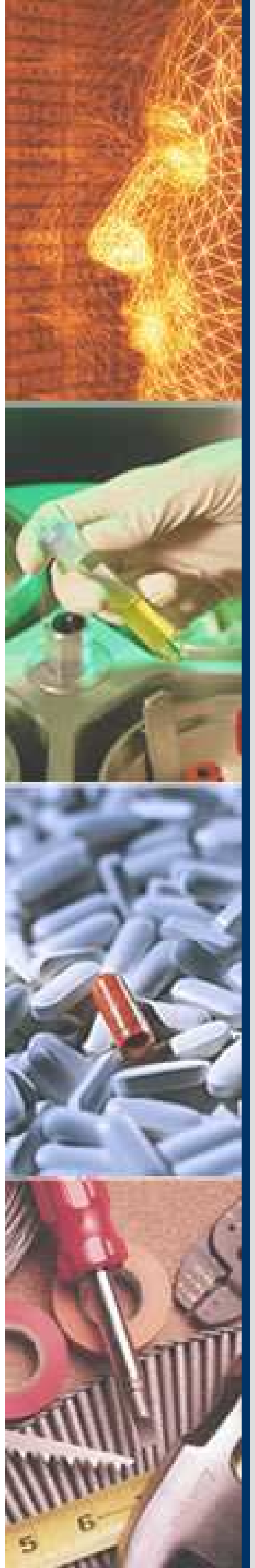
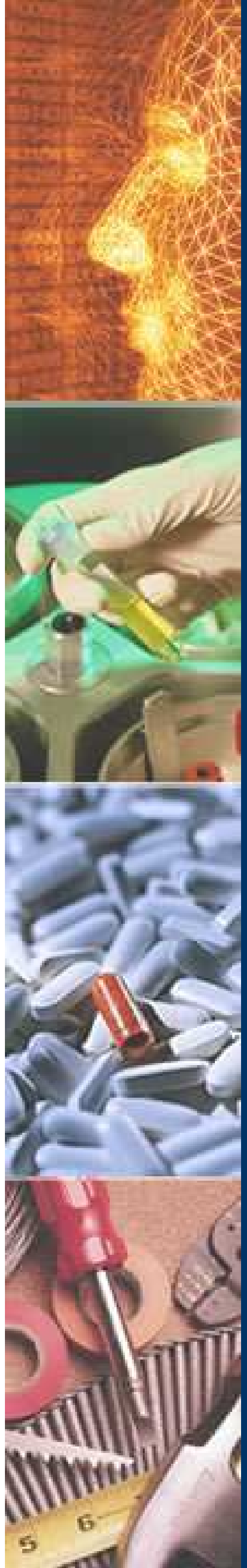


