Nesta seção você encontra artigos voltados para testes, processo, modelos, documentação, entre outros

Definição de ciclo de vida e processo de software

Um estudo de caso

Arthur Macedo
arthur_macedo@hotmail.com
Graduando em Ciência da Computação pela UNIFACS – Universidade Salvador. Atua também como Consultor de TI na COMMIT – Empresa Jr de Computação da UNIFACS. Atualmente é Analista de Suporte Jr na Provide IT – Suporte e Treinamento

Nicolli Rios
nicollirioss@gmail.com
Formanda em Ciência da Computação na Universidade Salvador (UNIFACS), Gerente de Projetos da COMMIT– Empresa Júnior de Computação da UNIFACS. Já estagiou na área de Teste/Qualidade de Software na Softwell. Atualmente é estagiária de Desenvolvimento de Software na BAHIAGÁS- Companhia de Gás da Bahia.

Eliseu Torres
eliseutorres.gx@gmail.com
Estudante de graduação em Ciência da Computação na Universidade Salvador (UNIFACS). Estagia com desenvolvimento de software na Secretaria Municipal de Gestão (Semge) – Prefeitura Municipal de Salvador e é pesquisador voluntário de Iniciação Científica na área de Redes Definidas por Software e OpenFlow na Universidade Federal da Bahia (UFBA).

Daniel Almeida
danielalmeida.ti@gmail.com
Estudante de graduação em Ciência da Computação na Universidade Salvador (UNIFACS) e Diretor Administrativo e Financeiro da Commit – Empresa Júnior de Informática da UNIFACS. Participou do Grupo de Engenharia de Software e Aplicações (GESA), na UNIFACS, enquanto bolsista de Iniciação Científica da FAPESB. Atualmente se prepara para estudar na University of Toronto como bolsista de graduação no exterior pelo programa Ciência sem Fronteiras.

Rodrigo Oliveira Spínola
rodrigo.devmedia@gmail.com
Editor Chefe – Engenharia de Software Magazine

Porque esse artigo é útil:

Neste artigo são descritos alguns dos principais tipos de ciclo de vida de software. Através de um estudo de caso são apresentadas as etapas de elaboração de um processo, que compreende desde a fase de escolha do ciclo de vida, até a etapa de especificação do processo de desenvolvimento do software (a atividade a ser realizada, os responsáveis, critérios de entrada, ferramentas etc).

O tema é útil para Projetistas de Software em geral, que têm dificuldades em perceber qual o ciclo de vida mais adequado para cada situação. A escolha do ciclo de vida é muito importante, pois definirá como o software será desenvolvido, como os requisitos vão ser coletados, além de definir quando o cliente receberá a primeira versão operacional do sistema.

A engenharia de software, que segundo a definição proposta pela IEEE é a utilização de uma abordagem sistemática, disciplinada e quantificável no desenvolvimento, na operação e na manutenção de software, tornou-se fundamental em função de algumas características da utilização do software no século 21:

- A incorporação de aplicações em todos os âmbitos da sociedade, tanto no ambiente corporativo como pessoal;
- O aumento da complexidade e da diversidade de requisitos e
- A dependência de indivíduos, empresas e governos para gerenciar negócios, tomar decisões e controlar operações cotidianas.

Nesse contexto, o processo de software é uma atividade essencial, já que é a base para o controle e gerenciamento de projetos, permitindo o uso de técnicas e ferramentas para a produção de produtos de software com qualidade e dentro das restrições estabelecidas.

Segundo Sommerville, um processo de desenvolvimento de software é um conjunto de atividades que, ao serem executadas, geram um produto de software. Normalmente um processo inclui pelo menos quatro atividades fundamentais:

- Especificação de software;
- Projeto e implementação de software;
- Validação de software;
- Evolução de software.

Devido à necessidade de se adaptar às necessidades e características da organização, não existe um processo ideal que se aplique a todos os contextos. As atividades básicas do processo tendem a ser complexas e incluir diversas subatividades a depender do prazo, custo e qualidade esperados, das características do software a ser desenvolvido e das pessoas envolvidas no projeto, dentre outros aspectos. Ao discutir e descrever processos, além de apresentar as atividades envolvidas, muitas vezes são incluídas informações sobre os produtos esperados a partir de uma ou mais atividades do processo e os papéis envolvidos nelas, entre outras.

