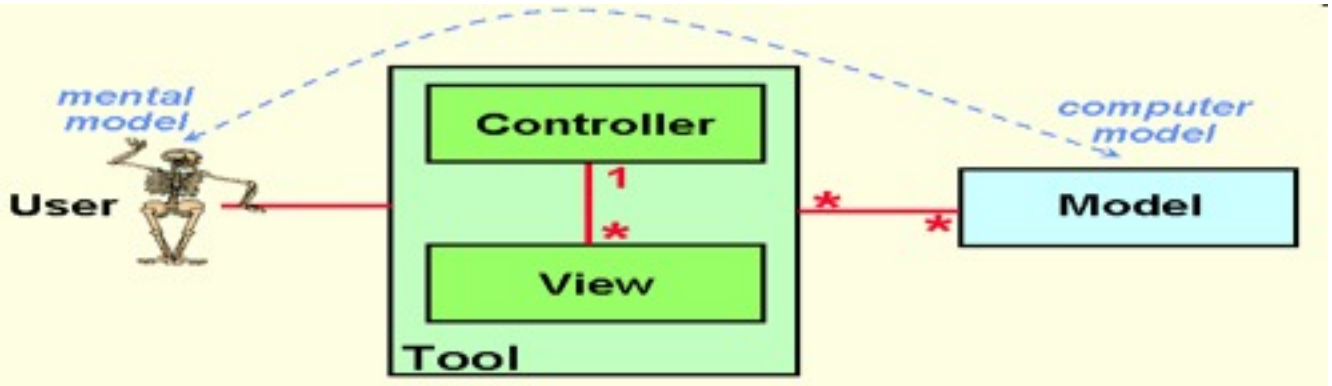
Model View Controller

CSPP 51023

Aiman Ashraf

Background

- Model View Controller (MVC) first originated in 1979 for the language Smalltalk. It was first described by Trygve Reenskaug in a paper called “Applications Programming in Smalltalk-80: How to use Model–View–Controller”
- As per him “*The essential purpose of MVC is to bridge the gap between the human user's mental model and the digital model that exists in the computer*”



