* View/Model/Controller
  + Controller – State Machine
  + Model - all the parameters
  + View - the Ascii output
* Game Loop change from “Input, Update, Render” to **State Machine**

How do we split up the work?

1. Get the Vision (What are we doing?)
2. Work Breakdown Structure / Psuedocode – Both at a High Level
3. UML Class Diagrams
4. Document our High Level Structure
5. Identify data structures
6. Assign Workload
7. Create Task Lists
8. Design/Create Sprint #1

Object-Orientated Programming - You take a group of related data, and you put them all inside of a class  
Signature – the prototype is the signature; the contract is what you pass in and what you get out.

|  |  |
| --- | --- |
| **Client** | **Server** |
| UI | Engine |
| Communications | Communications |

* Today:
  + Doc Hugh Level Structure
  + UML / DATA
  + Psuedocode
* Wednesday
  + Create task list
  + Assign Work Load
  + Design Sprint #1

|  |  |
| --- | --- |
| **Data We Will Need** | **Model For The Game** |
| Grain Amount | 1 Bushel per Acre |
| Grain Amount Change | 1 Person Plants 10 Acres |
| Grain To People | 20 Bushels Per Person to Feed |
| Grain Harvested Per Acre |  |
| Population |  |
| Max Acres Per Planter |  |
| Deaths |  |
| Land |  |
| Land Amount Change |  |
| Land Price Per Acre |  |
| Morale |  |
| Time |  |

|  |  |
| --- | --- |
| **Classes** | |
| Grain Class | Game Engine |
| Population Class |
| Land Class |
| Controller Class (Log In) |
| Game Class |
| Economic Model Class |
| Event Class |
| Communications Class | Client / Server |
| Message Class |
| UI Class | User Interface |

Keep an internal task list for everyone

|  |  |  |  |
| --- | --- | --- | --- |
| Who is Going to Do What? | | | |
| Kevin | William | Gary | Jordon |
| Communications (Client) | User Interface | Controller | Model |
| Communications (Server) |  |  | Event |
|  |  |  | Grain |
|  |  |  | Land |
|  |  |  | Population |