Rapid Development of Web Interfaces to Heterogeneous Systems

José Paulo Leal
zp@ncc.up.pt

Marcos Aurélio Domingues
marcos@ncc.up.pt

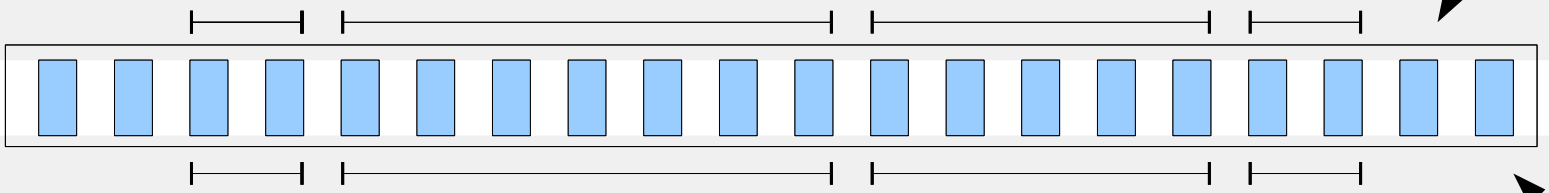
University of Porto
Portugal

Outline

1-Introduction

3-Implementation

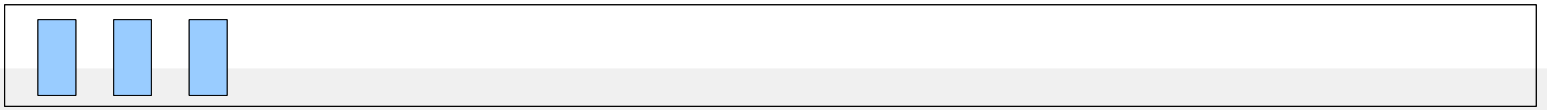
5-Conclusion



2-Architecture

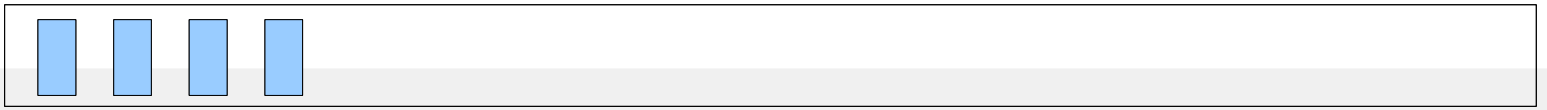
4-Case study

6-Future work



Intro: Motivation

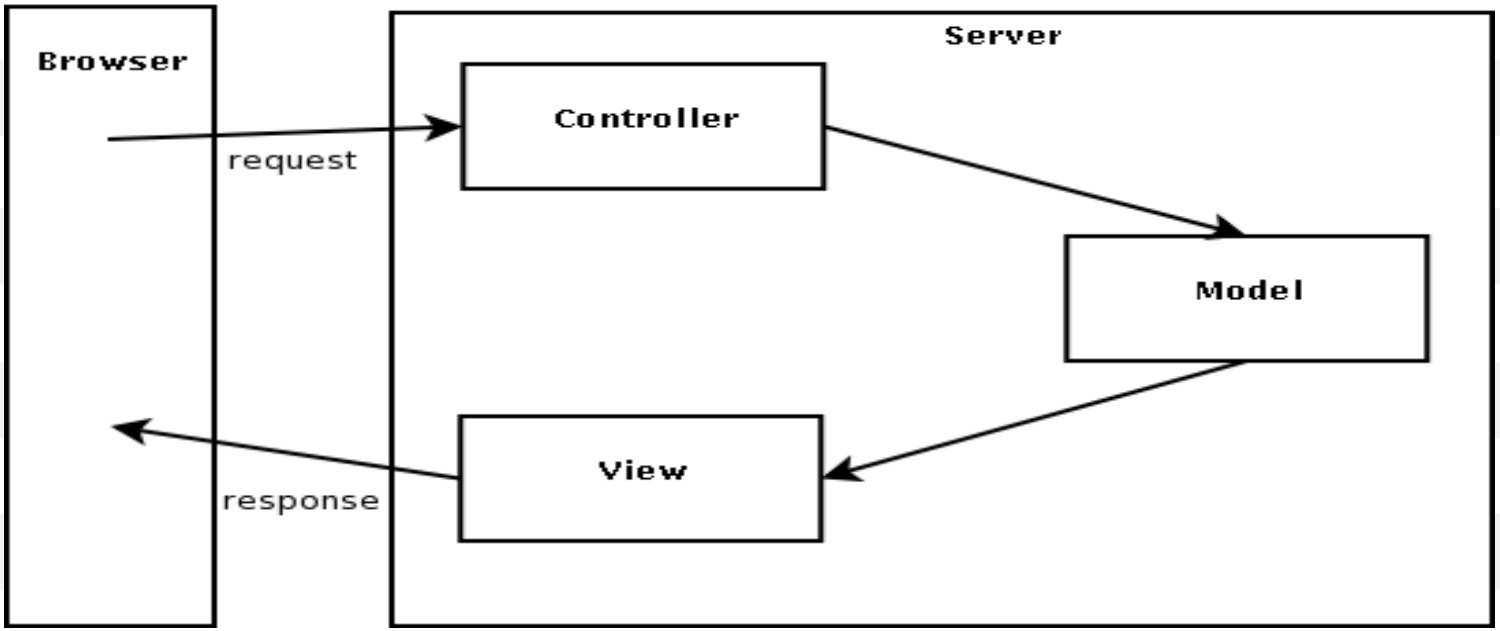
- Web interfaces to heterogeneous systems
  - Research systems
  - Legacy systems
- Web interfaces:
  - Easily deployable
  - Fairly good visualization and interaction
- Systems:
  - Command line interpreter
  - Text output



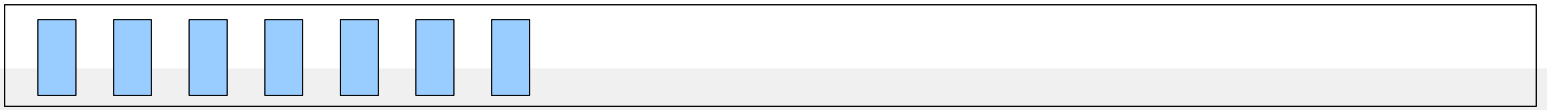
Intro: Approach

- Framework for web interface support
- Hot spots: XML configuration files
  - System parameters (state)
  - Interaction + state -> system commands
  - System output + state -> web formatting
- Simple communication with system

