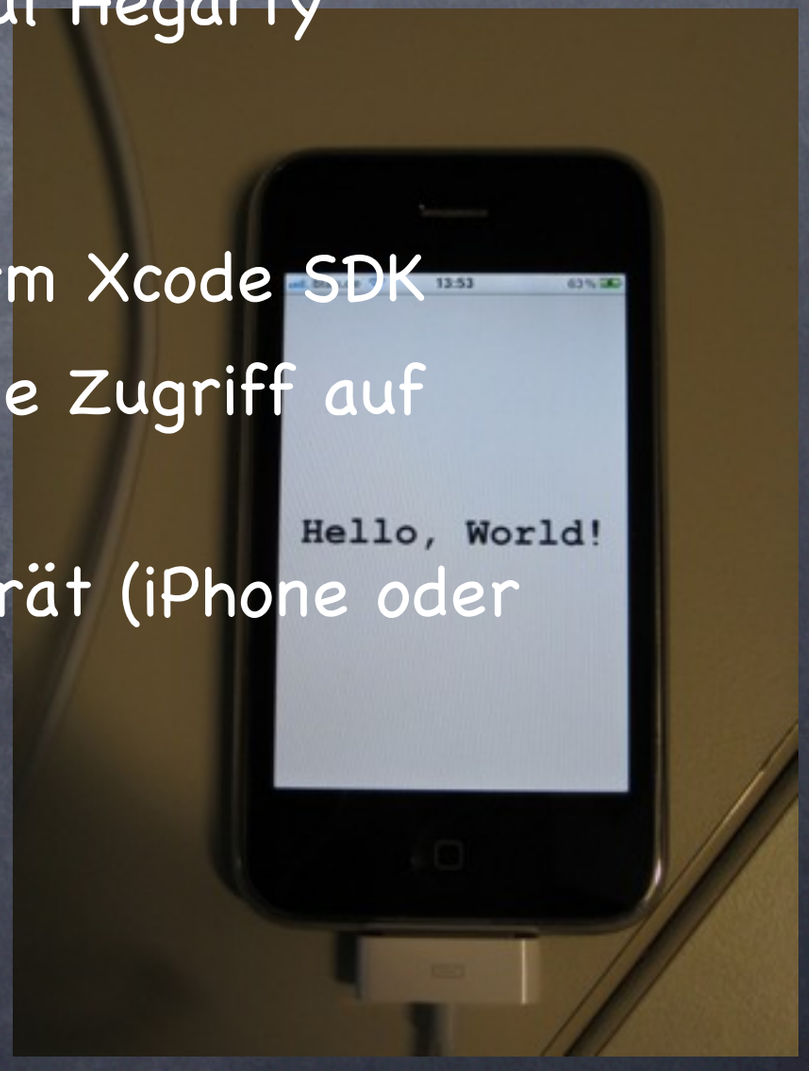
iOS Programmierung

Teil der Vorlesung Erdbebenfrühwarnsysteme:
Objective-C erlernen
mit xcode arbeiten
Applikationen für iPhone und iPad bauen

Einführung

- Grundlage: [iTunes U] verfügbarer Kurs der Stanford University (Fall 2010):
  - http://cs193p.stanford.edu
  - 17 Lektionen mit vielen Demos von Paul Hegarty
- Voraussetzung:
  - Mac (Intel based) als Entwicklungsplattform Xcode SDK
  - xcode: Apps im Simulator möglich (ohne Zugriff auf spezielle Hardware)
  - xcode: direkte Ausführung auf Zielgerät (iPhone oder iPad)
  - OOP Begriffe und Konzepte



Einführung

- Weitere Ressourcen:
  - Apple on-line documentation http://developer.apple.com
  - Big Nerd Ranch: http://bignerdranch.com
  - http://stackoverflow.com
  - http://www.iphonedevsdk.com

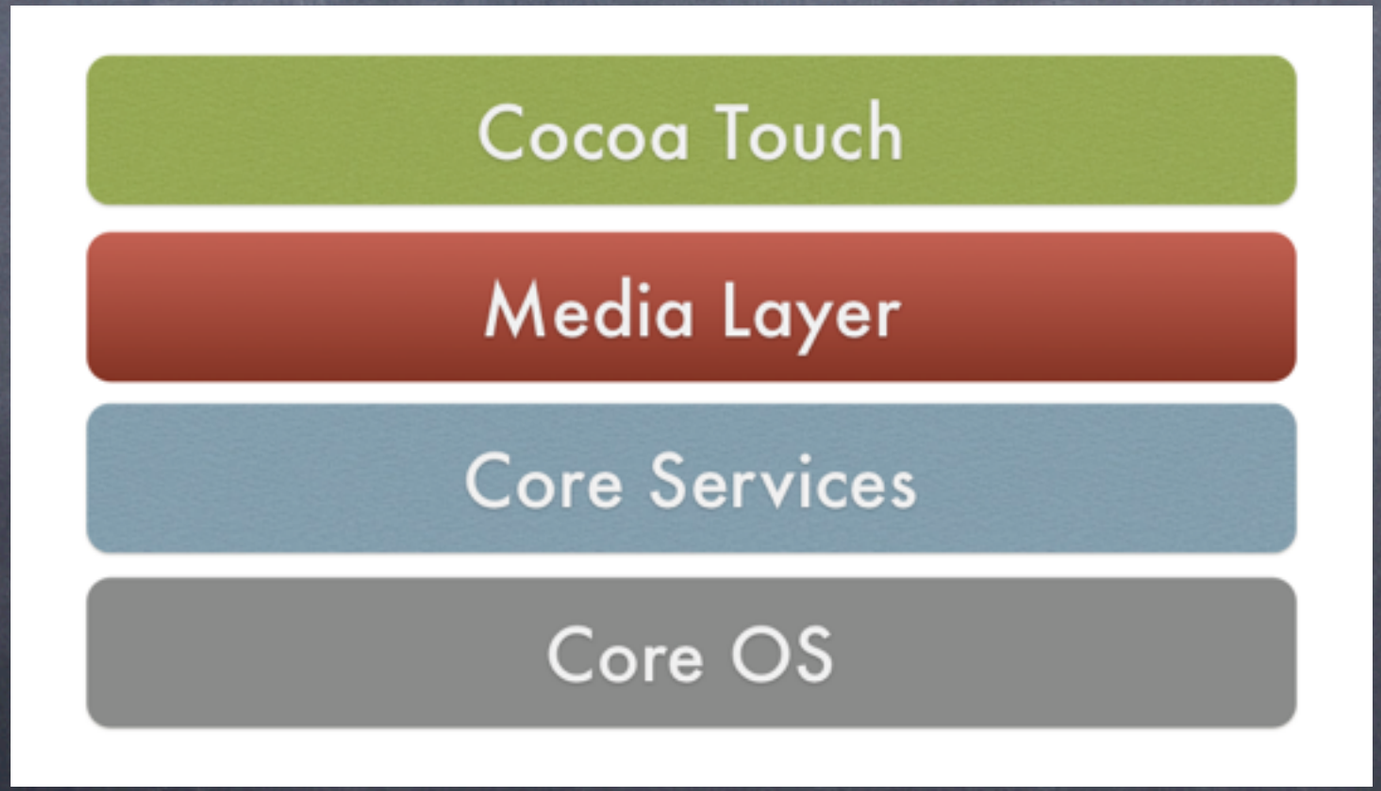
Plattform Komponenten

- Design Strategien
- Werkzeuge
- Sprache
- Frameworks

```
[display setTextColor:[UIColor blackColor]];
```

iOS Aufbau

- das Betriebssystem von iPhone/iPad
  - aktuelle Version 4.3.2
- Schichtenaufbau



iOS Aufbau

Core OS

- OSX Kernel
- Mach 3.0
- BSD Sockets
- Security
- Power Management
- Keychain Access
- Certificates
- File System
- Bonjour

iOS Aufbau

Core Services

- Collections
- AddressBook
- Networking
- File Access
- SQLite
- Core Location
- NetServices
- Threading
- Preferences
- URL Utilities

iOS Aufbau

Media

Core Audio

OpenAL

Audio Mixing

Audio Recording

Video Playback

JPEG, PNG, TIFF

PDF

Quartz (2D)

