Capítulo 1

Revisão Literária

Neste capítulo são apresentados os conceitos utilizados neste trabalho baseado na literatura. Na primeira seção são abordadas definições processuais de desenvolvimento de um software. Na segunda são revisados termos de direcionamento técnico para este processo. Na terceira, conceitos de controle e automação importantes, enquanto, na quarta, trabalhos relevantes que abordam assuntos comuns ao projeto aqui relatado.

1.1 Processo de Software

Um processo de software é composto por um conjunto de atividades relacionadas que levam à produção de um produto de software [2]. A utilização deste conjunto de atividades auxilia no aumento da qualidade do produto desenvolvido, bem como sua manutenção. De forma geral, um processo de software é composto de quatro grandes etapas:

- **Especificação de Software:** Devem ser definidas as funcionalidades e limitações de funcionamento do software a ser desenvolvido;
- **Projeto e Implementação do Software:** O software deve ser produzido buscando o atendimento das especificações;
- **Validação de software:** Deve ser validado para garantir o atendimento às demandas solicitadas;
- **Evolução do Software:** Deve ser evoluído e alterado conforme novas necessidades

1.1.1 Modelo em Cascata

Existem diversos modelos de processo de software, dentre eles o modelo cascata, também conhecido como ciclo de vida de software. Este modelo consiste no encadeamento de suas etapas, tendo o início de uma associado necessariamente ao final de outra. São elas:

- **Definição de requisitos:** São definidas restrições e metas do sistema por meio de entrevista com o usuário. Posteriormente são detalhadas a fim de se tornar especificação do sistema;

- **Projeto de Sistema e Software:** Etapa de definição de arquitetura geral do sistema. Envolve identificação e descrição das abstrações, módulos ou componentes, bem como o relacionamento entre eles;
- **Implementação:** O projeto de software é desenvolvido com seus conjuntos ou unidades de programas;
- **Integração e Testes de sistema:** Os módulos individuais do sistema elaborado são integrados e testados em conjunto, buscando assegurar o funcionamento conforme especificado;
- **Operação e manutenção:** O sistema é colocado em uso. São efetuadas possíveis correções de erros não descobertos em etapas anteriores. Expansões e melhorias de implementação das unidades também podem ser realizadas nesta etapa.

Este modelo dá grande importância à documentação gerada ao final de cada uma das etapas, geralmente amarrando o início da etapa seguinte à aprovação destes documentos. Ele também é consistente com outros modelos de processos de engenharia, tornando o processo mais visível para o nível gerencial.

1.1.2 Desenvolvimento Ágil

O desenvolvimento ágil se baseia na entrega de pequenos incrementos do produto de software, disponibilizando versões para o cliente, a cada período de tempo especificado (em geral, duas ou três semanas). Esse envolvimento do cliente durante o processo de construção possibilita o acompanhamento da evolução dos requisitos do sistema. Tem como outra forte característica a utilização de conversas informações, minimização da documentação, gerando mais responsividade do processo a possíveis mudanças nos requisitos do produto de software.

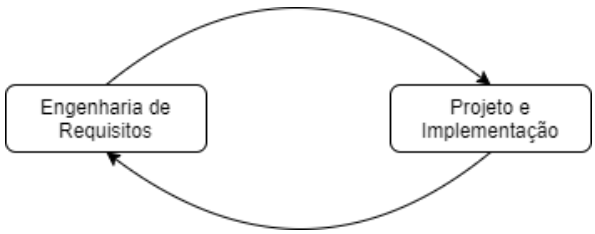

Figura 1.1: Métodos Ágeis

1.2 Arquitetura de Software

Arquitetura de Software é um conjunto de decisões significativas sobre a organização de um sistema de software, a seleção dos elementos estruturais e suas interfaces, a composição destes em subsistemas progressivamente maiores, bem como o estilo de arquitetura que orienta a organização e interação dos módulos envolvidos. A definição da arquitetura de um software passa por etapas e conhecimentos prévios necessários para uma estruturação correta do mesmo [1].

1.2.1 Modelagem orientada a Dados

Modelos construídos sob o paradigma de orientação a dados trazem a sequência de ações envolvidas no processamento daqueles dados, desde sua entrada, até sua saída. Pode ser realizada através de um modelo de sequências[1] destacando a movimentação dos dados daquele módulo ali representado. A figura 1.2 traz um exemplo desse tipo de modelo.

