Dynamic Reconfiguration as Safeguard Mechanism in the Distributed Control Lab

Workshop: Fault Tolerant Systems in Hard- and Software

Dipl. Inf. Andreas Rasche

Outline

- The Distributed Control Lab
  - Motivation, Architecture
- Foucault's Pendulum Details
- Dynamic Reconfiguration as safe-guard mechanism
- Evaluation of the approach
- Other Experiments in the DCL
- Conclusions

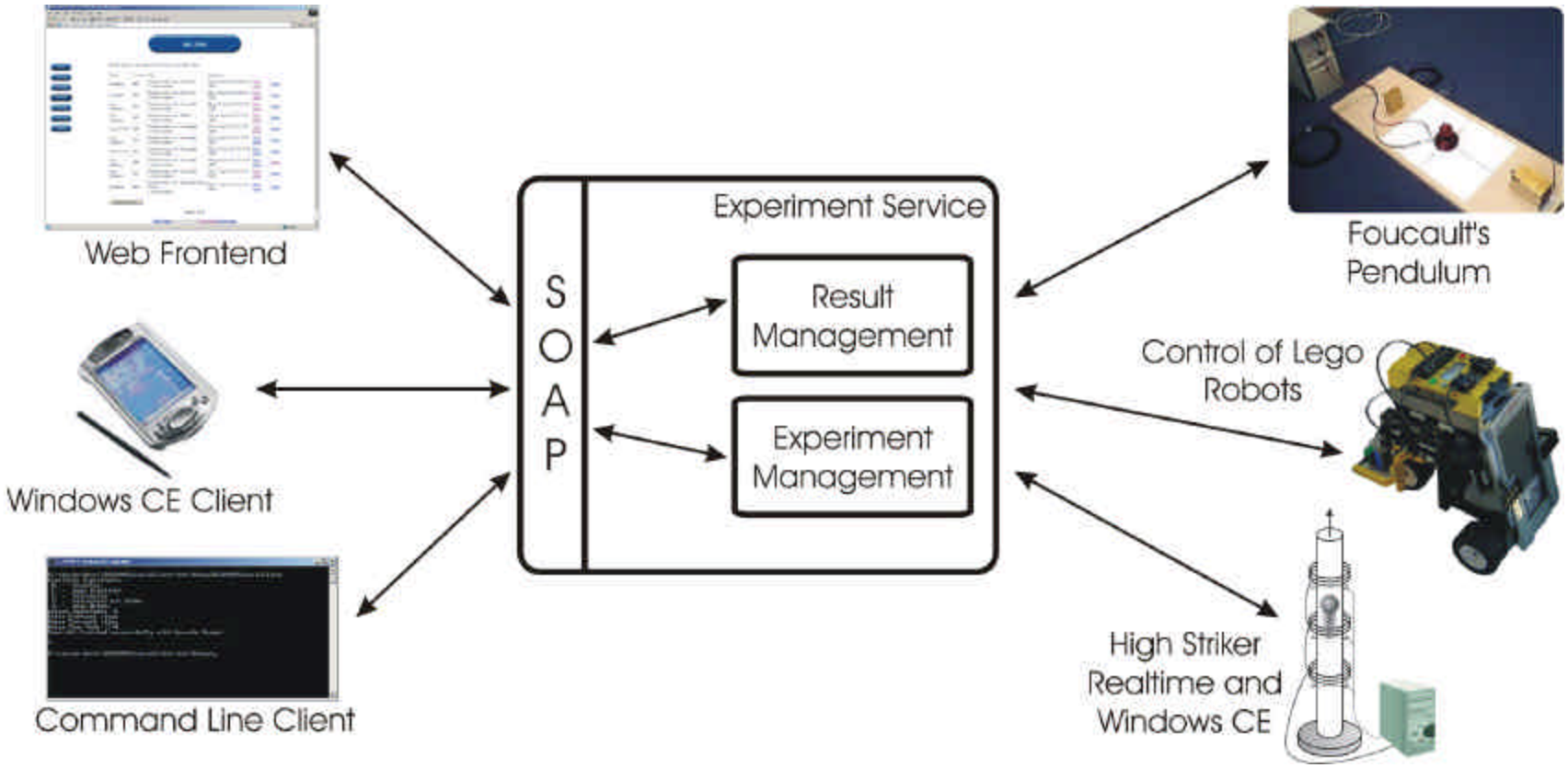
Distributed Control Lab

- 2001 project start at Hasso-Plattner-Institute
- Online access to physical experiments over the Web
- Practice of writing control algorithms for real-time control problems
- 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

