

Semana 8 : Livro - Projeto de Software: Do Requisito ao Blueprint

Site: [Boas-vindas ao Moodle do Ifes](#)
Curso: Fundamentos de Tecnologia da Informação
Livro: Semana 8 : Livro - Projeto de Software: Do Requisito ao
Blueprint

Impresso por: Marcelo de Oliveira Rodrigues
Data: quarta-feira, 15 out. 2025, 11:45



Índice

1. A Transição do "O Quê" para o "Como"

1.1. Do Requisito à Especificação Técnica

2. Stakeholders Envolvidos na Fase de Projeto

2.1. Papéis e Responsabilidades Chave

3. Tarefas e Trabalhos Executados

3.1. Design da Arquitetura do Software

3.2. Design de Banco de Dados

3.3. Interface e Experiência do Usuário (UI/UX)

3.4. Desenho de Componentes e Plano de Testes

4. Técnicas e Documentos de Modelagem

4.1. Utilização de Diagramas UML

4.2. Modelagem de Dados e Padrões Comprovados

5. Importância Estratégica e Saídas Esperadas

5.1. Benefícios Chave da Fase de Design

5.2. Saídas Esperadas



1. A Transição do "O Quê" para o "Como"

A criação de um software passa por diversas etapas cruciais, e a fase de projeto marca uma transformação fundamental no ciclo de vida do desenvolvimento. É o ponto onde a visão abstrata do produto, definida pela análise de requisitos, começa a ganhar forma técnica e estruturada. Esta fase transforma as ideias e requisitos em um blueprint, ou seja, um mapa detalhado, indispensável para a construção subsequente do software.



1.1. Do Requisito à Especificação Técnica

Se a fase anterior, a de análise, tinha como objetivo responder à pergunta "O Quê?" (ou seja, quais são as necessidades de negócio), a fase de projeto foca intensamente em responder ao "Como?".

Este é o momento de tomar decisões concretas sobre como o sistema funcionará internamente. Envolve a definição da arquitetura, a estrutura global e os componentes individuais do sistema. O principal resultado é a tradução das necessidades de negócio em uma especificação técnica detalhada que a equipe de desenvolvimento poderá seguir com clareza.



2. Stakeholders Envolvidos na Fase de Projeto

Embora a fase de projeto seja inerentemente técnica, exigindo um foco intenso da equipe de engenharia, a colaboração com outras partes interessadas (stakeholders) continua sendo vital. O sucesso do design depende de um equilíbrio entre a viabilidade técnica e o alinhamento com os objetivos de negócio e o orçamento. Nesta fase, a equipe técnica assume um papel mais proeminente.



2.1. Papéis e Responsabilidades Chave

Diferentes profissionais desempenham funções específicas para garantir que o blueprint seja robusto e coerente com os requisitos:

- **Arquiteto de Software:** Este profissional, geralmente um desenvolvedor sênior ou líder técnico, é o principal responsável por projetar a arquitetura geral do sistema. Ele é o tomador de decisões técnicas de alto nível.
- **Equipe de Desenvolvimento:** São os programadores e engenheiros que, futuramente, construirão o produto. Eles participam ativamente do processo de projeto, fornecendo feedback essencial sobre a viabilidade técnica das soluções propostas e auxiliando na estimativa do esforço necessário para a implementação.
- **Gerente de Projeto:** Continua a supervisionar o andamento geral. Seu papel nesta fase é crucial para garantir que o plano de design elaborado esteja alinhado com o cronograma estabelecido e o orçamento disponível para o projeto.
- **Analista de Sistemas/Negócios:** Mantém-se como a ponte de comunicação entre a equipe técnica e os stakeholders do negócio. Sua responsabilidade é garantir que, apesar de todas as decisões técnicas, o projeto final ainda atenda aos requisitos originais definidos.



3. Tarefas e Trabalhos Executados

A fase de projeto abrange uma série de atividades detalhadas que elevam o design do software de um conceito geral a uma descrição minuciosa, desde a sua arquitetura geral até o desenho de seus componentes menores.



3.1. Design da Arquitetura do Software

Esta é frequentemente descrita como a atividade mais crucial da fase, pois define a estrutura geral do sistema. As decisões tomadas aqui impactam a escalabilidade, manutenção e custo do software:

- **Estrutura do Sistema:** Decidir se o sistema será construído usando uma Arquitetura Monolítica (um único e grande aplicativo) ou Microsserviços (um conjunto de serviços menores e independentes).
- **Padrões de Design:** Determinar quais padrões serão utilizados para organizar o código, como o padrão MVC (Model-View-Controller).
- **Tecnologias:** Selecionar as linguagens de programação, os frameworks de suporte e os bancos de dados que serão utilizados na construção.



