Sistemas Distribuídos e Programação Concorrente

Introdução a Arquitetura de Software

Prof. MSc. Rodrigo Daniel Malara

Arquitetura de Software

INTRODUÇÃO

Introdução

- Histórico
- Contexto
  - Interessados ou Stakeholders
  - Perfil profissional da Equipe
  - Princípios e Boas Práticas Atuais
  - Processos de desenvolvimento de Software
- Definições
- Objetivos
- Atributos de Qualidade e Decisões Arquiteturais

Relevância

Quando podemos dizer que um projeto de desenvolvimento de software teve sucesso?

- Quando ele compila?
- Quando ele passa em todos os testes?
- Quando ele tem uma boa performance em produção?

O seu software obtém sucesso quando a sua empresa obtém sucesso usando o seu software!

A Arquitetura de Software deve focar primeiramente objetivos de negócio da empresa e depois em tecnologias!

Por quê um projeto de Software falha?

- Tipos de riscos associados a projetos de software
  - Requisitos
  - Políticos
  - Recursos disponíveis
  - Conhecimentos das equipes
  - Tecnologia utilizada
- Projetos geralmente falham devido a qualidades sistêmicas que não foram identificadas ou atingidas.
- Exemplos:
  - Performance, Vazão, Escalabilidade
  - Segurança
  - Extensibilidade e Flexibilidade
- Qualidades Sistêmicas são diretamente impactadas pela arquitetura
  Ou Requisitos não-funcionais

Mitigação de Riscos

- Adotar um processo de desenvolvimento de software bem estabelecido
- Endereçar os problemas mais difíceis primeiro
  - Casos de uso mais complexos
  - Sistemas distribuídos ao invés de locais
  - Integração com aplicações legadas
  - Gerenciamento de transações – integridade dos dados
    - Envolvimento de múltiplos mecanismos de armazenamento
  - Dividir para conquistar
- Comunicação adequada com interessados
- Utilizar uma estratégia de melhoria de processo de software (ex: CMMI)

Arquitetura vs Projeto

- A diferença está no contexto e como interpretar
- Um diagrama de arquitetura em um contexto pode se tornar um diagrama de projeto em outro contexto
- Arquitetura está preocupada com decisões globais
- Projeto está preocupado com decisões locais
- Arquitetura está preocupada com o projeto do sistema
  - Como os componentes irão se comunicar e os custos associados?
  - Quais são as interfaces? Elas seguem padrões?
    - Evitar Aprisionamento Tecnológico
  - Performance em nível de sistema
- Projeto está preocupado com partes do sistema
  - Como os componentes serão programados?
  - Quais algoritmos serão utilizados?

Histórico

- 1996: Publicação da versão 1.0 do DoDAF

  Department of Defense Architecture Framework
- 1995: Publicação da versão 1.0 do TOGAF

  “The Open Group Architecture Framework”
- 1992: Perry & Wolf

  “Foundations for the Study of Software Architecture”

Histórico

- 1989: M. Shaw
  "Software Architecture: Perspectives on an Emerging Discipline"
- 1987: Publicação da versão 1.0 do Zachman Framework
- 1978/79: David Parnas, program families

  "On the Design and Development of Program Families"
- 1972 (1969): Edsger Dijkstra

  "The Humble Programmer"

Histórico

- 1969: I.P. Sharp @ NATO Conferência de Engenharia de Software

  “Eu acredito que tenhamos algo diferente, além da Engenharia de Software [..] Essa é a Arquitetura de Software. Arquitetura é diferente da Engenharia"