PharmQuest's Functional Testing Harness

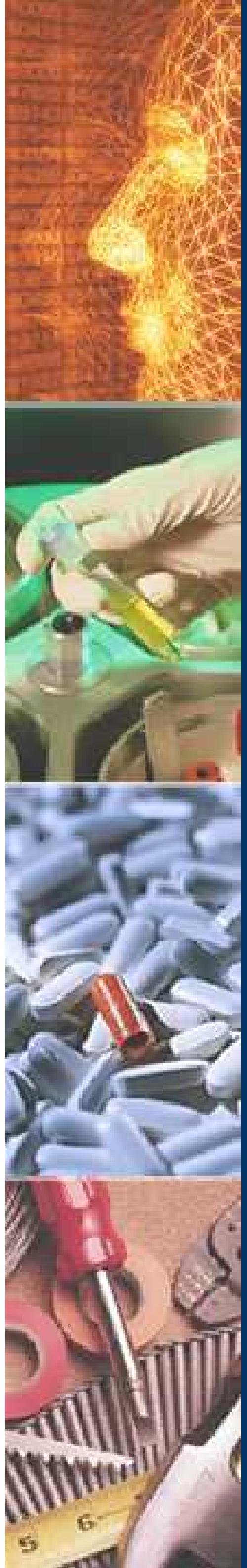
By

Joe Glenny Thomas



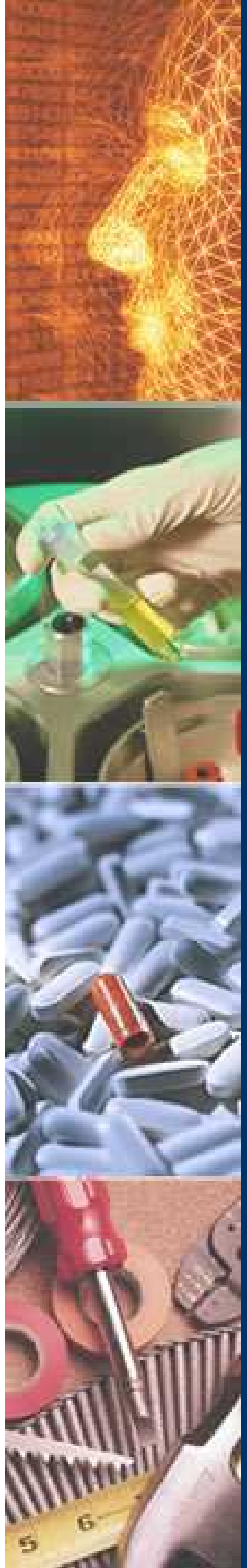
Importance of Functional Testing

- Enhanced software quality
- Unit testing is not enough
- We need to test the software as an actual user will use it
- Automated functional testing
  - Speed
  - Enables functional testing to be part of the build process



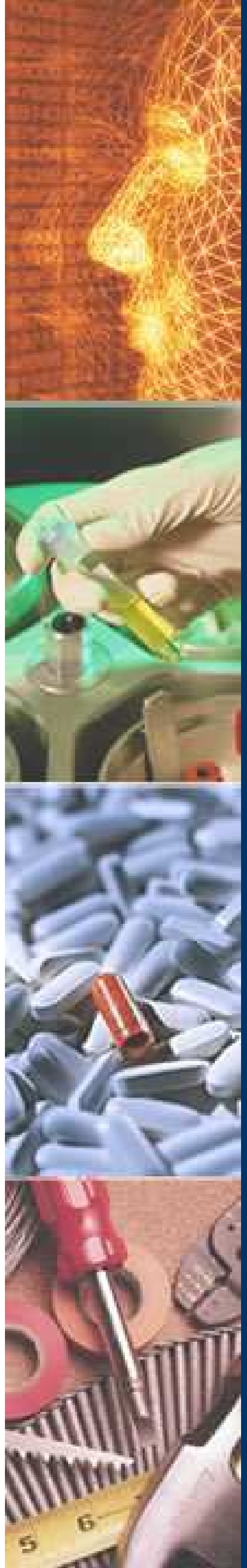
Writing Test Code

- Approaches for writing test code
  - Functionality driven
  - Data driven



Functionality-Driven Approach to testing

- Code/Script is written to execute every functionality in the software
- Various parameters are checked as the code executes
- This is a good approach if we need to test for parameters beyond the data displayed on the screen
- Large amount of code needs to be written if a screen has more than one data view
- The data to test for must also be encoded

