

GILVAN SOUSA DA SILVA RA: 6820508946

LEONARDO CÉSAR VALENTINO COSTA RA: 6247228721

MARCUS ALVES FERREIRA RA: 6277261905

RODRIGO DOURADO NUNES RA: 6818457769

MODELO DE DESENVOLVIMENTO DE SOFTWARE PARA UMA REDE DE DROGARIAS

BRASÍLIA

2016

GILVAN SOUSA DA SILVA

LEONARDO CÉSAR VALENTINO COSTA

MARCUS ALVES FERREIRA

RODRIGO DOURADO NUNES

MODELO DE DESENVOLVIMENTO DE UM SOFTWARE PARA UMA REDE DE DROGARIAS

Projeto apresentado ao Curso em Bacharelado de Sistemas de Informação da Instituição Faculdade De Negócios e Tecnologias da Informação – FACNET Anhanguera  
Orientador: Mariana Monteiro Nunes

BRASÍLIA

2016

**SUMÁRIO**

[**1. INTRODUÇÃO 4**](#_Toc446565869)

[1.1 PROBLEMA 5](#_Toc446565870)

[**2.** **OBJETIVOS** **6**](#_Toc446565871)

[2.1 OBJETIVO GERAL OU PRIMÁRIO 6](#_Toc446565872)

[2.2 OBJETIVOS ESPECÍFICOS OU SECUNDÁRIOS 6](#_Toc446565873)

[**3.** **JUSTIFICATIVA** **7**](#_Toc446565874)

1. INTRODUÇÃO

O projeto trata o modelo de desenvolvimento de um software, exemplificado em um sistema para uma rede de drogarias, onde são abordadas as etapas de desenvolvimento desse sistema seguindo desde concepção até a implantação.

Serão contempladas aqui também as regras de negócio encontradas em uma drogaria real, elucidando assim, as necessidades reais, as quais são muito evidentes, ainda mais se levarmos em conta o momento atual, em que mudanças são cada vez mais rápidas e influenciadoras, o que já traz a necessidade de utilizar ferramentas de software que sejam capazes de prestar auxílio.

Será levada em conta uma série de padrões ditados pela engenharia de software, os quais farão parte do tema central ser pesquisado. De acordo com a evolução do projeto, cada padrão que for utilizado será explicado por meio de pesquisa, para isso, buscaremos explicar o porquê utilizar determinado padrão, mostrando assim, as vantagens de se fazer uso desses padrões.

Este projeto pretende aplicar as melhores práticas da engenharia de software para resolver o problema da rede de drogarias, de forma a mostrar a importância da engenharia de software para que se possa ter um sistema de qualidade. Além de demonstrar como a sociedade envolvida no desenvolvimento de sistemas pode se beneficiar destas técnicas, como empresas, usuários de sistemas e estudantes. E tentar esclarecer para todos os estudantes, que venham a ler este projeto, a importância da engenharia de software.

# PROBLEMA

Atualmente os sistemas apresentam muitos problemas, derivados da má concepção e de processos ruins. Como criar sistemas que satisfação os clientes e atendam as especificidades de seus negócios?

1. OBJETIVOS

# OBJETIVO GERAL OU PRIMÁRIO

Desenvolver um sistema para uma rede de drogarias com o uso da engenharia de software.

# OBJETIVOS ESPECÍFICOS OU SECUNDÁRIOS

1. Estudar o negócio do cliente, procurar entender sua forma de trabalho e criar uma linguagem comum entre a equipe técnica e o cliente;
2. Pesquisar sobre sistemas semelhantes, para analisar pontos fortes e fracos, averiguar o que pode ser aproveitado do sistema atual, assim estabelecendo o que de melhor pode ser aplicado para o cliente;
3. Realizar os processos de engenharia de software, que serão estabelecidos para melhor atender ao cliente;
4. Implementar e testar o sistema.
5. JUSTIFICATIVA

As mudanças no atual cenário corporativo estão sempre acontecendo, surge então, a necessidade de as empresas possuírem sistemas de informações que sejam capazes de se adequar a essas mudanças. Portanto, sistemas obsoletos que não sejam flexíveis e escaláveis devem ser na medida do possível substituídos. Queremos assim, deixar mais transparente os benefícios trazidos pela modernização dos sistemas utilizados pelas empresas.