Distributed Control Lab Motivation

- study of system predictability, availability and security in context of middleware-based dynamic control systems
- Reach a predictable system behaviour in unstable environments
- Study techniques to prevent malicious code damaging physical equipment
  - Source code analysis
  - Compiler / Language based
  - Simulation
  - Online observation / Component replacement
- Foucault's Pendulum demonstrates usage of dynamic reconfiguration for online replacement of user control

The Distributed Control Lab



My Jobs

Home
Experiments
Live Video
My Jobs
My Settings
Latest News
Logout

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

Name	Location	State	Queued on		
Pendulum (Owner : Andreas Rasche)	HPI	Currently executed ...	Tue, 23 Sep 2003 10:51:27 GMT	View details	Cancel
Lego_Robots (Owner : Stefan Henze)	HPI	Finished (result code : StopSignal), 3 results available	Fri, 19 Sep 2003 16:59:36 GMT	View details	Delete
Lego_Robots (Owner : Lego Test Account)	HPI	Finished (result code : StopSignal), 3 results available	Fri, 19 Sep 2003 14:33:33 GMT	View details	Delete
Lego Simulator (Owner : Andreas Rasche)	HPI	Finished (result code : Successful), 4 results available	Fri, 19 Sep 2003 12:45:32 GMT	View details	Delete
Lego_Robots (Owner : Stefan Henze)	HPI	Finished (result code : StopSignal), 3 results available	Fri, 19 Sep 2003 11:18:49 GMT	View details	Delete
Lego_Robots (Owner : Andreas Rasche)	HPI	Finished (result code : StopSignal), 3 results available	Thu, 18 Sep 2003 16:28:56 GMT	View details	Delete
Lego_Robots (Owner : Andreas Rasche)	HPI	Finish_Experiment crashed	Thu, 18 Sep 2003 16:27:35 GMT	View details	Delete
Lego_Robots (Owner : Andreas Rasche)	HPI	Finished (result code : StopSignal), 3 results available	Thu, 18 Sep 2003 16:22:24 GMT	View details	Delete
Lego_Robots (Owner : Andreas Rasche)	HPI	Finished (result code : Failed), 2 results available	Thu, 18 Sep 2003 16:20:50 GMT	View details	Delete
Lego_Robots (Owner : Andreas Rasche)	HPI	Finished (result code : StopSignal), 3 results available	Thu, 18 Sep 2003 16:15:58 GMT	View details	Delete

Delete all entries

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

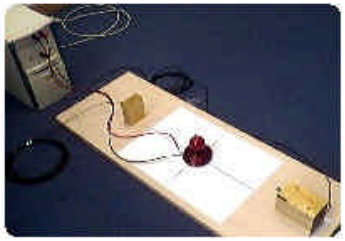
Foucault's Pendulum

- First installation 1848 by Leon Foucault in the Pantheon in Paris
- Demonstrates earth rotation
- 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

Experiments

Home
Experiments
Live Video
My Jobs
My Settings
Latest News
Logout

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 :

```
while(true)
{
        // Peak for Next Event
        se=pendel.GetNext();
        // New Event ?
        if(se!=null)
        {
                        // First time at this place ?
                        if(last==null) last=se;
                        // Kugel tritt ein
```