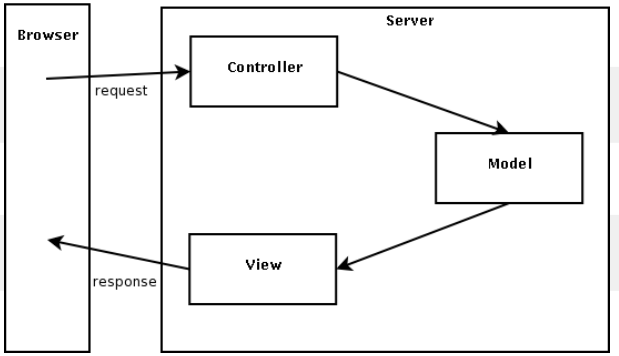
Architecture: MVC (1)



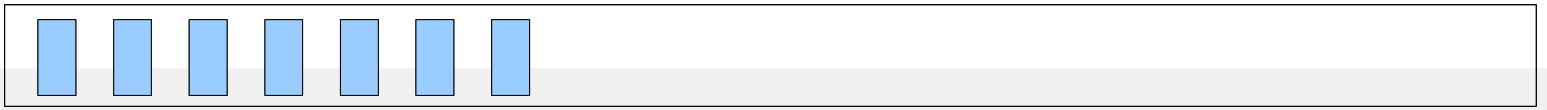
- Pattern Model-View-Controller
- Decouples GUI from logic
- Separates commands from presentation



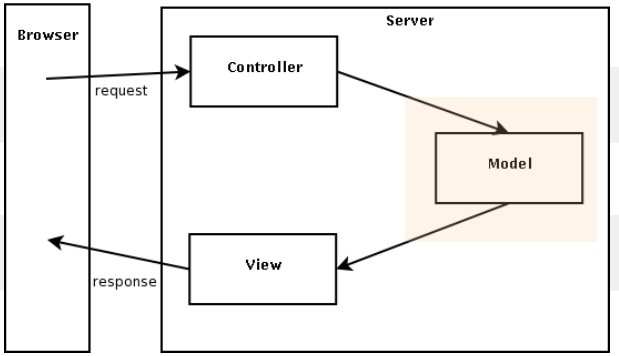
Architecture: MVC (2)



- Discovered for Smalltalk
- OO applications with GUI
- Rediscovered as “model2” for *webapps*
  - Views: JSP
  - Controller: servlets
  - Model: Java beans
- Frameworks implement MVC
  - Struts
  - Spring

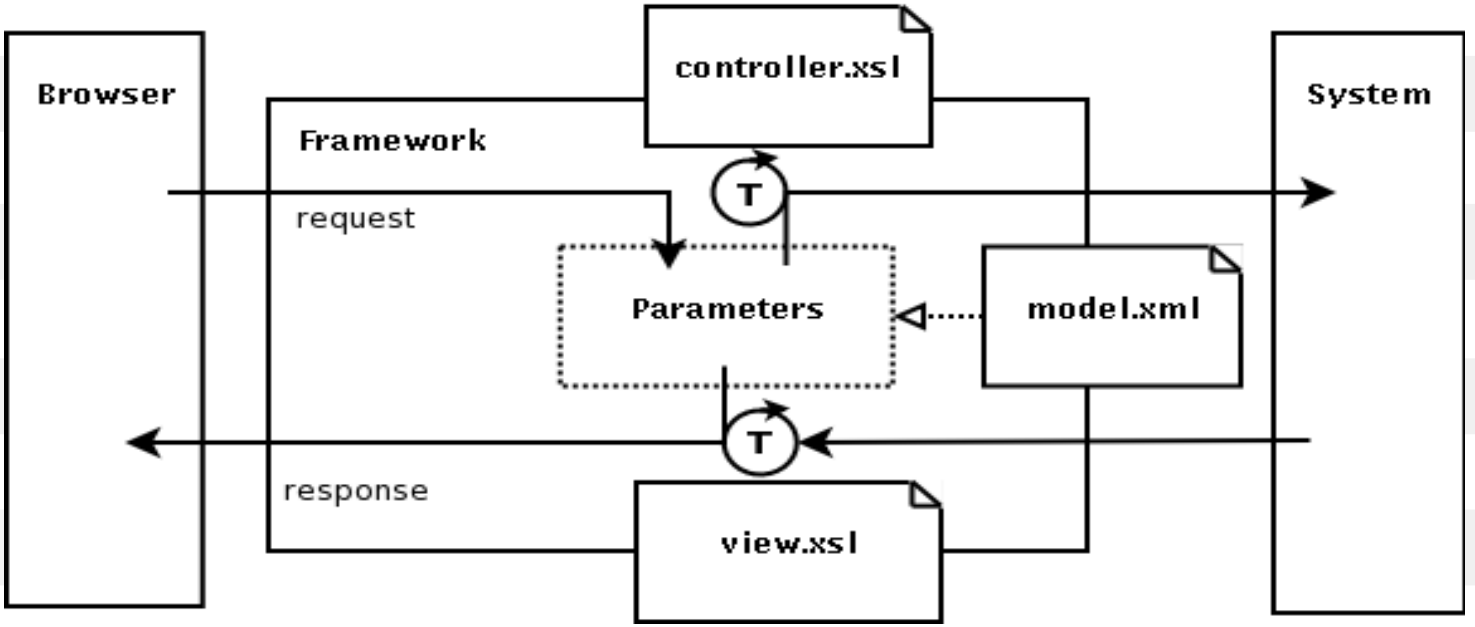


Architecture: MVC (3)

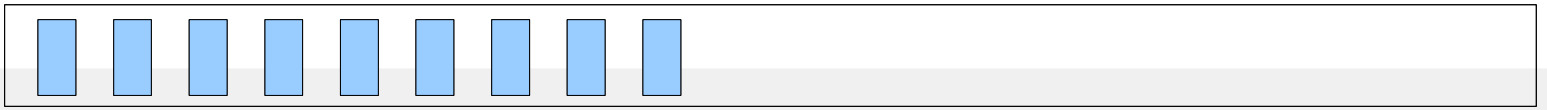


- System is the model
- Model as set of Java beans
  - Beans define **properties** (parameters)
  - Access trough setters/getters
  - Usually running in same process
- But system is a remote process
  - Avoid remote calling (e.g. RMI,RPC)
  - Execute command in system's interpreter
  - Process system output

