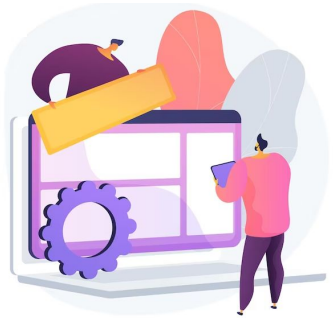
Disciplina: Desenvolvimento Web - *Back End*

Temática: Tópicos 4 e 5

Professor: Vinícius da Rocha Motta
vinicius.motta@uvv.br

A **arquitetura de um *software*** ou de um sistema de informação é a **organização** ou a **estrutura** dos **componentes** significativos do sistema de software, que **interagem por meio de interfaces**.

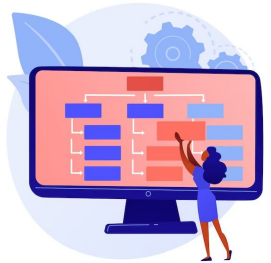


Uma **arquitetura** bem **projetada** deve atender aos **requisitos funcionais** e **não funcionais** de um sistema de informação e ser **flexível** para **atender a requisitos voláteis**.

A arquitetura **permite uma abstração reutilizável** do sistema de informação, caso haja **situações diferentes com características similares** (reuso de *software*).

Uma forma de **organizar a arquitetura** de um **sistema complexo** em **partes menores** é a utilização de **camadas de *software***.

Cada **camada** corresponderá a **um conjunto de funcionalidades** de um sistema de informação



UNIVERSIDADE VILA VELHA ESPÍRITO SANTO

As f**uncionalidades de alto nível** dependerão das **funcionalidades de baixo nível**, criando, assim, **coesão** dentro do sistema de informação.

Contexto de um cenário real para utilização de um sistema de informação

UNIVERSIDADE VILA VELHA ESPÍRITO SANTO

- O **modelo MVC** é um **padrão arquitetural com três camadas** que separa o sistema em **modelo**, **visão** e **controle**, oferecendo **abstração** e **agrupamento lógico** de **subsistemas**.

- Neste modelo as **camadas superiores dependem** das **camadas inferiores, isso** permite a modificação mais fácil do sistema.

- Ou seja, há **interdependência entre as camadas**, mas cada uma delas **possui graus de independência uma das outras**.

- Essa interdependência ocorre porque na arquitetura MVC **a divisão em camadas** permite a **melhor separação de responsabilidades** e, consequentemente, a **decomposição da complexidade da implementação do sistema de informação** (e diretamente, o **reuso deste sistema**), caso haja cenários de negócios similares, o que ajuda na **adaptabilidade**.

Contexto de um cenário real para utilização de um sistema de informação

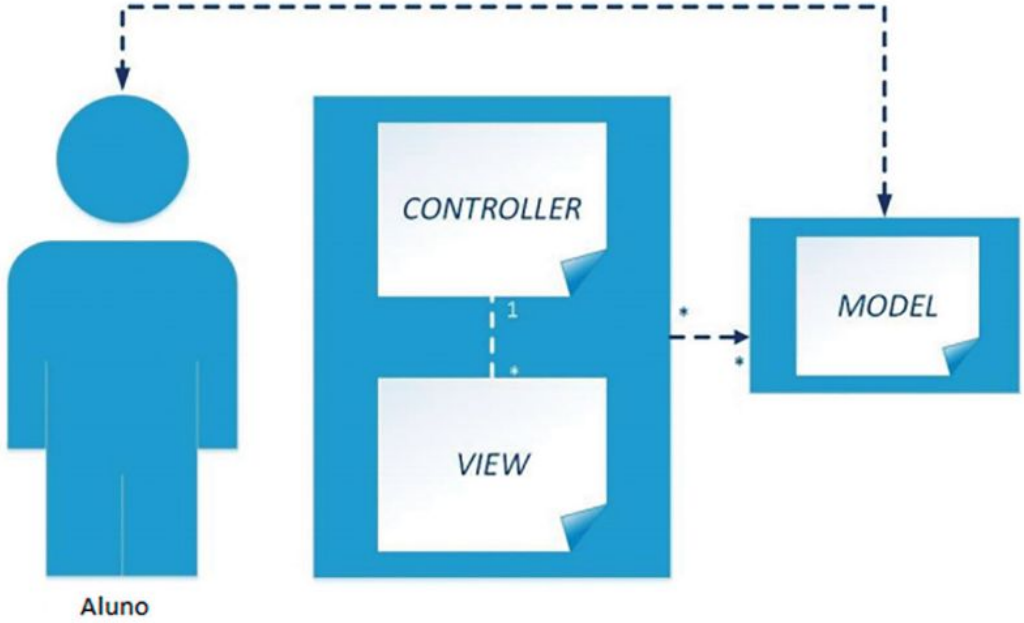


Figura 1. Arquitetura *Model-View-Controller*.