Upload your code file : Browse... Upload

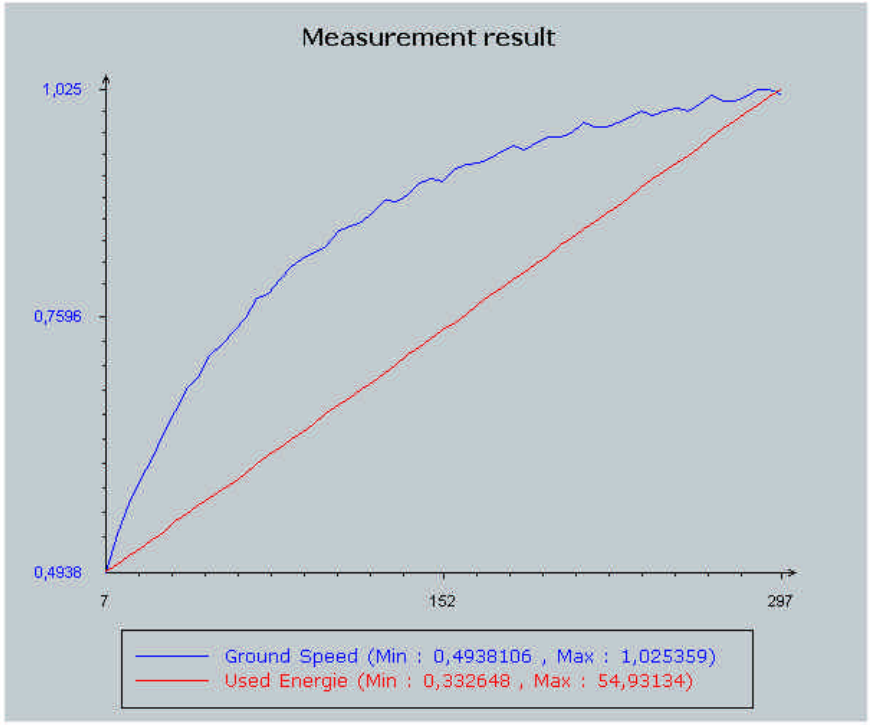
Start Job

back to top

```
long tperiod=0,tdi=0;
SimEvent se=null;
SimEvent[] last= new SimEvent[2];

bool setevent = false;
int state = 0;

int mode=0;
int magnetontime=0;

double s_k
```