Outro alvo importante deste projeto é a importância da utilização dos padrões engenharia de software, por se tratar de padrões aceitos internacionalmente, que já foram testados inúmeras vezes. Construir um software tem suas particularidades, por ser algo mais lógico do que concreto, a necessidade de ter um controle do projeto é ainda mais fundamental, por meio de prazos e custos bem definidos, documentação clara e abrangente, entre muitos outros fatores que são aqui abordados.

É claro que, uma empresa possuir sistemas de informações atuais e desenvolvidos seguindo todos os padrões não garantirá seu bom desempenho, mas, sem dúvida, facilitará enfrentar este mercado cada vez mais difícil, no qual turbulências podem surgir a qualquer momento, ainda mais levando em conta a atual situação da economia do país.

1. FUNDAMENTAÇÃO TEÓRICA
2. Engenharia de sofware

Nesta abordaremos os principais conceitos da Engenharia de Software respaldado por autores reconhecidos no cenário internacional.

Cada vez mais o mundo ganha um dinamismo que influencia muito a vida das pessoas, a forma obter informações alcançou um nível que no passado era inimaginável, e os sistemas de informações têm um papel fundamental nesse acontecimento. Dessa forma buscar formas de otimizar o seu processo de desenvolvimento torna-se cada vez mais necessárias, e é esse o principal o objetivo da Engenharia de Software.

Conforme Pressman “A Engenharia de Software é uma tecnologia dividida em camadas (ferramentas, métodos, processo e foco na qualidade) ” (PRESSMAN, 2011, p. 15).

“Engenharia de Software é a criação e a utilização de sólidos princípios de engenharia a fim de obter software de maneira econômica, que seja confiável e que trabalhe em máquinas reais. ” (FRIEDRICH LUDWIG BAUER)

“Engenharia de Software é uma disciplina cujo foco está em todos os aspectos da produção de software, desde os estágios iniciais da especificação do sistema até sua manutenção...” (SOMMERVILLE, 2013, p. 5)

Há vários exemplos de sistemas, em que foram gastos recursos consideráveis e que não atenderam às necessidades de seus usuários, e ainda por cima foram entregues fora do prazo estabelecido no cronograma. Esses acontecimentos são muitos graves, haja vista que as organizações estão cada vez mais dependentes dos sistemas de informações. Pressman descreve esta importância: “O software distribui o produto mais importante de nossa era – *a informação*. Ele transforma dados pessoais de modo que possam ser úteis num determinado contexto” (PRESSMAN, 2011, p. 31). Isso deixa muito claro o tamanho da importância da Engenharia de Software.

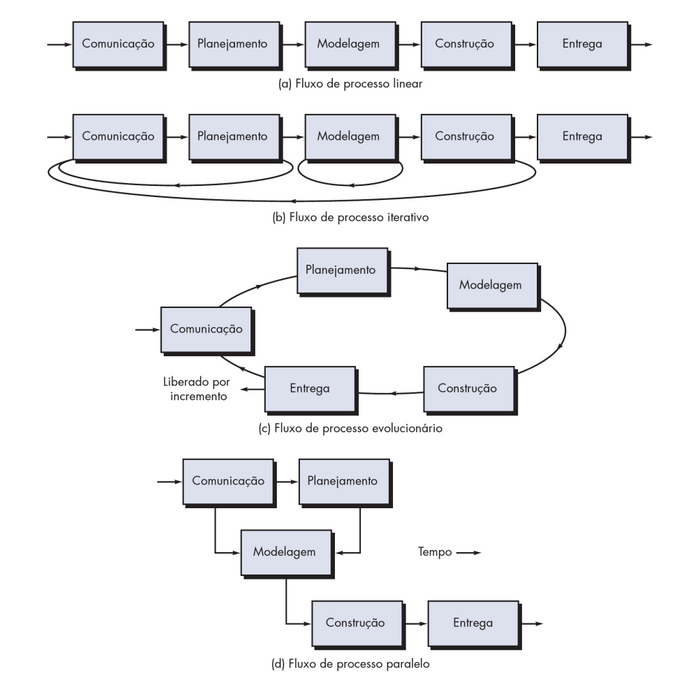
Um dos pontos mais importantes da Engenharia de Software é a divisão do desenvolvimento de software em fases definidas. Vejamos estas fases de acordo com alguns autores.