A Figura exemplifica o conceito arquitetural apresentado com um sistema escolar que fará o registro dos alunos no diário de classe e lançará suas notas.

Contexto de um cenário real para utilização de um sistema de informação

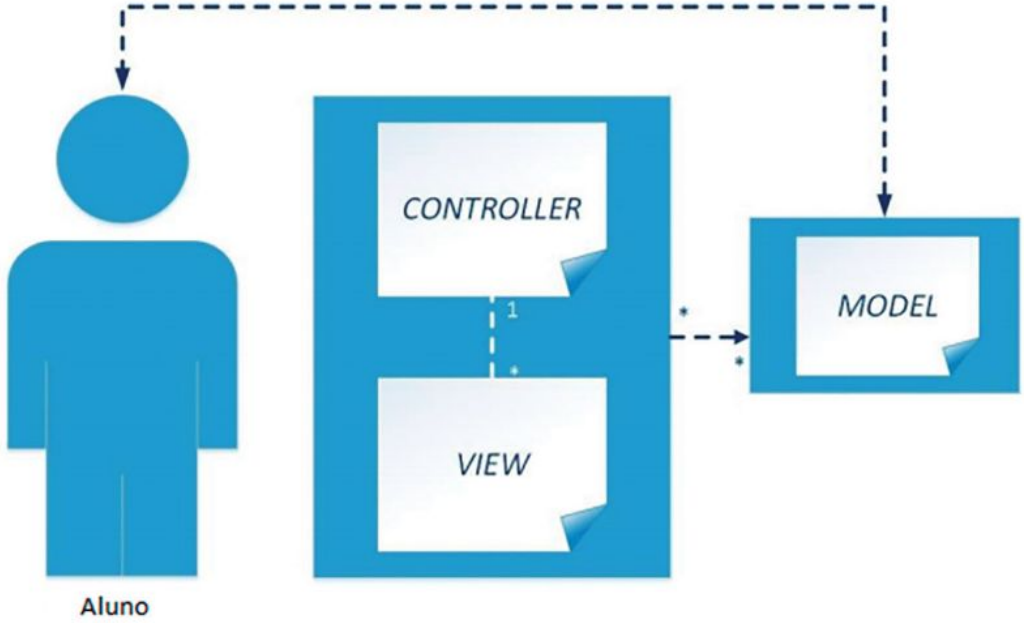


Figura 1. Arquitetura *Model-View-Controller.*

No exemplo apresentado temos a interação do aluno com um sistema de informação com as camadas ***Model***, ***Views*** **e** ***Controller***.

Contexto de um cenário real para utilização de um sistema de informação

UNIVERSIDADE VILA VELHA ESPÍRITO SANTO

- Temos, então, **três componentes** interagindo para que o aluno tenha o seu registro em sala de aula e possa consultar suas notas:
  - No componente ***Model***, temos o objeto da **aplicação** (modelo de negócio) ou **lógica** de como funciona o sistema (registrando os alunos e lançando as notas).
  - No componente ***View***, temos a **interface de visualização do usuário**, ou seja, a parte que o **usuário** (aluno, professor, pai do aluno etc.) utiliza para interagir com o sistema de informação. Neste caso, adotaremos uma interface web para facilitar nosso contexto e trazê-lo para a realidade atual.
  - Por fim, temos o ***Controller*** , responsável por trabalhar as **entradas de dados da View** e as reações (de acordo com as entradas) do modelo de negócio (***Model***) do sistema .

