema busca na base de dados os laudos produzidos no período fornecido pelo usuário e formata os dados de forma

	legível ao usuário. 5. O sistema exibe o relatório para o usuário. 6. O sistema retorna ao caso de uso UC5.
Fluxo Alternativo	Não há resultados 4.a. O sistema emite alerta para o usuário 2.b. O sistema retorna ao passo 1 do fluxo principal

SubFluxo 5.2	Emitir Laudos realizados por produtor
Objetivo	Emitir laudos produzidos por um produtor
Fluxo Principal	<ol style="list-style-type: none"> 1. O sistema apresenta ao usuário uma lista de produtores cadastrados. 2. O ator seleciona o produtor desejado. 3. O ator seleciona a operação "Emitir relatório". 4. O sistema busca na base de dados os laudos produzidos pelo produtor fornecido pelo usuário e formata os dados de forma legível ao usuário. 5. O sistema exibe o relatório para o usuário. 6. O sistema retorna ao caso de uso UC5.
Fluxo Alternativo	Não há resultados 4.a. O sistema emite alerta para o usuário 2.b. O sistema retorna ao passo 1 do fluxo principal

3.6 Medir Evolução

UC6	Medir Evolução
Atores	Administrador
Objetivo	Apresenta ao usuário comparações de resultados de análises dos laudos de um produtor, proporcionando conhecimento quanto a evolução da amostra.
Pré-requisito	Usuário logado no sistema com perfil de acesso Administrador Laudos Cadastrados
Fluxo Principal	<ol style="list-style-type: none"> 1. O sistema apresenta uma lista de produtores cadastrados 2. O ator seleciona o produtor desejado 3. O sistema exibe uma lista de laudos produzidos deste produtor 4. O ator seleciona os laudos necessários para medir a evolução 5. O ator seleciona a operação "Medir Evolução" 6. O sistema realiza busca pelos resultados dos laudos selecionados e formata de maneira legível ao usuário. 7. O sistema exibe o resultado 8. O sistema retorna ao caso de uso UC6

Fluxo Alternativo	Não há resultados 5.a. O sistema emite alerta para o usuário 5.b. O sistema retorna ao passo 1 do fluxo principal
--------------------------	---

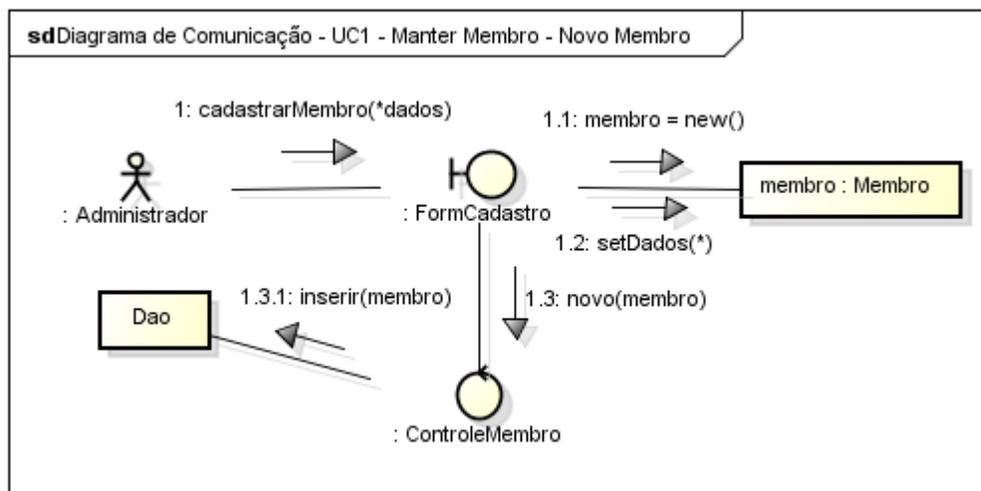
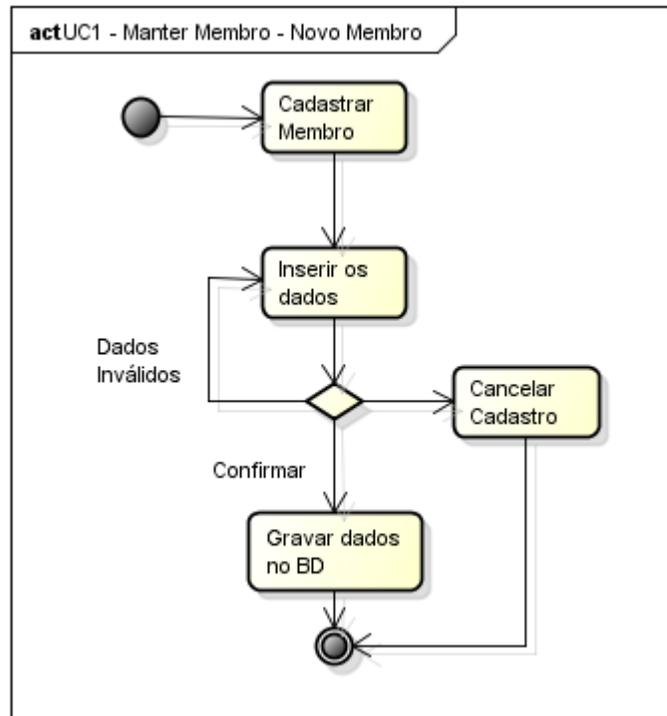
3.7 Visualizar Laudo