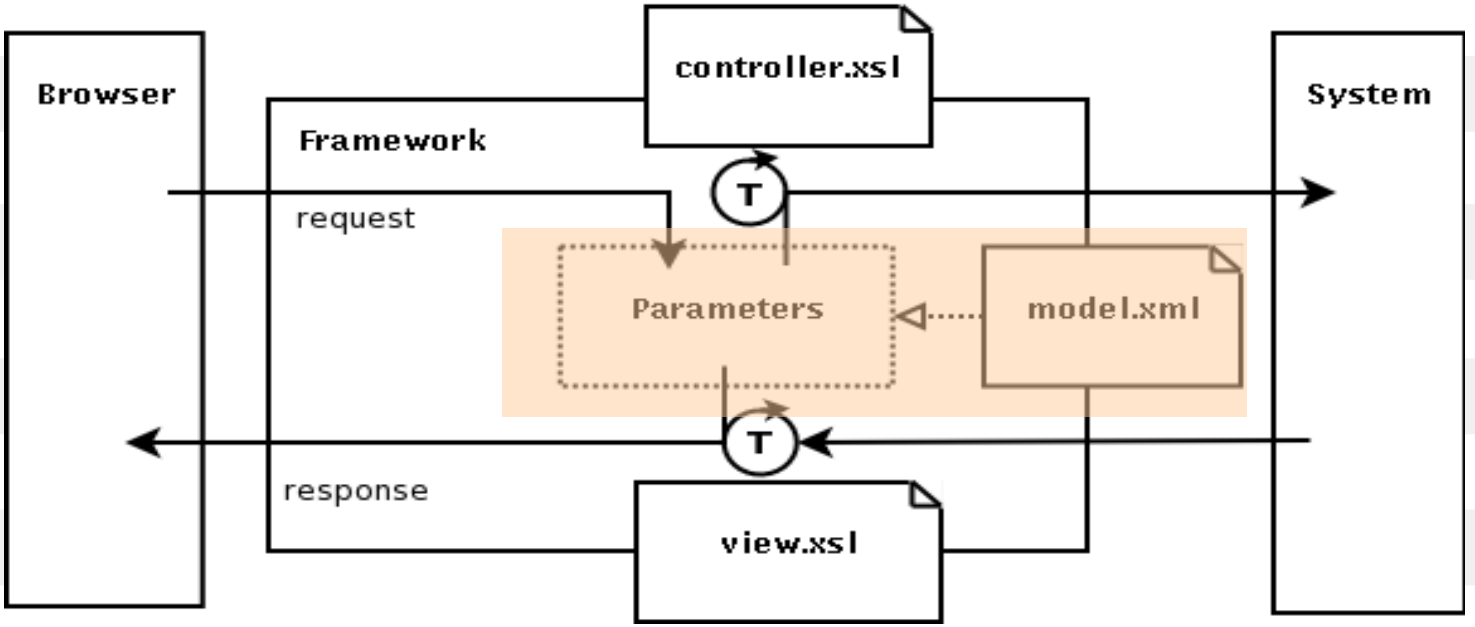
Architecture: framework (1)



- Framework hot spots inspired by MVC
  - Model abstract the system
  - Controller transform input to system
  - View transforms system output
- But are XML files, not software objects

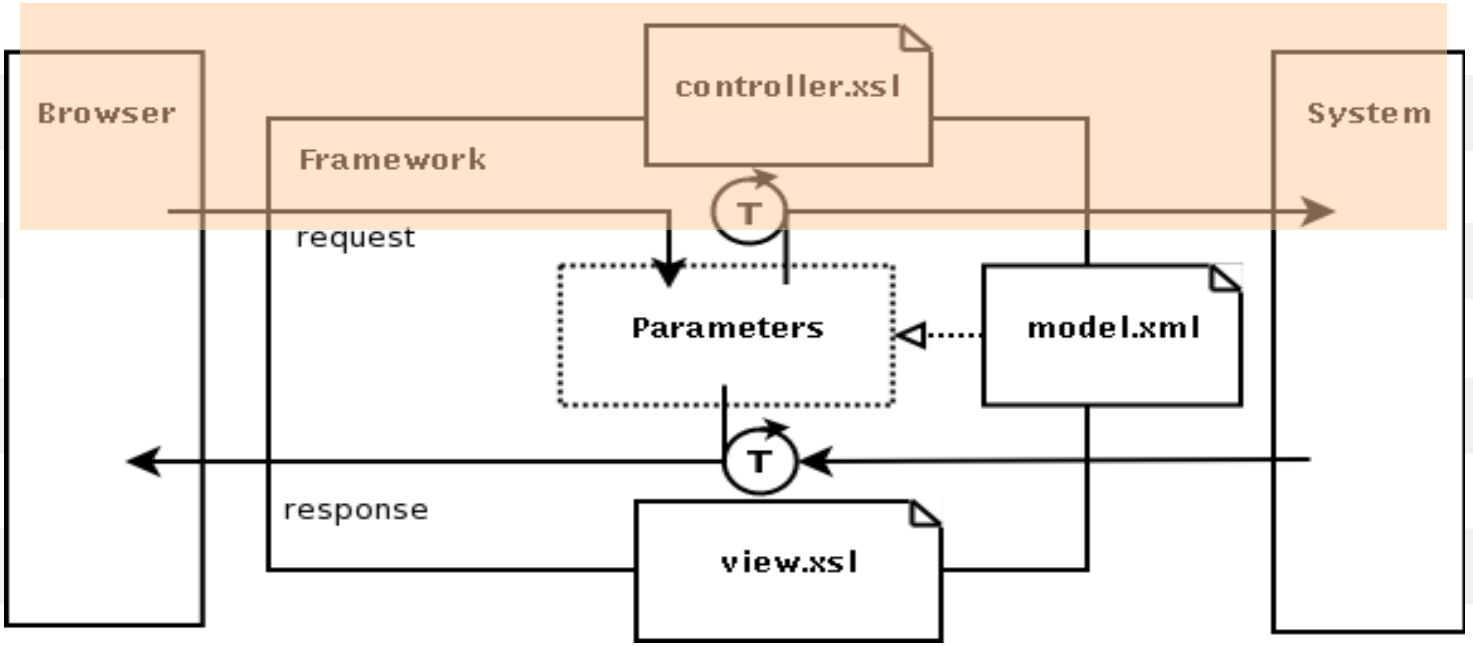


Architecture: framework (2)