Modelo MVC no cenário proposto

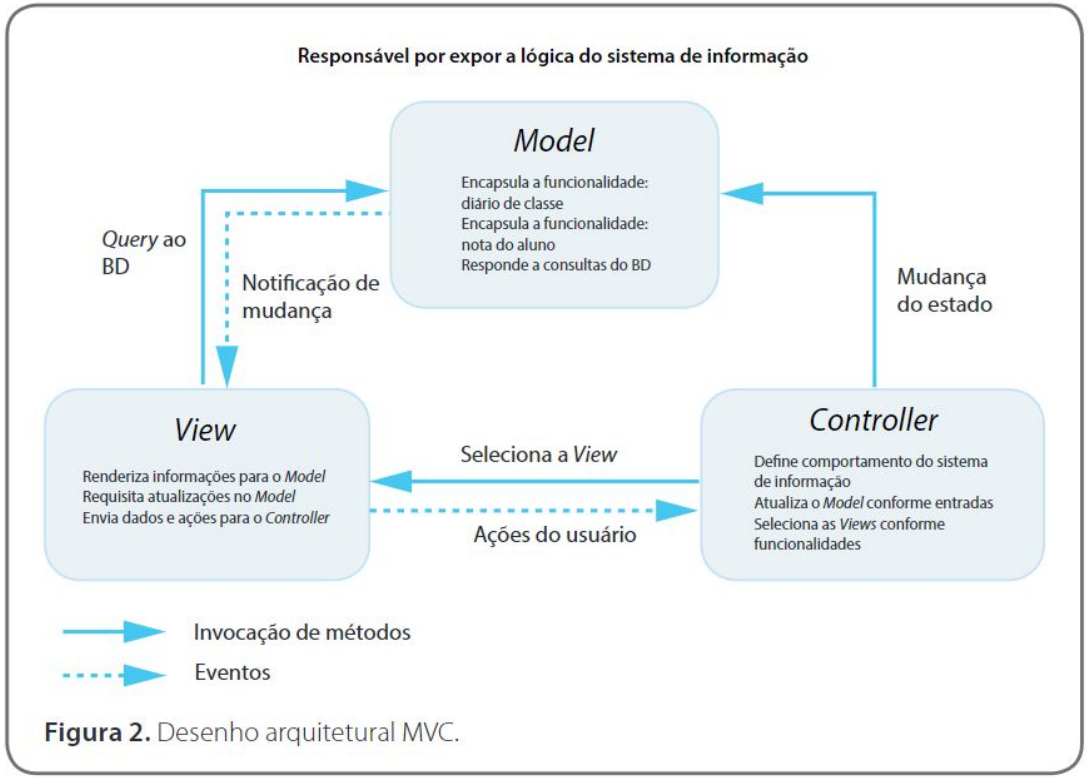


Figura 2. Desenho arquitetural MVC.

Conforme o exemplo proposto de um sistema de informação web para diário de classe e lançamento de notas, temos o desenho arquitetural MVC.

MVC

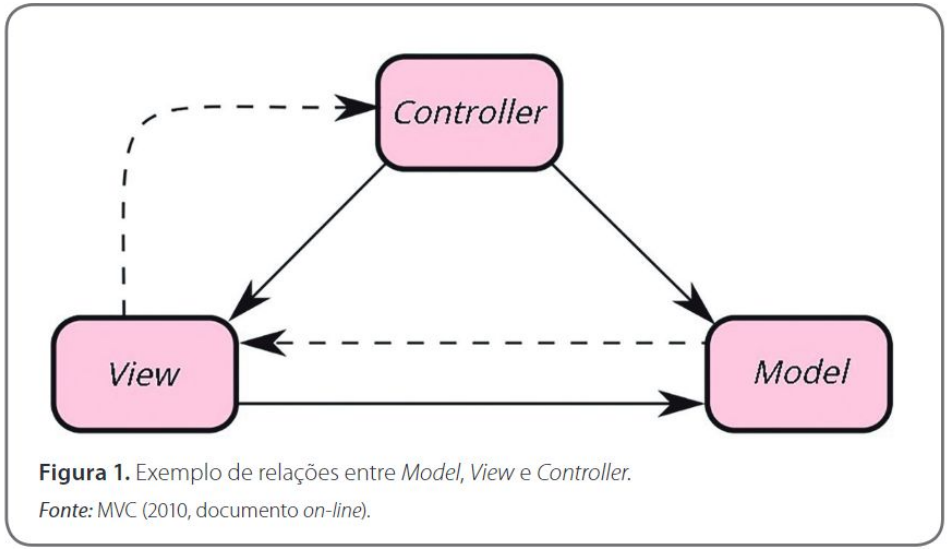


Figura 1. Exemplo de relações entre *Model*, *View* e *Controller*.

Fonte: MVC (2010, documento *on-line*).

O MVC faz a **separação das responsabilidades** entre componentes de uma interface gráfica, sendo estes responsáveis pela manutenção do estado da aplicação, denominado de ***Model***, componentes responsáveis pela **exibição** de parte deste modelo para o usuário, ou seja, a ***View***, e a respectiva coordenação entre **atualizações no modelo e interações com o usuário**, por meio do ***Controller***.