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

State Flow

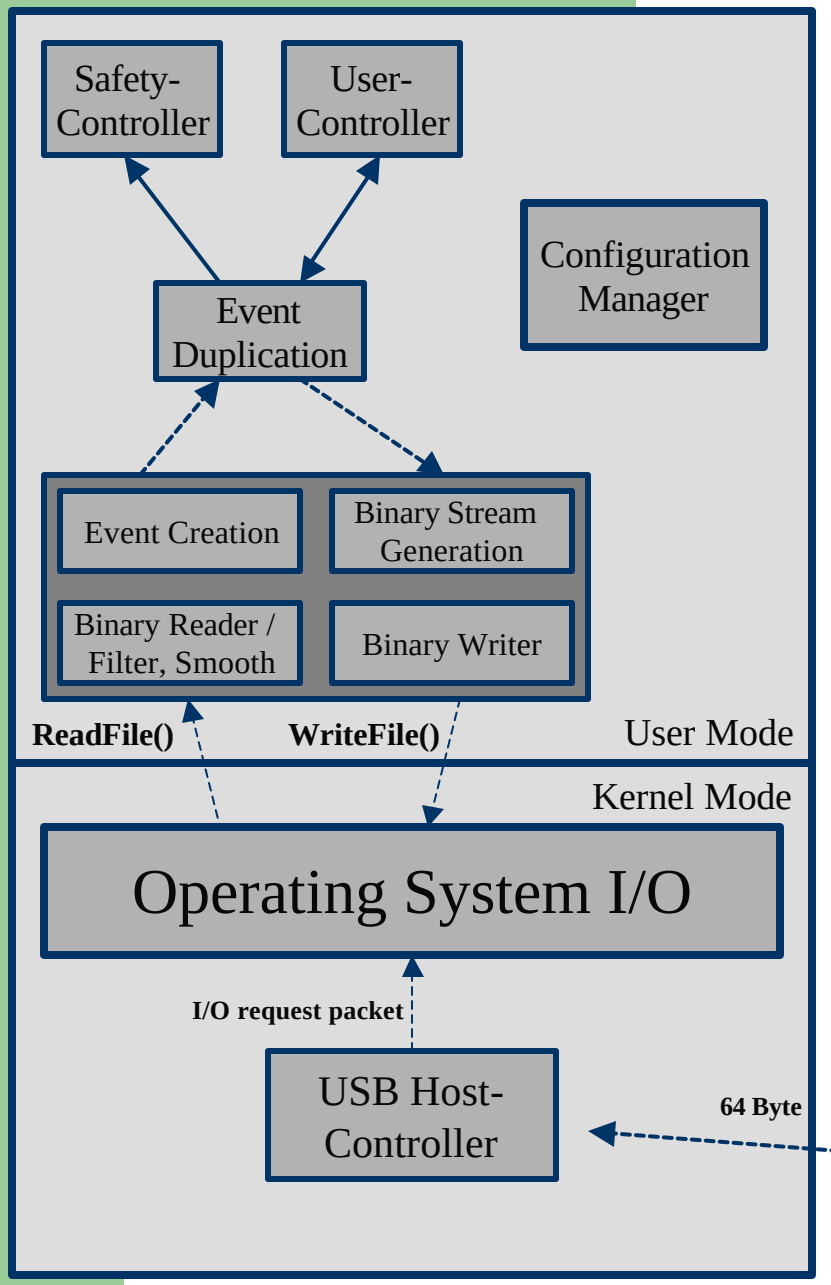


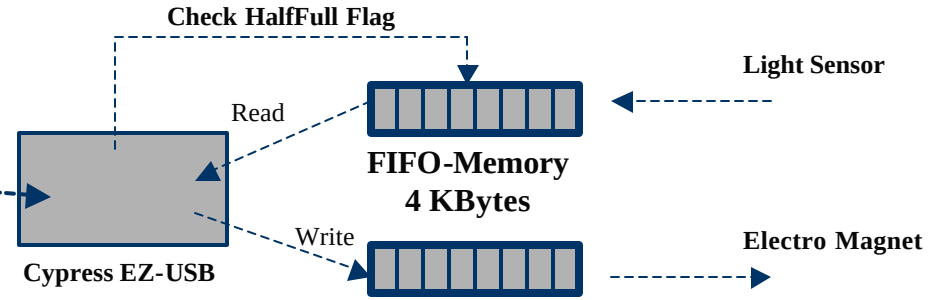
You can use the download link for viewing or saving the source data of the diagram.

Console Output

Pendulum Control – detailed

- Laser light barriers sampled into 4KByte FIFO-memorys with 23,4 kHz
- USB-Controller checks half-full-Flag
- 64 Byte blocks of data transferred via USB 1.1
- Real-Time OS-Threads process in-coming signals / produce out-going bit stream





Our Approach : Dynamic Reconfiguration as safe-guard mechanism

- Mapping of profiles to application configurations based on environmental conditions and component states
- Selection of application configuration according to conditions provides best service for a given situation
- Definition 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 (changed conditions)