Core Animation

OpenGL ES

iOS Aufbau

Cocoa Touch

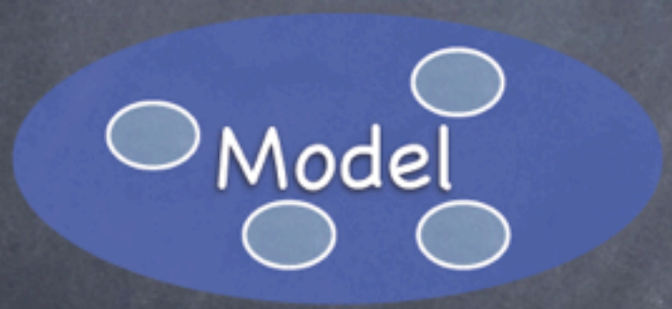
- Multi-Touch
- Core Motion
- View Hierarchy
- Localization
- Controls
- Alerts
- Web View
- Map Kit
- Image Picker
- Camera

MVC Model - View - Controller

- Programme in 3 Camps aufteilen!
  - Abhängigkeiten minimieren
  - Komponenten wiederverwenden
  - Skalierung ermöglichen
  - Apps bauen, die verstehbar, wartbar und gut strukturiert sind

MVC Model - View - Controller

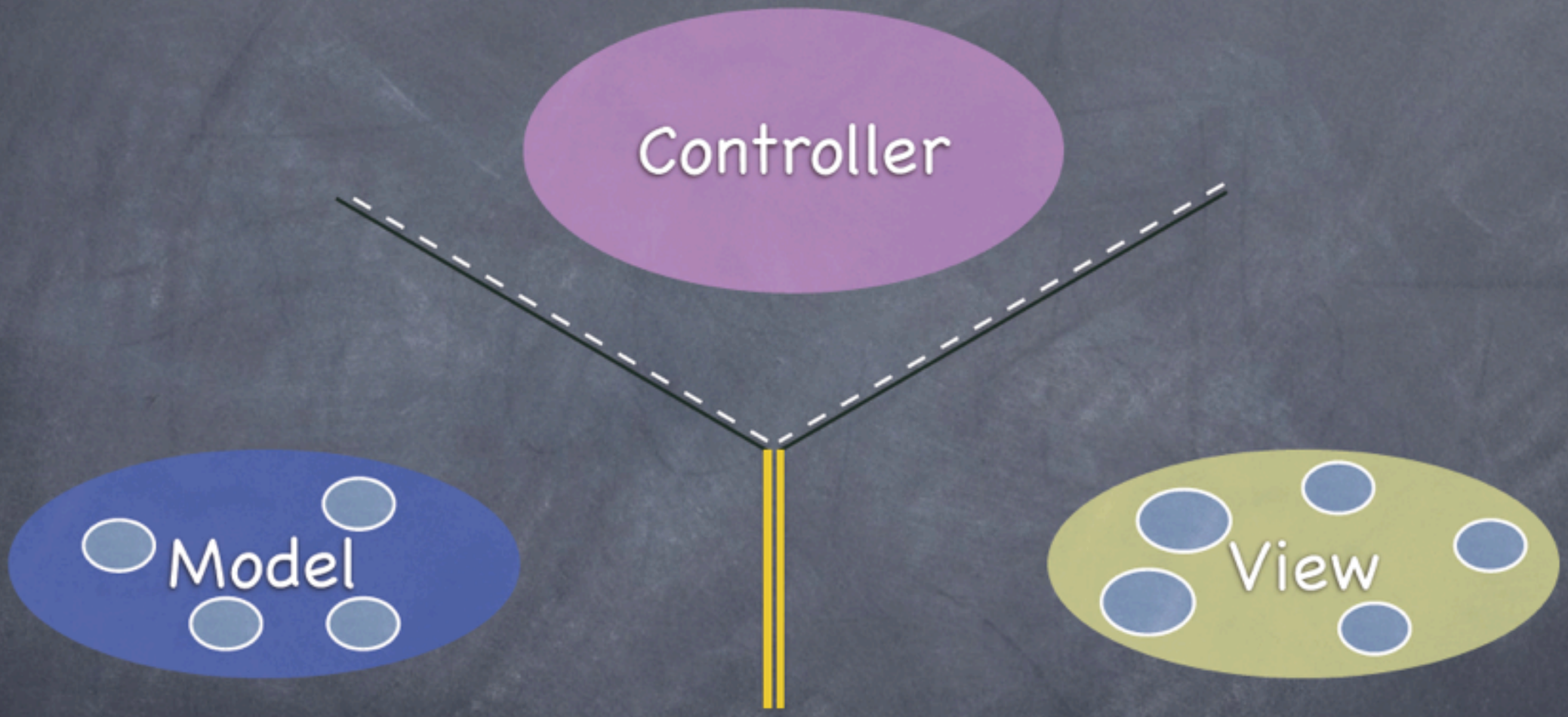
C: wie die Anwendung dem Anwender
präsentiert wird (UI Logik) plattformabhängig!



M: die Anwendung selbst
(nicht wie sie aussieht)

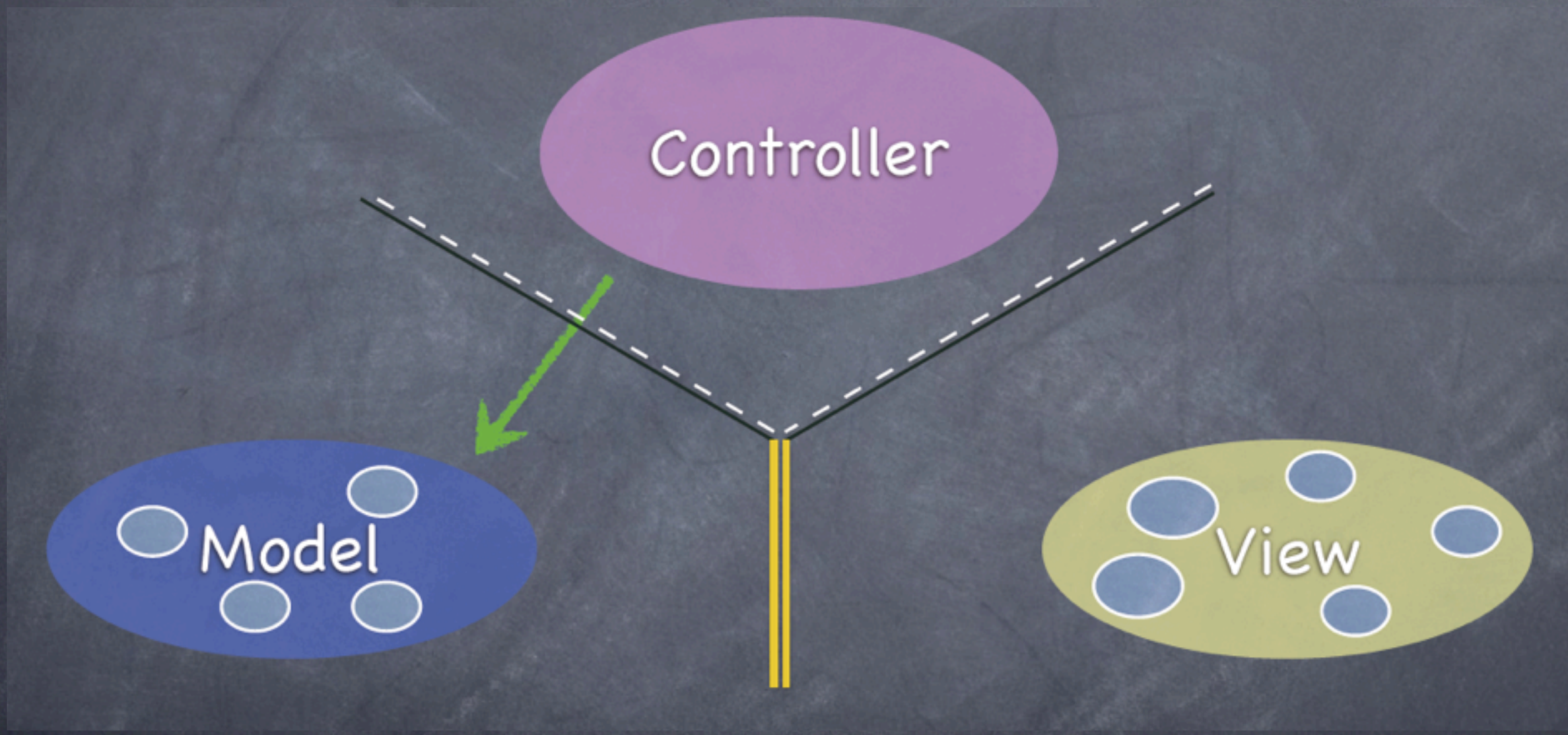
V: die generischen Komponeneten der
Darstellung
(gesteuert vom C)