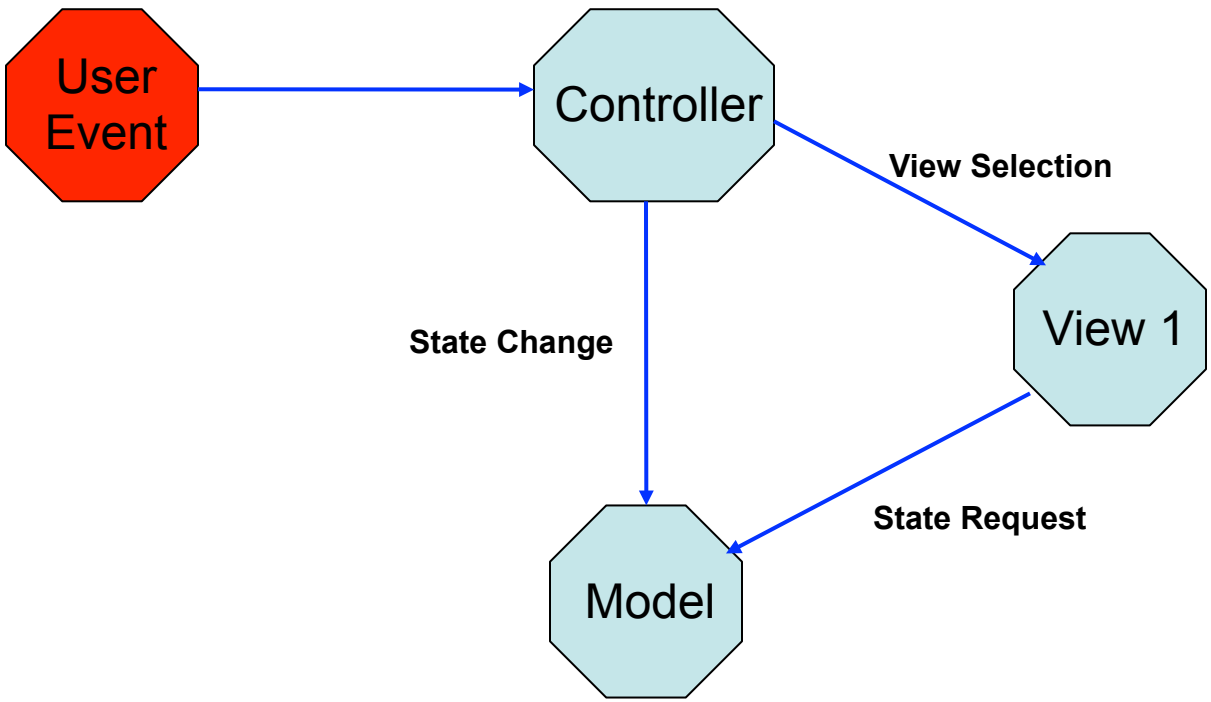
Motivations

- Separation of user interface and business logic.
  - Maintenance of code which contains both user interface and business logic is not fun.
  - User interface development is a different skill than business logic development.
  - There is usually the case of the same data being displayed in different ways.

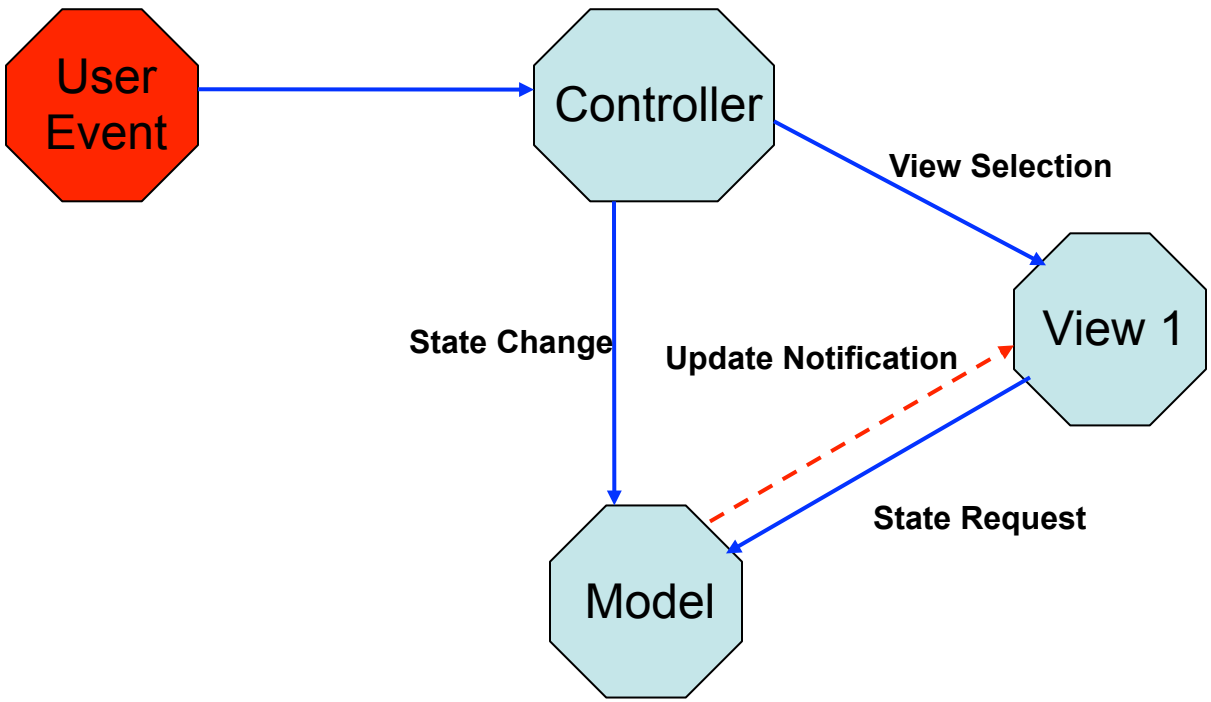
So, what is MVC?