Para Pressman, a Engenharia de Software é dividida em cinco etapas:

1. **Comunicação –** [...] A intenção é compreender os objetivos das partes interessadas para com o projeto e fazer o levantamento das necessidades que ajudarão a definir as funções e características do software.
2. **Planejamento –** [...] define o trabalho de engenharia de software, descrevendo as tarefas técnicas a ser conduzidas, os riscos prováveis, os recursos que serão necessários, os produtos a ser produzidos e um cronograma de trabalho.
3. **Modelagem –** [...] Cria-se um “esboço” da coisa, de modo que se possa ter uma ideia do todo – qual será seu aspecto em termos de arquitetura, como as partes constituintes se encaixarão e várias outras características. [...]
4. **Construção –** Essa atividade combina geração de código (manual ou automatizada) e testes necessários para revelar erros de codificação.
5. **Emprego –** O software é entregue ao cliente, que avalia o produto entregue e fornece feedback, baseado na avaliação.(PRESSMAN, 2011, p. 41)

1.1.2.1 Processos Genéricos

Pressman defende que essas etapas, que ele chama de processos, podem vários fluxos, vejamos na figura 1.1 esses exemplos.

Figura 1- Fluxos de processo

Fonte: PRESSMAN (2011, p. 54)

1.1.2.2 Modelo de Processo Prescritivo

Foram propostos alguns modelos de processos para situações mais específicas, diante de uma enorme necessidade.

“A história tem demonstrado que esses modelos tradicionais proporcionaram uma considerável contribuição quanto à estrutura utilizável no trabalho de equipes de software. Entretanto, o trabalho de engenharia de software e o seu produto permanecem à beira do caos”. (PRESSMAN, 2011, p. 59)

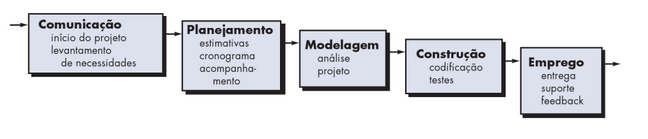
1.1.2.2.1 O Modelo Cascata

É mais indicado em situações nas quais os requisitos estão em um nível de definição elevado, e que por isso tendem a mudar pouco. Isso é mais comum em mudanças bem definidas e um sistema existente.

O Modelo Cascata “...sugere uma abordagem sequencial e sistemática para o desenvolvimento de software”. (PRESSMAN, 2011, p. 59)

Vejamos na figura 2 como evolui o Modelo Cascata, segundo Pressman.

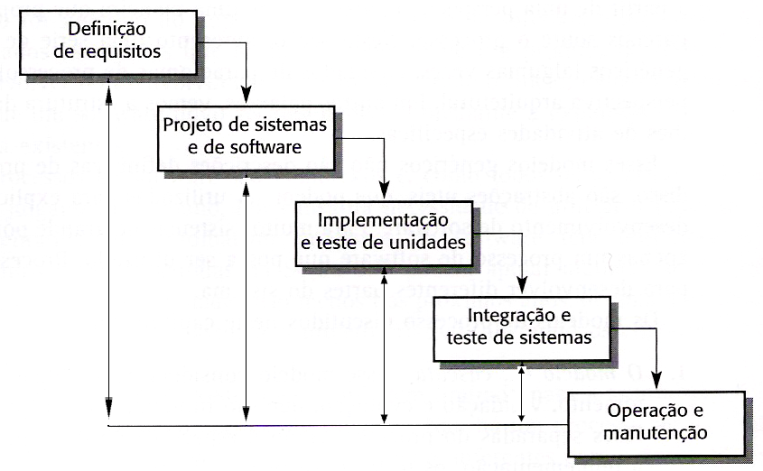
Figura 2- Modelo Cascata - Pressman



Fonte: PRESSMAN (2011, p. 60)