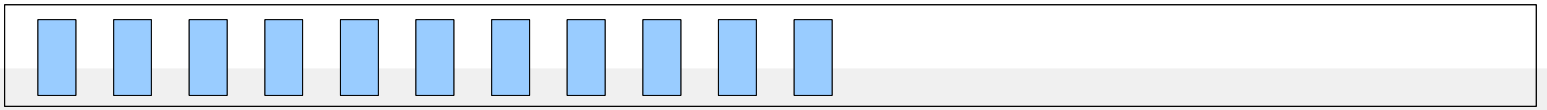


- Parameters (properties) as interface
- Loaded from XML to DOM object
- Bound to users sessions
- Parameters are interaction state

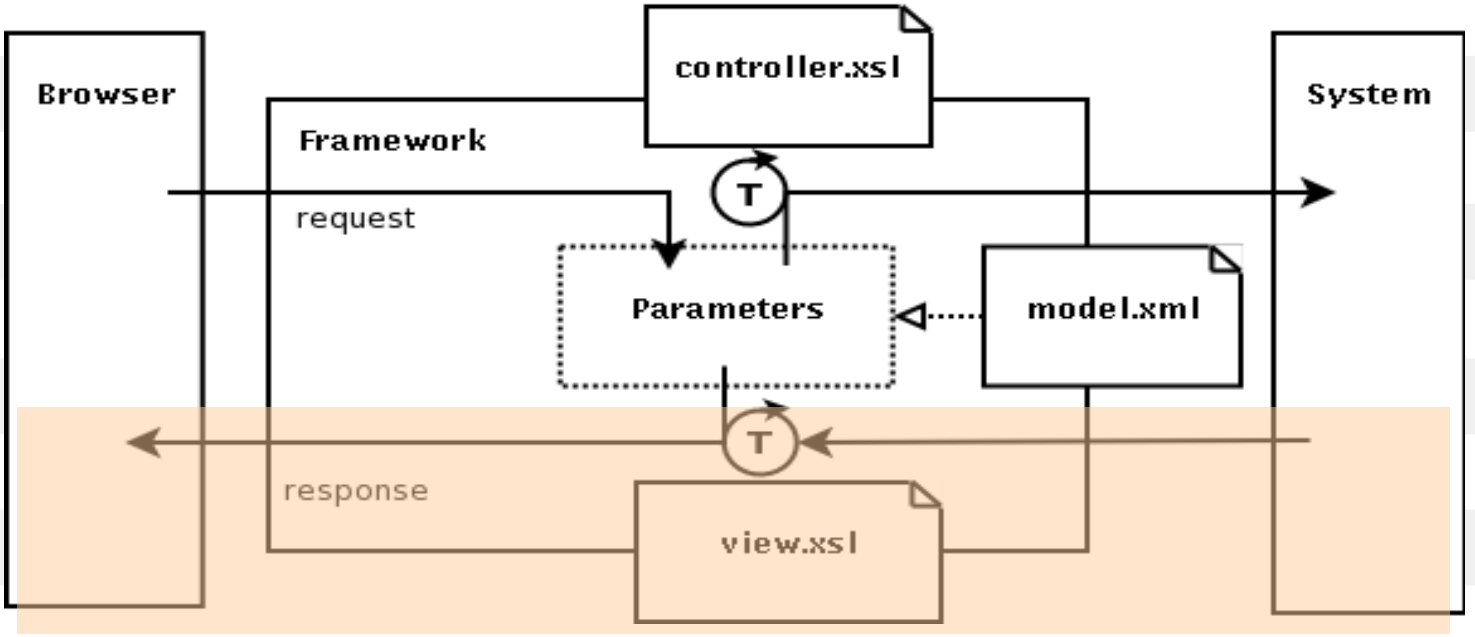
Architecture: framework (3)



- HTTP Requests
  - Changes parameters
  - Transforms them in system's commands
- Transformation uses `controller.xsl`