- The Model represents the data in the system. It responds to request for an information about its state. It also responds to instructions to change state.
- The View renders the model into a form suitable for interaction, typically a user interface element. There can be multiple views for the same model.
- The Controller receives user input and chooses the model and view to be used. So it is just like a conductor of the orchestra.

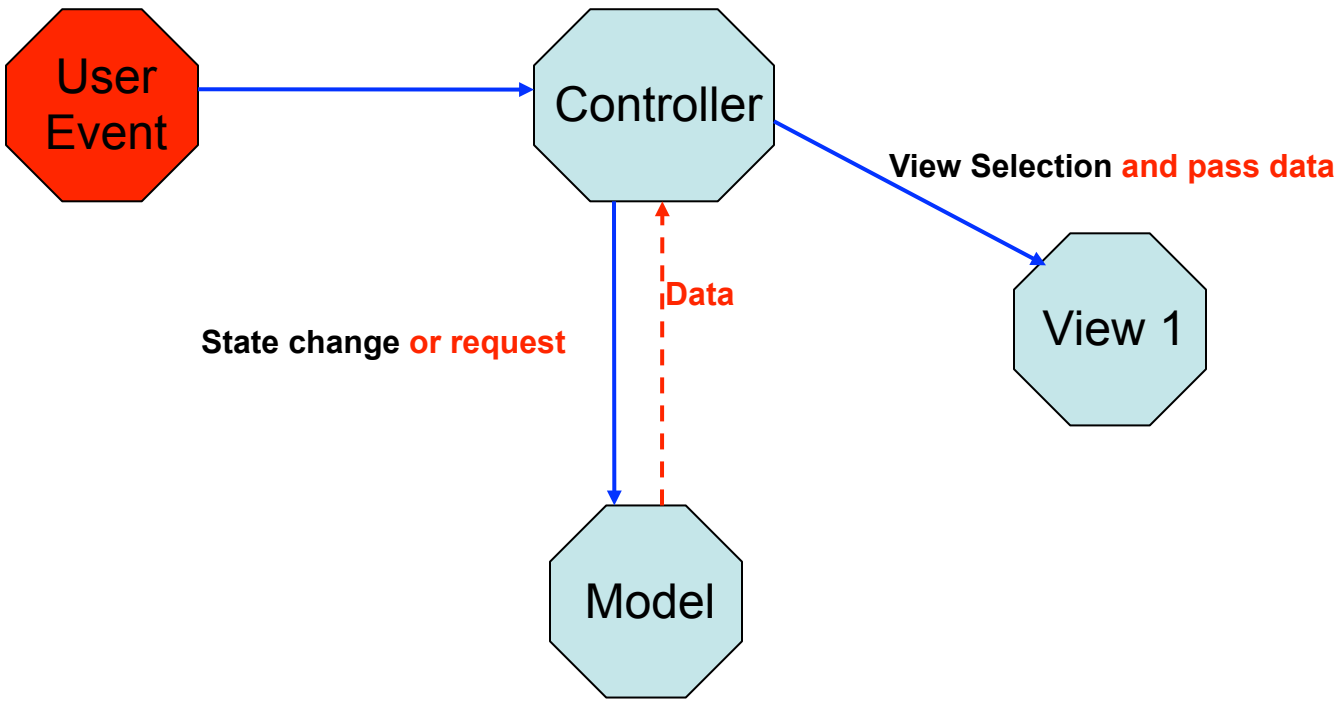
Classic MVC


User Event
Controller
View Selection
View 1
State Change
State Request
Model

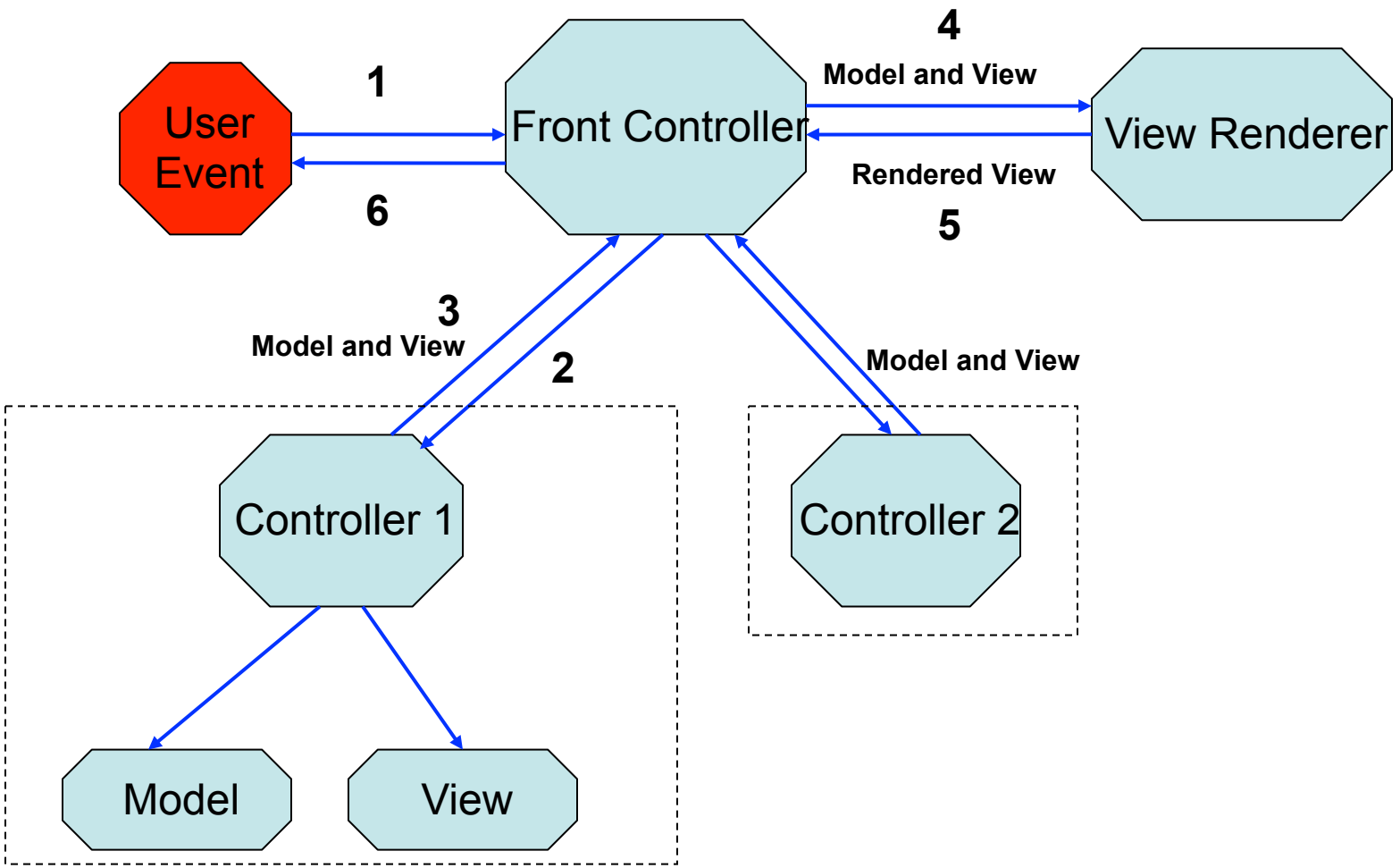
Other MVC forms (1)