O objetivo deste artigo é apresentar um estudo de caso em que um cenário de desenvolvimento de software é apresentado, suas características analisadas e um ciclo de vida de software e um processo de desenvolvimento de software são propostos.

Modelo de ciclo de vida de software

As atividades e tarefas definidas no processo são organizadas em um modelo de ciclo de vida de software que define como cada uma das tarefas e atividades se relaciona com o processo e umas com as outras. Assim, um ciclo de vida de software é uma representação simplificada do processo de software que, normalmente, apresenta as atividades envolvidas no processo, mas não apresenta os detalhes de cada uma delas. Além disso, um modelo de ciclo de vida define os princípios e diretrizes que guiarão a realização das fases e a estrutura e a filosofia segundo as quais os processos serão executados. Tidos como abstrações ou frameworks, os ciclos de vida são utilizados para serem ampliados e adaptados a fim de elaborar processos mais específicos. Segundo Pressman, um modelo de processo genérico ou ciclo de vida de software estabelece pelo menos cinco atividades metodológicas principais:

- Comunicação;
- Planejamento;
- Modelagem;
- Construção;
- Entrega.

Além dessas atividades, um conjunto de atividades de apoio é utilizado ao longo do projeto: controle e acompanhamento do projeto, administração de riscos, garantia de qualidade, gerenciamento de configuração e outras. Neste artigo apresentaremos os seguintes modelos de ciclo de vida:

- Cascata;
- Incremental;
- Prototipação.

Modelo Cascata

No modelo em cascata cada fase é executada de forma sequencial. O objetivo é que cada fase finalizada sirva como base para a próxima, ou seja, nenhuma fase deve ser iniciada sem que a anterior seja completamente finalizada.

O modelo em cascata é indicado em projetos onde os requisitos são muito bem definidos e espera-se que não exista nenhuma alteração do projeto no decorrer das fases. A **Figura 1** ilustra as principais fases do modelo em cascata, cujos objetivos são:

- Definição de Requisitos: compreender o problema e levantar as necessidades que precisam ser atendidas pelo software;
- Projeto de Sistema e Software: converter os dados da definição de requisitos em documentos que serão utilizados pelos desenvolvedores;
- Implementação e Teste Unitário: desenvolver cada parte "módulo" do sistema seguindo a documentação do projeto. Ao término, as partes serão submetidas a um processo de teste para verificar se atendem as especificações do projeto;
- Integração e Teste de Sistema: integrar cada parte desenvolvida para compor o sistema completo. Após a integração, o sistema é submetido a testes para verificar se o sistema atende a todos os requisitos. Sendo aprovado pela bateria de testes, o sistema pode ser entregue ao cliente.
- Operação e Manutenção: realizar a correção de erros apresentados no sistema que não foram detectados anteriormente e modificar o software para atender a novos requisitos.

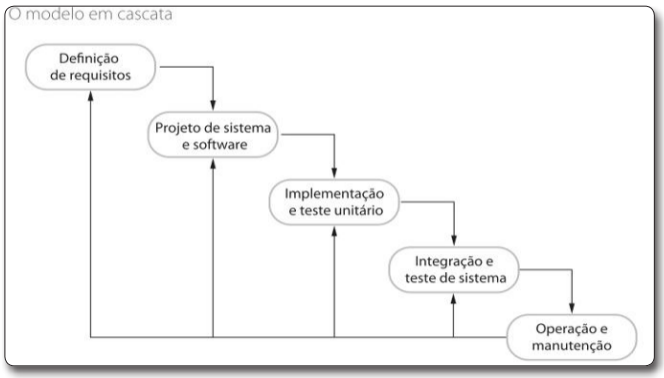

Figura 1. Modelo Cascata

Modelo Incremental

O modelo incremental adota desenvolvimento por estágios, onde os requisitos mais importantes são implementados primeiro, enquanto os outros, menos importantes, em versões posteriores. Gradativamente o cliente recebe partes funcionais do software aptas a entrarem em produção.

Como mostra a **Figura 2**, cada nova versão é incrementada à anterior, processo que se repete até a conclusão do projeto.