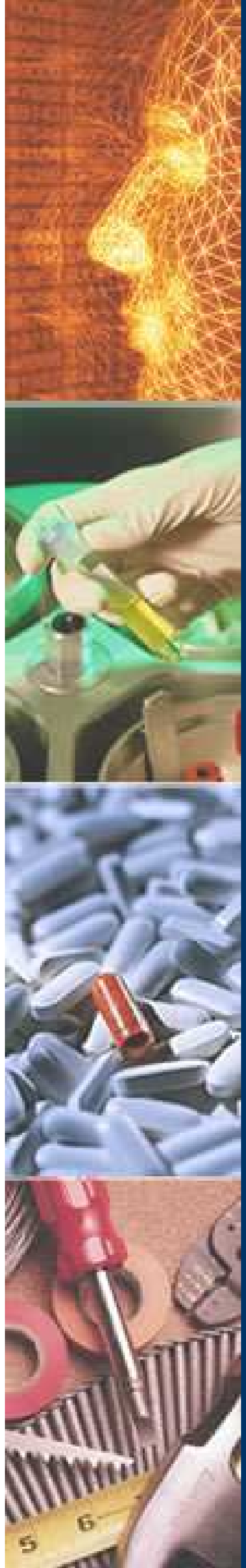


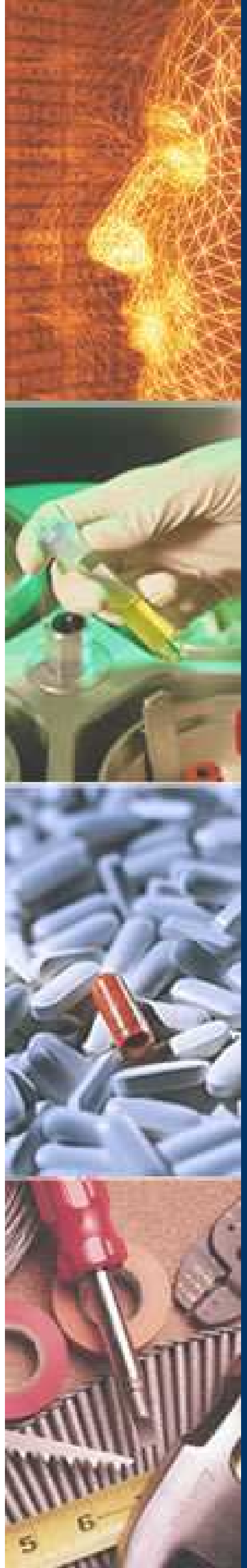
Data-Driven Approach to testing

- Contains a generic script to reach an end point
- The path to be taken and the final data to test is dictated by a specially formatted data file
- Incorporating a new view can be done by adding a data file
- This is a good approach if only the end data has to be verified
- Ease of use

Available functional testing tools

- FIT
- SuiteRunner (www.artima.com)
- Canoo WebTest
- WinRunner



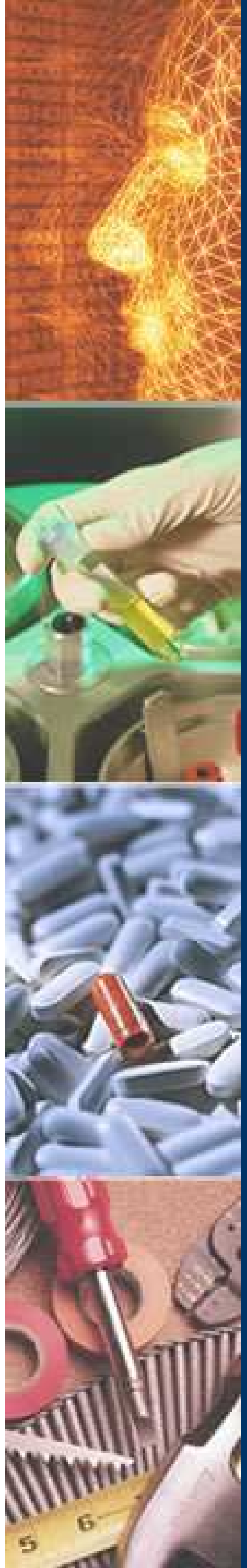


Our experience using Canoo Webtest

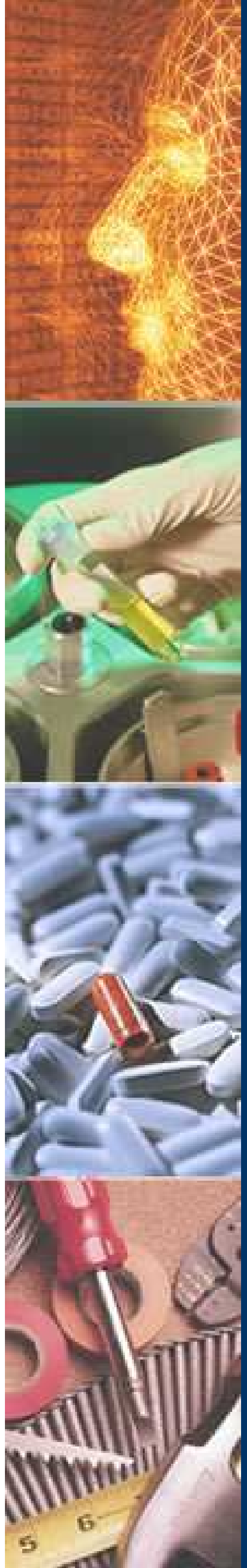
- Canoo is a testing framework built on top of HttpUnit
- It uses scripts written in xml
- Data to be compared with is represented as xml
- The comparison is done using xpath

Our experience using Canoo - Advantages

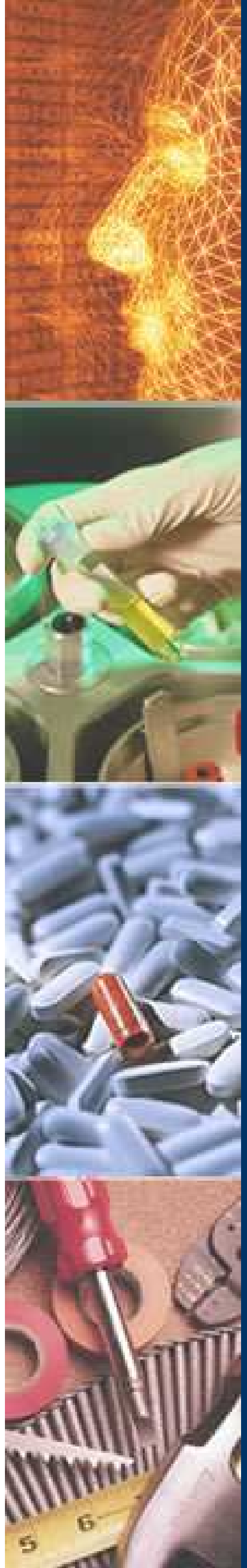
- Writing execution scripts was very easy using xml
- Knowledge of Java was not required to write test scripts
- Xpath allowed us to identify the element in the web response for comparison