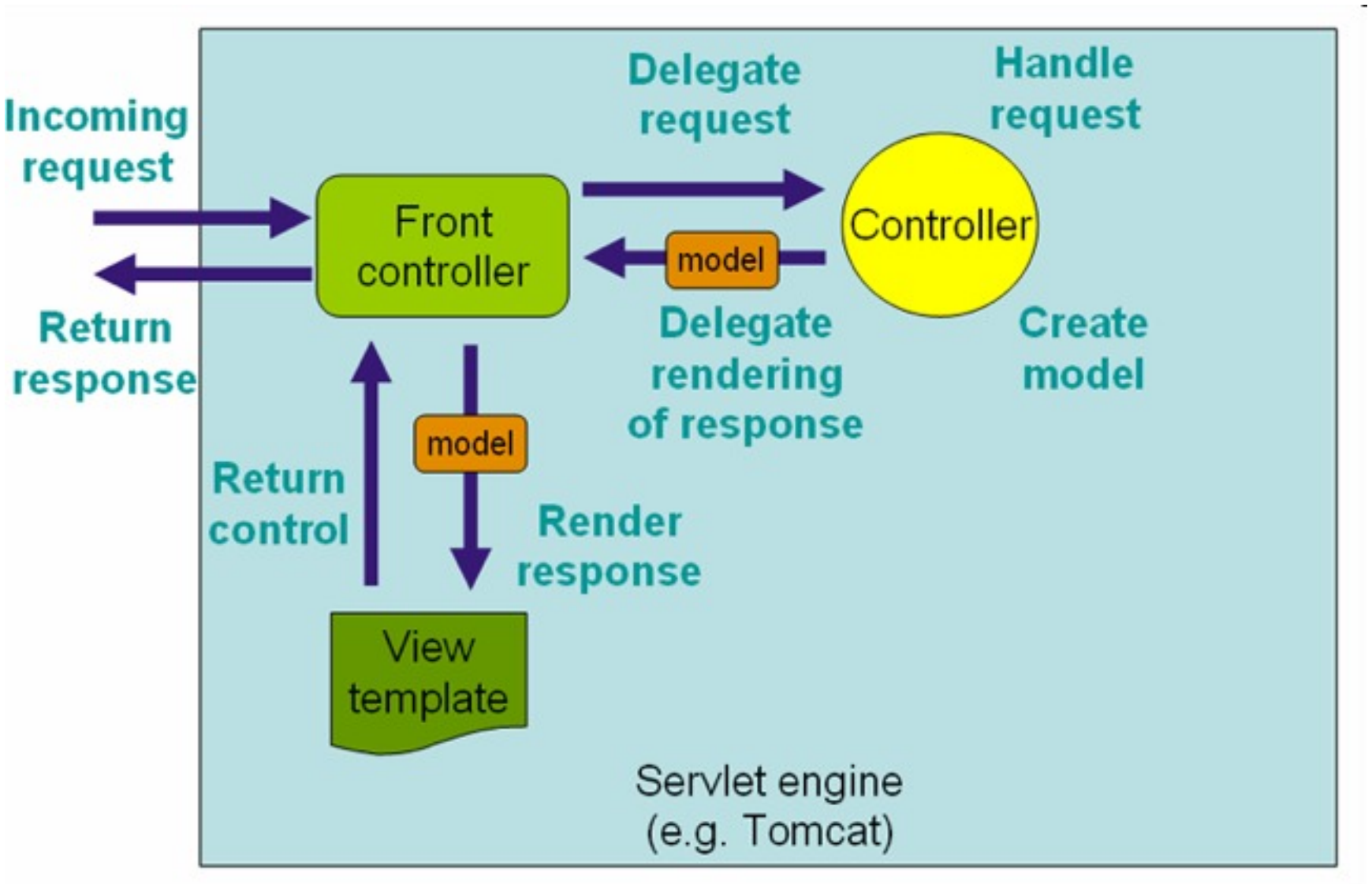
Other MVC forms (2)