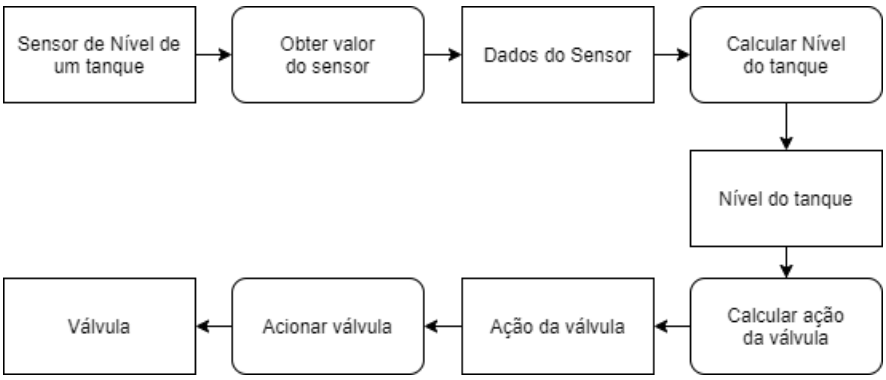

Figura 1.2: Modelo de sequência

1.2.2 Arquitetura em Camadas

Arquitetura que organiza o sistema em um grupo de camadas, onde cada uma delas oferece um conjunto de serviços. Essa noção de separação e independência é fundamental para que haja a possibilidade de alterações localizadas, facilitando as etapas de manutenção e sustentação do produto de software. A figura 1.3 traz uma representação genérica deste tipo de arquitetura.

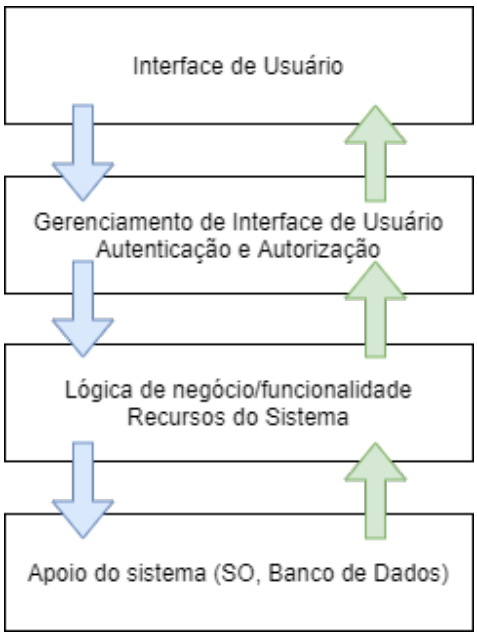

Figura 1.3: Arquitetura genérica em camadas

O relacionamento entre as camadas se dá de forma bem definida. Possui protocolos que indicam como a comunicação entre elas se dá. Uma camada só solicita serviços à inferior, e somente fornece à superior. Esse tipo de divisão facilita e colabora para o processo de desenvolvimento incremental. Uma arquitetura deste tipo muito utilizada e conhecida é a *MVC*, sigla do inglês para *Model-View-Controller*.

1.2.3 Orientação a Objetos

O conceito de OO (Orientação a objetos) consiste na estruturação de um código ou programa utilizando de abstrações para conectar o espaço do problema ao da solução. Os vários objetos construídos nessas abstrações se relacionam através de comandos. Cada um desses objetos possui uma *classificação* que o determina, chamada de *classe*.

Para ficar claro como os objetos se relacionam, os conceitos que seguem são importantes :

- **Encapsulamento:** Estratégia utilizada para garantir que detalhes internos do funcionamento de uma classe. Dessa forma, o conhecimento sobre a implementação interna de uma classe é desnecessária do ponto de vista da instância daquela classe.;
- **Herança:** Quando uma classe pode ser um *tipo* de uma outra classe, o conceito de herança pode ser aplicado. Se existe uma classe *veículo* e se deseja construir uma representação *motocicleta*, a segunda (classe filha) pode herdar características da primeira (classe pai), visto que *bicicleta* é um tipo de *veículo*;
- **Polimorfismo:** É o princípio pelo qual duas ou mais classes derivadas de uma mesma classe pai podem invocar métodos que têm a mesma identificação (assinatura) mas comportamentos distintos, especializados para cada classe derivada, usando para tanto uma referência a um objeto do tipo da superclasse.;

1.2.4 Modelo Relacional de Banco de Dados

Banco de dados é uma mecanismo de armazenamento que permite persistência de dados de uma aplicação, programa ou produto de software. O modelo relacional busca o aumento da independência de dados nos sistemas. O Modelo relacional revelou-se ser o mais flexível e adequado ao solucionar os vários problemas que se colocam no nível da concepção e implementação da base de dados. A estrutura fundamental do modelo relacional é a relação. Ela é constituída por um ou mais atributos que traduzem o tipo de dados a armazenar [3].

1.3 Conceitos de Processos industriais

1.4 Trabalhos Relacionados

Referências Bibliográficas

[1] RUMBAUGH J. JACOBSON BOOCH, G. *UML, Guia do Usuario*. 2ª edition, 2005.

[2] Ian SOMMERVILLE. *Engenharia de Software*. 9ª edition, 2011.

[3] Osvaldo Kotaro Takai. *IntroduÃ§Ã£o a Banco de Dados*. 2005.