

Operational Frameworks for e-Business (Part 1)

Gary Murphy

Hilbert Computing, Inc.

glm@hilbertinc.com

913-780-5051

Frameworks vs. Class Libs

- Class Libraries provide reuse of behavior at a granular level.
 - Objects don't have coupled behavior
 - Most Sun Microsystems packages are class libraries
- Frameworks enable reuse at a higher level
 - Reuse without strong O-O skills
 - Objects are coupled to create smart

Frameworks vs. IDEs

- Frameworks are for use by programmers
 - Minimal OO experience is needed to use
 - Extending the framework requires considerable OO skills...
 - ...however, it *is* extensible
- IDEs may be used by non-programmers
 - Generally not extensible

Frameworks Motivation

- Infrastructure Programming is not being done.
 - Applications programmers work on applications, not infrastructure
 - Operational organizations generally don't have programmers.
- Bridge the complexity gap between servlets and proprietary web technologies

Hilbert Frameworks - Why?

- Rich features for operational support.
 - Logging (with future Tivoli, et. al. integration)
 - Performance Metrics
 - Service Level (uptime) reporting
 - Remote diagnostics
- Better operational support means your customers have a more available application.

Hilbert Frameworks - Why?

- Faster time-to-market
- Encourages a consistent design style
- Richer implementation of operational components, such as logging
- Some complexity of thread management (e.g. JDBC prepared statements) is handled automatically

Goals of a Framework

- Easy for programmers to use
 - Easy things are easy
 - Hard things are easier than coding at a class library level.
- Smart, default behavior.
 - Lot of function with no configuration
 - Variations can be set via properties/attributes

Frameworks in a Nutshell

- Logging (Part 1)
- Performance metrics (Part 1)
- Servlets (Part 2)
- JavaServer Pages™ “Widgets” (Part 2)
- Relational Database Access (Part 2)
- ... plus some base classes (Part 1)

Using Hilbert Frameworks

- This code is copyrighted by Hilbert Computing, Inc.
- Open-source, royalty-free license
- Not GNU “copy-left”. You don’t have to give back any code derived from this code.
- Don’t directly modify anything in the Hilbert packages.

Scope of this Session

- How to Use
 - What is available
 - Programming examples
 - Goals behind the code
- Internals
 - This info is not required to use the framework
 - Focus on this at Colorado Software Summit

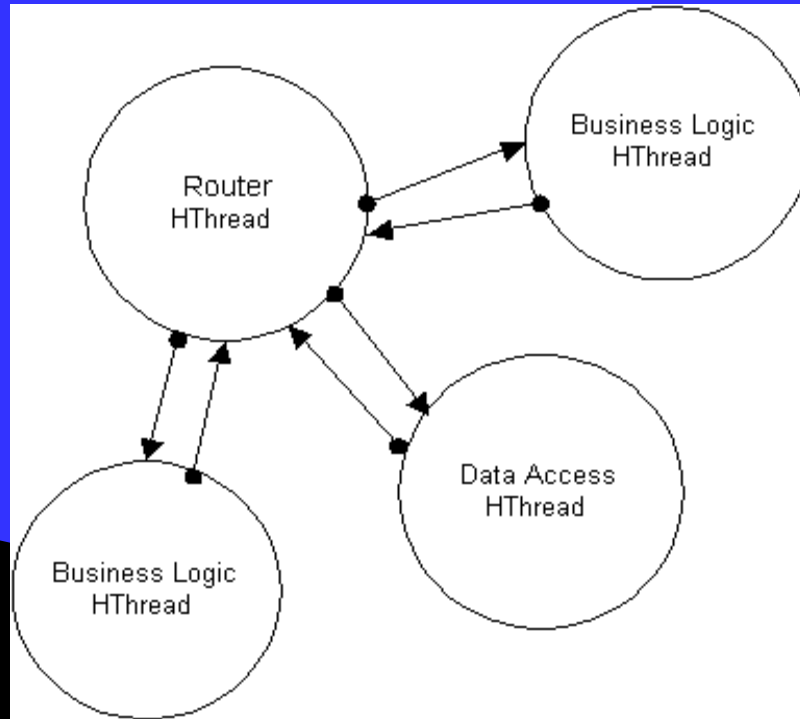
Hilbert Base Classes

- HThread - Asynchronous msg passing.
- HException - Wrappered exception.
 - Contains failing method name
 - Numeric code for legacy integration
 - Useful in handling exceptions on multiple threads and deferred exception handling

Base Classes - HThread

- HThread Supporting Classes
 - HWorkQue - This contains a Vector and implements a blocking FIFO queue that is synchronized across threads using wait()/notify().
 - HWorkItem - Wrappers an integer message id and an Object.