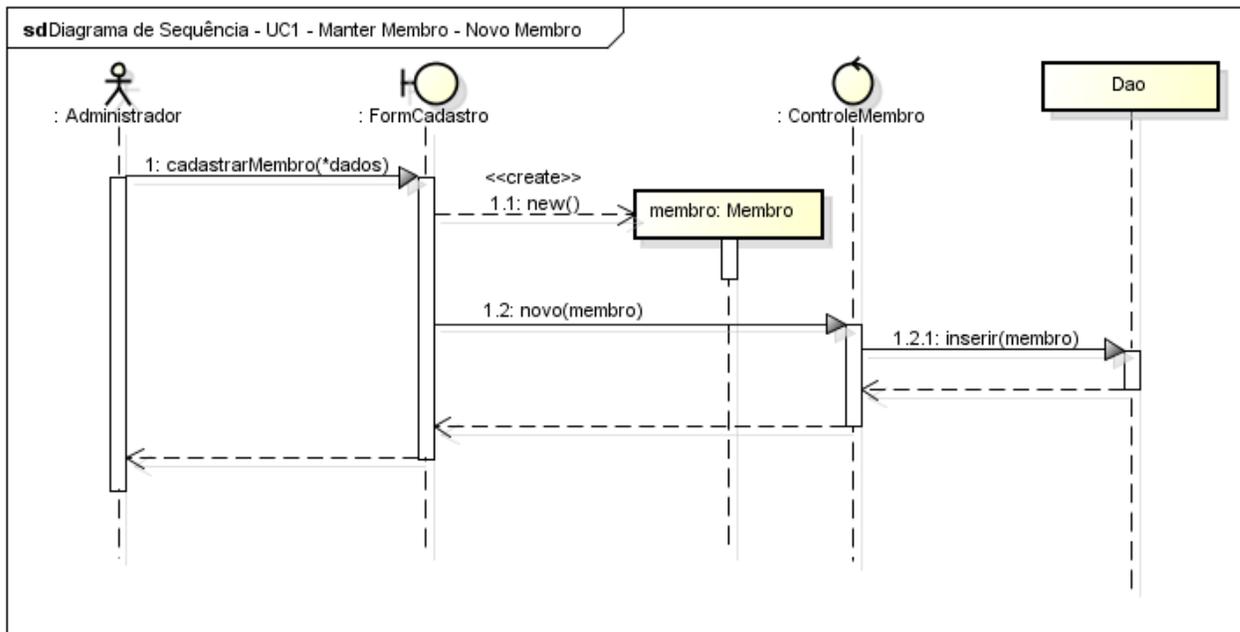
UC7	Visualizar Laudo
Atores	Produtor
Objetivo	Apresenta ao produtor os seus respectivos laudos para a visualização dos resultados das análises
Pré-requisito	Usuário logado no sistema com perfil de acesso Administrador Laudos Cadastrados
Fluxo Principal	<ol style="list-style-type: none"> 1. O sistema apresenta uma lista de laudos cadastrados 2. O ator seleciona o laudo desejado 3. O ator seleciona a operação "Visualizar Laudo" 4. O sistema realiza busca pelos resultados do laudo selecionado e formata de maneira legível ao usuário. 5. O sistema exibe o resultado
Fluxo Alternativo	Não há resultados 5.a. O sistema emite alerta para o usuário 5.b. O sistema retorna ao passo 1 do fluxo principal