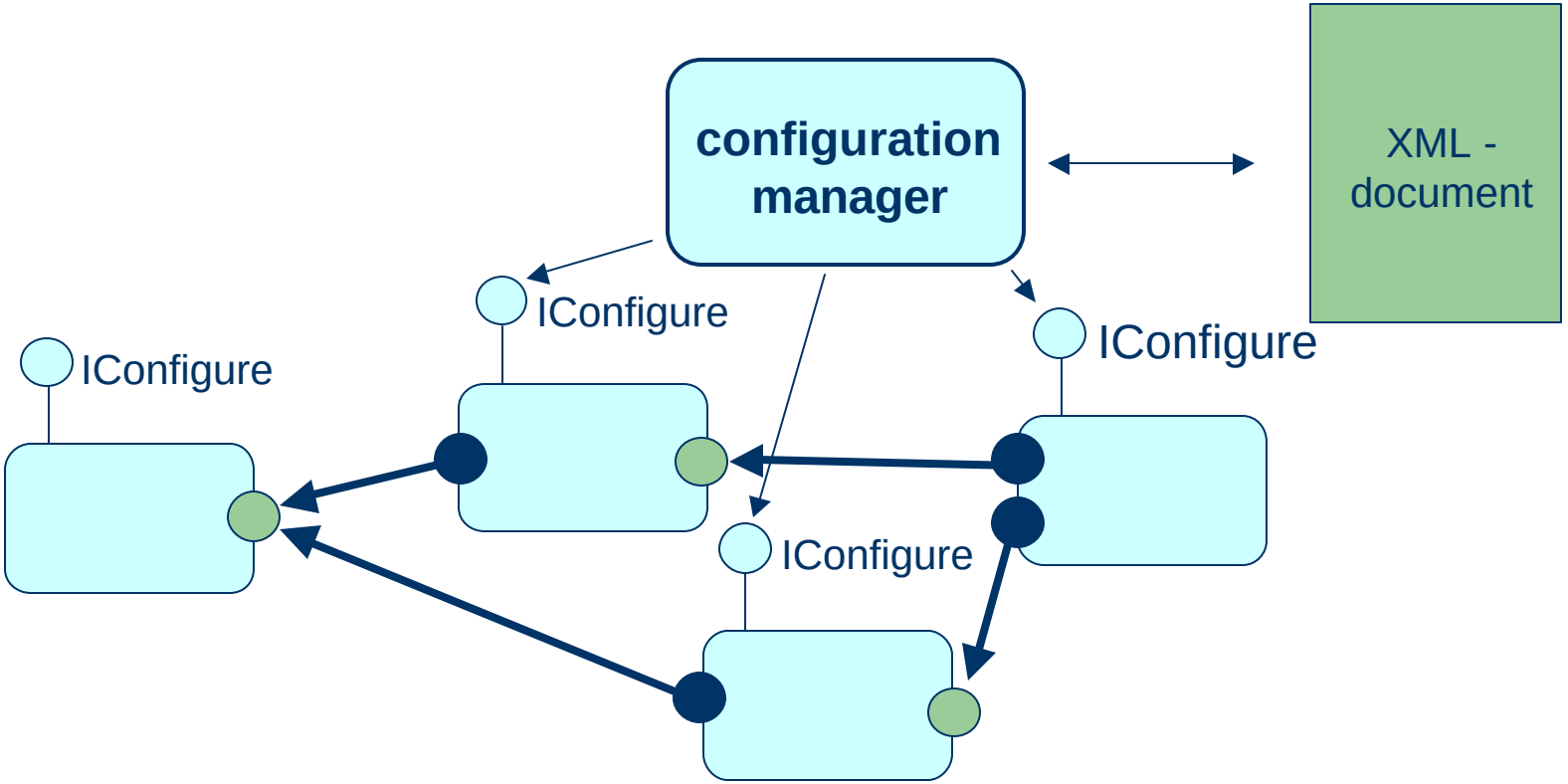
Description of configurations of component-based applications

- “A **Configuration** of a component-based application denotes the set of its parameterized components and the connections among them.”
- XML-based description language
- Configuration Description: List of components, their attributes, and connectors
- Support for a variety of component connectors

Configuration Framework

- Configuration Manager
  - Selects matching app configuration based on observed conditions and corresponding XML-configuration description
  - Instantiates/queries defined observers
  - Realizes distributed object activation
  - Enables adaptation of distributed applications using dynamic reconfiguration if required
- Standard reusable Observer-components
  - Network Bandwidth, CPU Power, Memory Consumption, Component State
- Components provide hooks for configuration management
  - Interface *IConfigure* must be implemented – can be automated