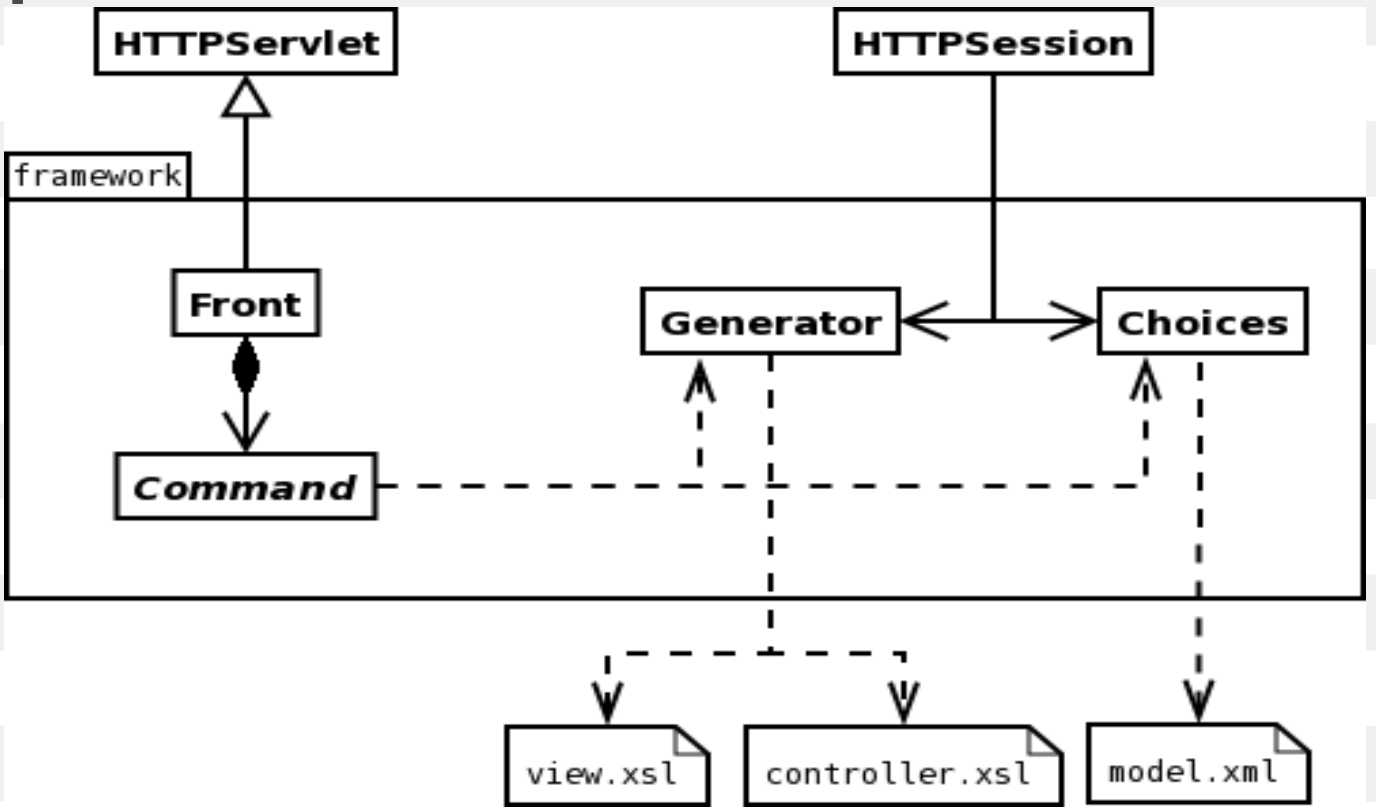


Architecture: framework (4)



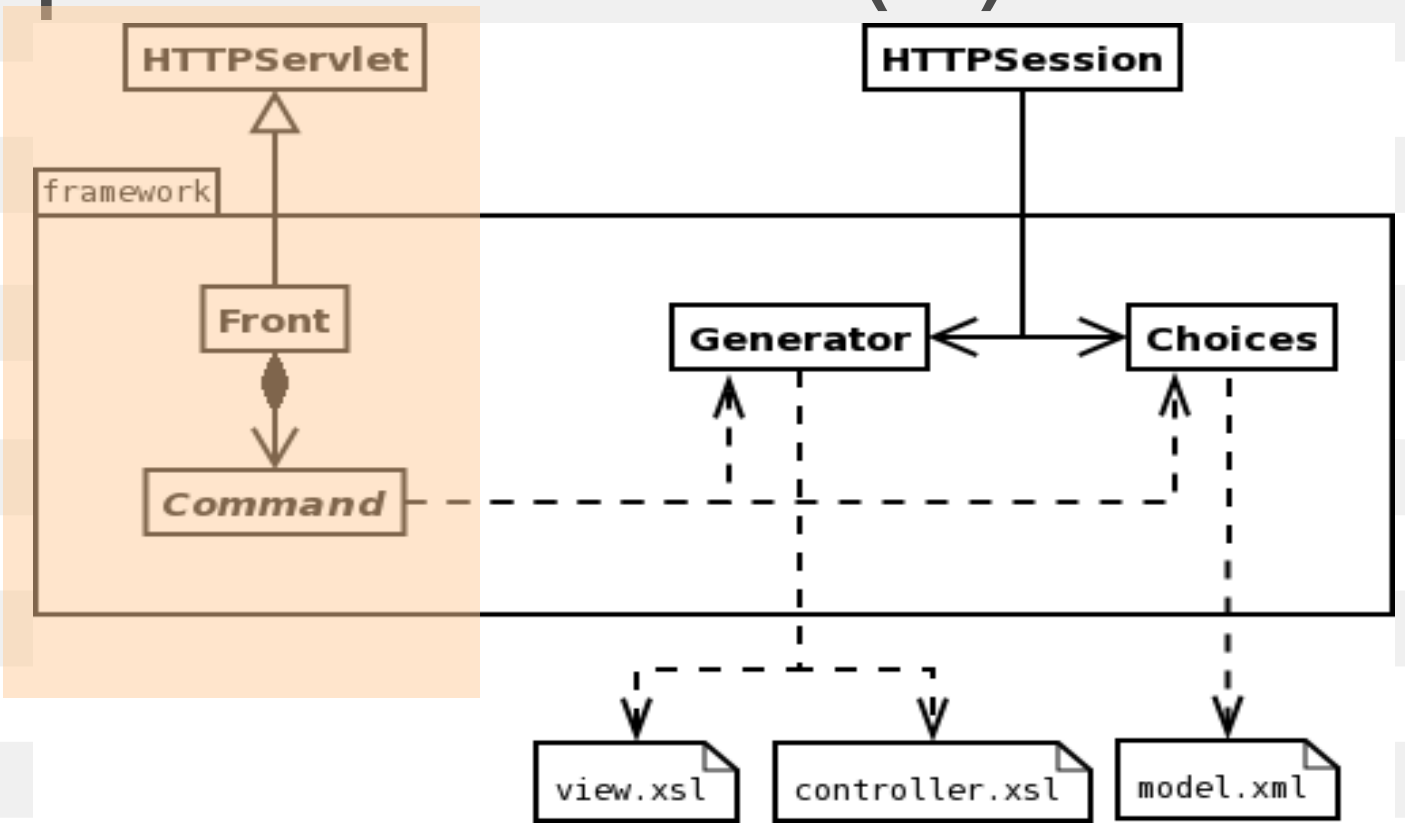
- System output converted to XML
- Transformed to HTML as response
- XSLT file acts as view definition

Implementation (1)