APÊNDICE D – MODELO DE DESIGN

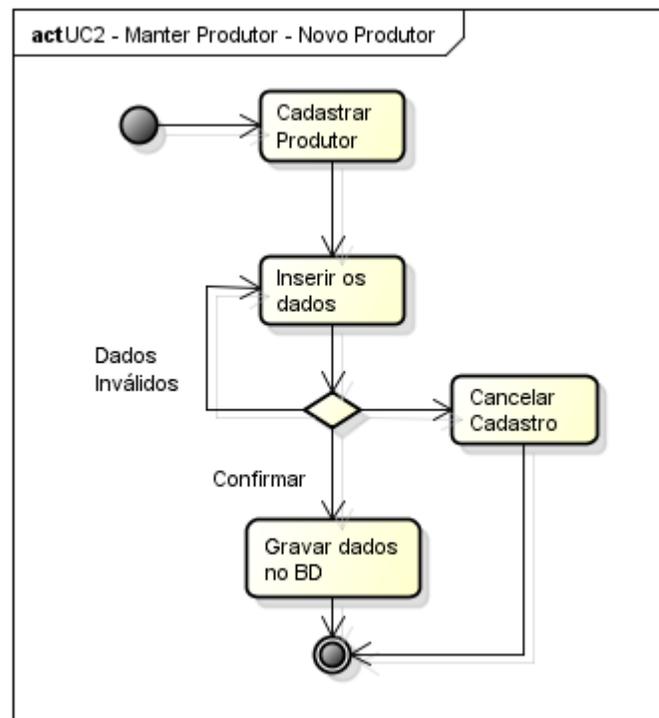
A seguir são apresentados alguns diagramas criados para este artefato.

