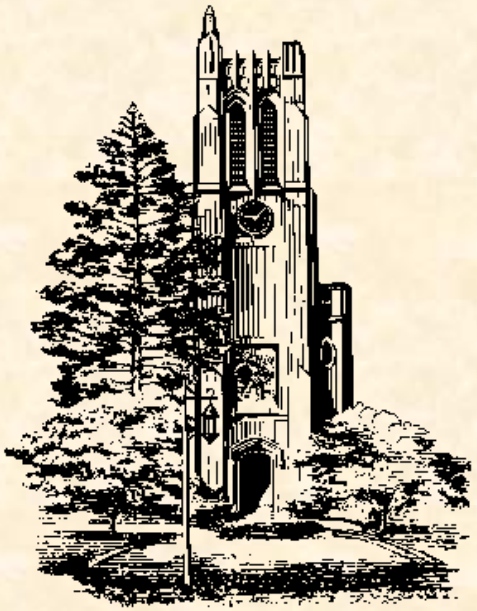
MICHIGAN STATE UNIVERSITY

Technical Specification / Schedule
IBM POWER Hypervisor hcall Test Suite



Team #4: IBM

CSE 498, Collaborative Design

Scott Judd
Jordan Fish
Joshua Roys

Department of Computer Science and Engineering
Michigan State University

Spring 08

Project Overview

- Hypervisor – A layer of code always memory resident in PPC Machines
- Mediates hardware access through hcalls
- Hcall => Hypervisor as syscall => OS kernel
- Hypervisor Layer allows multiple OS images (partitions) to share a single physical system
- Assertion – no partition can effect another
- Our goal is to design an intelligent way to test the Hypervisor for cases in which a partition effects another or causes system failure

Team #4: IBM

Functional Specifications

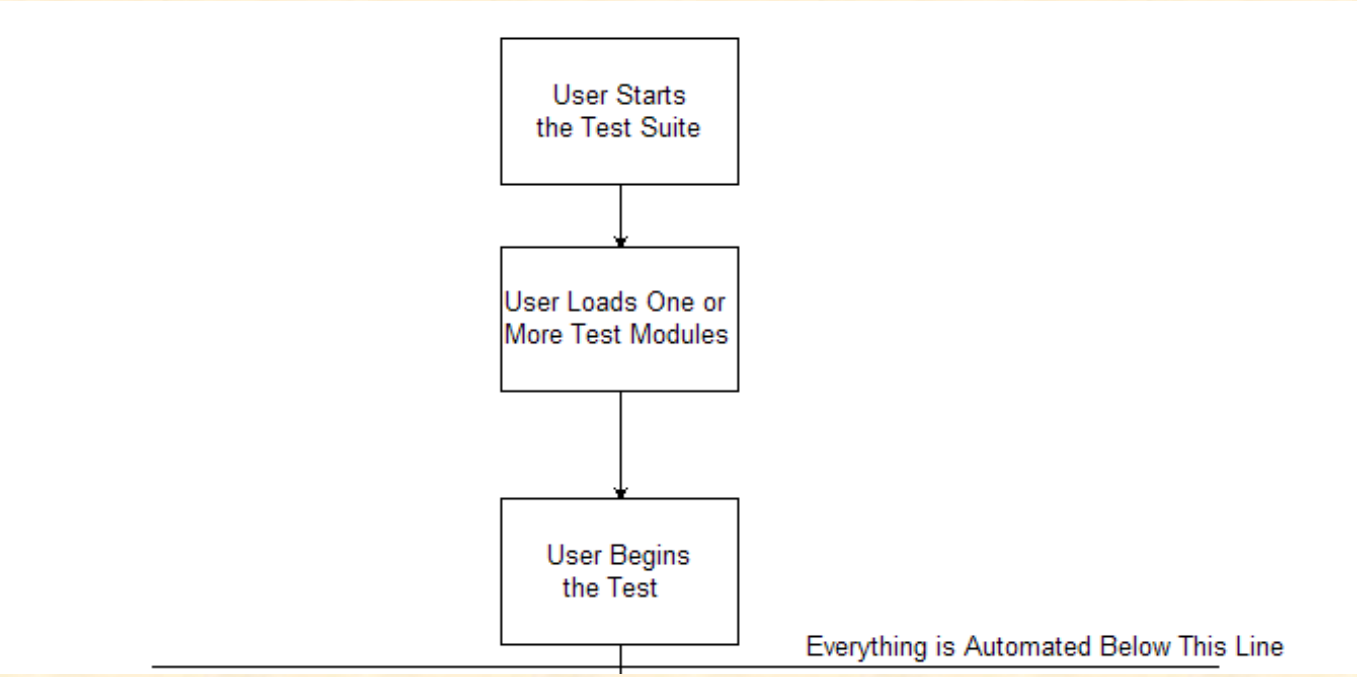
- Make Hypervisor Calls
- Retrieve Results
- Break tests in to modules
- Modules to test all hcalls (~100)
- Graphical reports of results
- Report of how thorough the tests are