Vejamos na figura 3 como evolui o Modelo Cascata segundo Sommervile.

Figura 3- Modelo Cascata - Sommerville



Fonte: SOMMERVILLE (2013, p. 20)

O modelo Cascata tem suas qualidades, como organizar e dividir responsabilidades no desenvolvimento de software. Entretanto, tem sofrido inúmeras críticas por não atender a complexidade que é desenvolver um software, ainda mais nos dias atuais.

“Para maioria dos sistemas, esse processo não oferece custo-benefício significativo”*.* (SOMMERVILLE, 2013, p. 21)

Pressman cita três principais defeitos:

* Projetos reais raramente seguem o fluxo sequencial[...]
* Frequentemente, é difícil para o cliente estabelecer explicitamente todas as necessidades[...]
* O cliente dever ter paciência. Uma versão operacional do(s) programa(s) não estará disponível antes de estarmos próximo do final do projeto[...] (PRESSMAN, 2011, p. 61)

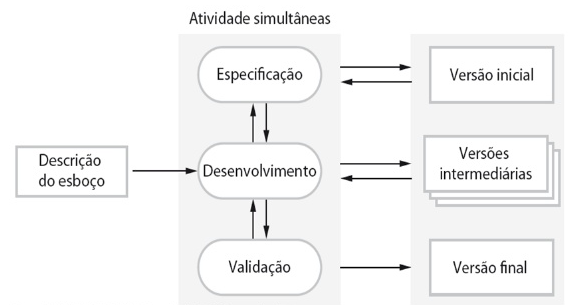
“Na média, 45% das características nos requisitos em cascata nunca são usadas e o cronograma e orçamentos iniciais, em cascata, variam até 400% do que ocorre na realidade”. (LARMAN, 2007, p. 61)

1.1.2.2.2 O modelo Incremental

Este modelo foi proposto inicialmente por Mills em 1980 e veio para superar a dificuldade que o modelo cascata tem em lidar com mudanças frequentes durante o processo de desenvolvimento de software, principalmente por conta da sua maneira de lidar com os requisitos, mais flexível considerando mudanças de requisitos algo esperado e normal. O Modelo Incremental pode ser definido como: “O desenvolvimento incremental é baseado na ideia de desenvolver uma implementação inicial, expô-la aos comentários dos usuários e continuar por meio de criação de várias versões até que um sistema adequado seja desenvolvido” (SOMMERVILLE, 2013, p. 21).

A figura 4 demonstra como o modelo incremental se comporta segundo Sommerville.

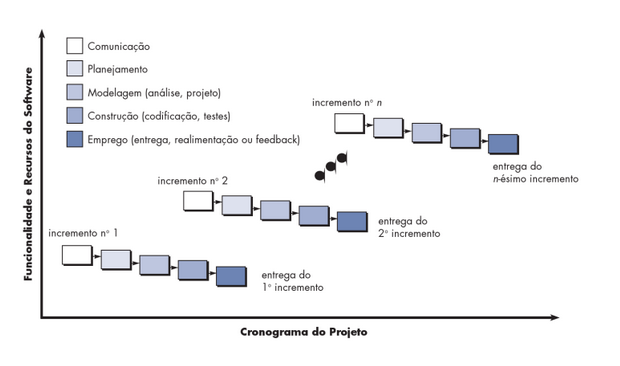
Figura 4- Modelo Incremental - Sommerville



Fonte: SOMMERVILLE (2013, p. 22)

A figura 5 demonstra como o modelo incremental se comporta segundo Pressman.