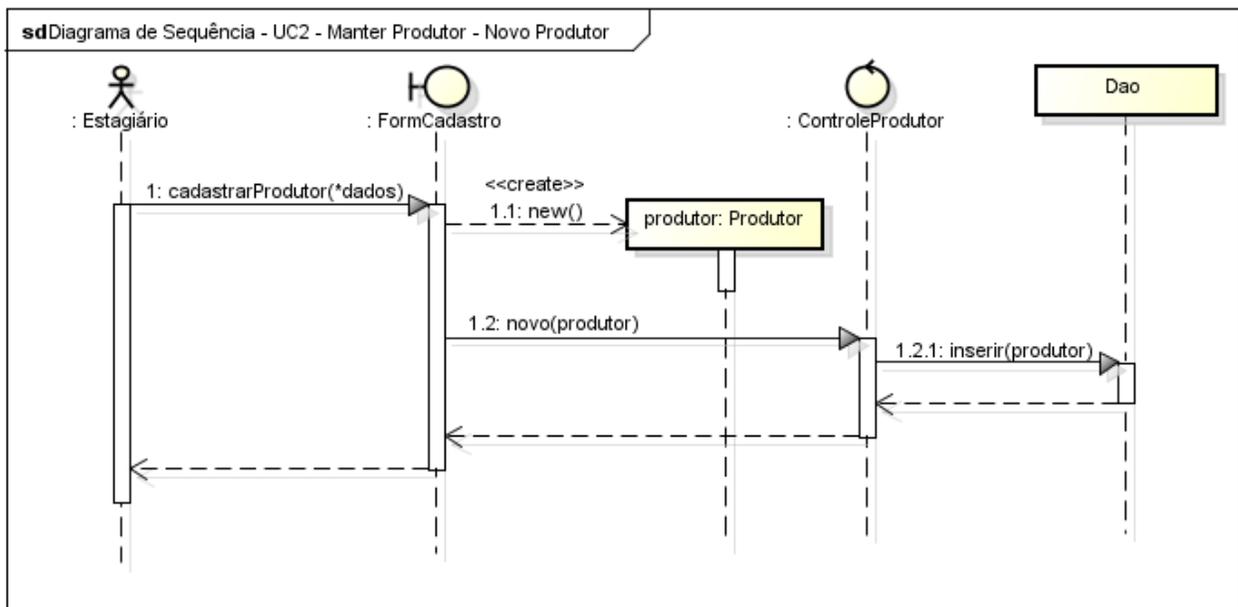
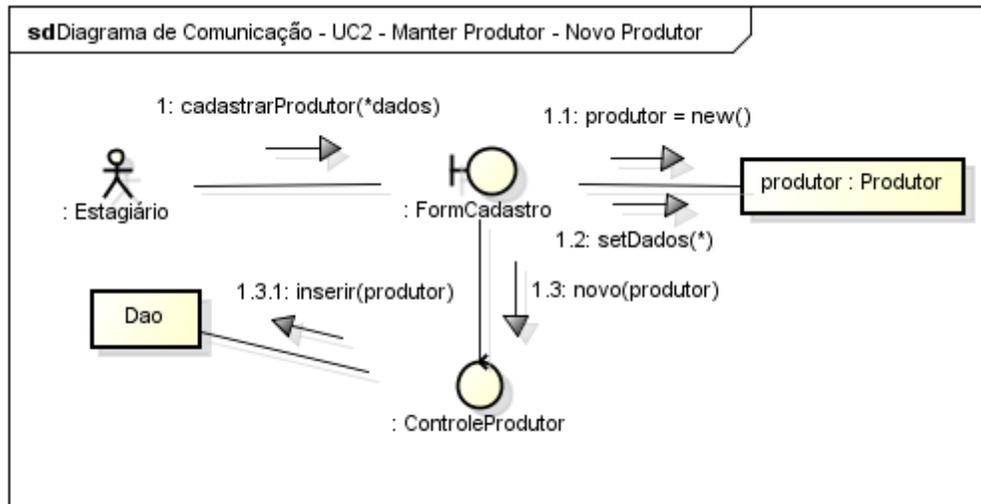
Diagramas Módulo Membro:



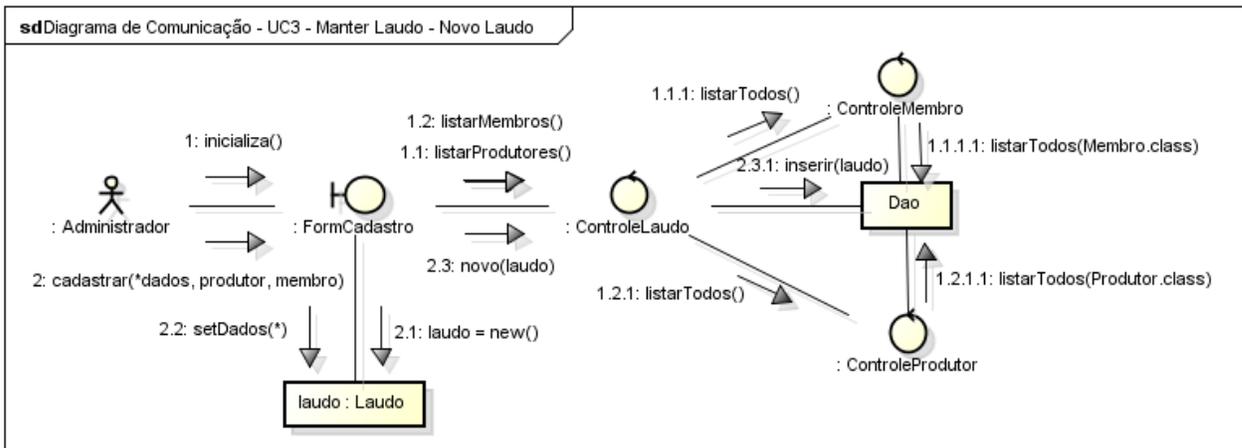
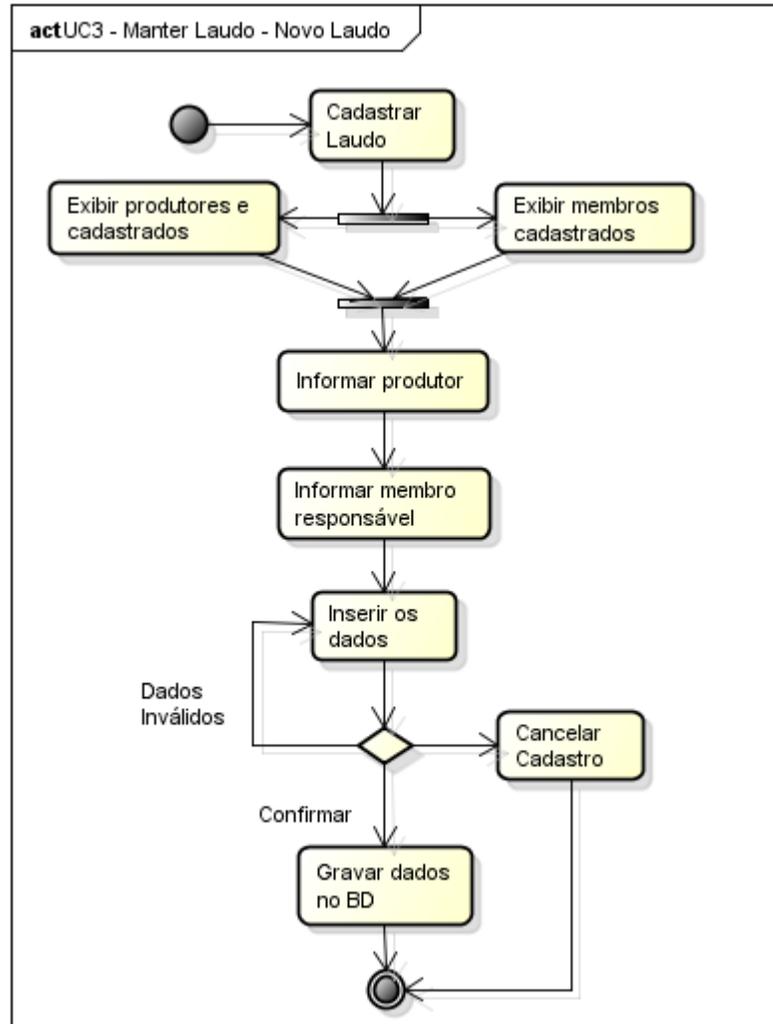