Our experience using Canoo - Disadvantages

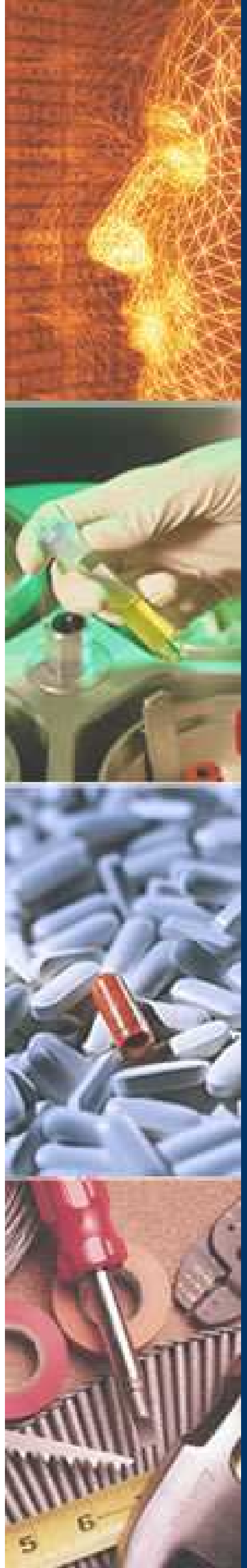


- Scripts had to be duplicated with different parameters
- Adding a new data view required duplication of scripts as well as creation of test data
- Xml files were very large and unweildy
- Javascript support in Canoo was a subset of Javascript support in HttpUnit



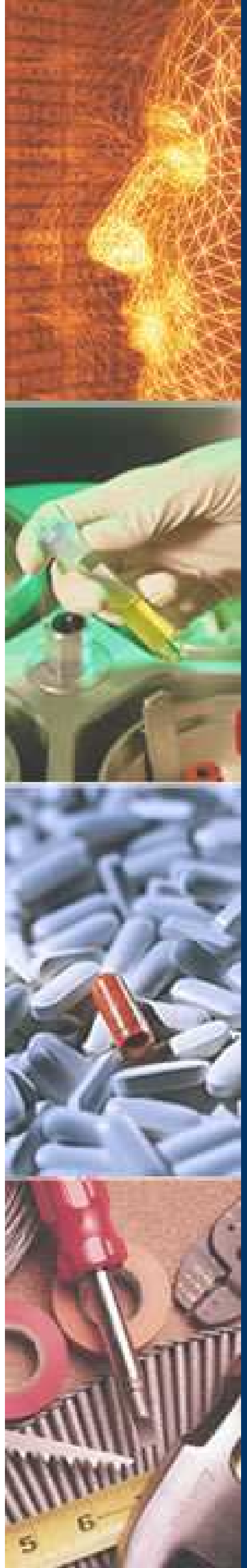
Our experience using Canoo - Code

- Webtest directories and files
- A Canoo script
- A Canoo data file



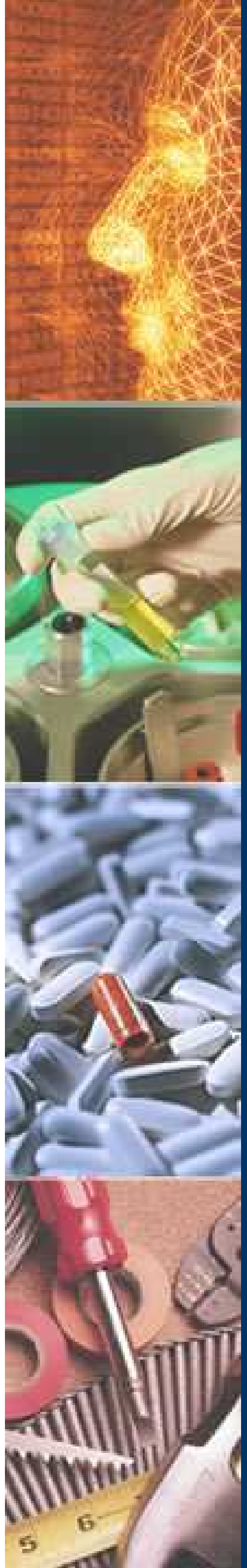
The PharmQuest Test Harness

- How it started
  - We needed a framework which could support more readable data files
  - Most of the tests in our software are for testing end data against expected data
  - A data driven approach would reduce the duplication of test scripts
  - Using HttpUnit directly would give us access to all its features
  - We wanted to code our test scripts in a language that could be executed without compiling
  - We wanted to code the test scripts in a language that was very powerful in text processing