Figura 5- Modelo Incremental - Pressman



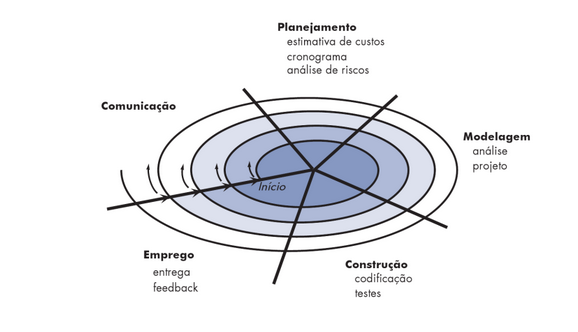
Fonte: PRESSMAN (2011, p. 61)

O modelo incremental se mostra eficaz em muitas situações, porém há algumas específicas em que não possível obter resultados satisfatórios com sua utilização, como em sistemas de bancários de grande porte, nos quais várias equipes de trabalho têm de estar envolvidas, exigindo dessa forma, uma arquitetura bem definida previamente.

1.1.2.2.**3 Modelo Espiral**

Segundo PRESSMAN 2011 p. 65 “o Modelo Espiral é um modelo de software evolucionário que acopla a natureza iterativa da prototipação com os aspectos sistemáticos e controlado do Modelo Cascata”. Neste modelo o cliente vai recebendo e testando o sistema em várias versões entregues, as primeiras podem ser um protótipo até chegar na versão final.

Figura 6- Modelo Espiral



Fonte: PRESSMAN (2011, p. 65)

O Modelo Espiral é eficaz na redução de riscos a que o processo de desenvolvimento de software está sujeito, por utilizar a prototipação em qualquer estágio.

Pressman explica o motivo do nome do modelo ser espiral:

Assim que esse processo evolucionário começa, a equipe de software realiza atividades indicadas por um circuito em torno da espiral no sentido horário começando pelo centro. Os riscos são considerados à medida que cada evolução é realizada[...](PRESSMAN, 2011, p. 41)

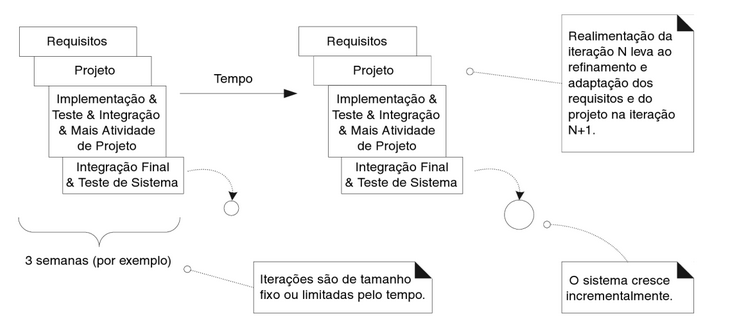
Como nenhum modelo é perfeito, o Modelo Espiral também tem suas limitações, como o fato de exigir avaliação de riscos, pois se algum importante não detectado e controlado poderá ocorrer sérios problemas.

1.1.2.2 Processo Unificado

O Processo Unificado é um processo para desenvolvimento de software iterativo que busca tirar proveito dos melhores recursos dos modelos mais tradicionais citados anteriormente. “Processo Unificado surgiu comoum processo iterativo popular para o de o desenvolvimento de software visando à construção de sistemas orientados a objetos”. (LARMAN, 2007, p. 46).

O Processo Unificado segue uma abordagem de desenvolvimento iterativa, que se baseia em incrementos e melhorias de acordo com a evolução do desenvolvimento, as mudanças são vistas como inevitáveis e essenciais para sucesso do projeto, ou seja, a adaptação faz parte do modelo. Vejamos na figura 7 como funciona o fluxo do desenvolvimento iterativo.

