Os padrões de projeto MVC, MVP e MVVM são arquiteturas amplamente utilizadas no desenvolvimento de software, especialmente em aplicativos móveis. Cada um deles possui características distintas que atendem a diferentes necessidades de design e desenvolvimento.

Model-View-Controller (MVC)

O **MVC** é um dos padrões mais antigos, criado na década de 1970. Ele divide a aplicação em três componentes principais:

* **Model**: Representa a lógica de negócios e os dados da aplicação.
* **View**: Responsável pela apresentação dos dados ao usuário.
* **Controller**: Atua como intermediário, recebendo entradas do usuário e atualizando o Model e a View.

Embora o MVC promova a separação de preocupações, ele pode levar a um "Controlador inchado", onde o controlador se torna complexo e difícil de manter.

Model-View-Presenter (MVP)

O **MVP** surgiu como uma melhoria ao MVC, buscando resolver a complexidade do controlador. Neste padrão, as responsabilidades são distribuídas da seguinte forma:

* **View**: Apresenta a interface gráfica e envia eventos ao Presenter.
* **Presenter**: Contém a lógica de apresentação, interagindo com o Model e atualizando a View. O Presenter é desacoplado da View, permitindo uma melhor testabilidade e manutenção.

O MVP é especialmente útil em aplicações onde a lógica de apresentação precisa ser separada da lógica de interface.

Model-View-ViewModel (MVVM)

O **MVVM** é um padrão mais recente, criado por engenheiros da Microsoft em 2005, e se popularizou no desenvolvimento Android a partir de 2015. Ele é estruturado da seguinte maneira:

* **View**: Similar ao MVP, é responsável pela interface gráfica e pela interação com o usuário.
* **ViewModel**: Contém a lógica de apresentação e mantém o estado da View. A ViewModel não tem conhecimento direto da View, permitindo um desacoplamento ainda maior.
* **Model**: Representa a lógica de negócios e os dados, como no MVC e MVP.

O MVVM se destaca pela sua capacidade de vinculação de dados, o que facilita a atualização automática da interface quando os dados mudam, tornando-o altamente testável e fácil de manter.

Comparação

* **Complexidade**: O MVC pode resultar em controladores complexos, enquanto o MVP e MVVM oferecem melhor separação de responsabilidades.
* **Testabilidade**: MVP e MVVM são mais testáveis que o MVC, com o MVVM geralmente sendo o mais fácil devido à vinculação de dados.
* **Uso**: O MVC é amplamente utilizado em aplicações web, enquanto o MVP e MVVM são mais comuns em aplicações móveis, especialmente no Android.