Arquitetura de Software

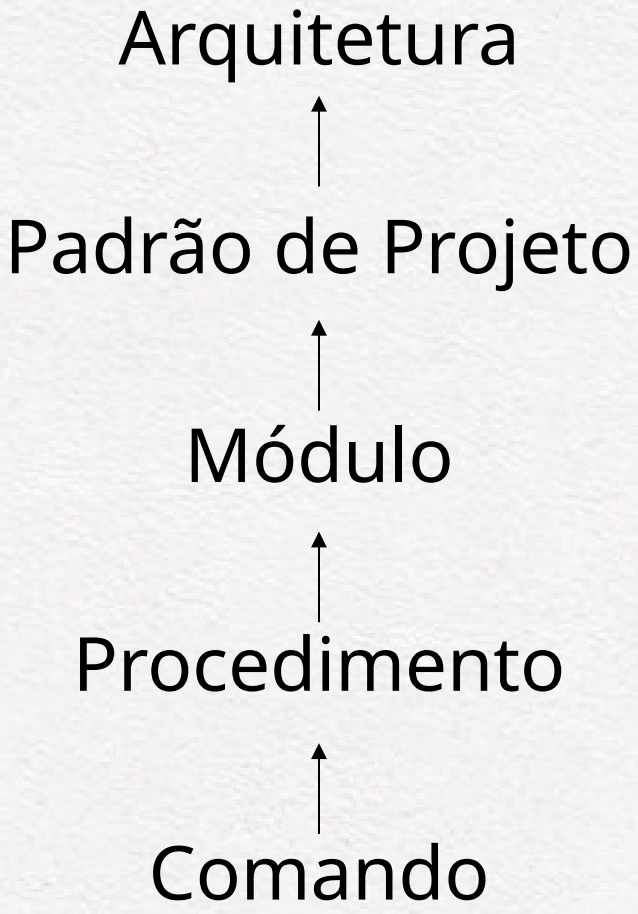
DEFINIÇÕES DE ARQUITETURA DE SOFTWARE

Definições de Arquitetura de Software

"A arquitetura de um componente de software define um sistema em termos de componentes computacionais e interações entre esses componentes"

De Shaw e Garlan, Software Architecture, Perspectives on an Emerging Discipline, Prentice-Hall, 1996.

Estrutura de Software



Definições de Arquitetura de Software

“A arquitetura de software de um Sistema é a estrutura ou estruturas do Sistema, que compreende elementos de software, propriedades externamente visíveis desses elementos e os relacionamentos entre eles.”

De Bass, Clements, e Kazman em ‘Software Architecture in Practice’, SEI Series in Software Engineering. Addison-Wesley, 2003.

Definições de Arquitetura de Software

- Sobre a definição de Bass et al:
- Assuntos importantes abordados na definição
  - Múltiplas estruturas de sistemas
  - Propriedades de componentes observáveis ou externamente visíveis

- Assuntos não abordados:
  - O processo
  - Regras e boas práticas
  - Modelos ou estilos arquiteturais

Definições de Arquitetura de Software

“Arquitetura é a organização fundamental de um sistema, representada pelos seus componentes, seus relacionamentos entre si e os princípios guiando seu projeto e evolução”

De IEEE Standard on the Recommended Practice for Architectural Descriptions, 2000..

Definições de Arquitetura de Software

- Arquitetura é conceitual
- Arquitetura é sobre coisas fundamentais
- Arquitetura existe em um contexto
- Arquitetura é um projeto de alto nível
- Arquitetura é a estrutura de um sistema, incluindo princípios que governam seu projeto e evolução no tempo

Arquitetura de Software é Importante

- A Arquitetura de Software é um veículo para comunicação entre interessados (stakeholders)
- A Arquitetura manifesta as decisões de projeto mais cedo
  - Restrições para implementação
  - Dita a estrutura organizacional de projeto
  - Inibe ou Ativa atributos de qualidade
- A Arquitetura é uma abstração de um Sistema
- Produtos similares podem compartilhar a mesma arquitetura
- Permite o desenvolvimento baseado em padrões
- Serve de base para treinamentos

Arquitetura de Software

O CONTEXTO ARQUITETURAL

O Contexto Arquitetural

- A Arquitetura de Software faz mais sentido e pode ser melhor elaborada levando-se em conta um contexto
- Esse contexto envolve
  - Interessados ou Stakeholders
    - Perfil professional da Equipe de Desenvolvimento e Operações (DevOps)
  - Recursos disponíveis
  - Requisitos de funcionamento sobre o software a ser desenvolvido
    - Ou Qualidades Sistêmicas
  - Processo de Desenvolvimento
  - Filosofias de Desenvolvimento e boas práticas acumuladas

O Contexto Arquitetural

- Interessados ou Stakeholders
  - Do lado do cliente
    - Dono da empresa ou diretores
    - Usuários
    - Gerentes dos Usuários
  - Do lado da equipe de desenvolvimento
    - Gerente de Projeto
    - Analista de Negócio
    - Arquiteto de Software
    - Desenvolvedor/Programador
    - Testador
    - Analista de Infraestrutura

O Contexto Arquitetural

- Interessados ou Stakeholders
  - O arquiteto deve considerar a disponibilidade e a qualificação da mão-de-obra para o desenvolvimento e manutenção da aplicação
  - De acordo com Robert C. Martin no livro Arquitetura Limpa

'O propósito principal de uma arquitetura é suportar o ciclo de vida de um Sistema. Boa arquitetura faz com que o Sistema seja fácil de entender, fácil de se desenvolver, fácil de se manter e fácil de se implantar. O maior objetivo é minimizar o custo total, durante toda a existência do Sistema e maximizar a produtividade do programador.'