MVC Model - View - Controller



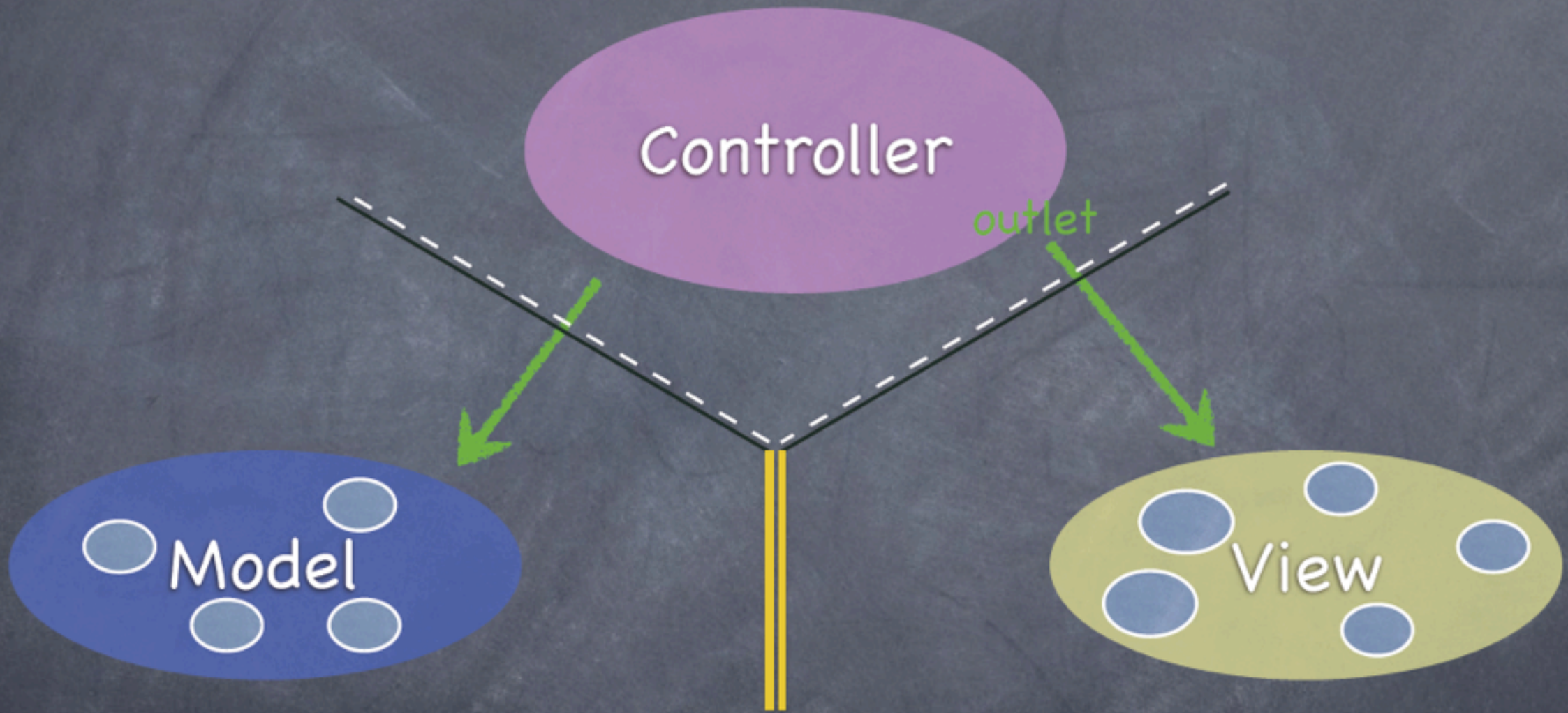
Kommunikation überschaubar halten
Wiederverwendbarkeit ermöglichen

MVC Model - View - Controller



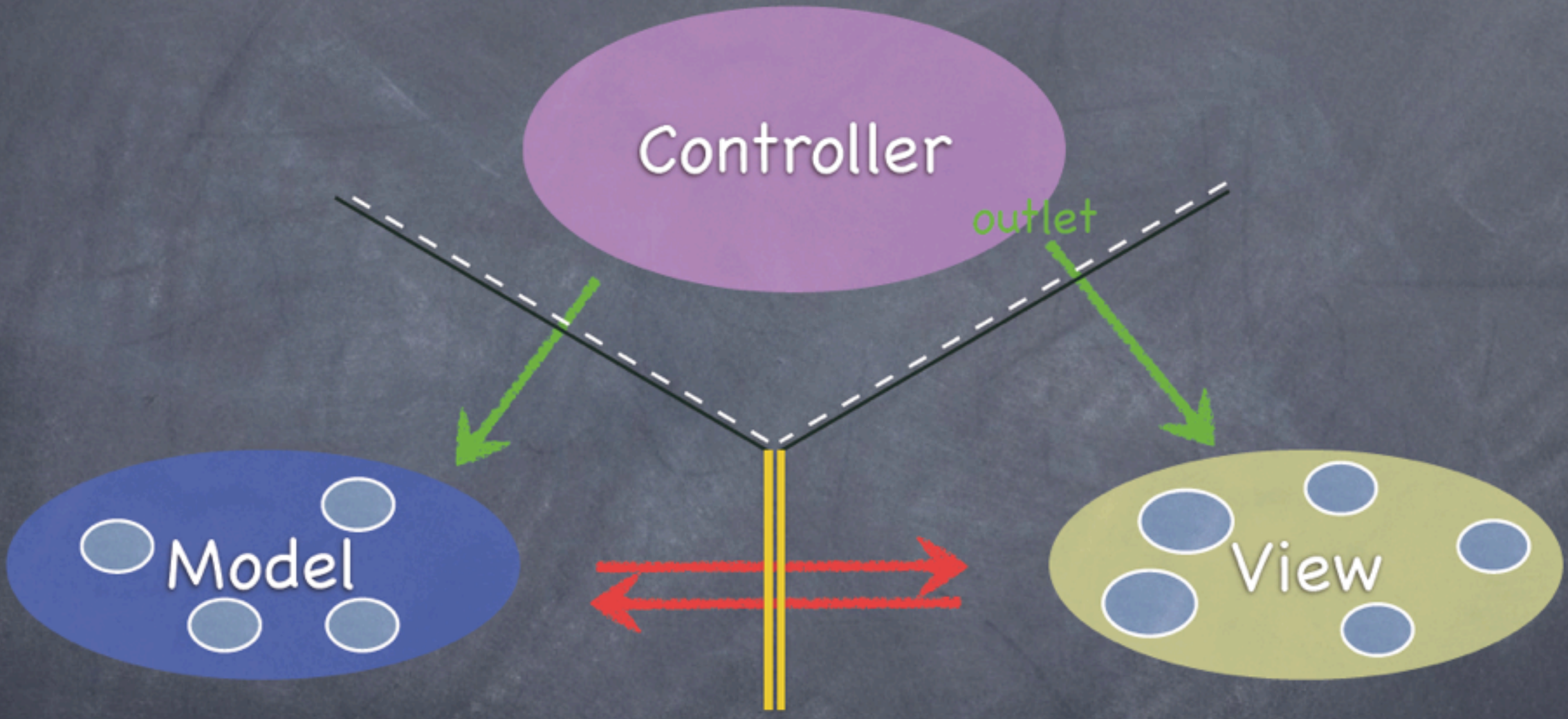
C kann Nachrichten direkt an M senden
(C importiert M's Headerfile)