Architecture for Adaptive Systems



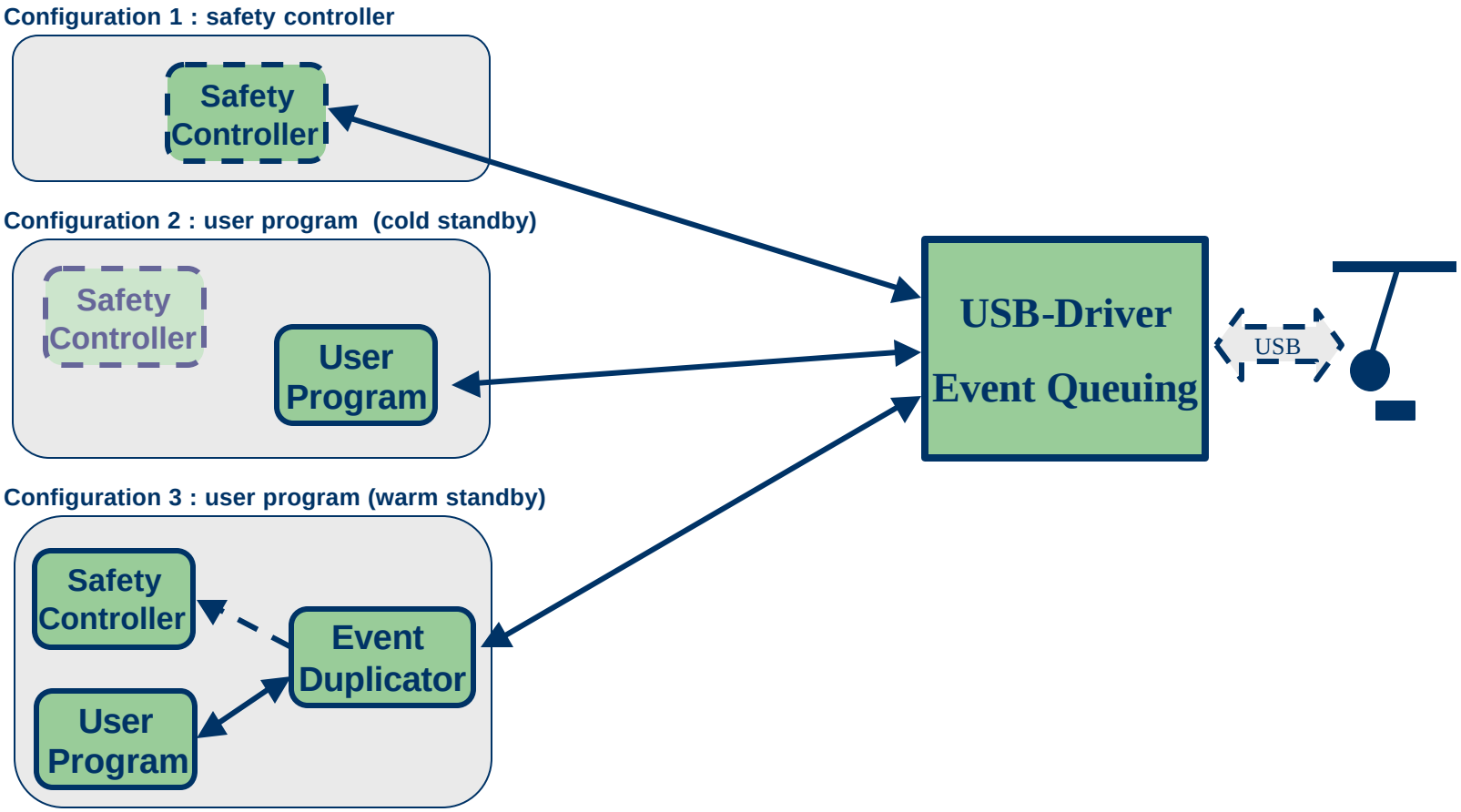
Our Reconfiguration Algorithm

- M.Wermelinger, J.Magee / J.Kramer
- Applications follow Actor Execution Model by G.Agha
  - Application consists of interconnected components
  - State of components changes only through interactions with other components
- Transaction Concept
  - Sequence of message exchanges over one connection
  - Initiator of a transaction is informed about its completion
  - Finishes in finite time
- Model matches wide range of typical applications
  - Including Client/Server-style applications
  - Control applications

