Real-time robotics and process control with Windows CE and .NET

2nd MiNEMA Workshop
Lancaster UK
30th Nov. – 1st Dec. 2004

Dipl. Inf. Andreas Rasche
Operating Systems and Middleware
Outline

- Motivation / Overview
- The Distributed Control Lab (DCL) Architecture
- Protecting the Lab from Malicious Code
  - Adaptability / Dynamic Reconfiguration in the Lab
- Experiments in the DCL
  - *Foucault’s Pendulum* – Control with .NET
  - *Higher Striker* – Real-Time Control with CE
  - *Industrial Programmable Logic Control* and CE
  - *Lego.NET* - .NET for Embedded Devices
- Conclusions
Motivation

- Dynamic Reconfiguration / Adaptability
  - Reach a predictable system behaviour in unstable environments
- “Extend the reach of middleware”
  - Interconnected middleware-components and embedded systems
  - Grid computing technologies
- Online access to physical experiments over the Web
- Study techniques to prevent malicious code damaging physical equipment
Distributed Control Lab

● 2001 project start at Hasso-Plattner-Institute
  – Teaching 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

Visual Studio Integration
Dynamic Reconfiguration for Protecting from Malicious Code

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

- **Background:**
  - Demonstrates earth rotation
  - Today many installations 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 minimal energy
  - Reaching the highest amplitude
The Pendulum Experiment

You can enter here the measurement result which is situated under the programming details documentation (get).

You can use one of the following code examples:

```java
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
Dynamic Reconfiguration of Control Algorithms

Configuration 1: safety controller
- Safety Controller
- USB-Driver

Configuration 2: user program (cold standby)
- Safety Controller
- User Program
- Event Queuing

Configuration 3: user program (warm standby)
- Safety Controller
- User Program
- Event Duplicator

Graph showing measurement distribution over reconfiguration time in ms.
Framework for Adaptive Applications
Reconfiguration Infrastructure

CoFRA
C1
CoCo
application configuration
circular wire: assemb-
application level
CoCo
component configurators
CoFRA
HOST A

C2
HOST B

C3
local assembly repository
binary components

CoFRA
HOST A

application assemblies

configura-
tion description

CoFRA
HOST B
„Higher Striker“ - Experiment

- Real-Time Control experiment
- Usage of Windows CE.NET real-time OS
  - Hard Deadlines: Smaller Buffers, Higher Sampling Frequency
  - Control delay caused by buffers must be minimized
- Combination of non-RT .Net and RT application
- Evaluation of real-time Linux
- CE-PC Windows Ce.Net 4.2
- Simulation of Control Jobs in our Grid-Infrastructure
- Parallel I/O / 38 kHz sample rate / 256 Byte hardware buffer
- COTS x86 PC: Intel Celeron 633 MHz, 128 MB RAM
- Custom control hardware prevents overheating of coils
- Hardware watchdog reboots control PCs if user control algorithm can not be stopped
Industrial Control in the DCL

- Beckhoff CX 1000 Software PLC running Windows CE.NET (small and medium enterprises)
  - Geode x86 compatible 200 MHz Processor, 128 MB RAM
  - Extensible I/O modules (digital, analog in/out, relay outputs)
  - CAN field bus communication modules

- Experiments in the DCL
  - Implementation of DIN EN 61131-3 Software for PLCs
    - Validity Checks with separate/ parallel running PLC programs
  - Interaction of native CE applications with PLC programs
  - Distributed Control and Configuration with connected .NET Services
Beckhoff Industrial-PCs and the DCL
Controlling a Fischertechnik Assembly Line
Lego.NET - .NET for devices

- Our gcc (Gnu Compiler Collection) frontend supports the full ECMA-335 standard and can parse any conformant .NET assembly.

- Port for Renesas/Hitachi H8-300 microcontroller underway
  - Extremely small footprint (32 KB memory)
  - Runtime Library based on free BrickOS operating system
CLI2RCX- .NET – Current State

• Version 1.0 release implements the following features of the .NET platform:
  ✓ primitive datatypes: bool, byte, short, int
  ✓ classes, including instance attributes and properties.
  ✓ static and instance methods, including parameters, local variables, and constructors.
  ✓ arithmetic operations
  ✓ control flow operations: conditional and unconditional branch instructions.

• Next steps: most value types (enums, structs, delegates, floats, doubles), strings, single-dimensional zero-based arrays (partially complete), multi-dimensional or non-zero-based arrays, inheritance, polymorphism, and late binding, interfaces, exceptions

• Download / Weblog:
  http://www.dcl.hpi.uni-potsdam.de/research/lego.NET/
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
- Adaptive Grid Services
  6th framework

- Increased Throughput
- Increased Response Time
Conclusions

- Adaptability will become the most sought after quality of future embedded and middleware systems
- Our focus is on dynamic reconfiguration
- .NET
  - code access security and dynamic reconfiguration allows for safe code execution of mobile code in our lab
  - Malicious Code Problem: .NET and dynamic reconfiguration usable for small embedded devices
- rtLinux, Windows CE
  - Experimental evaluation of heterogeneous RT control environments