Distributed Control Lab

Virtual Lab 2004
Setúbal, Portugal
Andreas Rasche, Bernhard Rabe, Peter Tröger, and Andreas Polze
Outline

- Motivation / Overview
- The Distributed Control Lab (DCL) Architecture
- Protecting the Lab from Malicious Code
- Experiments in the DCL
  - Foucault’ Pendulum
  - Higher Striker
- DCL - Grid Computing Integration
- Conclusions
Motivation

- Online access to physical experiments over the Web
- Test-bed for interconnected middleware-components and embedded systems
- Reach a predictable system behaviour in unstable environments
- Study techniques to prevent malicious code damaging physical equipment
- Foucault's Pendulum demonstrates usage of dynamic reconfiguration for online replacement of user control
Distributed Control Lab

- 2001 project start at Hasso-Plattner-Institute
- Practice of writing control algorithms for real-time control problems
- Study of system predictability, availability and security in context of middleware-based dynamic control systems
- Extensible architecture for hosting physical control experiments
- Investigation of algorithms for user code observation and replacement of control components
- Experiment: physical installation and specific control software
The Distributed Control Lab
## My Jobs

The following jobs are registered for your account in the system:

<table>
<thead>
<tr>
<th>Name</th>
<th>Location</th>
<th>State</th>
<th>Queued on</th>
<th>View details</th>
<th>Delete</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pendulum</td>
<td>HPI</td>
<td>Currently executed</td>
<td>Tue, 23 Sep 2003 10:51:27 GMT</td>
<td>View details</td>
<td>Cancel</td>
</tr>
<tr>
<td>Lego_Robots</td>
<td>HPI</td>
<td>Finished (result code: StopSignal), 3 results available</td>
<td>Fri, 19 Sep 2003 16:59:36 GMT</td>
<td>View details</td>
<td>Delete</td>
</tr>
<tr>
<td>Lego_Robots</td>
<td>HPI</td>
<td>Finished (result code: StopSignal), 3 results available</td>
<td>Fri, 19 Sep 2003 14:33:33 GMT</td>
<td>View details</td>
<td>Delete</td>
</tr>
<tr>
<td>Lego_Simulator</td>
<td>HPI</td>
<td>Finished (result code: Successful), 4 results available</td>
<td>Fri, 19 Sep 2003 12:45:32 GMT</td>
<td>View details</td>
<td>Delete</td>
</tr>
<tr>
<td>Lego_Robots</td>
<td>HPI</td>
<td>Finished (result code: StopSignal), 3 results available</td>
<td>Fri, 19 Sep 2003 11:18:49 GMT</td>
<td>View details</td>
<td>Delete</td>
</tr>
<tr>
<td>Lego_Robots</td>
<td>HPI</td>
<td>Finished (result code: StopSignal), 3 results available</td>
<td>Thu, 18 Sep 2003 16:28:56 GMT</td>
<td>View details</td>
<td>Delete</td>
</tr>
<tr>
<td>Lego_Robots</td>
<td>HPI</td>
<td>FinishExperiment crashed</td>
<td>Thu, 18 Sep 2003 16:27:35 GMT</td>
<td>View details</td>
<td>Delete</td>
</tr>
<tr>
<td>Lego_Robots</td>
<td>HPI</td>
<td>Finished (result code: StopSignal), 3 results available</td>
<td>Thu, 18 Sep 2003 16:22:24 GMT</td>
<td>View details</td>
<td>Delete</td>
</tr>
<tr>
<td>Lego_Robots</td>
<td>HPI</td>
<td>Finished (result code: Failed), 2 results available</td>
<td>Thu, 18 Sep 2003 16:20:50 GMT</td>
<td>View details</td>
<td>Delete</td>
</tr>
<tr>
<td>Lego_Robots</td>
<td>HPI</td>
<td>Finished (result code: StopSignal), 3 results available</td>
<td>Thu, 18 Sep 2003 16:15:58 GMT</td>
<td>View details</td>
<td>Delete</td>
</tr>
</tbody>
</table>
Research Focus: Problem: Malicious Code

- Investigation of Solution for malicious code detection
  - Source Code Analysis
  - Language limitations / special compiler
  - Simulation before execution on physical experiment
  - **Analytic Redundancy**
    - Online observation of user programs
    - Replacement of user programs before reach of uncontrollable state
    - Dynamic Reconfiguration of component-based control application
    - Monitoring of environmental settings and component states
Foucault’s Pendulum

- Demonstrates earth rotation – popular experiment
- Today many installation including one in UN-building in New York
- Problem: Pendulum must be kept swinging
- Solution: electro magnet
  under an iron ball
- **Experiment:** Find best control algorithm to keep the pendulum swinging
  - Using minimum energy
  - Reaching the highest amplitude
The Pendulum Experiment

You can enter here the program to steer the magnet, which is situated under the pendulum. The necessary programming details are explained in this documentation (german).

You can use one of the following code examples:

```java
while(true) {
    se=pendel.getNext();
    // New Event?
    if(se!=null) {
        // First time at this place?
        if(last==null) last=se;
        // Rugel tritt ein
```

Upload your code file: [Browse] [Upload]

Start Job
```cpp
long period=0,ia=0;
SimEvent serial;
SimEvent[] list=new SimEvent[2];
bool preprocess = false;
in state = 0;

int node=0;
in magnetostatic=0;

double s_k

... [incomplete text, use download for full version]
```

**State Flow**

**Measurement result**

![Graph showing measurement results with annotations](image)

You can use the download link for viewing or saving the source data of this diagram.
Pendulum Experiment Control Configurations

Configuration 1: safety controller

Configuration 2: user program (cold standby)

Configuration 3: user program (warm standby)
Measurements:
Abnormal Termination of User Program
Our Approach: Dynamic Reconfiguration as safe-guard mechanism

- Mapping of profiles to application configurations based on environmental conditions and component states
- XML-based description of
  - observer: monitoring of environmental settings and component states
  - profiles: mapping of environmental conditions to application configurations
  - configurations of component-based applications
- Online monitoring of environment and components
- Change of application configuration using dynamic reconfiguration if required
- Configuration Manager instantiates, sets attributes, connects and starts components – performs reconfiguration
- Dynamic reconfiguration based on XML-based configuration description using an algorithm of M. Wermelinger
  - Based on blocking of connections between components
"Higher Striker" - Experiment

- Real Time Control experiment
  - Parallel I/O / 38 kHz sample rate / 256 Byte buffer
- Use of Real Time OS
  - Smaller Buffers, Higher Sampling Frequency
  - Short control delay
  - COTS x86 PC
    - Intel Celeron 633 MHz, 128 MB RAM (max 64 MB usable)
    - 10 Mbit/s LAN (NE 2000 PCI)
- Combination of non-RT .Net and RT application
- CE-PC Windows Ce.Net 4.2
DCL - Grid Integration

- Heterogeneous
  - X86, Itanium, PowerPC
  - Windows 2000/XP, Linux, Mac OS X

- DRMAA – Job Submission and Control for Clusters and Grids

- GLOBUS

- IDLE-Time
  - Condor
  - Sun Grid Engine, Condor
Conclusions

- DCL: environment for remote experiment access based on COTS Operating System and Middleware
- Experiences in teaching real-time embedded systems
- Safety against malicious code demonstrated
- Analytic Redundancy / Runtime observation of user control at Foucault’s Pendulum applicable
- Replacement of faulty control algorithms using dynamic reconfiguration
- Usage of Grid Computing Technologies to increase lab performance