Camada *Model*

UNIVERSIDADE VILA VELHA ESPÍRITO SANTO

- É a camada principal, responsável pela **manutenção do estado da aplicação**.

- Esta camada **promove a representação dos dados** por meio de acesso (leitura/escrita).

- Esta camada **gerencia não somente os dados, mas também os comportamentos fundamentais da aplicação (funcionalidades)**, que são representados pelas **regras de negócio**.

Camada *Model*

UNIVERSIDADE VILA VELHA ESPÍRITO SANTO

- **Encapsula as principais funcionalidades e dado**s do *software*, além de ser **responsável** por **notificar as *Views* e os *Controllers***, quando houver alguma mudança em seu estado.

- Estas notificações permitem que as ***Views*** produzam saídas atualizadas e que os ***Controllers*** alterem o conjunto de comandos disponíveis

Camada *Controller*

UNIVERSIDADE VILA VELHA
ESPÍRITO SANTO

- É responsável por receber todas as requisições do usuário e processá-las.

- Seus **métodos** são chamados de ***actions*** e são **responsáveis** por cada ***View***.

- As ***actions*** do ***Controller*** **controlam** qual ***Model*** será usado e qual ***View*** dele será mostrada ao usuário. Ou seja, é possível e**nviar comandos para o *Model*** pelo ***Controller***, **atualizando seu respectivo estado**.

- O ***Controller*** também pode **enviar comandos** para a ***View*** para alterar a apresentação da ***View*** da camada ***Model***.

Camada *Controller*

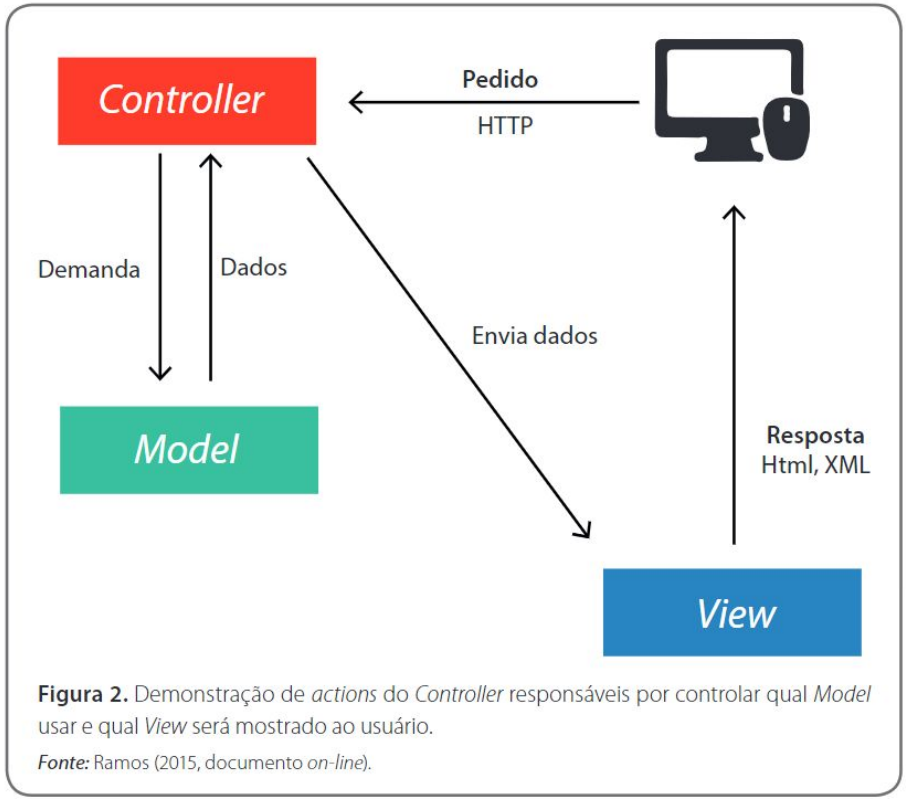


Figura 2. Demonstração de *actions* do *Controller* responsáveis por controlar qual *Model* usar e qual *View* será mostrado ao usuário.

Fonte: Ramos (2015, documento *on-line*).