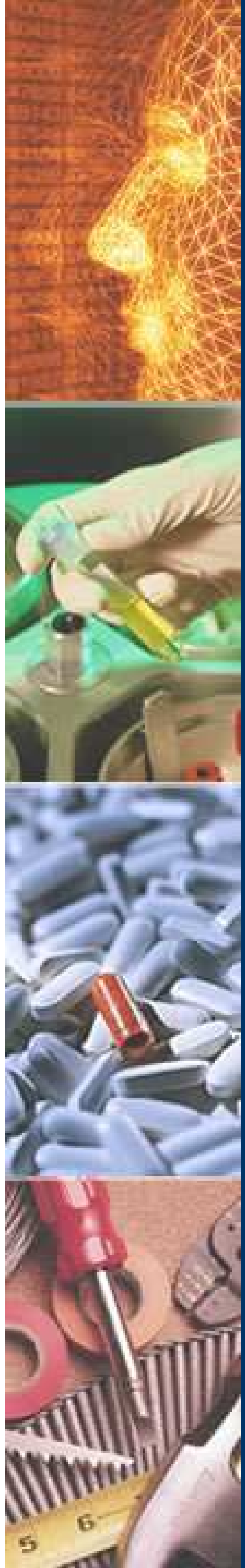
Choice of HttpUnit

- Web based software needs a tool to simulate a Web User Agent.
- HttpUnit does a great job
  - Reasonable support for javascript
  - Provides a web response as an object heirarchy
  - Developed in Java



Choice of Jython

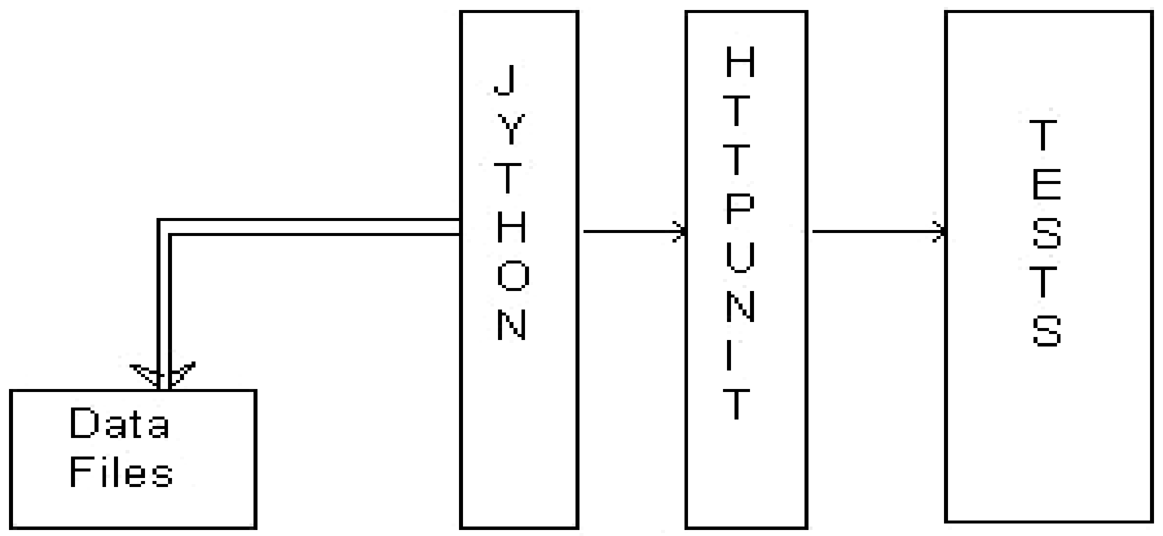
- Easy to manipulate data files using Python (since we have chosen a data driven approach)
- Need for interface to java classes