Essa fase em que o cliente recebe incrementos do sistema é chamada de prévia funcional.

A implementação deste modelo pode gerar algumas complicações. Problemas com documentação, por exemplo, pois são realizadas diversas mudanças durante o projeto, necessitando de um trabalho árduo de atualização da documentação. Outra dificuldade é a dependência entre os estágios (incrementos), já que o processo de junção de um incremento a outro é bastante complicado e minucioso.

A redução de riscos, maior visibilidade sobre o processo, descobertas de problemas logo no início, melhor previsão do tempo do projeto, são consideradas como vantagens do modelo incremental. Riscos de atrasos no projeto, alto número de incrementos, alto grau de dependência entre os incrementos, risco de faltar recursos financeiros (por parte do cliente) para a finalização do projeto são encarados como algumas das desvantagens do modelo.

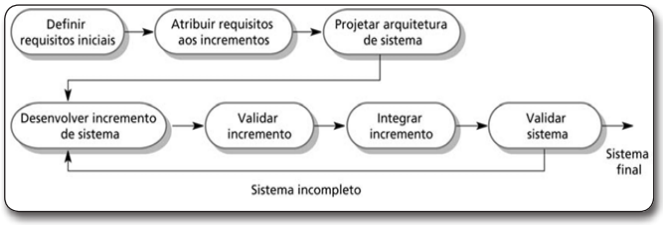

Figura 2. Modelo Incremental

Prototipação

Desenvolvedores costumam utilizar protótipos na construção de software com diferentes objetivos: na construção de modelos, simulações, implementações parciais e testes. Prototipação pode ser classificada como ciclo de vida ou pode ser adotada como parte de outros ciclos de vida.

Este ciclo de vida funciona da seguinte maneira: na primeira fase ocorre a definição inicial dos requisitos, onde a equipe responsável pelo projeto se reúne com o cliente para conhecer os requisitos iniciais do sistema de acordo com suas necessidades. Com base nesse levantamento preliminar dos requisitos é desenvolvido um protótipo do sistema e este é apresentado ao cliente para que ele tenha uma breve noção de como ficará o software, podendo sugerir mudanças ou não.

Com base nas alterações solicitadas pelo cliente, o protótipo é refinado. Esse protótipo passa por uma nova avaliação do cliente, que dará um feedback a respeito dos requisitos contidos no sistema. Com base nisso o protótipo é finalizado ou, caso sejam solicitadas mais mudanças pelo cliente, ele volta a sofrer ajustes até que o cliente se mostre satisfeito, como mostra na **Figura 3**.

Há diferentes tipos de protótipos:

• Protótipo descartável: Tendo a aprovação do cliente, o protótipo é abandonado e inicia-se a fase de construção do sistema. Posteriormente são feitos os testes e documentações necessárias no sistema e é entregue ao cliente o sistema pronto. Neste caso, o protótipo serve apenas para o cliente ter uma ideia geral de como será o sistema. Esse tipo de protótipo serve também como treinamento dos desenvolvedores, de forma que se conheça ainda mais as necessidades do cliente.

• Protótipo incremental: também conhecido como entrega por estágio. Normalmente, os requisitos mais importantes são desenvolvidos em primeiro plano e os demais requisitos só serão acrescentados em novas versões. Ao final de cada estágio, uma versão operacional é produzida e incrementada até a conclusão. Esse tipo de protótipo possui diversas vantagens como redução de riscos, problemas podem ser descobertos logo no início, maior visibilidade sobre o processo. Porém, há um grande esforço na atualização constante da documentação.

• Protótipo evolutivo: Após a aprovação do cliente, o sistema é desenvolvido a partir do protótipo aceito.