A Camada de Controller:

- Atende aos comandos do usuário
- Seleciona o *Model* e a *View* para interagir com o *Model*
- O usuário interage com os *Controllers* pelas *Views*
- Cada *View* interpreta eventos e entradas enviadas
- Mapeia ações do usuário em comandos para o *Model* ou *View*

O Controller:

- Define o comportamento do *software*
- Interpreta ações do usuário e mapeia chamadas ao modelo
- Responsável pelo controle do fluxo da aplicação
- Orquestra as manipulações do Model e interações com o usuário

Em resumo, o **Controller é uma camada intermediária entre a View e o Model**.
Geralmente, **há um Controller para cada View, mas pode haver vários Controllers para uma mesma View (FACADE)**.

Fluxo entre cada parte do MVC

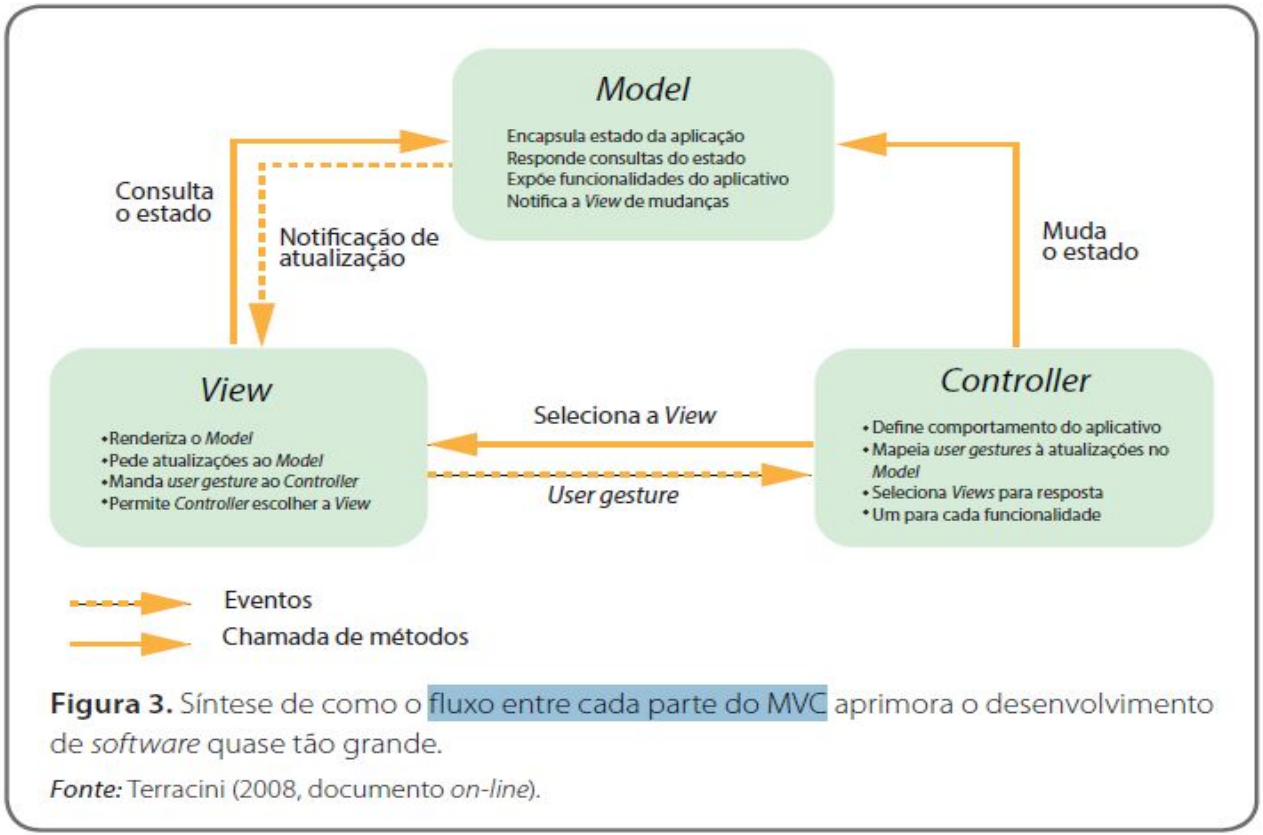


Figura 3. Síntese de como o fluxo entre cada parte do MVC aprimora o desenvolvimento de *software* quase tão grande.

Fonte: Terracini (2008, documento *on-line*).

A Figura ilustra um exemplo de interação entre *Model*, *Controller* e *View*.