Message Passing



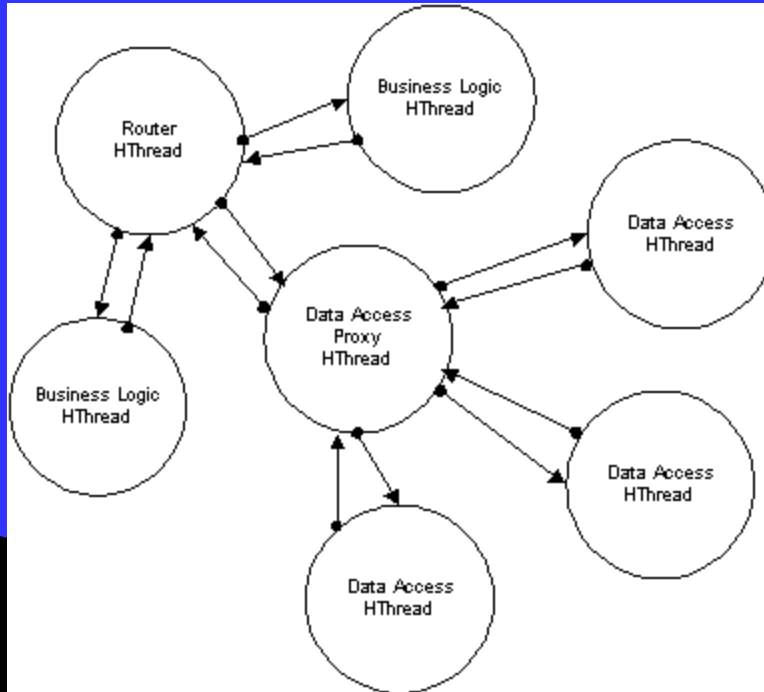
High Parallelism

No Critical Sections

No thread joins

Loose Coupling

Message Passing



Changed Thread Model

Increased Parallelism

No impact to application

Base Classes - HException

- `<soapbox>`
 - Exceptions should contain enough information to diagnose immediately
 - During development, before you fix the bug, add information to the exception then fix the bug.
- `</soapbox>`

Source Code

(new)

Descriptive exceptions:

```
HHtmlFormVisitor.handleElementOption(...)
```

```
HApplicationProperties.configure(...)
```