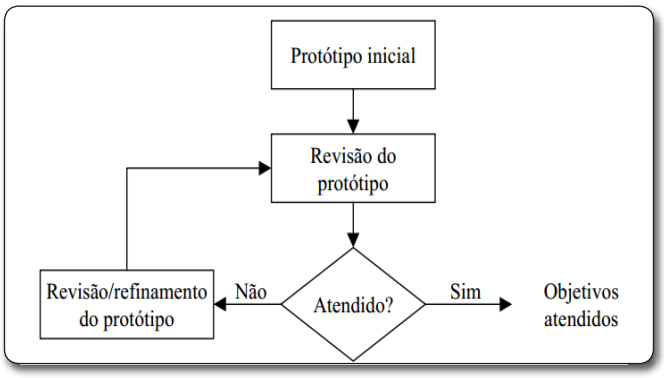

Figura 3. Processo de Prototipação

É aconselhável escolher esse tipo de ciclo de vida quando o cliente não sabe identificar seus requisitos de entrada, processamento e saída. O protótipo não só o ajudará a identificar suas necessidades, como pode ser utilizado também para determinar a viabilidade técnica, custo e prazos para o projeto. O protótipo reduz a possibilidade de insatisfação do cliente após o projeto criado, uma vez que o mesmo acompanhará todo o processo de desenvolvimento. Porém, como o protótipo é construído somente em âmbito visual (feito em HTML ou outra linguagem visual), não contendo nenhuma implementação de código, o cliente poderá ficar frustrado por achar que a maior parte já foi feita ou até mesmo quando comparar o desempenho do protótipo com a versão final do sistema.

Estudo de caso

O estudo de caso apresenta um cenário fictício de um projeto de desenvolvimento de software. Este cenário será analisado e, com base nesta análise, será proposto um modelo de ciclo de vida de software e definido um processo de desenvolvimento para o projeto.

O cenário que deve ser considerado pela equipe de desenvolvimento do projeto contempla os seguintes itens:

• Os requisitos do projeto não podem ser segmentados e, por questões contratuais, os requisitos devem ser estabelecidos

de maneira completa, correta e clara no inicio do projeto;
• A equipe de desenvolvimento possui pouca experiência em desenvolvimento de projetos de software
• O cliente possui um prazo definido para a entrega do produto, mas não há necessidade de entregas intermediárias do produto;
• A organização cliente possui dificuldade na definição e identificação dos requisitos do projeto.

A partir deste conjunto de restrições, será visto na próxima seção o ciclo de vida proposto para o projeto.

Ciclo de vida definido

A escolha de um modelo de ciclo de vida deve levar em consideração alguns pontos relevantes, a saber: natureza do projeto e da aplicação, experiência da equipe de desenvolvimento, metodologias e ferramentas a serem usadas e controles e produtos que precisam ser entregues.

No momento da escolha do ciclo de vida mais adequado para o cenário citado, levou-se em consideração a impossibilidade de segmentação dos requisitos e a necessidade de definir requisitos no início do projeto, assim como a pouca experiência da equipe de desenvolvimento e o fato de que o cliente não necessita de entregas intermediárias. O ciclo de vida escolhido foi o modelo cascata. Contudo, uma adaptação da prototipagem foi incluída na fase de definição de requisitos, permitindo que o sistema venha a sofrer alterações ao longo do projeto sem problema algum e auxiliando assim o cliente a ter uma visão geral do projeto final, a fim de facilitar a descoberta de novos requisitos, já que o mesmo possui dificuldade em identificá-los.

O modelo cascata é o mais antigo e consequentemente, o mais utilizado. Ele oferece uma maneira de tornar o processo mais visível, facilita o planejamento e o uso de revisões ao fim de cada fase.

Já o uso de prototipação é ideal para identificar requisitos. Na primeira fase é desenvolvido um protótipo para o cliente experimentar, podendo sugerir mudanças ou não. Com base nessa avaliação, o protótipo é refinado e passa por outras avaliações por parte do cliente, até que o mesmo se mostre satisfeito. A partir dessa aprovação, o protótipo é descartado e inicia-se a construção do projeto com base no feedback do cliente.

Nesse contexto, explicaremos cada fase do modelo de ciclo de vida definido, como mostra a **Figura 4**.

Definição de Requisitos e Prototipação:

Primeiramente ocorre a especificação dos requisitos. Nesta atividade a equipe responsável pelo projeto se reúne com o cliente para conhecer os requisitos iniciais do sistema de acordo com as necessidades do mesmo através de entrevistas, workshops de requisitos e *brainstorms*. Com base nesse levantamento preliminar dos requisitos é desenvolvido um protótipo descartável do sistema e este é apresentado para validação por parte dos *stakeholders*. Assim, o cliente poderá entender melhor suas próprias necessidades e auxiliar os analistas de sistemas na definição dos requisitos de maneira apropriada.