MVC Model - View - Controller



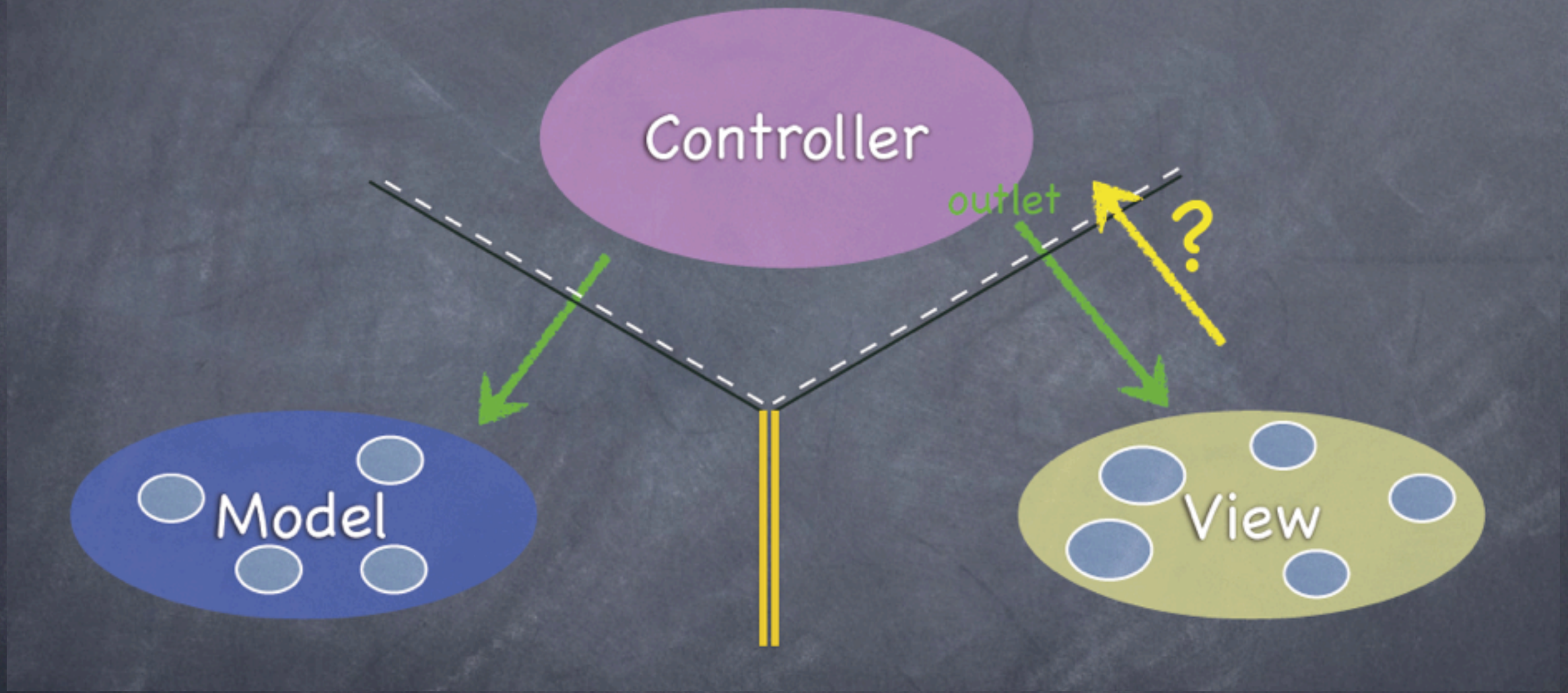
C kann Nachrichten direkt an V senden
(C hält Zeiger auf V-Objekte)

