SOEN 387 Web-based Enterprise Application Design

Stuart Thiel

Concordia University
Department of Computer & Software Engineering

Fall, 2014

Outline

Frameworks

Frameworks

- People say framework a lot
- Everyone wants to write their own framework
- I don't suggest that!
- But sometimes you do something that's not a framework
- Maybe you do it a lot
- And then you realize you can pull out a framework

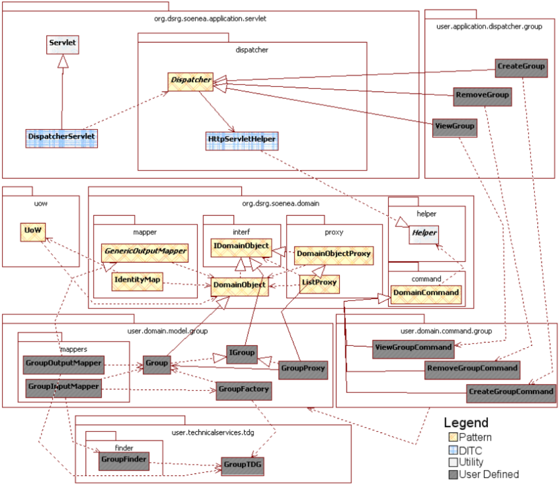
What Frameworks Give You

- People redefine the word framework to suit themselves
- It really does imply something architectural
- This course covers high-level architectural patterns for webapp development
- If what I say here is true, then web frameworks of all flavors and languages should reflect these patterns

SOENEA

- We're going to look at SoenEA
- It's kind of cheating in terms of highlighting the patterns we've covered
- I wrote/adjusted it specifically to suit these patterns
- I grabbed a couple things like Command Annotations from Spring
- I used it on well over a hundred different webapps/software development projects

Applying SoenEA



What Did SoenEA Give Us

- Concrete base-classes/interfaces for everything save
- A very convenient Helper Utility to take care of basic stuff
    - Template Views
    - Input Mappers
    - Factories
    - Finders
    - TDGs
- DomainCommand gives strong support for for fields and field/assignment via annotations, and is one of my favorite features. . . that I basically stole from Spring because a Spring developper using SoenEA wanted them
- DomainCommand also gives basic parameter validation Annotations
- UoW is the Identity Map for everything and just automagically takes care of stuff

What You Have To Give SoenEA

- Output Mappers are no longer static
- This is to take advantage of some dynamic instantiation stuff
- You need to explicitly map between Domain Objects and their corresponding OutputMappers in Servlet Initialization
- You need to explicitly use the UoW as an IM in your Input Mappers and Factories

How to Start

- You can start with Dispatchers and work down (very TDD)
- You can start with Domain Objects and work up
- You can start with Template Views and work down (also very TDD)
- Pick an approach and be consistent and you'll be ok

How to Use SoenEA

- Inherit from SoenEA base-classes
  - Dispatchers
  - Commands
  - Domain Objects (interfaces and proxies too)
  - Output Mappers
- Register Domain Objects with their Output Mapper in servlet initialization
- Use UoW as an IM in InputMapper
- Initialize UoW when you start a transaction for a request
- Commit when you're done (maybe in Dispatcher?)
- Rollback when things fail (maybe also in Dispatcher?)
- Most other stuff is the same