Team #4: IBM

System Components

- Hardware Platforms
  - Linux Server at IBM (Testing Platform)
  - Linux HMC Machine at IBM (Creates Partitions)
  - AIX Machine at IBM (TestHarness)
  - Linux Machine at MSU (Programming Platform)
- Software Platforms / Technologies
  - Hypervisor Layer
  - TestHarness (composed of Perl Scripts)
  - Graphical User Interface (written in Java)

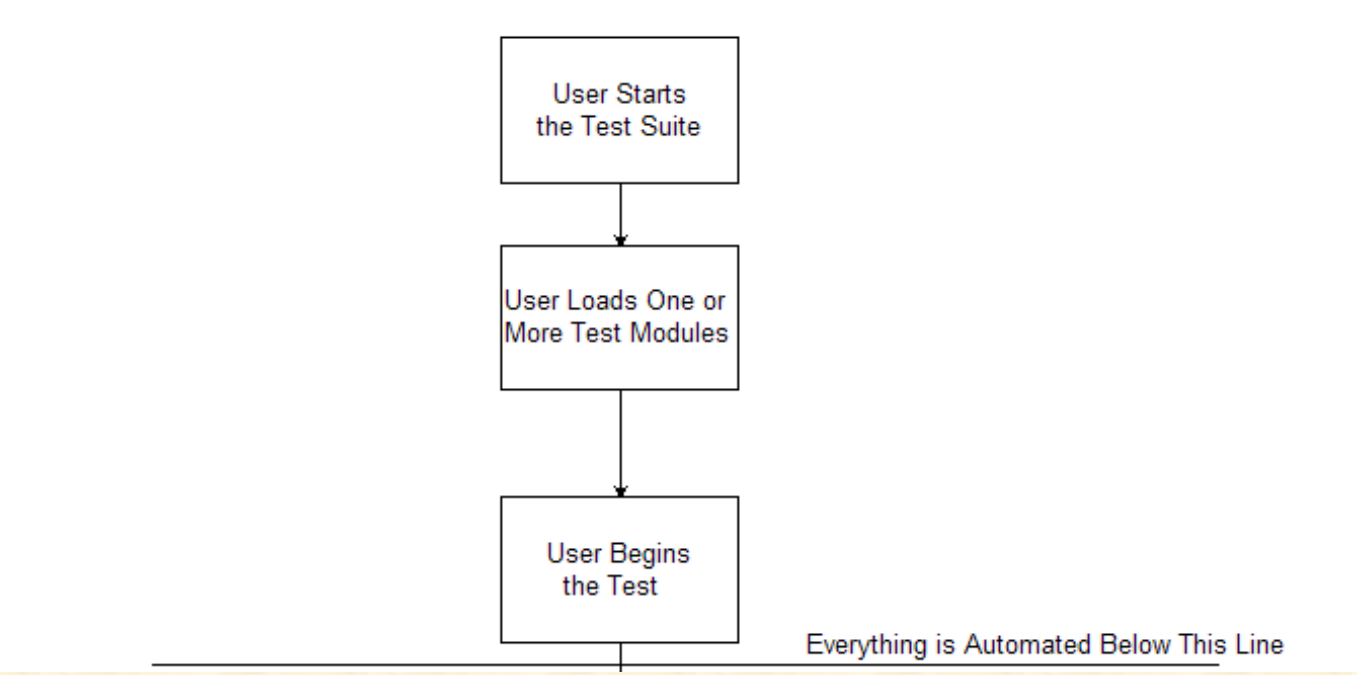
Use Case Flowchart



This is all the user does
Everything afterwards is automated
(Continued on Next Slide)