3.2. Design de Banco de Dados

O banco de dados é considerado o coração do sistema por armazenar e gerenciar todos os dados essenciais. Esta etapa envolve projetar as tabelas, definir as relações entre elas e determinar a forma mais eficiente de armazenar e recuperar os dados. Para visualizar essa estrutura, é fundamental a criação de um Modelo de Dados, frequentemente representado por um Diagrama de Entidade-Relacionamento (DER).



3.3. Interface e Experiência do Usuário (UI/UX)

Embora a fase de análise possa ter gerado protótipos de baixa fidelidade para validar os requisitos, a fase de projeto aprofunda este design. A equipe de UI/UX trabalha para definir a aparência e o fluxo de navegação do software, criando artefatos como:

- **Wireframes:** Esboços simples que definem a estrutura básica.
- **Mockups:** Designs mais detalhados, incorporando cores, tipografia e imagens.
- **Protótipos interativos de alta fidelidade.**



3.4. Desenho de Componentes e Plano de Testes

Para tornar o sistema gerenciável, ele é dividido em módulos e componentes menores. Para cada um desses componentes, a equipe deve documentar suas responsabilidades específicas, como ele se comunicará com outros componentes e quais são os requisitos técnicos necessários para sua implementação.

Adicionalmente, antes que a codificação efetiva comece, um Plano de Testes é elaborado. Este plano detalha os tipos de testes a serem realizados (incluindo testes de unidade, integração, sistema e aceitação), define os responsáveis por sua execução e especifica as ferramentas que serão utilizadas.



4. Técnicas e Documentos de Modelagem

Para que o design se materialize em um plano concreto, são produzidos diversos documentos e diagramas. Estes artefatos visuais e formais são cruciais para comunicar as decisões técnicas de maneira inequívoca.



4.1. Utilização de Diagramas UML

A UML (Unified Modeling Language) é amplamente utilizada para visualizar e documentar o projeto de software. Alguns dos diagramas mais comuns produzidos incluem:

- **Diagrama de Classe:** Essencial para mostrar as classes do sistema, detalhando seus atributos e métodos, e as relações existentes entre elas.
- **Diagrama de Sequência:** Descreve a interação dinâmica entre os objetos e especifica a ordem exata em que as mensagens são enviadas dentro de um fluxo de execução.
- **Diagrama de Componentes:** Representa como os diferentes componentes de software se organizam e se conectam dentro da arquitetura.



4.2. Modelagem de Dados e Padrões Comprovados

Outras técnicas essenciais garantem a solidez estrutural e funcional do sistema:

- **Modelagem de Dados:** O Diagrama de Entidade-Relacionamento (DER) é uma ferramenta fundamental para estruturar o banco de dados. Ele define as entidades (objetos do mundo real), seus atributos específicos e as relações que as conectam.
- **Padrões de Design (Design Patterns):** A equipe utiliza padrões de design de software, que são soluções comprovadas para resolver problemas de design recorrentes. A aplicação destes padrões não só economiza tempo, como também melhora a qualidade do código e facilita significativamente a manutenção futura.



5. Importância Estratégica e Saídas Esperadas

A fase de projeto transforma a visão do produto em um plano concreto e executável. A ausência de um design sólido é um risco significativo, pois pode levar a problemas de arquitetura e um sistema difícil de evoluir. Pelo contrário, um projeto bem-sucedido permite que a fase de implementação ocorra de forma fluida, garantindo que a equipe de desenvolvimento tenha total clareza sobre o que e como construir.



5.1. Benefícios Chave da Fase de Design

Investir tempo e recursos na fase de projeto traz vantagens estratégicas claras:

- **Redução de Custos:** Alterações feitas no design são significativamente mais baratas do que correções realizadas em código já pronto.
- **Evita Retrabalho:** Como todos os detalhes técnicos são definidos previamente, o retrabalho durante a programação é minimizado.
- **Facilita Manutenção:** Sistemas com uma arquitetura bem definida e documentada são inerentemente mais fáceis de evoluir e dar manutenção.
- **Melhora a Comunicação:** O processo de design formaliza as decisões, melhorando a comunicação e o entendimento entre a equipe técnica e os stakeholders do negócio.



5.2. Saídas Esperadas

Ao final da fase de Projeto, a equipe deve ter produzido um conjunto completo de artefatos que constituem o blueprint final para a construção. Os entregáveis esperados incluem:

- Documento de design de software (SDS).
- Protótipos de interface (wireframes e mockups).
- Modelagem do banco de dados concluída.
- Diagramas arquiteturais, como diagramas de componentes, classes e implantação.
- Definição clara de APIs e contratos de comunicação entre os módulos.
- Padrões de qualidade, segurança e codificação a serem seguidos.

