**Resumo do Capítulo 6**
No Capítulo 6, se fala sobre os desafios e soluções que aparecem no desenvolvimento de sistemas distribuídos, especialmente quando se trata de usar protocolos de comunicação como TCP e UDP. Um dos maiores perrengues que os desenvolvedores enfrentam é a necessidade de mudar entre diferentes protocolos sem quebrar o funcionamento do sistema. Uma solução bacana que aparece é o padrão de design Fábrica, que permite criar objetos sem que o código do cliente precise entender os tipos exatos. Isso dá uma flexibilidade maior, já que a gente pode trocar implementações sem precisar mexer em outras partes do código, centralizando a lógica de como as coisas são criadas.

Um exemplo de como implementar esse padrão é pelo método `ChannelFactory.create()`, que ajuda a alternar entre diferentes maneiras de comunicação. O capítulo também sugere que dá para expandir essa ideia usando fábricas abstratas, que podem criar diferentes tipos de objetos que estão relacionados. Essa flexibilidade é super importante em ambientes ágeis, onde as necessidades e tecnologias mudam o tempo todo, permitindo que os desenvolvedores se concentrem em criar soluções robustas e escaláveis.

No final, a discussão sobre o padrão Fábrica mostra como ele é essencial na arquitetura de sistemas distribuídos. Ao centralizar a criação de objetos e permitir mudanças nas implementações sem afetar o código já existente, esse padrão se torna uma ferramenta muito útil para os desenvolvedores. Isso não só torna o processo de desenvolvimento mais eficiente, mas também facilita a manutenção e a evolução dos sistemas ao longo do tempo.

---

**Resumo do Capítulo 7**
O Capítulo 7 começa com uma introdução sobre o que é arquitetura de software, destacando a importância de um design bem estruturado, que envolve componentes como pacotes, módulos e serviços. Esses elementos são fundamentais para a estrutura de um sistema, e a sua importância pode variar dependendo do que o projeto busca alcançar. O capítulo também discute decisões cruciais na arquitetura, como a escolha da linguagem de programação e do banco de dados, que podem ter impactos duradouros e criar desafios no futuro.

Entre os padrões arquiteturais que são discutidos estão a Arquitetura em Camadas e o Model-View-Controller (MVC), que ajudam a organizar a estrutura dos sistemas de maneira eficiente. O capítulo faz uma distinção entre padrões e estilos arquiteturais, embora ambos sejam considerados padrões dentro do contexto. A complexidade dessas decisões é ilustrada por meio da famosa discussão entre Tanenbaum e Torvalds sobre a arquitetura de sistemas operacionais, mostrando as consequências das escolhas entre microkernels e kernels monolíticos.

Uma parte interessante é a seção sobre microsserviços, que analisa as limitações das arquiteturas monolíticas, que podem enfrentar problemas de desempenho devido à interdependência entre módulos. Com os microsserviços, onde os módulos funcionam de forma independente, as empresas conseguem aumentar a escalabilidade e a resiliência dos sistemas. Porém, essa arquitetura traz novos desafios, como latência e gerenciamento de

transações distribuídas. O capítulo termina com a introdução ao conceito de anti-padrões arquiteturais, como a "big ball of mud", que ilustra a falta de organização e os problemas que surgem quando um sistema não tem uma estrutura arquitetônica clara, como acontece em um estudo de caso de um sistema bancário que cresceu de maneira desordenada.