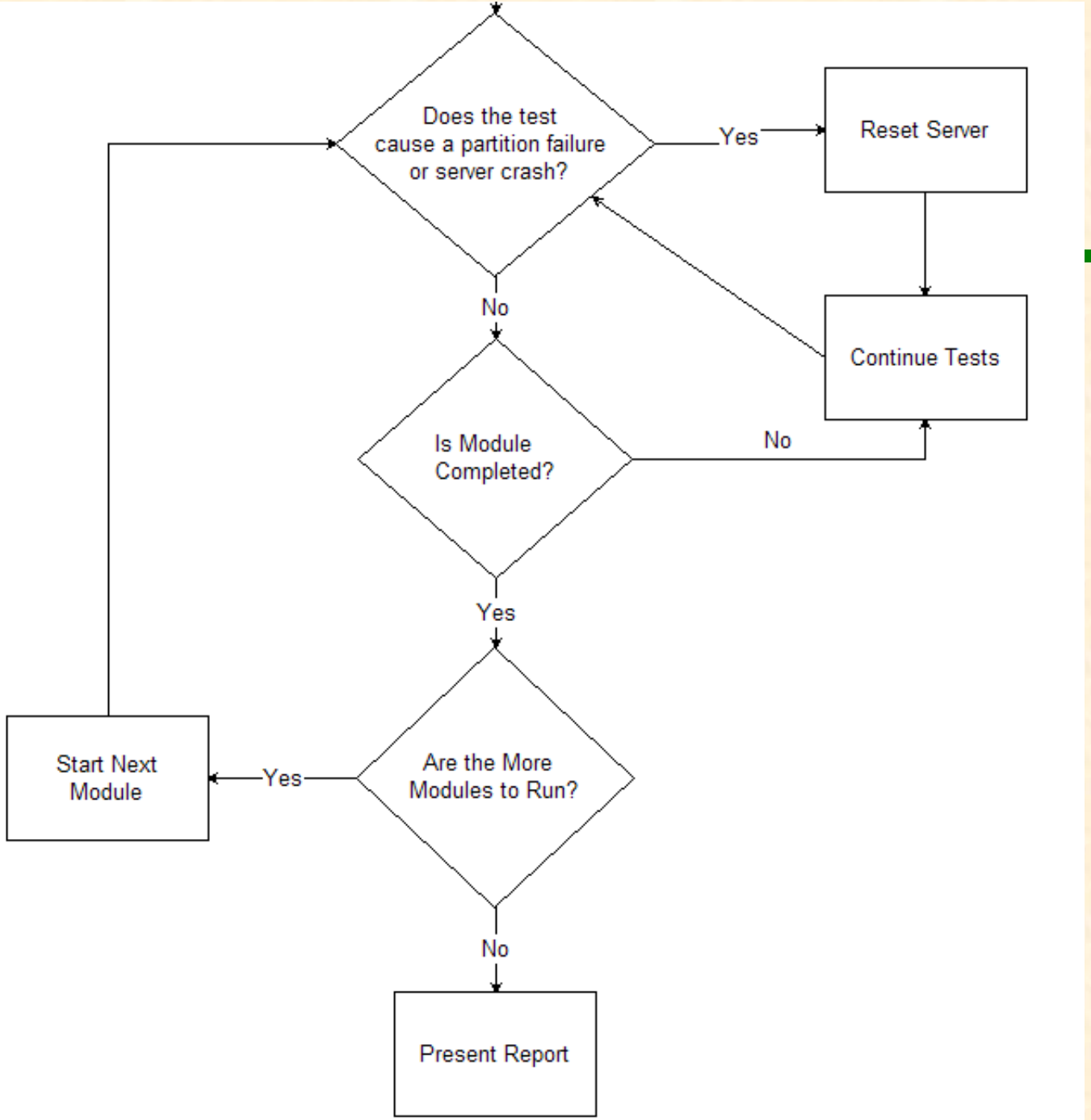
Team #4: IBM

Use Case Flowchart



This is all the user does
Everything afterwards is automated

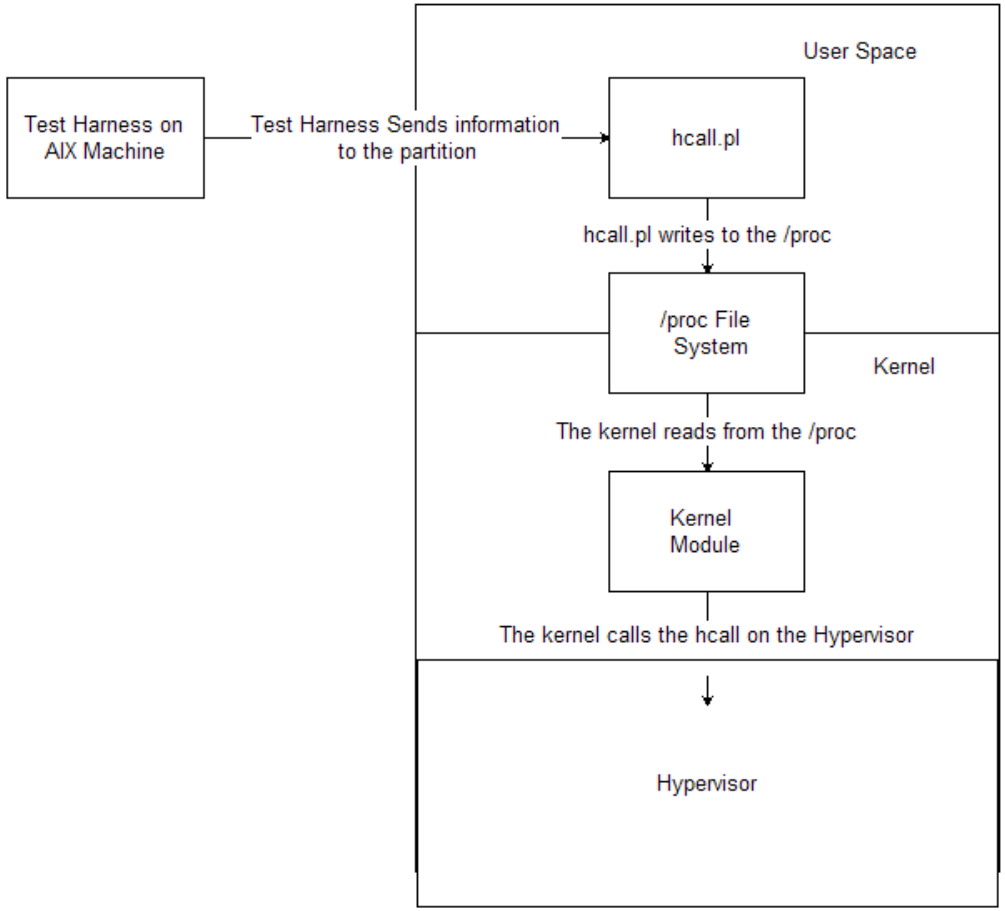
Team #4: IBM


Does the test cause a partition failure or server crash?
Yes
Reset Server
No
Continue Tests
Is Module Completed?
No
Yes
Are the More Modules to Run?
Yes
Start Next Module
No
Present Report