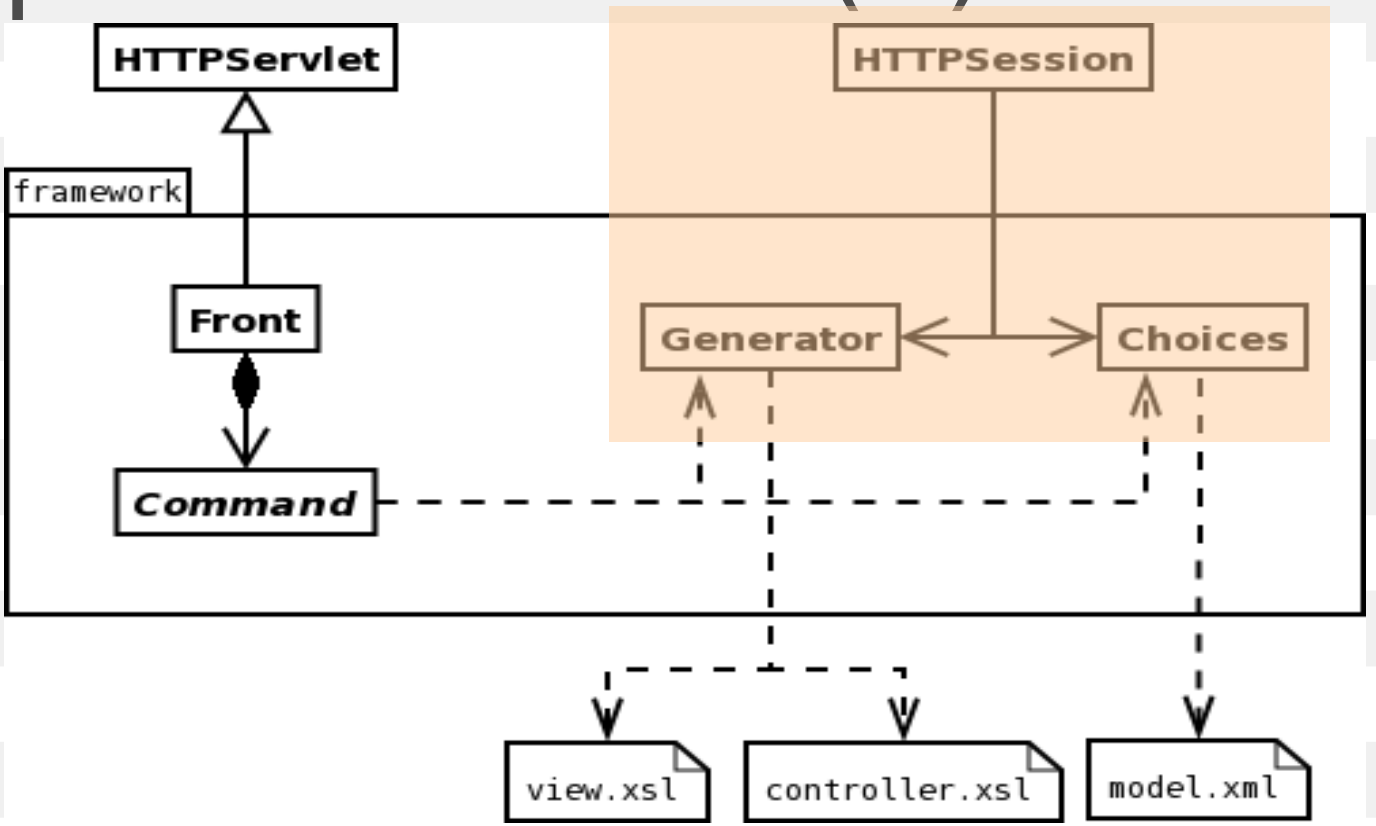
- Framework implemented
  - Java *webapp*
  - Tomcat *servlet* container
- UML class diagram (conceptual)

Implementation (2)



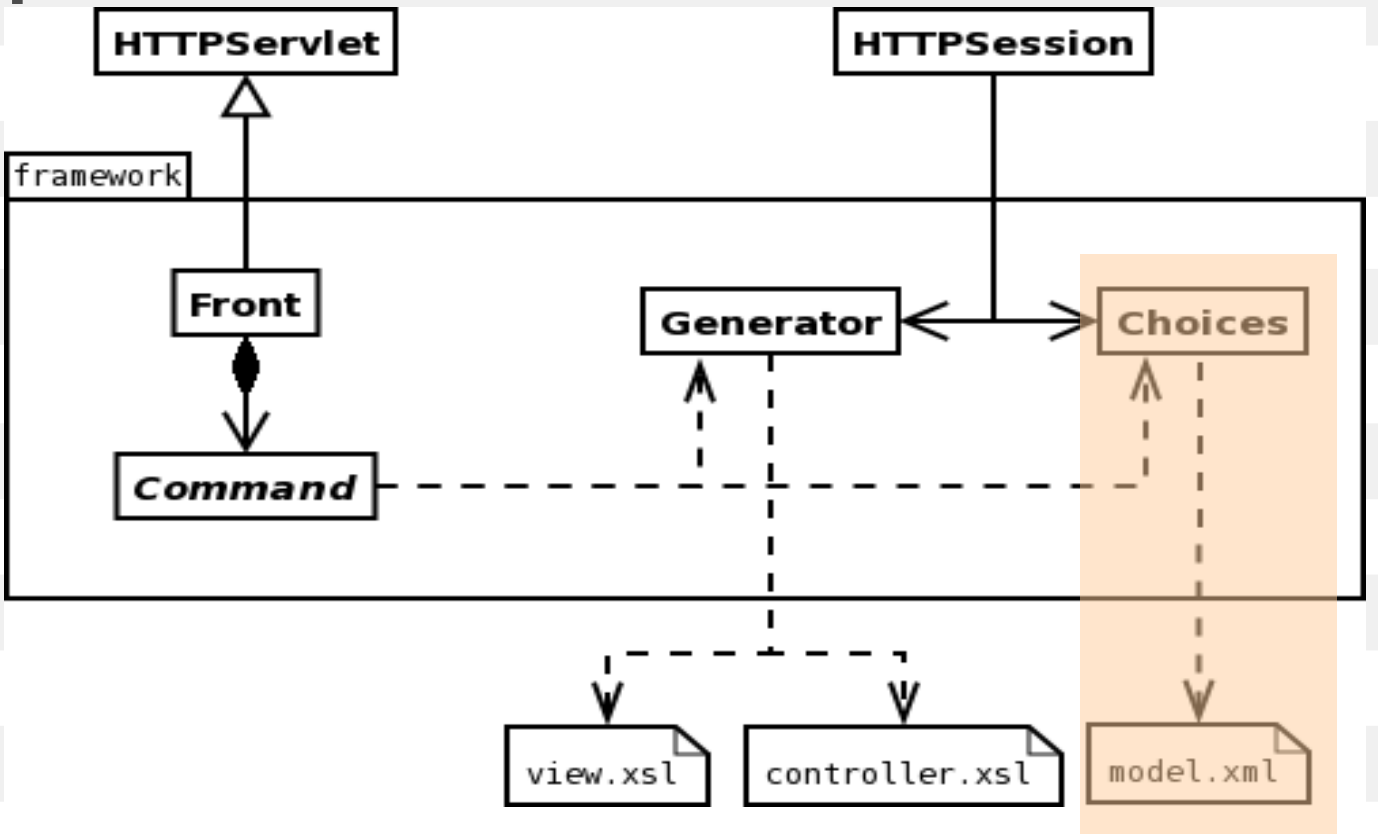
- Single `Front` controller is HTTP Servlet
- `Commands` process requests
- Depend on other framework classes

Implementation (3)