MVC Model - View - Controller



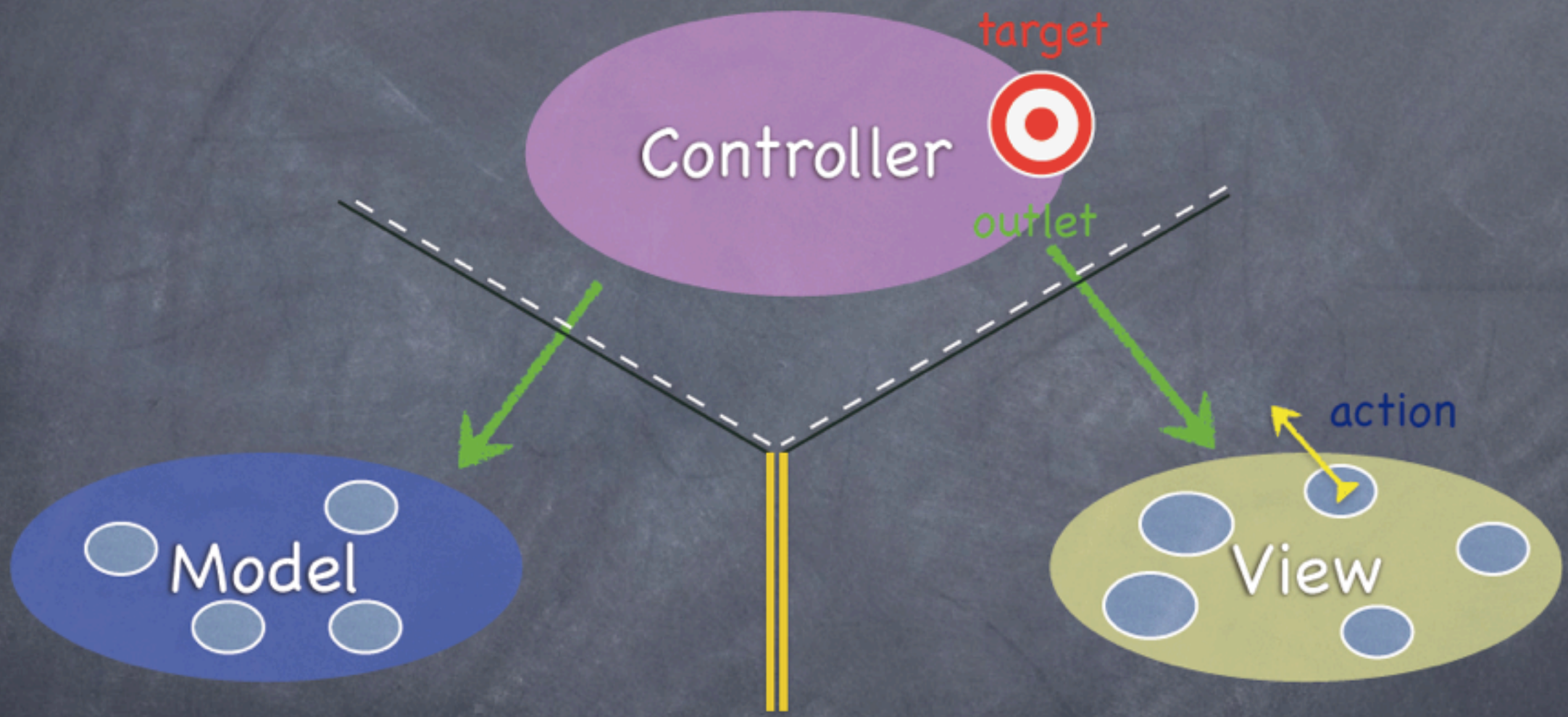
M und V sprechen NIEMALS miteinander

MVC Model - View - Controller



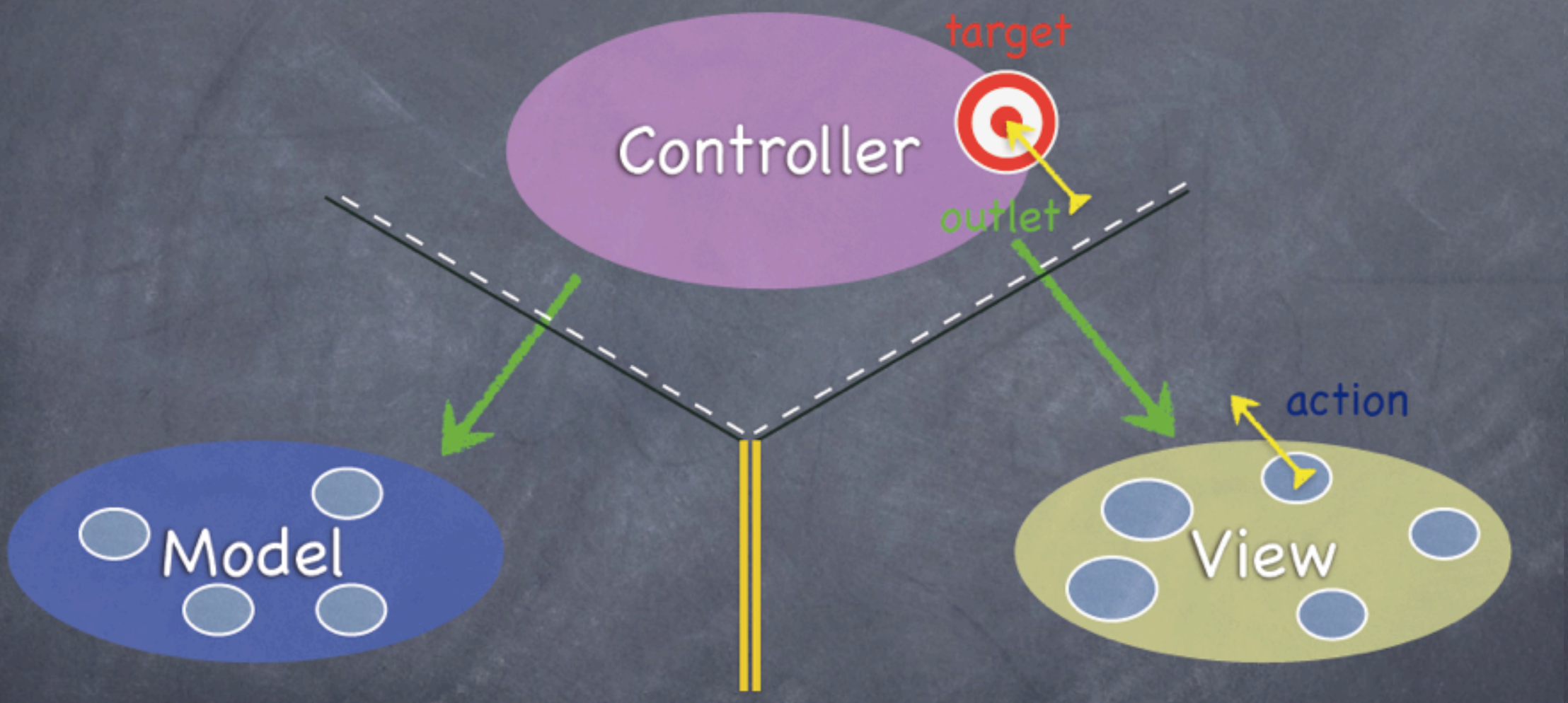
Kann V Informationen an C übermitteln?

MVC Model - View - Controller



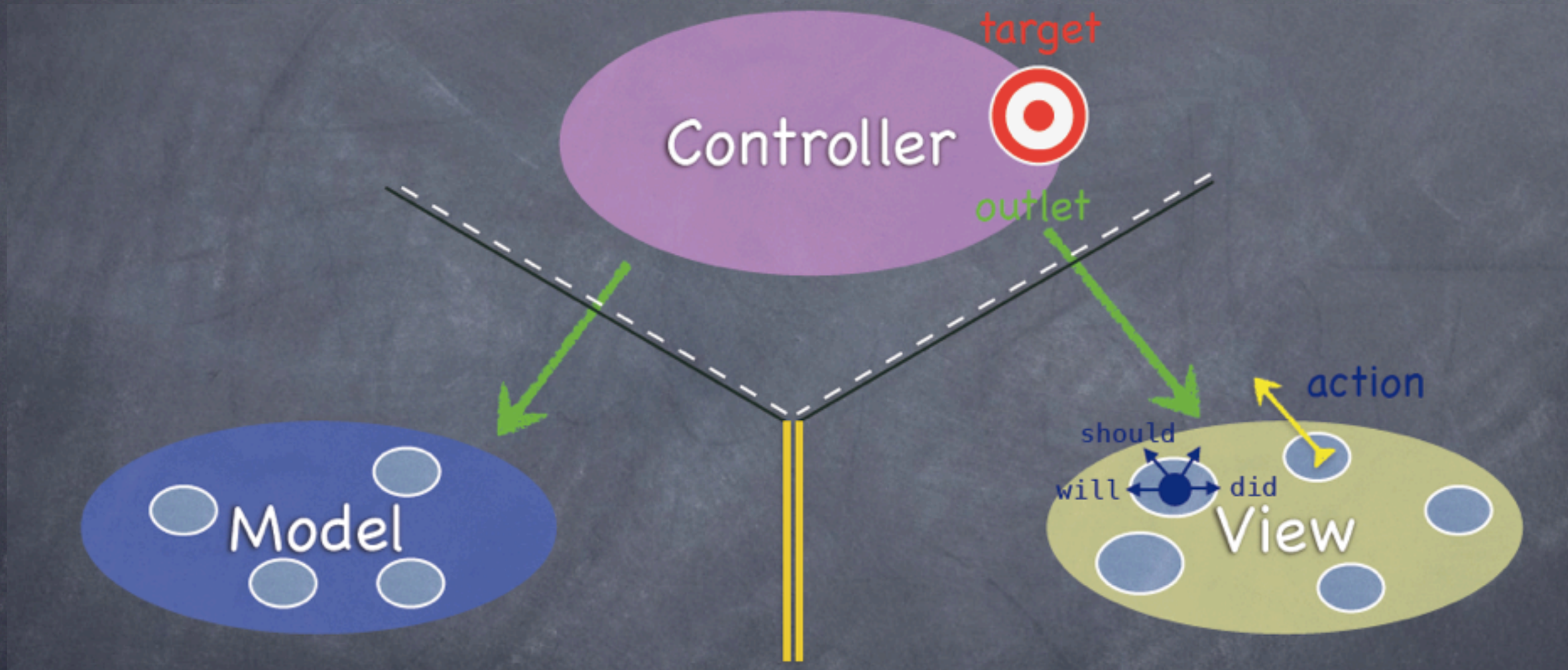
C kann ein Ziel anbieten, auf welches V ‚schießen' kann (ohne C zu kennen!), indem V eine Action in C bekannt gemacht wird

MVC Model - View - Controller



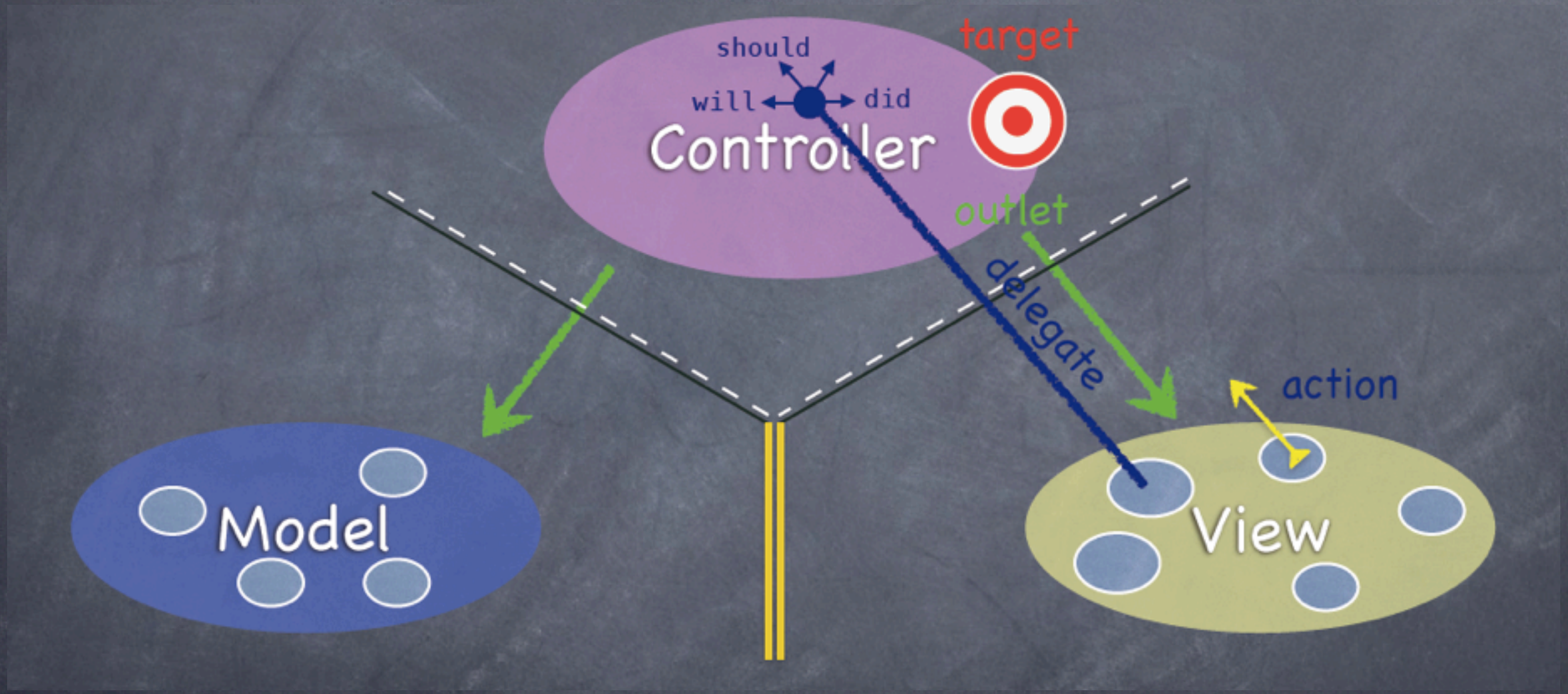
bei Ereignissen in V wird die Action gerufen