Team #4: IBM

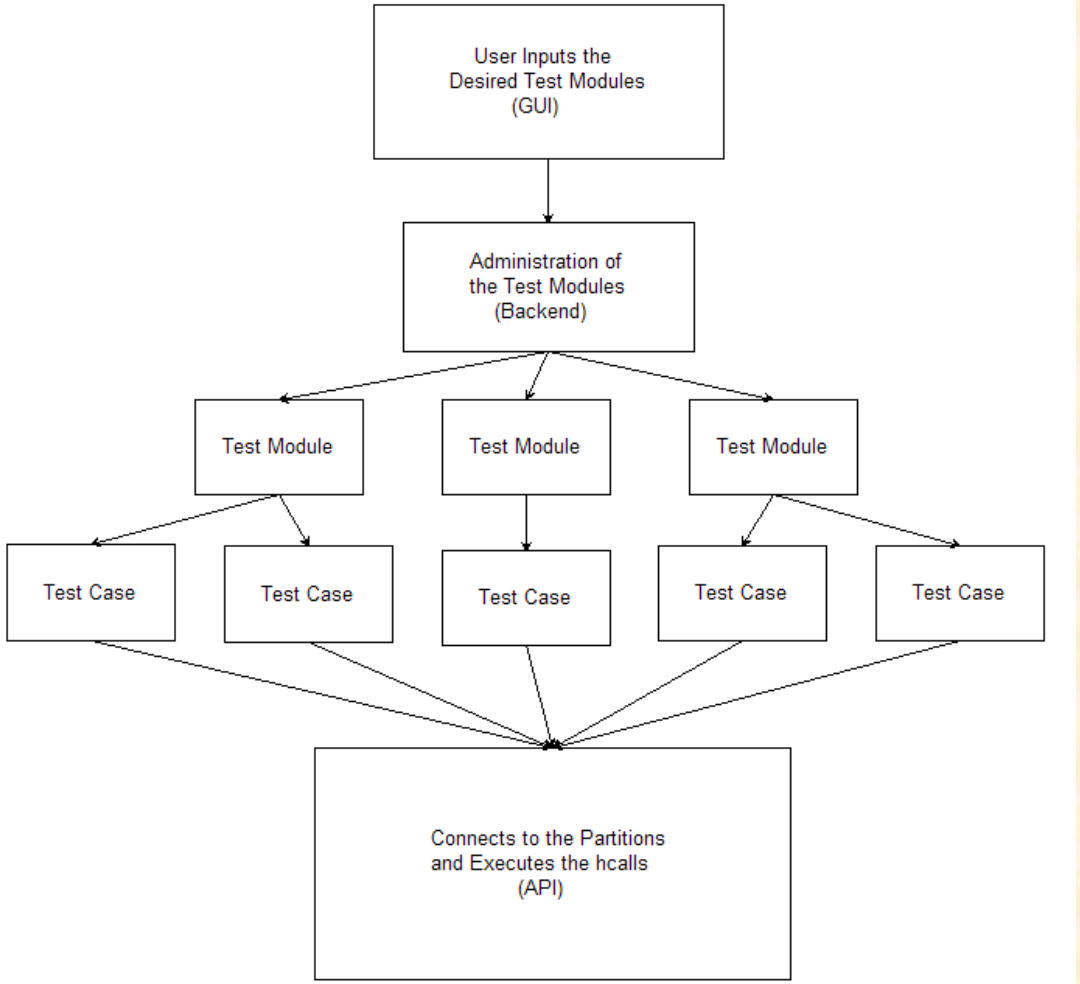
Architecture Illustrated (Hcalls made from a partition)

- The Test Harness executes hcall.pl remotely in the partition
- hcall.pl writes to the /proc File System
- The Kernel Module makes the hcall stored in /proc and places the results back in the file system.
- hcall.pl returns the result to the Test Harness