Asynchronicity

```
HThreadPool.run()
```

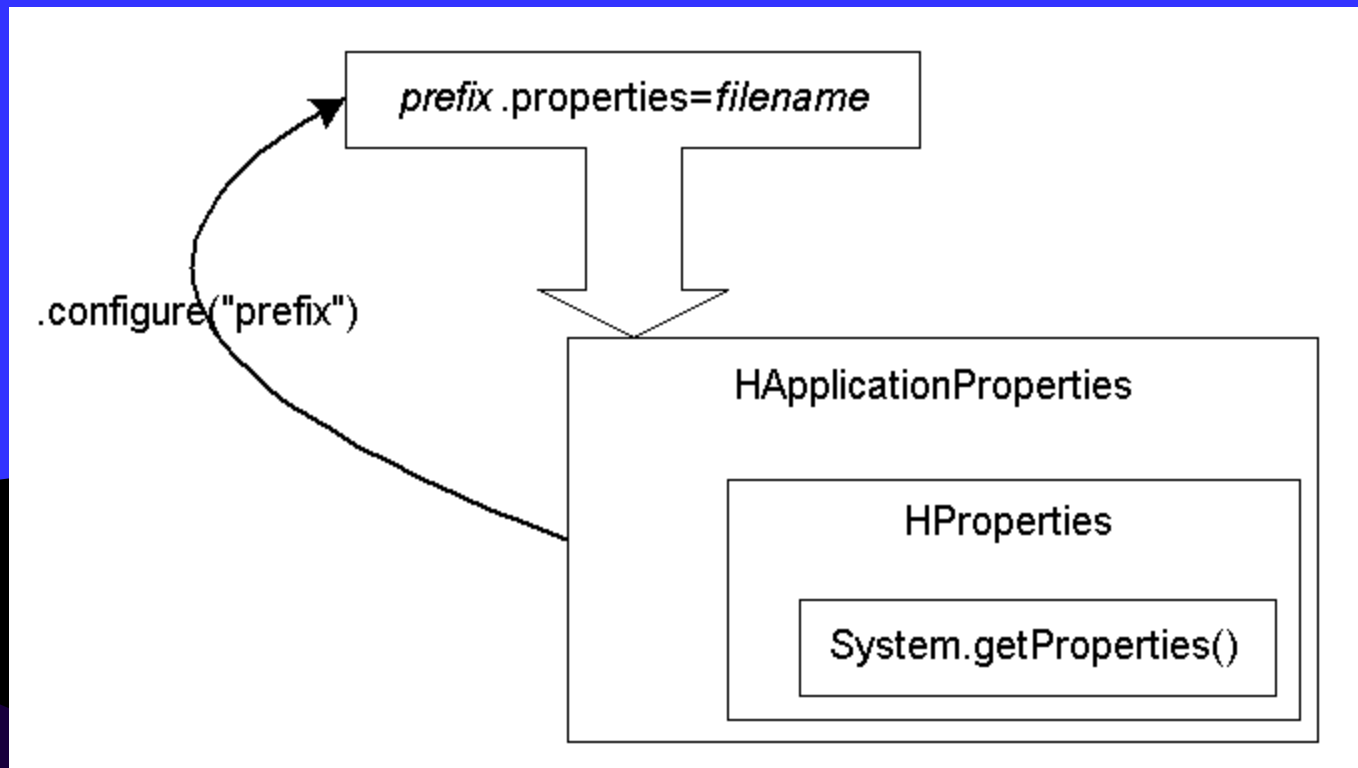
```
HThreadPoolThread.handleRunRequest()
```

Peter Haggag has a session on exception handling

Hilbert Base Classes

- HProperties/HApplicationProperties
 - Automatically load themselves from CLASSPATH
 - Layered into System properties
 - Configurable per application. Important since multiple servlets share JVMs

Base Classes - Properties



Properties Configuration

- properties contains key/value pairs from:
 - hilbert.properties
 - application.properties
 - demo.properties

```
HApplicationProperties properties =  
    new HApplicationProperties();  
properties.configure("demo");
```

Base Classes - Properties

- `'hilbert.properties'` and `'application.properties'` are maintained by the systems admin staff.
- `'application.properties'` points to the application configuration which is maintained by the programmer or application owner.
- The above allows for ACLs or filesystem permissions to be set appropriately.

Hilbert Base Classes

- HMailClient - Layers on JavaMail™

```
HMailClient client = new HMailClient(userid, password);  
Message message =  
    client.createMessage("glm@hilbertinc.com", "Subject");  
message.setText("This is the body");  
client.send(message);
```

Hilbert Base Classes

- HSymbolTable
 - Maps strings to integers and vice-versa
 - Useful for mapping POST/GET variables (e.g. “request” into the numeric equivalent for use in a switch statement)

Hilbert Security Classes

- This is **not** a new security application
 - Uses underlying authentication of services (e.g. database authentication)
 - A subject (aka “user”) contains credentials.
 - Each service uses credentials in a manner appropriate for its services
- Modelled after Sun JAAS v1.0
 - Will call JAAS services directly in Java 2

Logging Framework

- Must be simple for programmers to use

```
HLogger log = new HLogger("Demo");  
log.logInformation("Something happened");  
log.logException(exception);
```

- Don't slow the application execution

Logging - Design Points

- Multithreaded to not slow the application
 - Logged events are placed on another thread so your application continues to run
- Pluggable backend handler
 - The handling of the message is separate from the application.
 - Messages are passed as XML documents