MVC Model - View - Controller



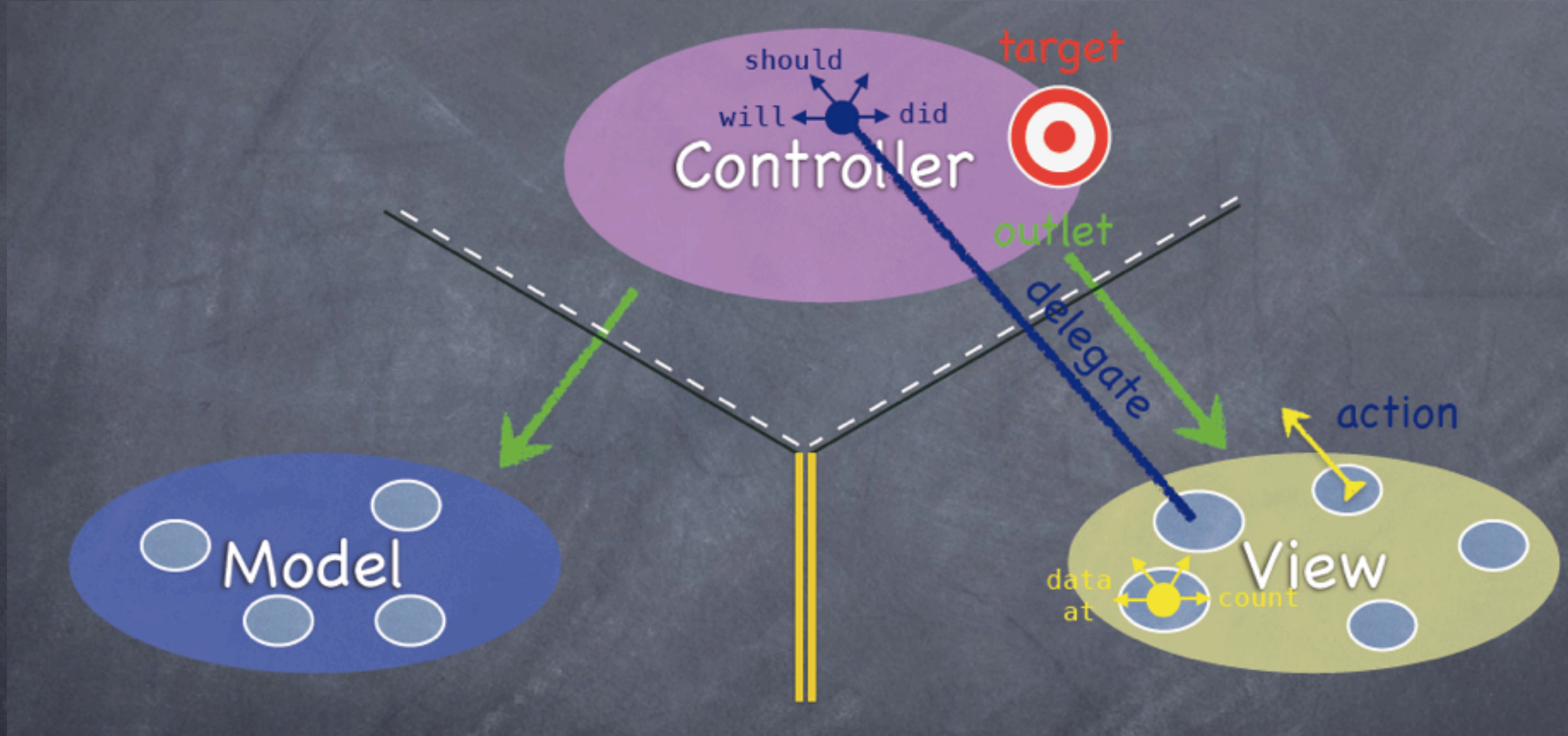
manchmal muss V mit C synchronisiert werden,
wenn Dinge geschehen sollen/sind

MVC Model - View - Controller



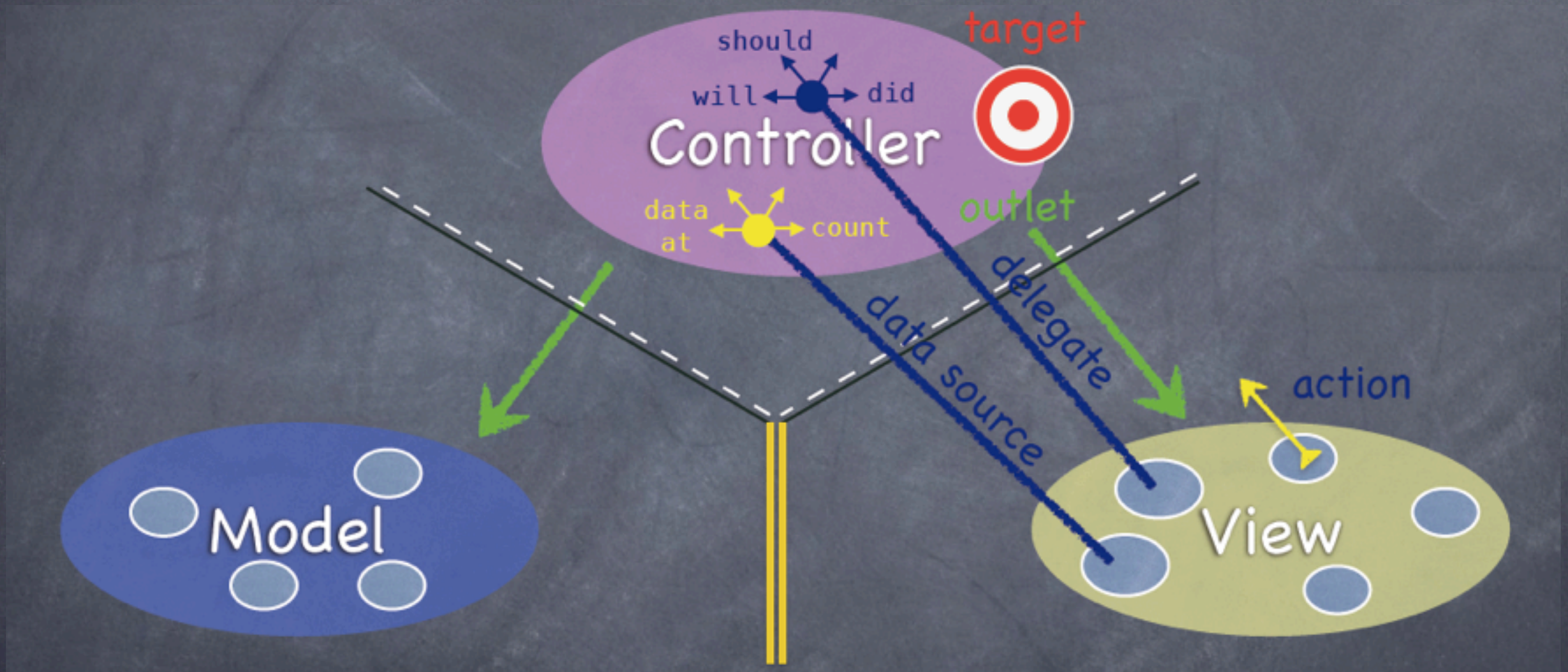
dazu erklärt sich ein C zum Delegate von V,
delegates erfüllen vordefinierte Protokolle