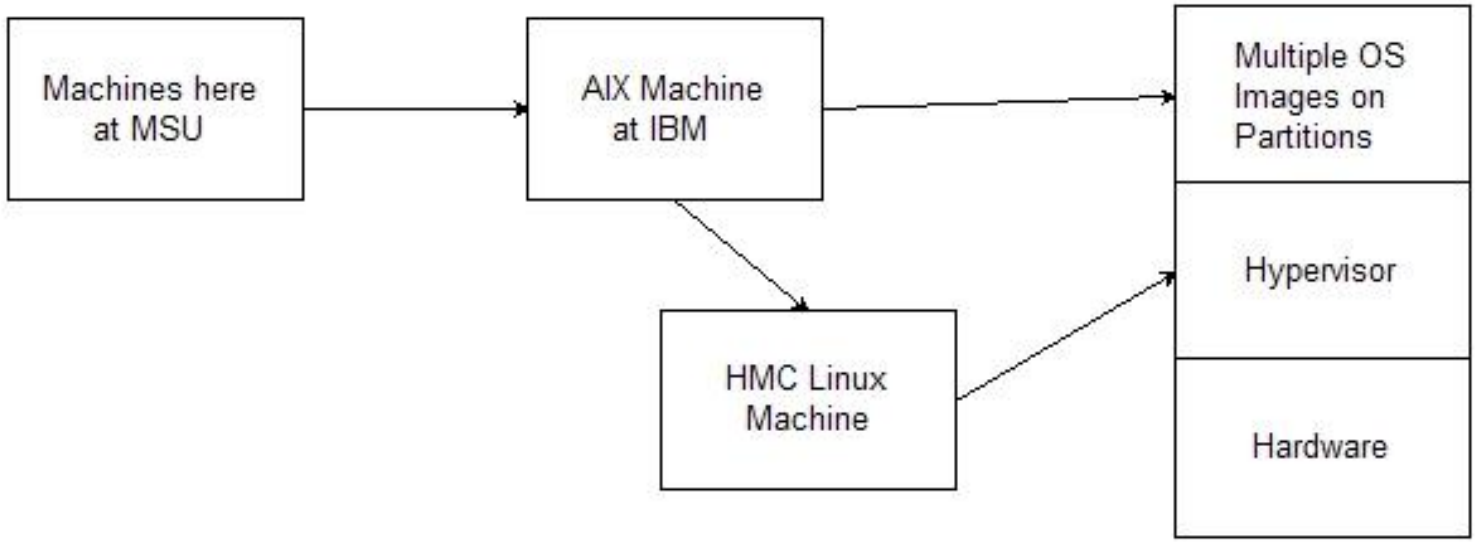
Team #4: IBM

Architecture Illustrated (Test Harness Design)