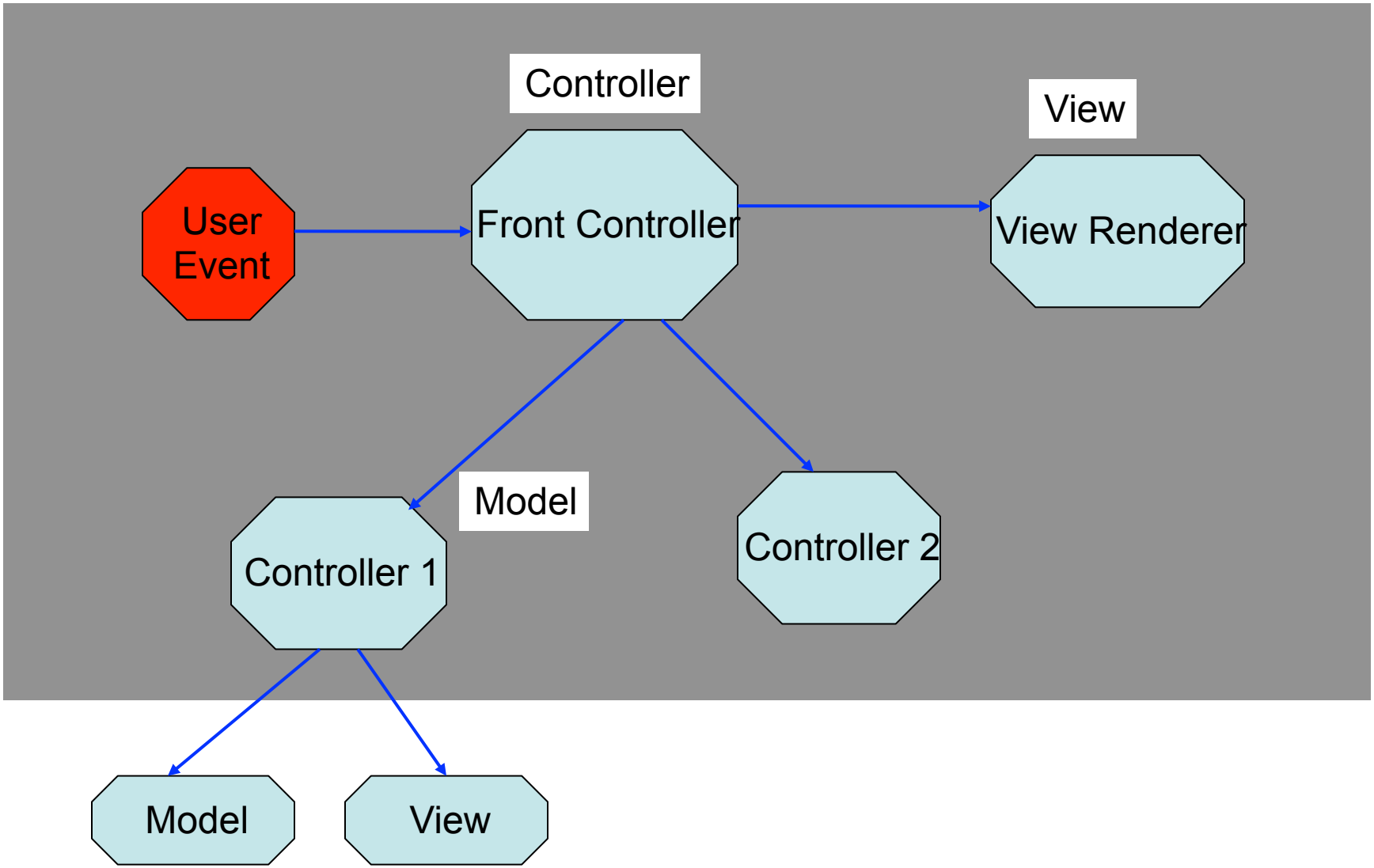
Front Controller



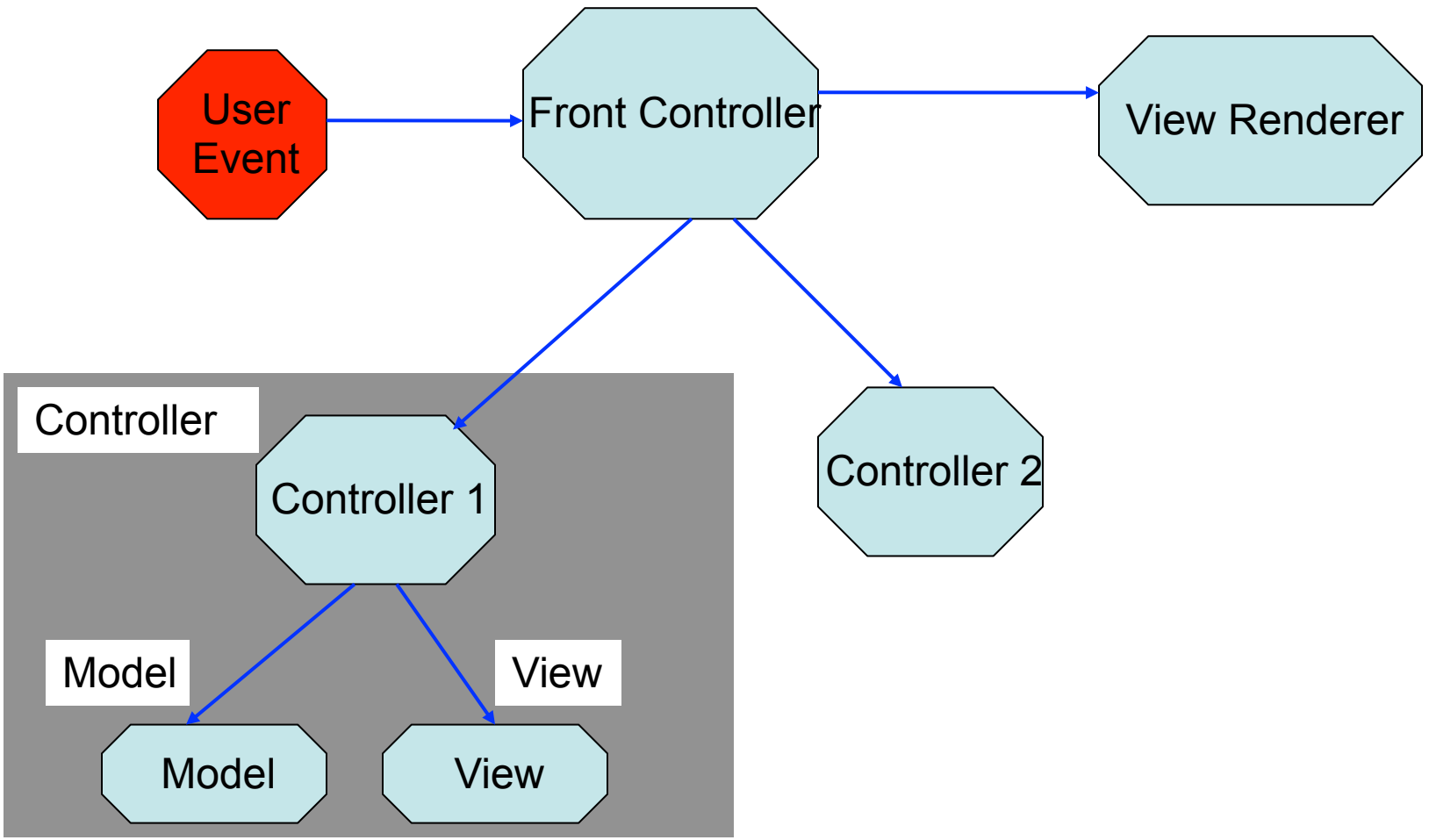
Spring MVC


Incoming request
Return response
Front controller
Delegate request
Handle request
Controller
model
Delegate rendering of response
Create model
model
Return control
Render response
View template
Servlet engine (e.g. Tomcat)

MVC Perspective 1



MVC Perspective 2



Spring MVC

```
public interface Controller {
    /*Process the request and return a ModelAndView object which the
    DispatcherServlet will render.*/
    ModelAndView handleRequest( HttpServletRequest request,
    HttpServletResponse response);
}
public class ModelAndView {
    Model model;
    View view;
}
public interface View {
     public void render(java.util.Map model, HttpServletRequest request,
         HttpServletResponse response)
}
public class Model {
```

To wrap up

- MVC is an architecture rather than a design pattern.
- MVC is an intuitive and natural way of solving software design problems.
- MVC is everywhere.

Thanks