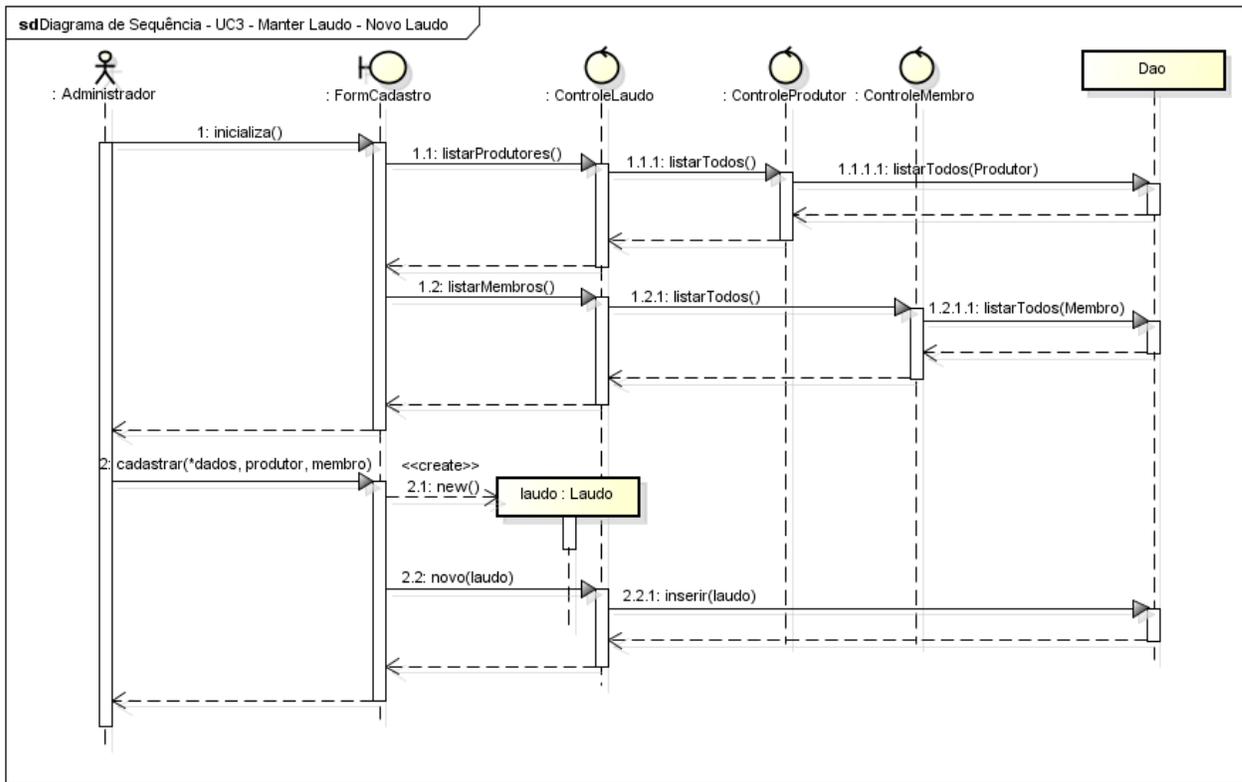
Diagramas Módulo Produtor:



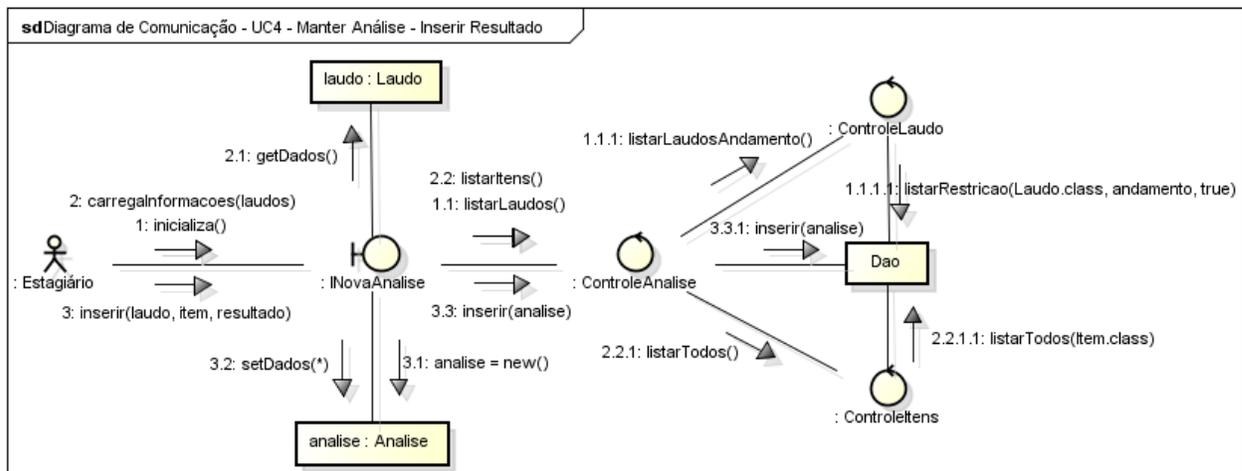


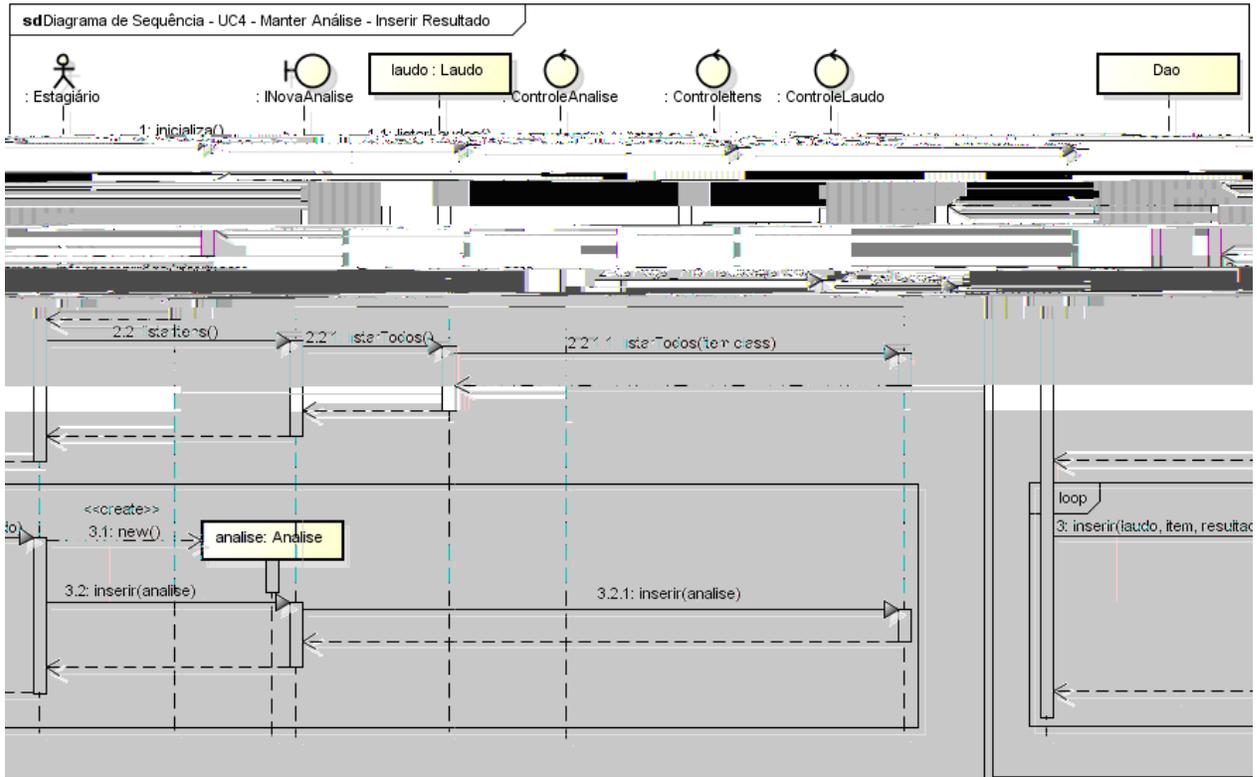
Diagramas Módulo Laudo:





Diagramas Módulo Análise:





ANEXO A – ENTREVISTA: ELICITAÇÃO DE REQUISITOS

Processo de negócio

- Qual é o problema?
- Porque esse problema existe?
- Qual é o motivo para querer resolver este problema?
- Como você gostaria de resolver o problema?
- Qual é o valor de uma solução bem-sucedida?
- Quais serviços/produto você produz? Para quem?
- Quais dados são armazenados e quais resultados são gerados?
- Como esses dados e resultados são armazenados atualmente?

Usuário

- Quantos usuários utilizariam o sistema?
- Quem são os usuários?
- Quais são as suas responsabilidades principais?

Projeto

- Quais são suas expectativas para a utilidade do produto?
- Quais problemas de negócios este produto poderia criar?
- Quais riscos poderiam existir para o usuário?
- Qual ambiente o produto encontrará?
- Quais são suas expectativas de utilidade?

Outros Requisitos

- Quais os dados necessários para o cadastro?
- O sistema deverá utilizar relatórios?
- Que tipo de cópia impressa e documentação on-line você precisa?

- Se houver, quais são os padrões e requisitos normativos ou ambientais que devem ser suportados?
- Você se lembra de algum outro requisito que deveríamos conhecer?

Hand-off

- Há outras perguntas que eu deveria fazer a você?
- Se eu precisar fazer perguntas de acompanhamento, posso ligar?
- Você estaria disposto a participar em uma revisão de requisitos?