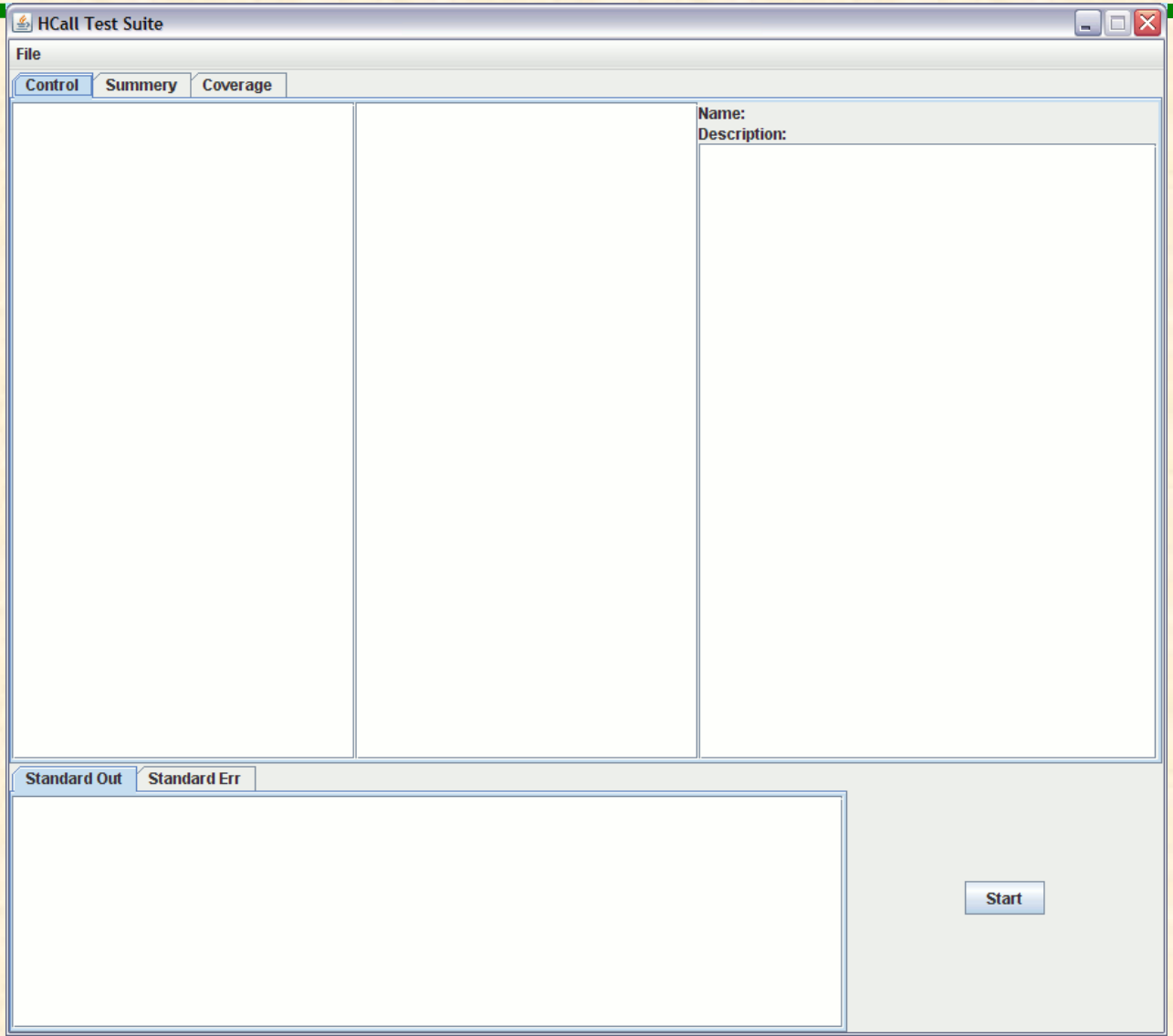
Team #4: IBM

Architecture Illustrated (How the Hardware Connects)