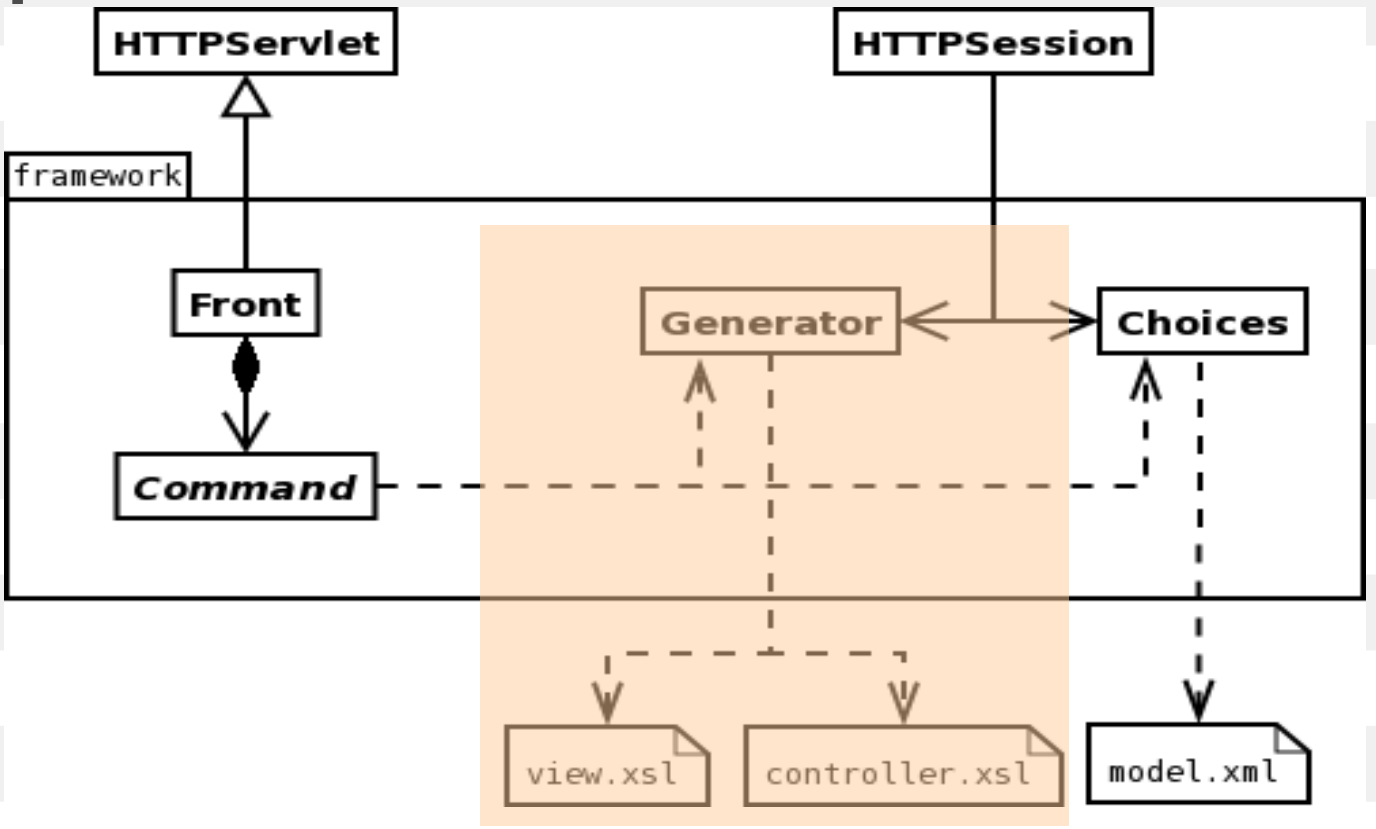
- Sessions control `Generator` and `Choices`
  - On creation they are initiated
  - On termination they are disposed
- Connection to a system process

Implementation (4)



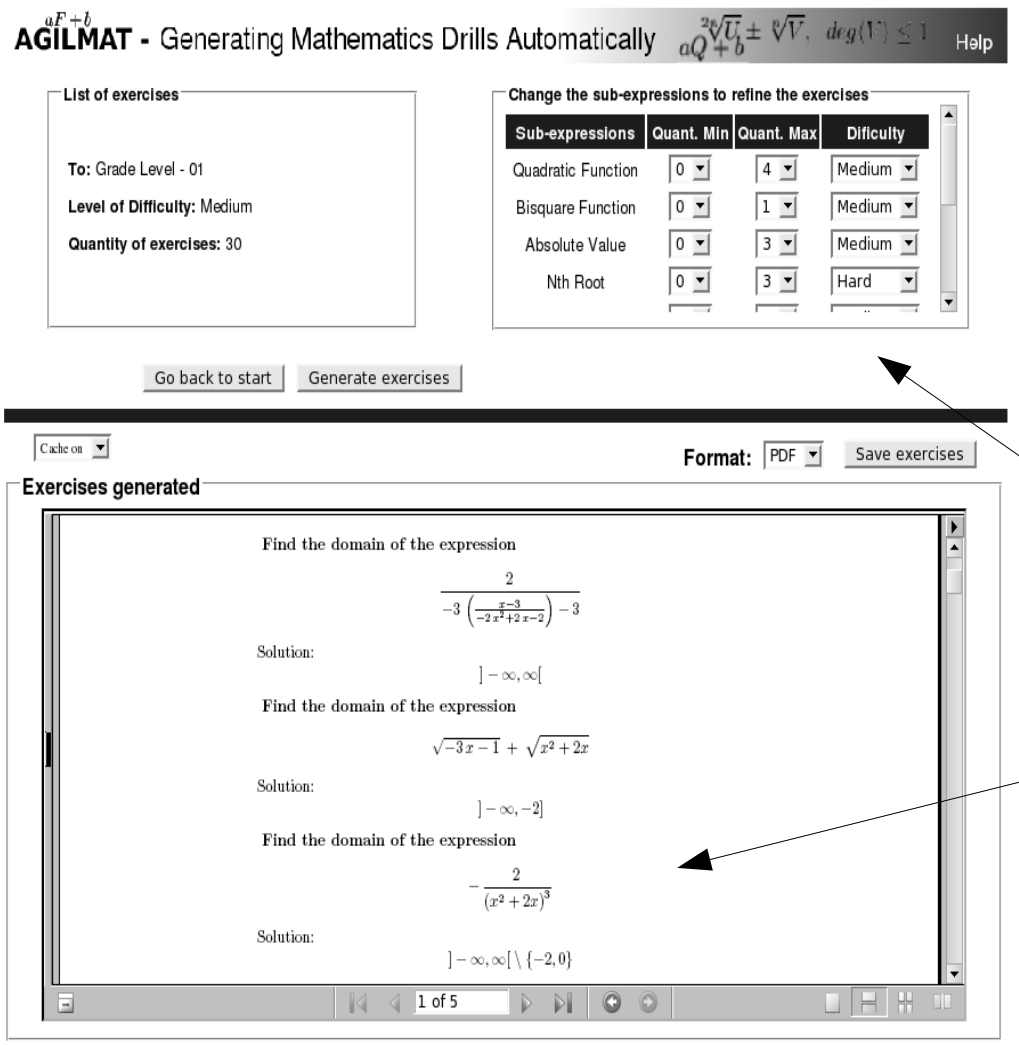
- `Choices` hold current parameter state
- DOM loaded from valid `model.xml`
- Controls life cycle of system connection