Com base nas alterações solicitadas pelo cliente, o protótipo é refinado. Esse protótipo passa por uma nova avaliação do cliente, que dará um *feedback* a respeito dos requisitos contidos no sistema. Com base nisso o protótipo é finalizado ou, caso seja solicitada mais mudanças pelo cliente, ele volta a sofrer modificação e assim por diante até o cliente se mostrar satisfeito.

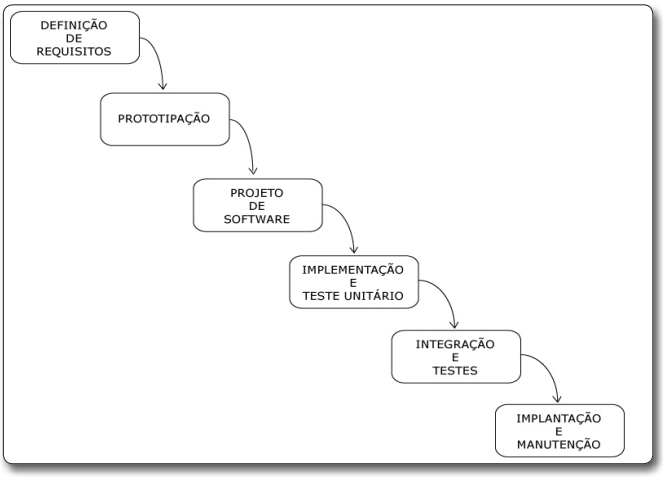

Figura 4. Ciclo de vida definido

Projeto de Software:

Após a aprovação do cliente, o protótipo é abandonado e os projetistas de software podem projetar o software e elaborar os diagramas de classe, sequência e entidade relacionamento, além de um documento de arquitetura de software.

Implementação e Teste Unitário:

A partir de toda a documentação gerada na primeira fase, assim como os casos de uso, os desenvolvedores podem codificar e documentar o software e executar testes unitários sobre as funcionalidades desenvolvidas. Posteriormente, os códigos são inspecionados por outro desenvolvedor e os defeitos identificados podem ser corrigidos ainda na etapa de implementação.

Integração e Testes:

Os módulos desenvolvidos são integrados e testados em nível de sistema. Os analistas de teste são responsáveis por elaborar o planejamento de teste e os casos de teste, enquanto os testadores executam os testes e elaboram relatórios de teste detalhando o comportamento do software e registrando os defeitos encontrados na ferramenta de *bugtracking*. Com base nos casos de uso, casos de teste e relatórios de teste, os desenvolvedores corrigem os incidentes (defeitos identificados). Após as correções necessárias, espera-se que o software esteja pronto para a sua implantação e utilização pelo cliente.

Implantação e Manutenção:

A equipe de desenvolvimento realiza a preparação do ambiente de produção conforme a especificação no documento de arquitetura de software e realiza a implantação, entregando ao cliente o software em funcionamento, bem como todos os produtos (artefatos) necessários.

Após a implantação, novos defeitos podem ser identificados ou solicitações de mudança podem ser feitas por parte do cliente, o que exige uma nova execução de todo o ciclo a fim de realizar as mudanças necessárias no software.

Processo de desenvolvimento definido

A partir do ciclo de vida elaborado, foi definido um processo de desenvolvimento. Este processo não objetiva estar completo, mas demonstrar como devemos proceder para elaborá-lo aderente ao ciclo de vida definido. Observe nas **Tabelas** de **1** a **6** que definimos um conjunto de atividades a serem realizadas pela equipe de desenvolvimento para cada fase do ciclo de vida.

Na definição de cada atividade descrevemos o nome da atividade, uma breve descrição de seu objetivo, os critérios de entrada, responsáveis e participantes envolvidos em sua execução, pré-atividade, artefatos requeridos e gerados, e ferramentas que podem ser utilizadas para apoiar a realização da atividade.

