Modelagem de Sistemas: Conceitos Essenciais

Marco Mendes
masmendes@pucminas.br
Junho de 2021

1 MODELAGEM DE SISTEMAS

Construímos modelos de sistemas complexos porque não é possível compreendê-los em sua totalidade. Isso se deve aos limites da capacidade humana de compreender complexidades. Com a ajuda da modelagem, podemos delimitar o problema que estamos estudando, restringindo nosso foco a um único aspecto por vez. Em essência esse é o procedimento de "dividir-para-conquistar", do qual Edsger Dijkstra expressou tão perfeitamente, ainda anos 60.

"Ataque um problema difícil, dividindo-o em vários problemas menores que você pode solucionar. ", Dijkstra Edsger

Com o auxílio da modelagem somos capazes de ampliar o intelecto humano. Um modelo escolhido de maneira adequada permitirá a quem usa a modelagem trabalhar em níveis mais altos de abstração. Em resumo, modelos são usados para simplificar os problemas de engenharia de software

A modelagem de sistemas tem quatro benefícios primários:

- **Ajudar na Compreensão de Sistemas Complexos -** A importância dos modelos aumenta à medida que os sistemas se tornam mais complexos. Por exemplo, uma oca indígena pode ser construída por algumas pessoas em uma aldeia em poucos dias sem ferramentas ou técnicas formais de engenharia. No entanto, à medida que evoluímos para casas e, depois, para arranha-céus, a necessidade de um projeto torna-se evidente. O projeto da Oca do Ibirapuera, de Oscar Niemayer, por exemplo, somente pode ser construído com um grande projeto de engenharia e muitos modelos.
- **Explorar e Comparar as Alternativas de Design a um Baixo Custo -** Modelos simples podem ser criados e modificados a um baixo custo para explorar alternativas de design. Idéias inovadoras podem ser capturadas e revisadas por outros desenvolvedores antes de investir em um desenvolvimento de código caro. Quando ligada ao desenvolvimento iterativo, a modelagem visual ajuda os desenvolvedores a avaliar mudanças de design e a comunicar essas mudanças a toda a equipe de desenvolvimento.
- **Prover uma Fundação para Implementação -** Modelos podem ser automatizados por ferramentas e técnicas diversas e permitem que uma parcela da implementação seja completamente automatizada. Ferramentas de engenharia direta e reversa são exemplos concretos deste aspecto.
- **Capturar Requisitos com Precisão -** Antes de construir um sistema, é essencial capturar os requisitos arquiteturais. A especificação dos requisitos utilizando um modelo preciso ajuda a assegurar que todos os investidores entendam e concordem com os requisitos. Um modelo que separa o comportamento externo do sistema da implementação o ajuda a se concentrar no uso pretendido do sistema, sem ficar perdido nos detalhes de implementação.

2 OS QUATRO PRINCÍPIOS DA MODELAGEM DE SISTEMAS

1. **A escolha dos modelos** a serem criados possui um enorme impacto em como o problema é atacado e como a solução é definida.

2. Cada modelo poderá ser expresso em **diferentes níveis de precisão**.

3. Os melhores modelos são **conectados à realidade**.

4. **Nenhum modelo único é suficiente**. Qualquer sistema não-trivial será mais bem investigado por meio de um pequeno conjunto de modelos quase independentes de vários pontos de vista. Todo sistema complexo é melhor alcançado através de diferentes visões de um modelo. Uma visão única não é suficiente. Todo modelo pode ser expresso em diferentes níveis de fidelidade. Todo sistema complexo é melhor alcançado através de diferentes visões de um modelo. Uma visão única não é suficiente. Todo modelo pode ser expresso em diferentes níveis de fidelidade. A linguagem UML, por exemplo, define vários diagramas gráficos para representação de modelos.

3 MODELAGEM ÁGIL

A modelagem não deve ser percebida como um aspecto burocrático que requeira o uso de ferramentas caras e gere documentos pesados. A escola de modelagem ágil, popularizada em (Ambler, Agile Modeling: Effective Practices for eXtreme Programming and the Unified Proces, 2002) gerou frutos também na área de arquitetura de software. Mais recentemente, o conceito de arquiteturas ágeis (Ambler, Agile Architecture: Strategies for Scaling Agile Development). Modelos ágeis, conforme a figura 3, podem apoiar fortemente a comunicação de arquiteturas entre times técnicos.

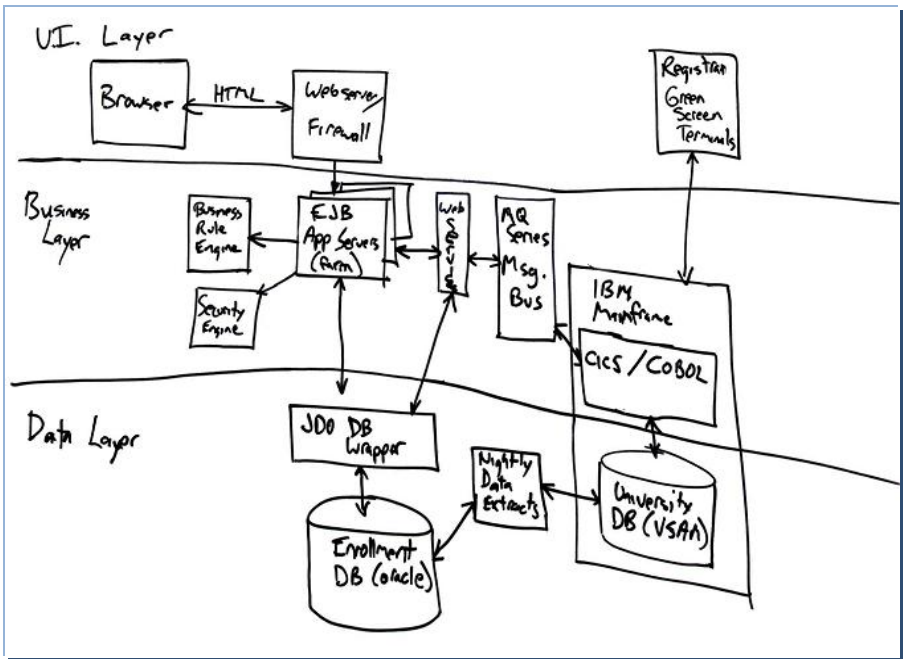

Figura 3: Arquiteturas ágeis (Fonte: Scott Ambler, Agile Modeling)