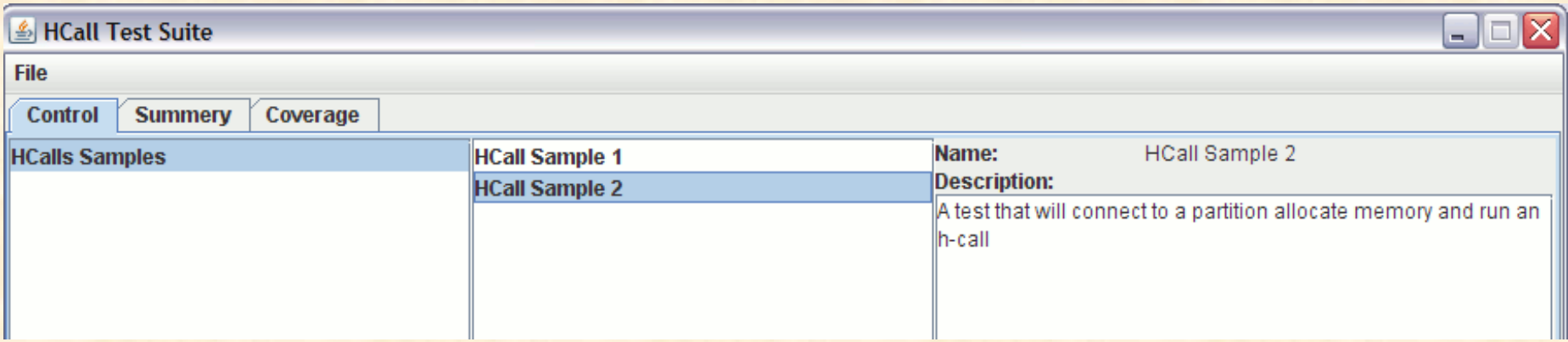
Team #4: IBM

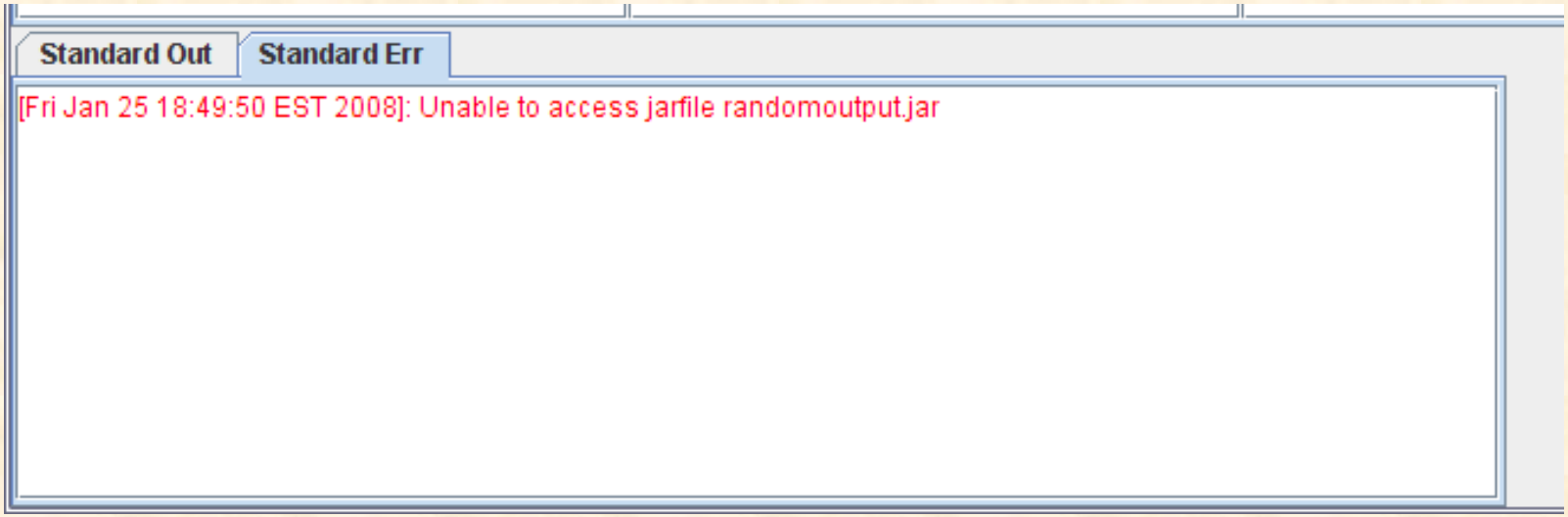
Architecture Illustrated (GUI)



Team #4: IBM

Architecture Illustrated (GUI)





Team #4: IBM

Risks

- Perl
  - Test Suite Written in Perl
  - Learn Perl
- IBM
  - Long distance and slow response times
  - A lot of contact with firm and concise times
- Hypervisor Technology
  - Hypervisor only runs on PPC systems
  - Use an IBM server
- Developing Pseudo-Random Test Cases
  - Can not test all possible hcall parameters
  - Produce a set of logical tests cases

Project Schedule

- January 23:
  - Test Module Prototype Compete
- January 28:
  - First Prototype Test Module Written
- January 30:
  - First Prototype Test Module Working as Intended
- February 1:
  - Hcall Modules Complete
- February 4:
  - Extending Autoharness Partition Failure Detection Begins
- February 6:
  - All Modules Framed

Team #4: IBM

Project Schedule

- February 8:
  - One Working Alpha Module Completed Apiece
- February 11:
  - Alpha GUI Producing Meaningful Output
- February 13:
  - Half of all Test Modules Complete
- February 15:
  - Alpha Presentation Complete
- February 15:
  - Autoharness Partition Failure Detection Complete

Project Schedule

- February 18:
  - Alpha Presentation
- February 20:
  - All Test Modules Working for Random Input
- February 27:
  - Pseudo Random Test Case Development Begins
- February 29:
  - Narrow Down Pseudo Random Heuristic to Three Candiates
- March 3:
  - Miscellaneous Modules Implemented with Pseudo Random Heuristic

Project Schedule

- March 3-7:
  - Project Report Preparation
- March 5:
  - GUI Report Generation Begins
- March 7:
  - Pass/Fail Report Complete
- March 10:
  - Project Report Due
- March 10:
  - Working Pseudo Random Heuristic Implementation
- March 12:
  - Partition Failure Report Complete

Project Schedule

- March 15:
  - Finalized Pseudo Random Heuristic
- March 17:
  - Beta Presentation of GUI Frontend with Report Generation Complete
- March 21:
  - Coverage Report Generation Complete
- March 24:
  - Test Modules Moved to Pseudo Random Heuristic Phase 1
- March 26:
  - Test Modules Moved to Pseudo Random Heuristic Phase 2

Project Schedule

- March 28:
  - Test Modules Moved to Pseudo Random Heuristic Phase 3
- March 31:
  - Pseudo Random Heuristic Implementation Complete
- April 4:
  - Release Candidate Complete
- April 7:
  - Final Presentation Design Begins
- April 7-13:
  - Project Report Preparation

Project Schedule

- April 14:
  - Project Report Due
- April 16:
  - Video Finished
- April 16:
  - Design Day Preparation Begins
- April 21:
  - Project Video Due
- April 21-23:
  - Design Day Rehearsals
- April 24:
  - Design Day