Camada *View*

- Responsável pela interação com o usuário
- Representação visual da lógica descrita no modelo
- Permite apresentar dados de diversas formas
- Recebe instruções do *Controller* ou notifica e recebe informações do *Model*

Diferença entre arquitetura em três camadas e MVC

- Na arquitetura em três camadas, a comunicação passa obrigatoriamente pela camada intermediária (*Controller*)
- Na arquitetura MVC, a camada *View* se comunica diretamente com o *Model* e o *Controller*

Interações entre as camadas do MVC

UNIVERSIDADE VILA VELHA ESPÍRITO SANTO

- *View* interage com o *Controller* por meio de eventos
- *Controller* notifica o *Model* sobre mudanças de estado
- *Model* modifica seu estado e notifica *Views* e *Controllers*

Exemplo de interação no modelo MVC

UNIVERSIDADE VILA VELHA ESPÍRITO SANTO

- *View* pode fazer um pedido diretamente para a camada *Model*
- O modelo MVC é aplicado em diferentes plataformas atualmente

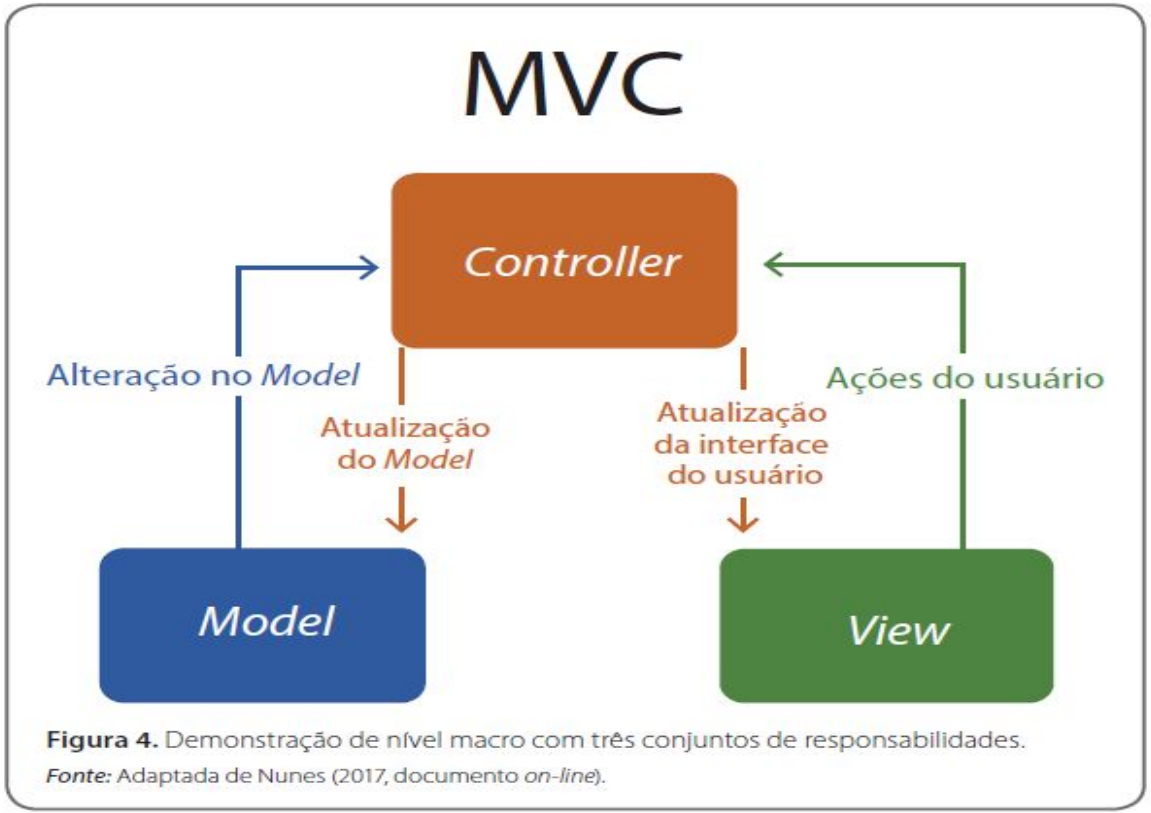


Figura 4. Demonstração de nível macro com três conjuntos de responsabilidades.

Fonte: Adaptada de Nunes (2017, documento *on-line*).

Na Figura, podemos ver as possíveis interações da arquitetura MVC aplicada à plataforma mobile Android.

UNIVERSIDADE VILA VELHA
ESPÍRITO SANTO

Dúvidas?

vinicius.motta@uvv.br