Figura 7- Desenvolvimento Iterativo e Evolutivo



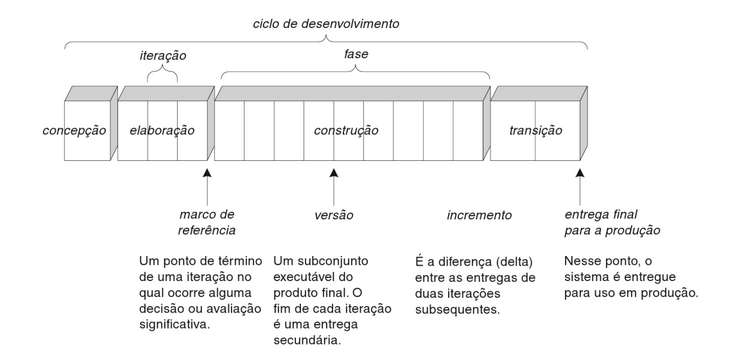
Fonte: LARMAN (2007, p. 48)

LARMAN descreve as quatro fases do Processo Unificado:

1. **Concepção –** visão aproximada, casos de negócio, escopo e estimativas vagas.
2. **Elaboração –** visão refinada, implementação iterativa da arquitetura central, resolução dos altos riscos, identificação da maioria dos requisitos e do escopo e estimativas mais realistas.
3. **Construção –** implementação iterativa dos elementos restantes de menor risco e mais fáceis e preparação para a implantação.
4. **Transição –** testes beta e implantação. (LARMAN, 2010, p. 61)

Vale ressaltar que o processo pode ser composto de várias iterações de acordo a evolução, vejamos na figura 8 mais detalhes.

Figura 8- Cronograma do Processo Unificado

Esse modelo é seguido por uma metodologia de desenvolvimento de software muito difundida pertencente à IBM, o RUP (Rational Unified Process), que será mais detalhado posteriormente.

Fonte: LARMAN (2007, p. 62)

1. CRONOGRAMA DE DESENVOLVIMENTO

**Quadro 1** – Calendário de execução das atividades do Projeto e do Trabalho de Conclusão de Curso

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ATIVIDADE** | **2016** | | | | | **2016** | | | | |  |
| **JAN** | **FEV** | **MAR** | **ABRIL** | **MAI** |  | **AGOS** | **SETEM** | **OUTUB** | **NOVEM** | **DEZEM** |
| Escolha do tema | X |  |  |  |  |  |  |  |  |  |  |  |
| Revisão bibliográfica | X | X | X | X | X |  | X | X | X | X | X |  |
| Elaboração do projeto | X | X | X | X |  |  |  |  |  |  |  |  |
| Elaboração de questionários, tópico de entrevistas etc. \* |  | X | X | X |  |  |  |  |  |  |  |  |
| Entrega do projeto |  |  |  |  | X |  |  |  |  |  |  |  |
| Elaboração da monografia (TCC) |  |  |  |  |  |  | X | X | X | X | X |  |
| Realização dos capítulos |  |  |  |  |  |  | X | X | X |  |  |  |
| Coleta e análise de dados / amostragens \* |  |  |  |  |  |  | X | X | X | X |  |  |
| Realização da conclusão e introdução |  |  |  |  |  |  |  |  | X | X |  |  |
| Correção de textos | X | X | X | X |  |  | X | X | X | X | X |  |
| Elaboração de elementos pré e pós-textuais |  |  |  |  |  |  |  |  | X | X |  |  |
| Entrega da monografia |  |  |  |  |  |  |  |  |  |  | X |  |
| Defesa da monografia |  |  |  |  |  |  |  |  |  |  | X |  |

1. REFERÊNCIas

SOMMERVILLE, Ian. **Engenharia de Software**. 9ª ed. São Paulo. Pearson. 2013

PRESSMAN, Roger S. **Engenharia de Software.** 7ª ed. São Paulo. Bookman. 2011

LARMAN, Craig. **Utilizando UML e Padrões.** 3ª ed. São Paulo. Bookman. 2007