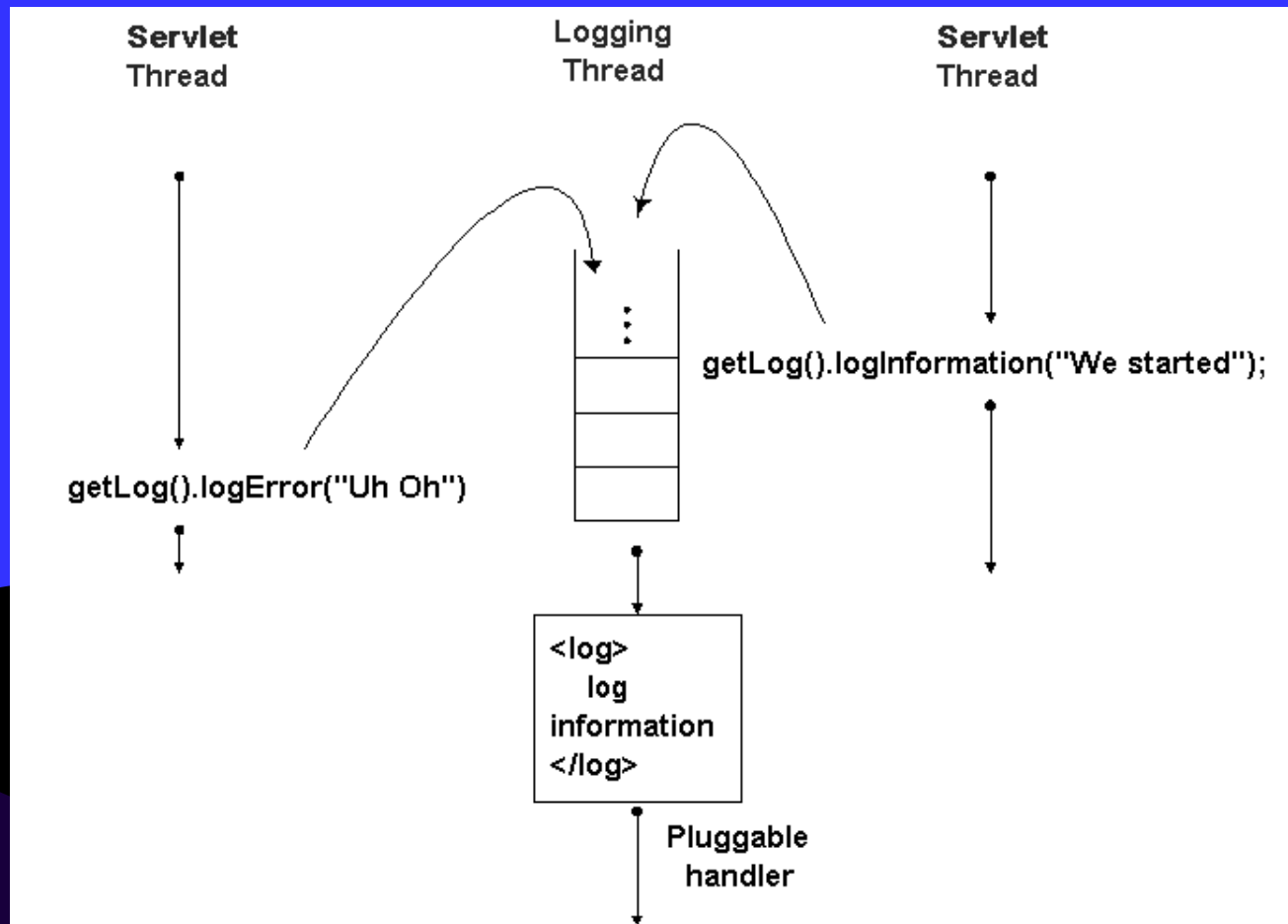
Logging - Design Points

- Self-metering
 - If a loaded system is logging at a rapid rate, the logging frameworks need to catch up.
 - Multiple log requests are packaged into a single XML document
- Flexible
 - There are 16 levels of logging in four categories: Info, Warning, Error, Exception