MVC Model - View - Controller



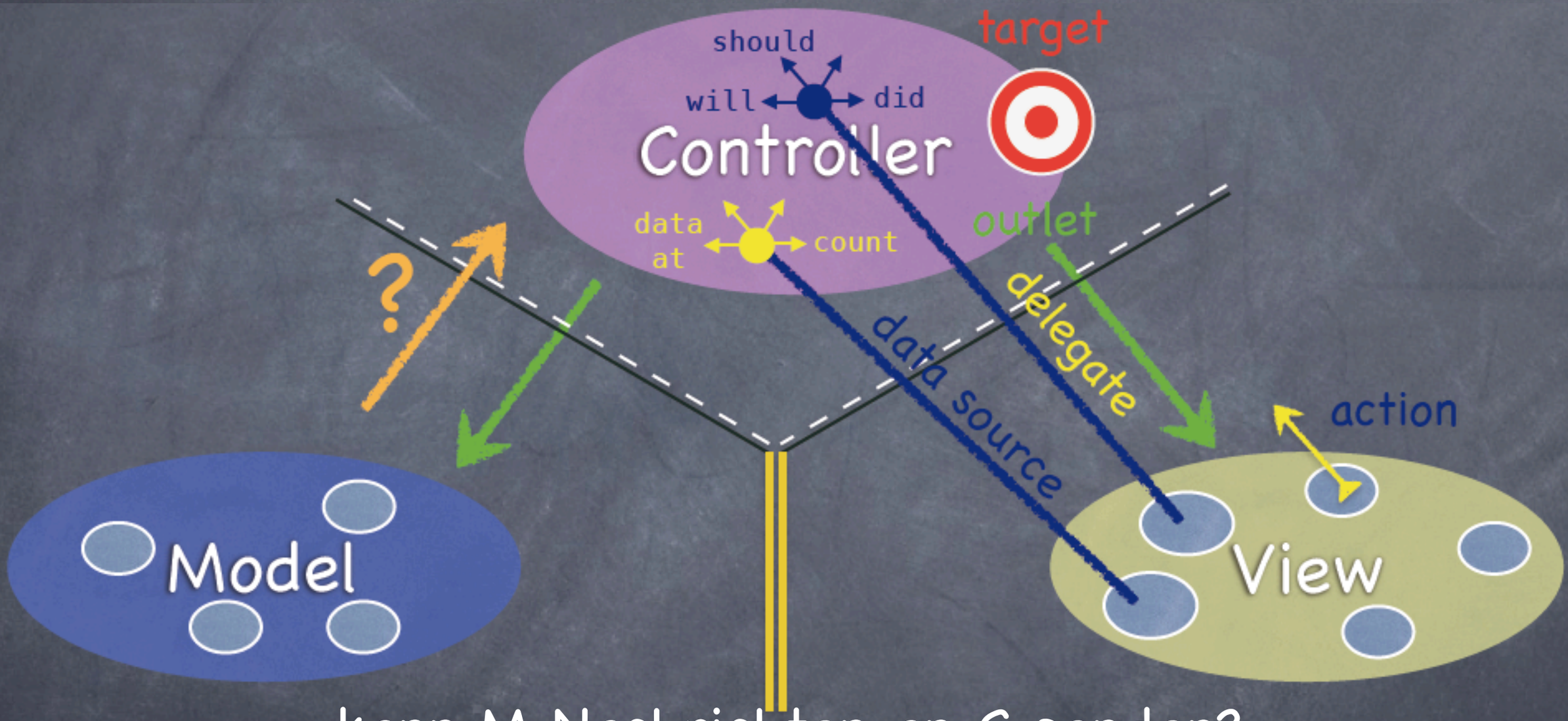
V's besitzen die Daten NICHT, die sie anzeigen

MVC Model - View - Controller



C (nicht M) fungiert als Data Source von V
(ebenfalls nach vordefinierten Protokollen)

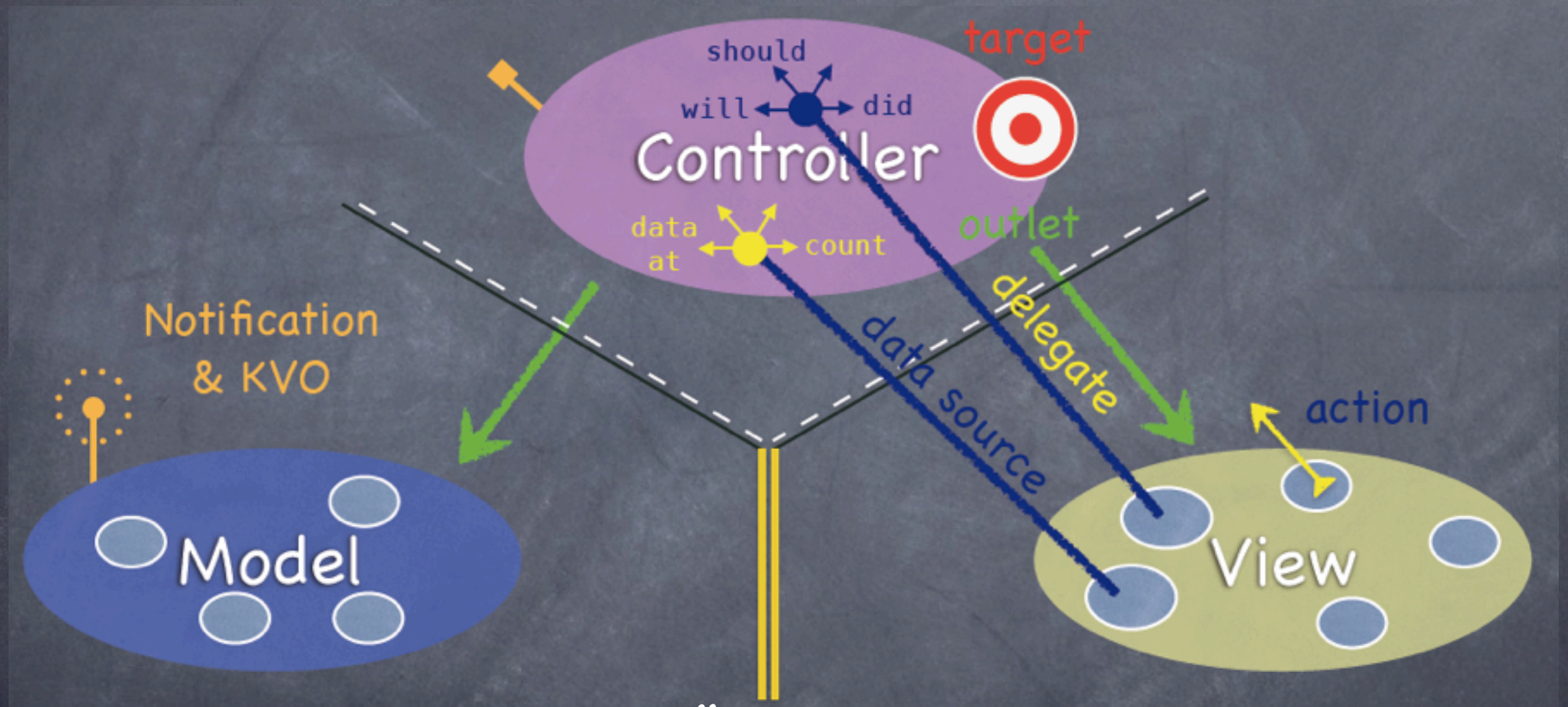
MVC Model - View - Controller



kann M Nachrichten an C senden?