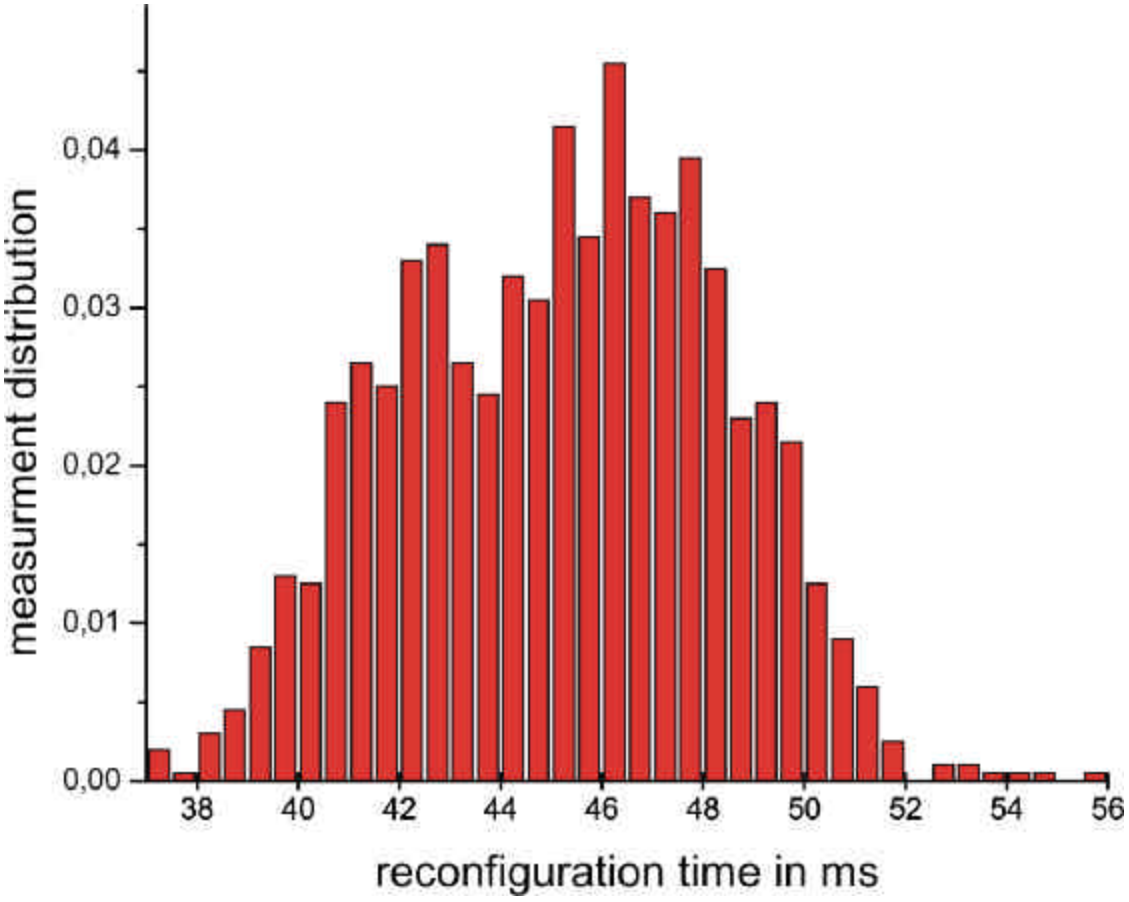
Dynamic Reconfiguration - Steps

- Start, Parameterization of new components
- Turn application into reconfigurable state
  - No pending requests
  - Block all connections involved in reconfiguration
    - Prohibit new transactions over identified connections
    - Wait for all ongoing transactions to complete
  - Blocking has to be ordered because of dependent transactions
- Parameterization of changed components
- Reconnect/Start all components
- Remove old components