4 MODELO DE VISUALIZAÇÃO 4+1

A ideia do modelo de visualização 4+1 foi proposto em (Kruchten, 1995) e se baseia nos seguintes princípios.

- Um **modelo** é uma simplificação (abstração) da realidade, que permite entender melhor o sistema que será criado;
- Uma **visualização** é uma representação do sistema, como um todo, da perspectiva de um conjunto de interesses relacionados;
- Uma **preocupação** pode se referir ao desenvolvimento do sistema, sua operação ou qualquer outro aspecto, que seja crítico ou de alguma forma importante para um ou mais envolvidos;
- Um **envolvido** é um indivíduo, equipe, ou organização com interesses em conceitos relativos ao sistema.

Uma arquitetura tem na maioria das vezes muitos envolvidos diferentes. Exemplos incluem: Usuário final, cliente, administrador de sistema, gerente de projetos, analista de suporte, desenvolvedor, arquiteto ou testador. Para isso, precisamos capturar múltiplas realidades, múltiplas visões e múltiplos modelos de desenho técnico existentes.
As visualizações de arquitetura podem ser vistas como abstrações ou simplificações dos modelos construídos, nas quais você torna mais visíveis as características importantes, deixando os detalhes de lado.

Visualização de casos de uso
A visualização de casos de uso mostra um subconjunto do modelo de casos de uso, um subconjunto de casos de uso e atores significativos para a arquitetura.

Visualização lógica
A visualização lógica mostra um subconjunto do modelo de design significativo em termos de arquitetura, ou seja, um subconjunto das classes, subsistemas, pacotes e realizações de caso de uso.

Visualização de implementação
A finalidade da visualização de implementação é captar as decisões de arquitetura tomadas para a implementação e gestão da configuração. Normalmente, a visão de implementação contém:

- uma enumeração de todos os subsistemas no modelo de implementação.
- diagramas de componentes que ilustram como os subsistemas são organizados em camadas e hierarquias.
- ilustrações de dependências de *importação* entre subsistemas.

Visualização de implantação
A visão de implantação mostra a distribuição física do processamento no sistema. A visão de implantação ilustra a distribuição do processamento em um conjunto de nós do sistema, incluindo a distribuição física dos processos e threads.

Visualização de processos

Esta visualização ilustra a decomposição do processo do sistema, incluindo o mapeamento de classes e subsistemas para processos e linhas de execução (threads). Esta visualização captura aspectos de infraestrutura de sistemas operacionais e são usadas para modelar software básico.

5 APLICAÇÃO DO MODELO DE VISUALIZAÇÃO 4+1

Este método assume como ponto de partida (o elemento +1) um conjunto de *cenários significativamente relevantes* para a modelagem arquitetural. Um cenário é um caminho ou percurso percorrido por um usuário na sua interação como o sistema sob desenho, extraído de um requisito arquiteturalmente significativo. Por exemplo, em um projeto que lide com a integração de um sistema entre plataformas baixas e altas, podemos elencar um cenário de transporte de informações (interoperabilidade) para análise. Uma lista de casos de uso pode ser um ponto de partida para que estes cenários sejam elencados.

Para cada cenário elencado, devemos explorá-lo em diversas outras visões UML conforme os requisitos arquiteturais do projeto.

Considerações do Modelo de Visualização 4+1
Nem todas as visões são obrigatórias. Algumas dicas para escolher que visões usar incluem:

- Em sistemas de baixa complexidade técnica, normalmente podemos ignorar a visão de implantação e implementação.
- Em sistemas de informação, normalmente podemos ignorar a visão de processos. A visão de processos tem aplicabilidade em sistemas de software básico, sistemas embarcados e dispositivos móveis.

Podemos elencar cinco vantagens primárias do uso deste método.

- As múltiplas representações capturam os interesses de múltiplos *stakeholders*.
- As múltiplas representações permitem realizar validações cruzadas e expor falhas prematuras no modelo.
- As visões estão centradas nos cenários de risco do projeto e portanto representam uma estratégia de mitigação de riscos técnicos.
- As visões são independentes e podem ser escolhidas conforme o contexto do problema sendo atacado (uso de duas, três, quatro ou cinco visões).
- Finalmente, estas visões permitem realizar um elo entre a visão do usuário (casos de uso) e o código. Sem estas visões, o desenvolvedor realizaria suposições diversas e provavelmente cada caso de uso teria um determinado formato. Em outras palavras, teríamos um sistema sem coerência arquitetural (estratégica). Metaforicamente, estaríamos criando um animal com patas de elefante, asas de morcego e cabeça de tamanduá.

6 Referências Adicionais

Kruchten, P. B. (1995). The 4+ 1 view model of architecture. *IEEE software*, *12*(6), 42-50.

Fowler, M. (2014). *UML Essencial: um breve guia para linguagem padrão*. Bookman editora.

Eeles, P., & Cripps, P. (2009). *The process of software architecting*. Pearson Education.