Fase 1 - Definição de Requisitos	
Atividade:	**Identificar requisitos**
Descrição:	Conversa entre gerentes de projeto e clientes para analisar suas necessidades
Critérios de Entrada:	Cliente com pelo menos uma ideia geral do projeto
Responsáveis:	Gerentes de projeto e Analistas de Requisitos
Participantes:	Cliente, analista de requisitos e gerentes de projeto
Pré-atividade:	Agendamento de uma reunião para discutir a respeito do produto contratado pelo cliente
Artefatos Requeridos:	Pré-conhecimento dos requisitos ou pelo menos uma ideia geral do projeto pelo cliente
Artefatos Gerados:	Possíveis necessidades do cliente descobertas pelo gerente de projetos através da reunião
Ferramentas:	Material para anotação
Atividade:	Especificação de casos de uso iniciais
Descrição:	Elaboração da especificação dos requisitos identificados utilizando casos de uso
Critérios de Entrada:	Itens escolhidos na reunião com o cliente
Responsáveis:	Analista de requisitos
Participantes:	Analista de requisitos
Pré-atividade:	Identificar Requisitos
Artefatos Requeridos:	-
Artefatos Gerados:	Especificação de casos de uso
Ferramentas:	Material de anotação e ferramentas de modelagem de classes/banco

Tabela 1. Fase 1 – Definição de requisitos

Fase 2 - Prototipação	
Atividade:	**Elaborar Protótipo**
Descrição:	Desenvolvimento do primeiro protótipo para análise do cliente
Critérios de Entrada:	Gerentes de projeto e desenvolvedores cientes dos requisitos
Responsáveis:	Desenvolvedores de software
Participantes:	Analistas de requisitos e desenvolvedores
Pré-atividade:	Especificação de casos de uso iniciais
Artefatos Requeridos:	Especificação de casos de uso definida na fase 1
Artefatos Gerados:	Protótipo do sistema desenvolvido com base nas necessidades do cliente
Ferramentas:	Ferramenta de desenvolvimento e design
Atividade:	Avaliação do protótipo
Descrição:	Protótipo será avaliado pelos gerentes de projeto com base nas necessidades do cliente
Critérios de Entrada:	Conhecimento prévio das necessidades do cliente para avaliar se o protótipo realmente está aderente a elas
Responsáveis:	Gerente de projeto
Participantes:	Gerente de projeto e desenvolvedores
Pré-atividade:	Elaborar protótipo
Artefatos Requeridos:	Protótipo desenvolvido na atividade anterior
Artefatos Gerados:	Protótipo avaliado de acordo com as necessidades do cliente e supervisionado pelo gerente de projeto
Ferramentas:	Material de anotação
Atividade:	Validação do protótipo pelo cliente
Descrição:	Análise, incremento e exclusão de requisitos do projeto pelo cliente
Critérios de Entrada:	Conhecimento prévio do cliente das suas necessidades citadas na fase 1 para avaliar se o protótipo realmente supre as mesmas
Responsáveis:	Gerentes de projeto e analista de requisitos
Participantes:	Gerentes de projeto, analista de requisitos e cliente
Pré-atividade:	Avaliação do protótipo
Artefatos Requeridos:	Protótipo pré-avaliado pelos gerentes de projeto
Artefatos Gerados:	Protótipo validado pelo cliente, pedido de alterações e/ou descoberta de novos requisitos.
Ferramentas:	Material de anotação
Atividade:	Refinamento do protótipo
Descrição:	Realizar alterações no protótipo com base na avaliação do cliente
Critérios de Entrada:	Novas necessidades do cliente
Responsáveis:	Desenvolvedores
Participantes:	Desenvolvedores e Analistas de requisitos
Pré-atividade:	Validação do protótipo pelo cliente
Artefatos Requeridos:	Pedido de alterações e protótipo já desenvolvido
Artefatos Gerados:	Protótipo ajustado
Ferramentas:	Ferramentas de desenvolvimento e design

Tabela 2. Fase 2 - Prototipação

Fase 3 - Projeto de Software	
Atividade:	**Projeto Técnico do Software**
Descrição:	Definição de arquitetura e desenvolvimento do projeto técnico do software.
Critérios de Entrada:	Ter requisitos definidos
Responsáveis:	Projetista de Software
Participantes:	Analista de Requisitos e Projetista de Software
Pré-atividade:	Validação do protótipo pelo cliente
Artefatos Requeridos:	Casos de Uso e Protótipos
Artefatos Gerados:	Diagramas de Classe, Diagrama de Sequência, Diagrama Entidade Relacionamento (DER) e Documento de Arquitetura de Software
Ferramentas:	Software para modelagem em UML, IDE

