Model-View-Controller (MVC)

Dan Olsen, Developing User Interfaces (1998)
Model-View-Controller (MVC)

- **Model**: system state
  - This is the core, underlying representation
- **View**: what users see
  - Takes in the model, and generates output to the user
- **Controller**: how stuff gets changed
  - Takes in user input, generates a change in the model

- User side of the equation parallels this…
  - Mental model: user state [model]
  - Actions (input): what the system sees [view]
  - Perceptions: updates user model [controller]
Multi-User Systems
Multi-user Architecture
Model-View-Controller (MVC)
Model-View-Controller (MVC): Multi-User Systems
MVC: **Local** Multi-User Systems
MVC: Remote Multi-User Systems
MVC: Remote Multi-User Systems

Requires locking/sync of state on remote server
MVC: **Remote** Multi-User Systems

- **View**
- **Client Controller**
- **Model**

Requires locking/sync of state on remote server.
Questions?
Terms / Concepts
Multi-User Systems

Distributed UIs
Model: Represents Joint+Individual State

The model as a state representation does not change much.

However, with the addition of more users, the state becomes more complex.

The model is responsible for tracking both the overall state of the system, as well as the user-specific state (which may not be shared equally among others).
View: User Specifiers (View Multiplexing)

(implicit specifiers)

(explicit specifiers)
Controller: Joint Input

combined / predicated actions
- user can jointly interact, or modify others' potential interactions, which must be handled by the controller
Questions?

(I’m looking to extend this… how should I? What would you want to know?)