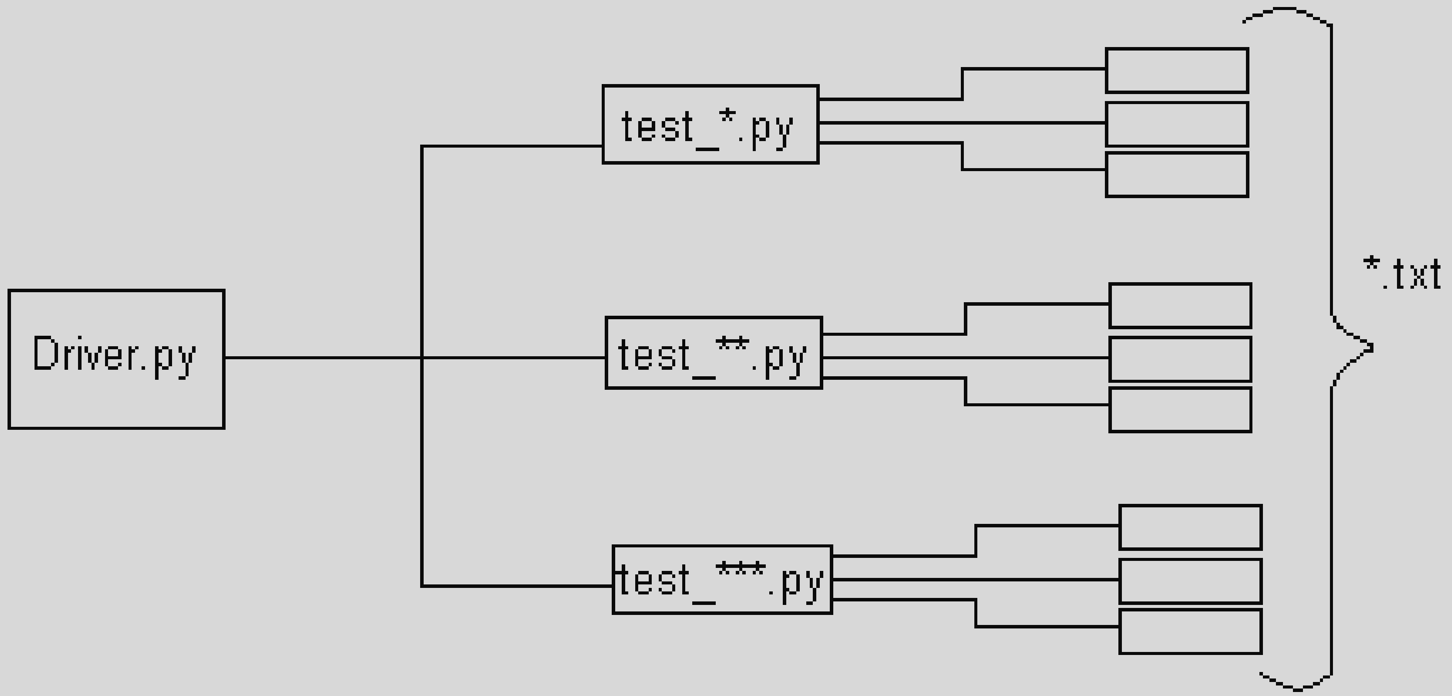
The PharmQuest Test Harness

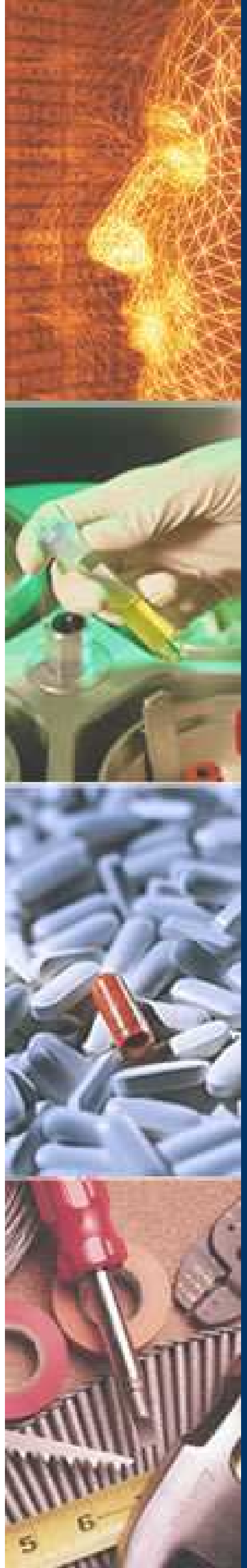
- The design
  - Designed data files that would encapsulate the data as well as the path to reach the data
  - HttpUnit was to be used directly
  - Python has better support for text processing than Java
  - Python does not need compilation
  - Jython gives access to Java code through Python
  - Designed to suit our application.