O Contexto Arquitetural

*"Requisitos definem **o que** um sistema deve fazer e sob **quais restrições**" Engenharia de Software Moderna por M. Valente*

- Requisitos relacionados com o que um sistema deve fazer, ou seja, suas funcionalidades — são Requisitos Funcionais.
- Os requisitos relacionados com a segunda parte — sob quais restrições — são Requisitos Não-Funcionais ou de Qualidade.
- **Requisitos Funcionais** são de responsabilidade de **Analistas de Negócio**
- **Requisitos Não-Funcionais** são de responsabilidade de **Arquitetos de Software**

O Contexto Arquitetural

- A noção de qualidade é central na arquitetura de software
- Uma arquitetura de software busca garantir as qualidades de um Sistema o mais cedo possível
- Algumas qualidades são observáveis pela execução:
  - Performance, Segurança, Disponibilidade,
  - Usabilidade, Escalabilidade
- Outras não são observáveis durante a execução
  - Manutenibilidade, Portabilidade, Reusabilidade, Testabilidade

O Contexto Arquitetural

Exemplos de Qualidades ou Requisitos Não-Funcionais:

- Performance ou Desempenho

  “Ao registrar um item sendo vendido, a descrição e preço devem aparecer em, no máximo, 2 segundos”

- Segurança

  "Apenas usuários com privilégios de acesso de poderão visualizar históricos de transações de clientes."

- Disponibilidade

  "O sistema estará disponível pelo menos 99,7% do tempo em dias de semana entre meia-noite e 06:00 e pelo menos 99,95% entre 16:00 e 18:00"

O Contexto Arquitetural

Exemplos de Qualidades ou Requisitos Não-Funcionais:

- Escalabilidade

  “O Sistema deve desempenhar bem com 50 usuários simultâneos e poder atender até 1.000.000 de usuários simultâneos apenas com redimensionamento de hardware”

- Interoperabilidade

  “O Sistema de Gerenciamento de Ativos deve ser capaz de integrar com leitores de QR-Code 1D e 2D e imprimir em impressoras de códigos de barra padrão Zebra”

- Confiabilidade

  “Nenhuma atualização de dados em memória permanente pode ser perdida"

O Contexto Arquitetural

Exemplos de Qualidades ou Requisitos Não-Funcionais:

- Usabilidade

  "Um novo usuário deverá ser capaz de fazer um pedido de compra de um novo produto químico após não mais do que 30 minutos de orientação"

- Manutenibilidade

  "Um programador com pelo menos 6 meses de experiência no suporte ao produto deverá ser capaz de resolver um defeito simples em não mais do que 1 hora de trabalho"

- Portabilidade

  "O produto deverá ser implantável em sistemas operacionais Linux ou Windows"

O Contexto Arquitetural

Exemplos de Qualidades ou Requisitos Não-Funcionais:

- Reusabilidade

  "O produto deverá usar componentes corporativos existentes sob forma de Enterprise JavaBeans. Novos componentes deverão ser EJBs"

- Testabilidade

  "Testes de Unidade e de Aceitação deverão ser completamente automatizados"

- Tolerância a falhas

  "O sistema deve fazer log dos pagamentos autorizados via cartão de crédito em 24 horas, mesmo com falhas de energia ou de dispositivo"

O Contexto Arquitetural

- Recursos Disponíveis
- O arquiteto deve considerar os recursos financeiros tangíveis e intangíveis para:
  - Contratação de equipe ou empresa para o desenvolvimento,
  - Adquirir licenças de software e ferramentas de desenvolvimento
  - Contratar ou adquirir infraestrutura computacional
  - Analisar possibilidade de reuso de recursos já existentes

O Contexto Arquitetural

- Processo de Desenvolvimento de Software

"Um processo de desenvolvimento de software é um conjunto de atividades, parcialmente ordenadas, com a finalidade de obter um produto de software. É estudado dentro da área de Engenharia de Software, sendo considerado um dos principais mecanismos para se obter software de qualidade e cumprir corretamente os contratos de desenvolvimento, sendo uma das respostas técnicas adequadas para resolver a crise do software."

- Exemplos:
  - SCRUM, eXtreme Programming, Processos ágeis unificados

O Contexto Arquitetural

- Filosofias de Desenvolvimento e boas práticas
  - Reuso de código
  - Projeto e Desenvolvimento Orientado a Objetos
  - Padrões de Projetos
    - GoF ‘Gang of Four’
    - Java Enterprise Edition
    - Model View Controller
  - SOLID
  - Teorema CAP
  - YAGNI
  - DRY
  - KISS
- A serem estudados nas próximas aulas

Referências Bibliográficas

- GALLOTTI, G.M.A. **Arquitetura de Software**. Editora Pearson, São Paulo, 2016.

- ERL, T. **SOA: Princípios de Design de Serviços**.São Paulo, Prentice Hall, 2009.

- SOMMERVILLE, Ian. **Engenharia de Software**. 10. Ed. – São Paulo: Pearson Education do Brasil, 2018.