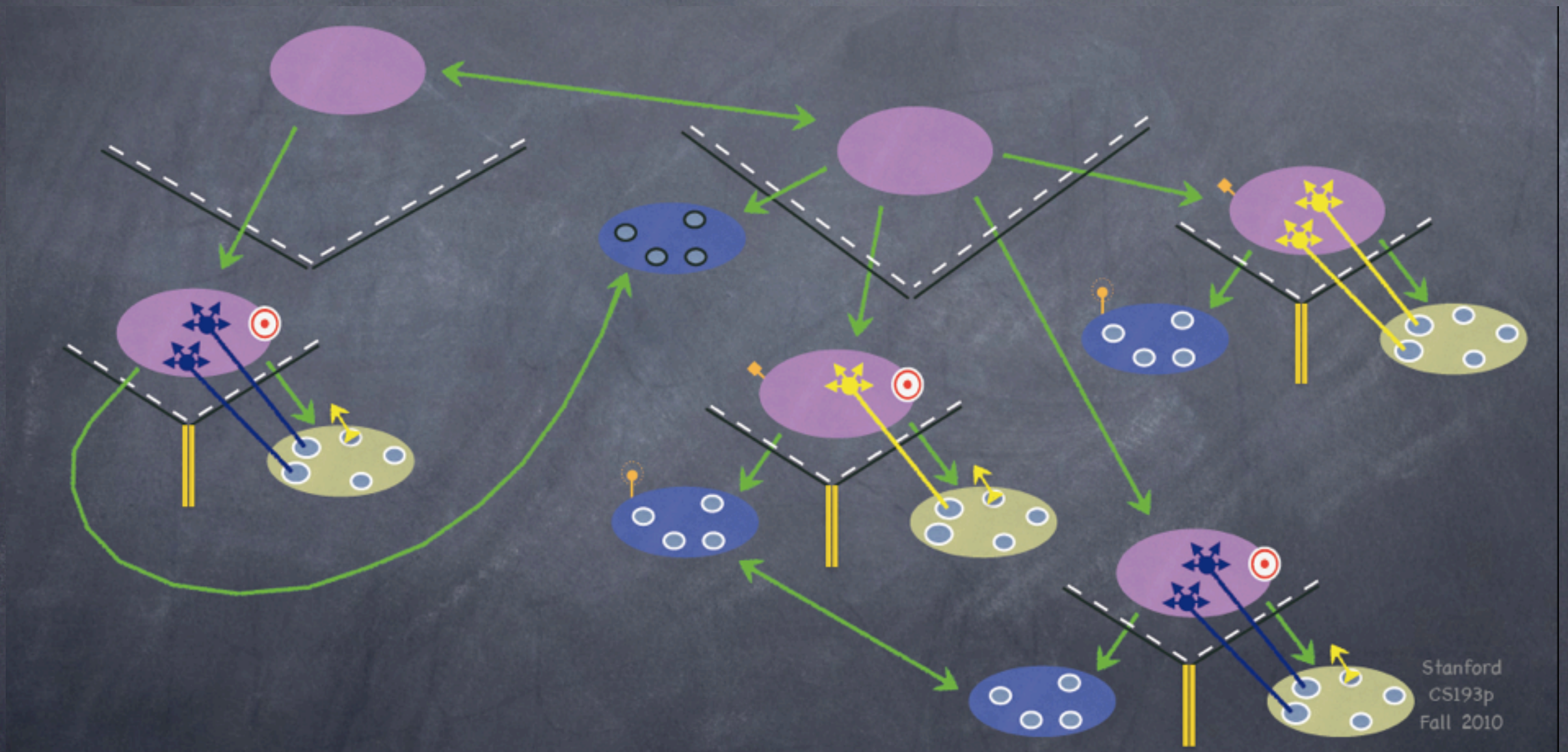
NEIN: M muss UI-unabhängig sein

MVC Model - View - Controller



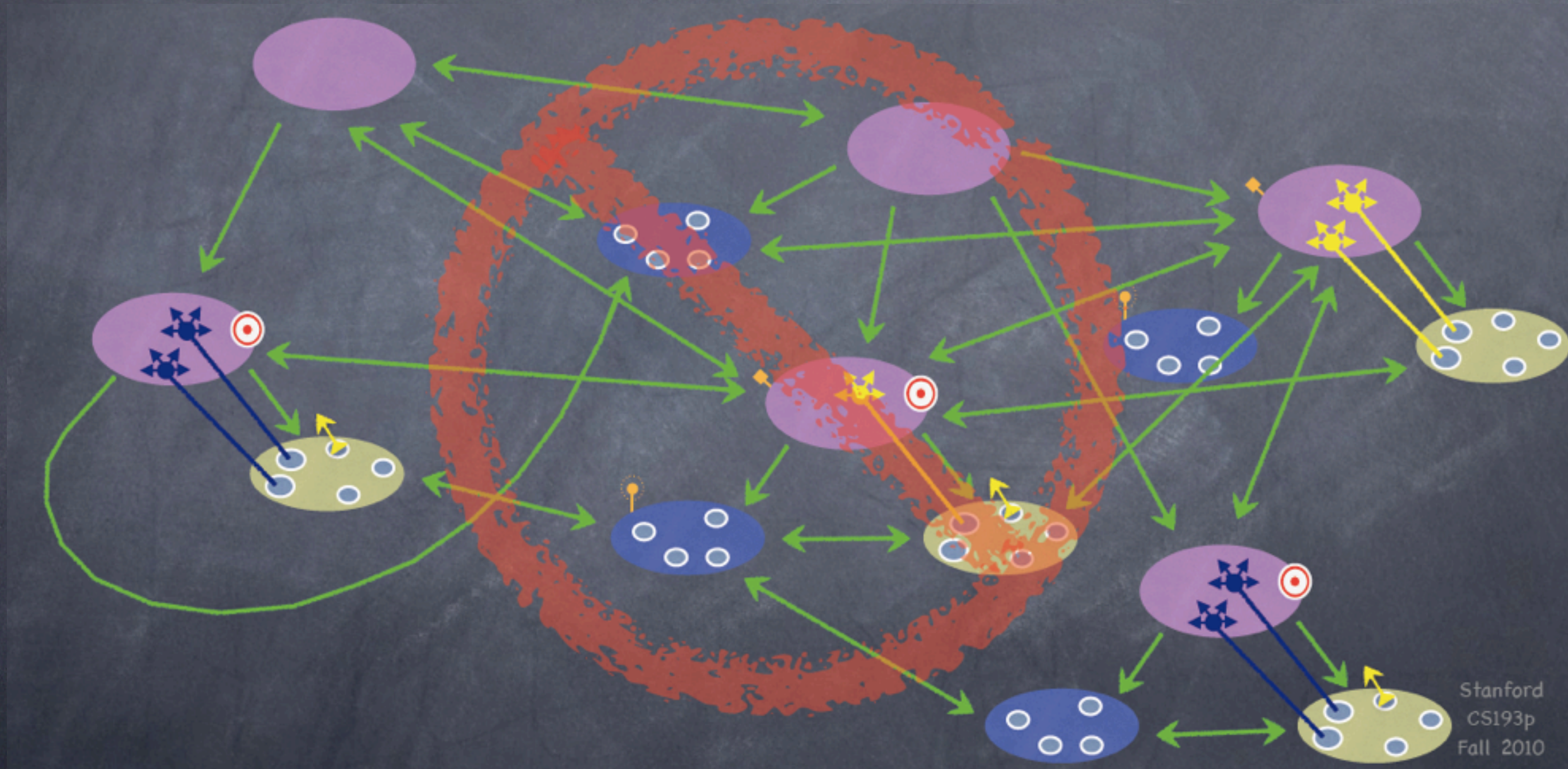
manchmal müssen aber Änderung in M an C übermittelt werden: M kann Broadcast senden auf welches C reagieren kann, auch V kann auf Broadcasts lauschen aber nicht auf solche von M!

MVC Model - View - Controller



komplexere Anwendungen haben viele MVC-Camps, wobei nur die C's untereinander kommunizieren

MVC Model - View - Controller



damit bleibt die Kommunikation
überschaubar/verstehbar/wartbar/skalierbar

MVC Model - View - Controller

- NIE: Code im falschen Camp ansiedeln!
- IMMER: neue Views generisch halten

- DEMO: `HelloWorld.xproj`