Tabela 3. Fase 3 – Projeto de Software

Fase 4 - Implementação e Teste Unitário	
Atividade:	**Codificação**
Descrição:	Codificação com base nos casos de Uso, protótipo e documentação arquitetural.
Critérios de Entrada:	Ter elaborado o projeto do software.
Responsáveis:	Desenvolvedores
Participantes:	Desenvolvedores
Pré-atividade:	Projeto técnico de software
Artefatos Requeridos:	Casos de Uso, Diagrama de Classes, Diagrama de Sequência, Diagrama Entidade Relacionamento e Documento de Arquitetura de Software
Artefatos Gerados:	Código fonte
Ferramentas:	IDE, Editor de textos, Ferramenta de acesso/interface com Banco de Dados
Atividade:	Teste unitário e inspeção de código
Descrição:	Execução de testes unitários e inspeção do código testado por outro desenvolvedor.
Critérios de Entrada:	Código liberado para teste
Responsáveis:	Desenvolvedores
Participantes:	Desenvolvedores
Pré-atividade:	Codificação
Artefatos Requeridos:	Código fonte e Casos de Uso
Artefatos Gerados:	Relatório de Inspeção de Código
Ferramentas:	Ferramenta para automação de testes, IDE
Atividade:	Correção de Defeitos
Descrição:	Correção dos defeitos identificados nos testes unitários ou na inspeção de código.
Critérios de Entrada:	Ter realizado a avaliação de qualidade do código
Responsáveis:	Desenvolvedores
Participantes:	Desenvolvedores
Pré-atividade:	Teste unitário e inspeção de código
Artefatos Requeridos:	Código fonte e Casos de Uso
Artefatos Gerados:	Código fonte
Ferramentas:	IDE, Editor de textos

Tabela 4. Fase 4 – Implementação e Teste Unitário

Fase 5 - Integração e Testes	
Atividade:	**Planejamento de Testes**
Descrição:	Elaboração e documentação de estratégias, objetivos e procedimentos de teste.
Critérios de Entrada:	-
Responsáveis:	Analista de Testes
Participantes:	Analista de Testes
Pré-atividade:	Correção de defeitos (fase 4)
Artefatos Requeridos:	Casos de Uso
Artefatos Gerados:	Planejamento de Testes.
Ferramentas:	Editor de textos
Atividade:	Elaboração de Roteiros de Teste
Descrição:	Elaboração de casos de teste para cada caso de uso, conforme o Planejamento de Testes.
Critérios de Entrada:	Ter o planejamento dos testes elaborado
Responsáveis:	Analista de Testes
Participantes:	Analista de Testes
Pré-atividade:	Planejamento de testes
Artefatos Requeridos:	Casos de Uso e Planejamento de Testes
Artefatos Gerados:	Casos de Teste
Ferramentas:	Editor de textos
Atividade:	Execução de Testes
Descrição:	Execução dos casos de teste e documentação dos resultados.
Critérios de Entrada:	-
Responsáveis:	Testador
Participantes:	Testador, Analista de Testes
Pré-atividade:	Elaboração de roteiros de teste
Artefatos Requeridos:	Casos de Teste, Planejamento de Testes e Casos de Uso, Software
Artefatos Gerados:	Relatórios de Execução de Testes
Ferramentas:	IDE, Editor de textos, Ferramenta para Bugtracking
Atividade:	Correção de Defeitos
Descrição:	Correção dos defeitos identificados nos Testes
Critérios de Entrada:	Bateria de teste executada
Responsáveis:	Desenvolvedores
Participantes:	Desenvolvedores
Pré-atividade:	Execução de testes
Artefatos Requeridos:	Código fonte, Casos de Uso, Casos de Teste e Relatórios de Teste
Artefatos Gerados:	Código fonte
Ferramentas:	IDE, Editor de textos, Ferramenta para Bugtracking

Tabela 5. Fase 5 – Integração e Testes