Diagram



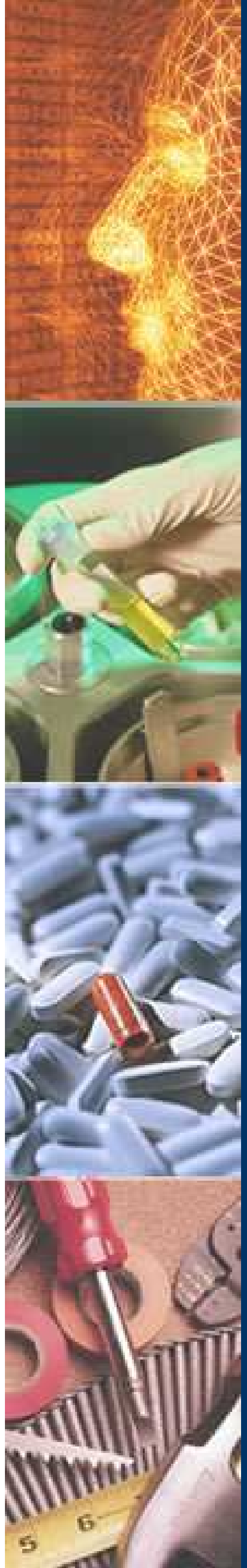
PharmQuest Test Harness – Detailed Architecture





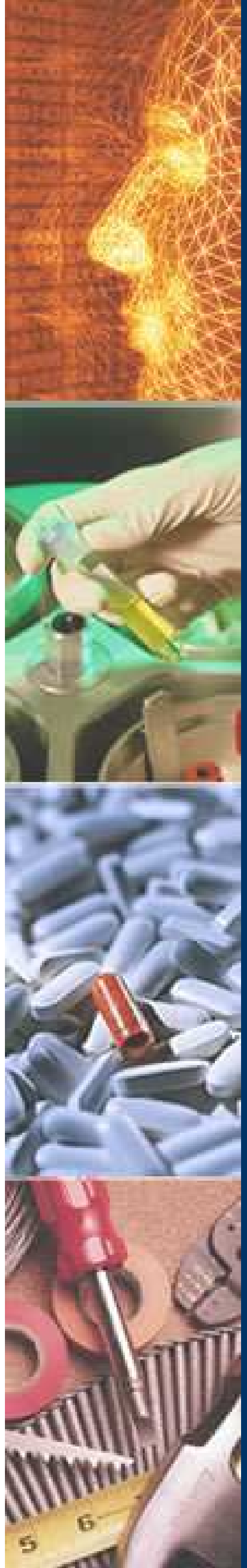
PharmQuest Test Harness

- Core scripts
- Data file format



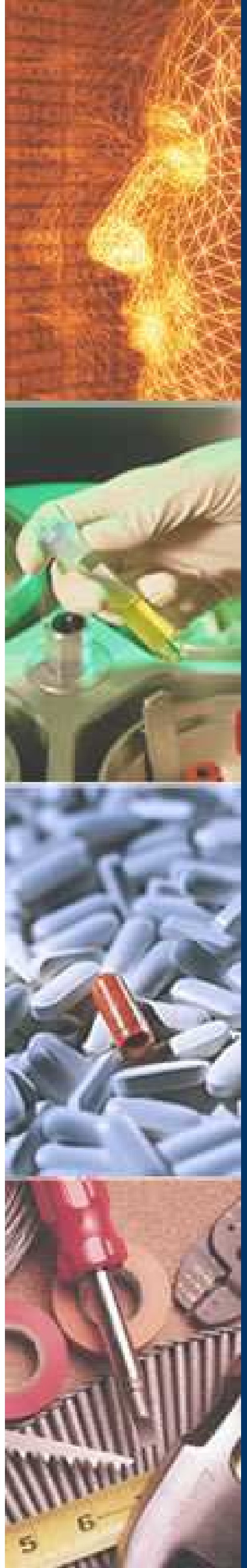
Advantages

- Quick creation of test scripts using Python
- Python has a smaller learning curve
- The data driven approach allows us to write new tests by adding data files that use the same generic code
- We allow for running a subset of the entire test suite using naming conventions
- Tailored to our requirements