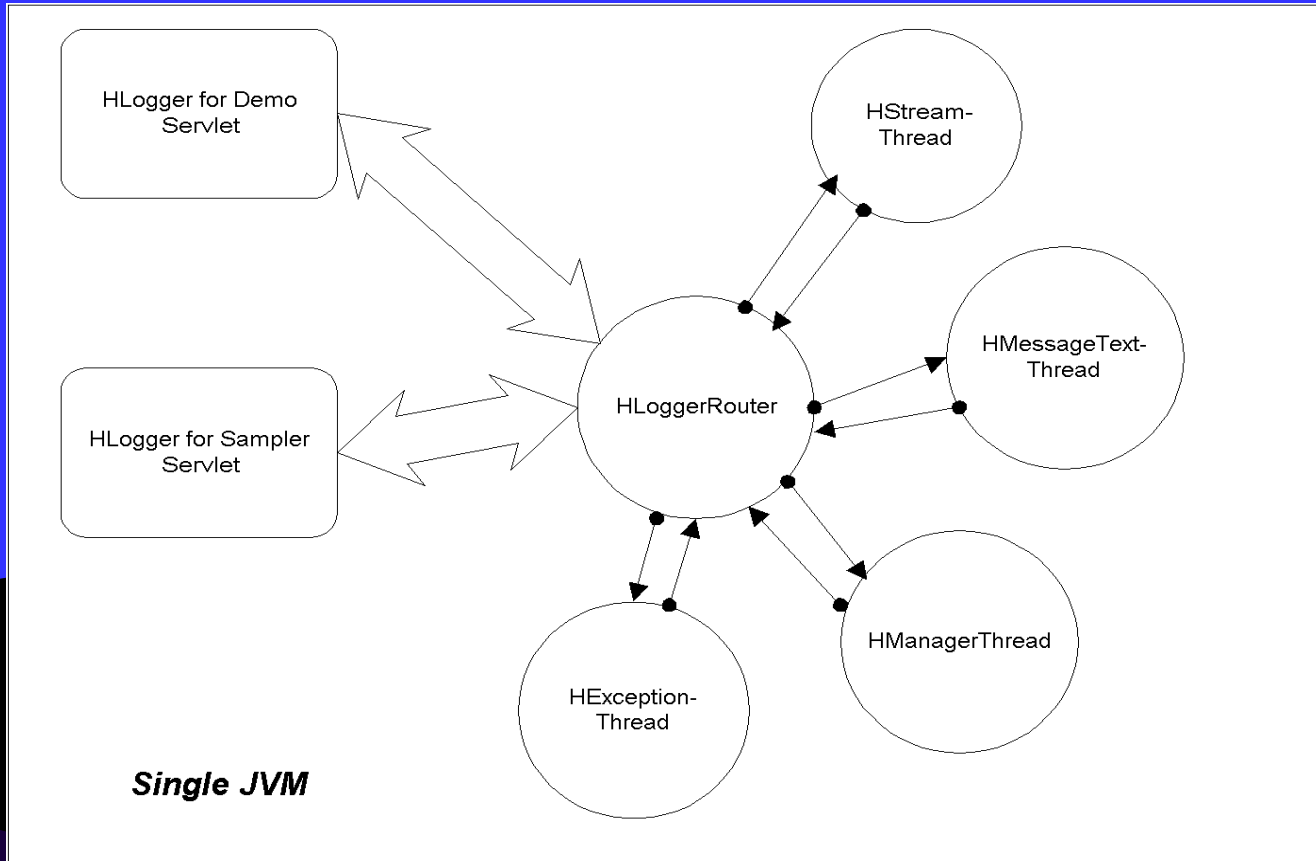
Logging - Implementation

- Uses asynchronous message passing for inter-thread communication.
- Interfaces:
 - HLoggerInterface - This interface defines the logging facilities.
 - HSegmentHandler - Interface for handling a log entry.

Logging - Internals



Logging Internals



Logging - Implementation

- Available classes:
 - HLogger provides all of the multithreaded handling described previously
 - HStderrSegmentHandler - write to stderr
 - HNetSegmentHandler - write to network
 - HHttpSegmentHandler - write to servlet
 - HLogMonitorSegmentHandler - Formatted for the Log Monitor application

Source Code

(new)