Pendulum Experiment Control Configurations



Measurements:
Abnormal Termination of User Program

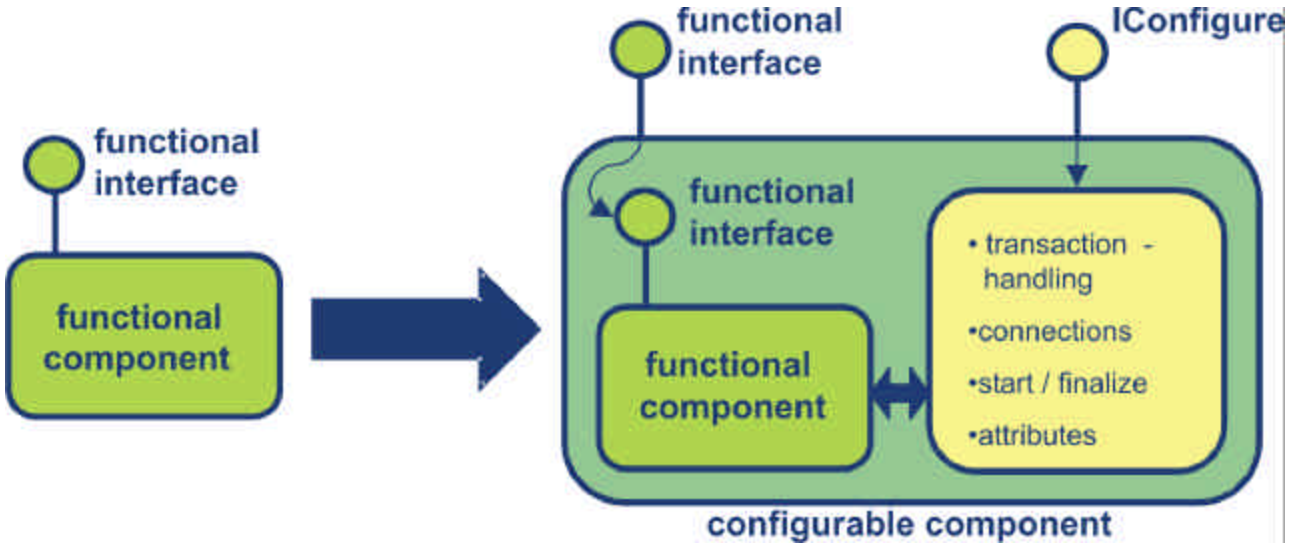


Configuration – a cross-cutting concern (AOP)

- Additional configuration-specific code has to be added to involved components
  - Handling-/Blocking Transactions
  - Start / Stop of component processing
  - Connection handling
  - Implementation of the IConfigure interface
- This code cross-cuts functional component code!
- We use aspect-oriented programming for automatic addition of non-functional configuration specific code
- **Usage of LOOM.Net – Aspect Weaver for .NET**
  - **based on (binary) components**

Making a Component Configurable

- Automatic implementation of configuration hooks
- Component programmer only has to mark transactions and provide access to connection references



Future Work

- “Higher Striker” – Real Time Control Experiment
  - Running on a RTOS (Ce.Net, eCos, rtLinux)
  - Small buffers possible / short delay
  - Parallel Port I/O
  - Sampling rate 38 kHz
- Port of .Net Environment (parts) to Lego-Robots
- Additional experiments / simulations

Related Work

- “Verbund Virtuelles Labor” project at University Reutlingen / Germany
- iLab project (WebLab) at MIT
- Virtual Lab at University of Hagen /Germany
- Tele-Laboratory at University of Pisa
- Tele-Lab / Simplex architecture

Conclusions

- DCL : environment for remote experiment access based on COTS Operating System and Middleware
- 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
- Measured reconfiguration times highly acceptable for control applications