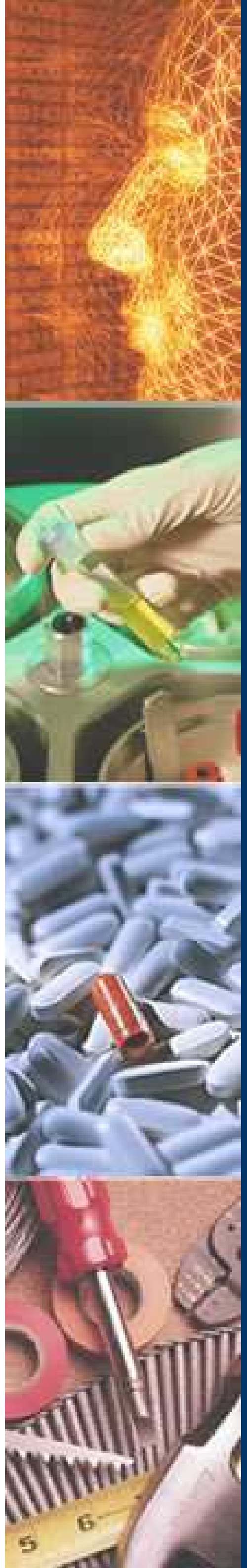
Drawbacks

- The data file format increases in complexity as the path to reach the data becomes complex
- Cannot test for parameters other than data efficiently



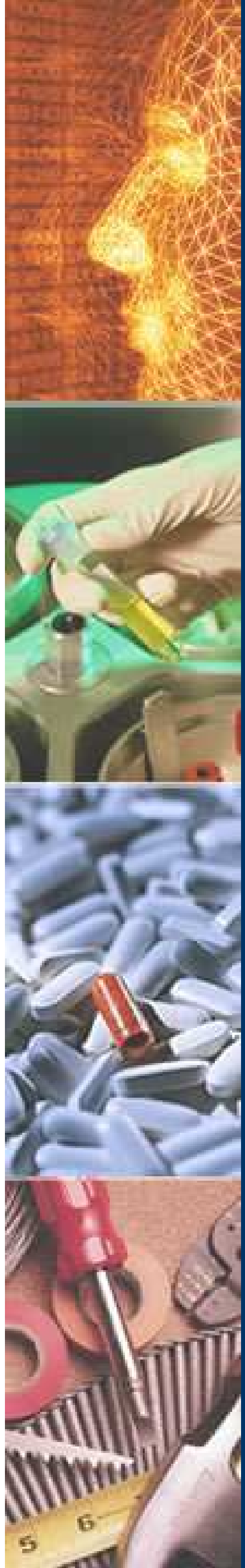
Future direction

- Extend the core engine so that enhancements can be made by changing the data format as well as by hooking in new classes
- Create an interface similar to Junit for status reporting
- Make it usable with CruiseControl



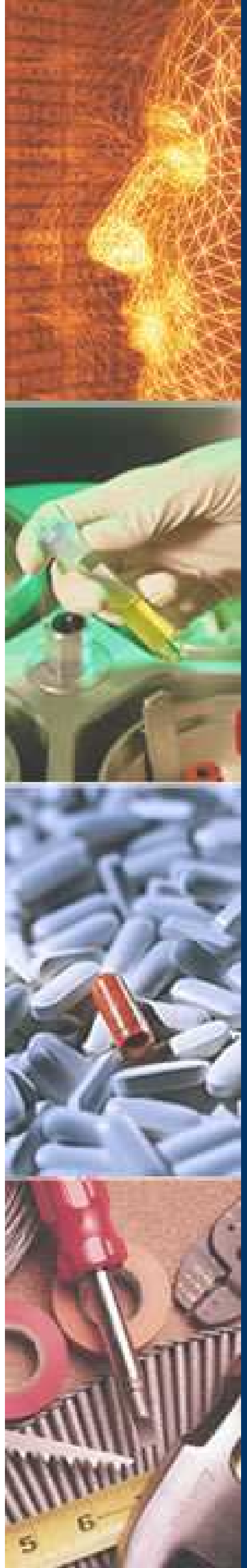
Agile Methodologies

- We used the agile technique to develop the Test Harness
- We made a quick good design and did the simplest thing that worked
- The design and code was refactored as more functionality was incorporated and it eventually matured into a framework



References

- FIT
- SuiteRunner
- Canoo Webtests
- HttpUnit
- Jython
- PharmQuest



Thank You

```
ERROR: syntaxerror
OFFENDING COMMAND: --nostringval--

STACK:

(PharmQuest’s Functional Testing Harness)
/Title
()
/Subject
(D:20050904170207)
/ModDate
()
/Keywords
(PDFCreator Version 0.8.0)
/Creator
(D:20050904170207)
/CreationDate
(njain)
/Author
-mark-
```