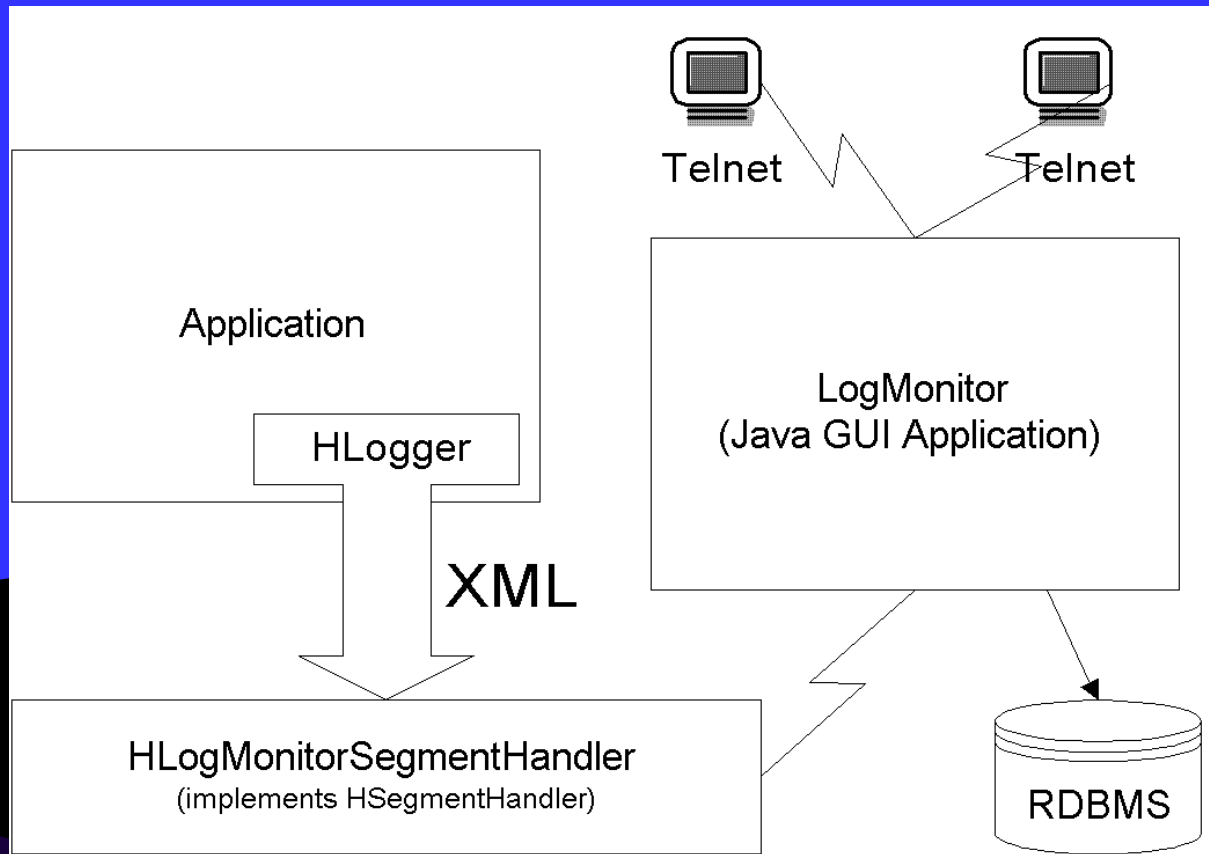
Log.xml

HDOMParser (delegates to XML4J)

HLogVisitor

See Kelvin Lawrence's presentation on processing XML for more on the subject.

Logging - Example Plugin



Demonstration Servlet

- See code.
- See code run.
- Run code. Run.

Metrics Framework

- Must be simple to use:

```
HMetrics metrics = new HMetrics();  
metrics.start();  
metrics.createTimingMetric("Test","run");  
metrics.createCounter("Test", "hits", 5);  
for (int i = 0; i < 25; ++i)  
    metrics.increment("hits");  
metrics.stop("runtime");
```

Metrics - Design Points

- Architecturally identical to the logging frameworks.
- Multithreaded, asynchronous message queueing to not slow the application
- Pluggable backend handler

Metrics - Implementation

- Supported metrics include:
 - Timers - Measures elapsed time
 - Counters - Counts
 - Concurrency - Counter with a maximum value
 - Rates - Counts over an elapsed time
- Framework associates no context or meaning to the metrics data itself

Metrics - Implementation

- Disposition is distinct from the metric itself
 - The disposition determines when I write the metric
 - Can be time-based or counter-based
- Let me explain...

Metrics - Implementation

- ... what really goes on

```
HMetrics metrics = new HMetrics();  
metrics.start();  
metrics.createTimingMetric("Test","run");  
metrics.createCounter("Test", "hits", 5);  
for (int i = 0; i < 25; ++i)  
    metrics.increment("hits");  
metrics.stop("run");
```

Metrics - Implementation

- HMetricsMonitorSegmentHandler
 - Separate application with telnet capability
 - Logs to a database
 - Pluggable metrics handlers on a per-application basis since the metrics frameworks associates no context to the metric.