Dynamic (Re-) Configuration as Safeguard Mechanism in dynamically changing environments
Overview and Motivation

Mobile systems are challenging:

- Dynamically changing parameters of communication (Quality-of-Service)
- Varying computational power
  - among different mobile end systems (Notebook, Pocket PC, cellular phone)
- One-size-fits-all type of services are not acceptable
- State of the art: device-specific information encoding
  - Different representations for the same device are typically not supported (i.e., image vs. text vs. voice)
  - No chance to react on changing communication settings
The Problem illustrated
Towards Configurable ad-hoc Services for Mobile Users

• XML-based description language for system configuration
  – Description of service profiles
• Separation of implementation and configuration
• Toolset for automatic system configuration
• Support for mobile devices in networks with dynamically changing communication settings

• Proof-of-concept scenario using COM+ and Windows CE
How to Configure a Service?

- **Service**: component-based software which delivers information upon client's request
- **Services are made of**:
  - Components + attributes (i.e.; jpeg-filter with given compression)
  - Connectors (i.e.; RPC, shared memory interconnect, TCP/IP socket)
- **Service configuration consists of**:
  - Component instantiation and configuration
  - Selection of the appropriate interconnection mechanism (plumbing)
- **XML-based configuration description**
  - Rules for dynamic re-configuration when environment changes
Configuration Process

- Configuration manager interconnects components as described in the chosen configuration profile
The Distributed Control Lab – a configurable robotic system

Variety of robot configurations
• Different actuators/sensors
• Experimentation with control algorithms
• Coordinated actions of multiple robots

Dynamic reconfiguration as a safeguard mechanism
Distributed Control Lab – Structure

RemoteDCL Systemsicht der Softwareebenen der Experimente sowie RCX-Hardware

Browser

Website and MasterControl

IDCLControl

RemoteDCL

DCLControl RCX

RCXRunnerControl

Tracker Component

Tracker

Tracker

Tracker

LEGO Tower

Tracking Hardware (Webcam)

RemoteDCL

VideoStreamer

Webcam

Webcam

Webcam
Web-based Robotics Experiments

Welcome to the Distributed Control Lab !!!

NQ-C-Befehlsreferenz

```c
void main()
{
    OnForward(OUT_C); // Vorfahren
    Wait(1000);
    Off(OUT_C);
    OnRev(OUT_C); // rückwärts
    Wait(1000);
    Off(OUT_C);
}
```

- `OnForward(OUT_C)`: Vorfahren
- `OnRev(OUT_C)`: Rückwärts
- `Wait(1000)`: Wartezeit 1 Sekunde
- `Off(OUT_C)`: Ausschalten

Internet + .NET Remoting

ASP.NET & .NET Remoting

www.dcl.hpi.uni-potsdam.de
Dynamic Re-Configuration – Experimental Evaluation of .NET

How expensive is dynamic re-configuration?

- Measurements based on Pentium High Resolution Performance Counter (Win32-Api)
- Measurement overhead (**unmanaged call**) – within the µ-second range
- Example application: Client/Server system
  - Server propagates attribute via get-method
- 5000-10000 measurements per experiment
Reconfiguration timeline

External event changes execution environment

Measurement
Polling period
Configuration manager notices change
Loading of new components
Adjustment of new components‘ attributes
Blocking of connection requests
Adjustment of component‘s attributes
Interconnection of components
Initialization of new components
Removal of old components

Reaction on env. change
Detection of env. change
Reconfiguration phase
Blackout period
- Adjustment of a single attribute value
- Local application
- Client/server are separate processes

![Graph showing probability distribution of experimental results for blackout periods in ms.](image)
Adding a remote component residing in a separate process

- **Blackout-period**
- **Duration of re-configuration**
Scaling up

Number of components involved

Duration in ms

- Blackout period
- Reconfiguration period
- Loading of addl. components

Number of components involved
Foucault‘s Pendulum