Implementation (5)



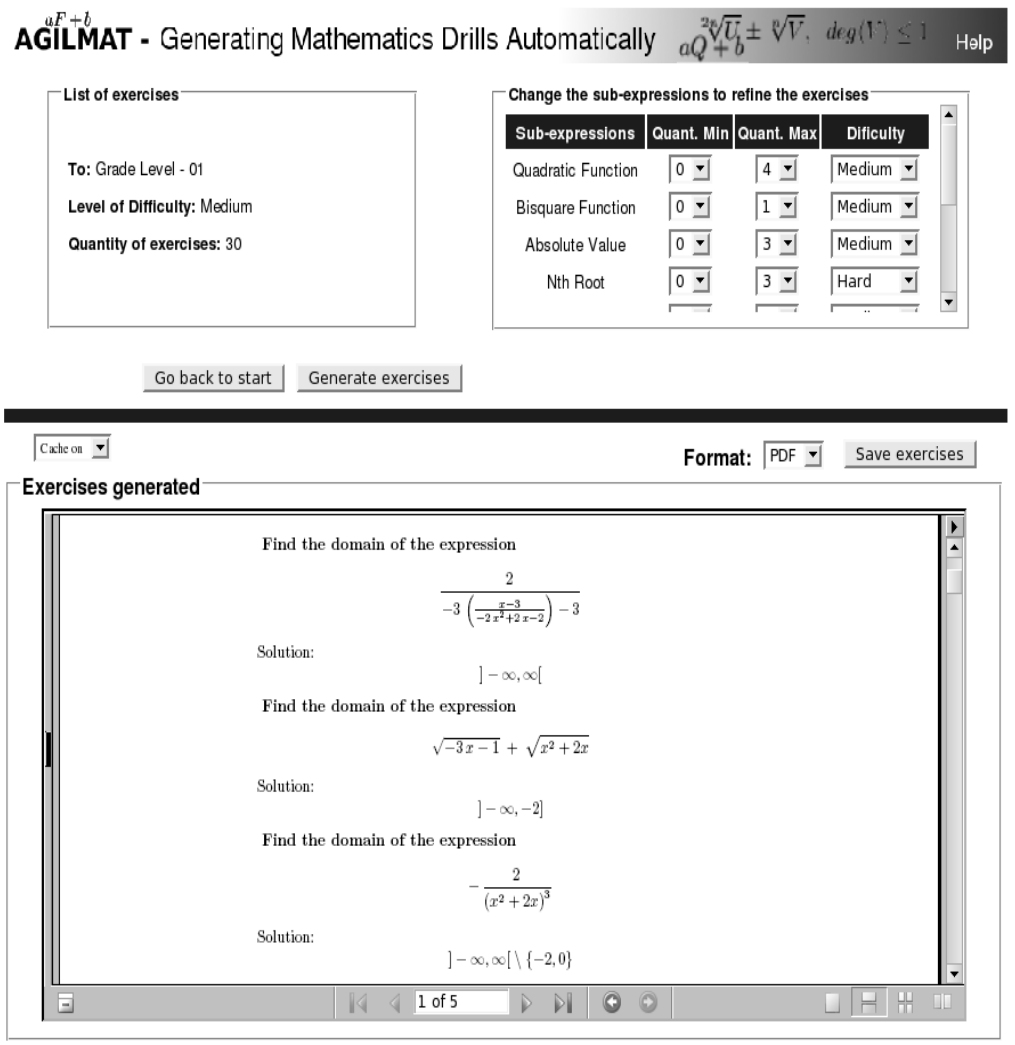
- `Generator` transforms:
  - `Choices` to commands: `controller.xsl`
  - System output to HTML: `view.xsl`
- HTML Cache avoids command execution

Case study (1)

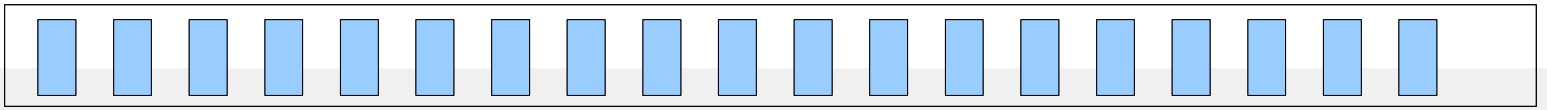


- Math exercise sheets generator
- Constraint Logic Program System
- Many parameters
- Produces
  - LaTeX -> PDF
  - QTI + MathML
- Available on-line

Case study (2)



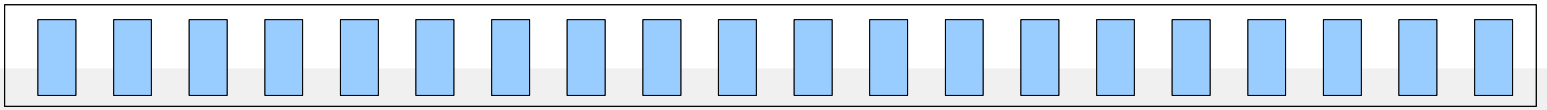
- Web wizard
- `Controller.XSL` produces Prolog queries/predicates
- Changes in DTD of `model.xml`
- Term to XML conversion
- Cache system



Conclusions

+ System/GUI decoupling
+ Configuration of
  - ✔ Parameters
  - ✔ Commands
  - ✔ Presentation
+ Efficient

- Model language
- Cache invalidation



Future work

- More applications
- Highly interactive web interfaces (Ajax)
- Improve cache invalidation