The European Robot Initiative for Strengthening the Competitiveness of SMEs in Manufacturing

The 3-day-deployable integrated robot system
The 3-day-deployable integrated robot system

**Day 1**
Install Devices

**Day 2**
Configuring I/O, Systems

**Day 3**
Generate Trajectories

Architecture and technologies for **Plug-and-Produce** of robot workcells

Robot task generation based on product and process data
Points of view on Plug and Produce

What is shown: “the objects”

User Interface
Intelligent gripper
Robot guide device
Bin picking process

Research

What was invented for integrating “the objects”

PC - Cell controller
Configuration module
UPnP communication protocol
XIRP communication protocol

End User
Industrial

Stuttgart – May 7, 2009
1. Origin and meaning of Plug and Produce

2. Plug and Produce – Layers
   a. Application
   b. Configuration
   c. Communication

3. Plug and Produce in a bin picking application
1. Origin and meaning of Plug and Produce
Fast Installation and Configuration

• Setup of a robotics application using conventional technologies is **time demanding** and needs different specialists.

• For SMEs, which work typically with small to medium lot sizes, **fast installation and configuration** of a robotic system for new products and processes are strategic.

• The **mission** of this project is to offer to the user **easy** means for **programming** a robot cell **without** the need to do the **configuration** in **SME** environments.
Looking around

• In the office world it is very easy to install and use new devices. For example, to install a printer to your PC, you just plug it in.

• The entire configuration is then done automatically and your application will offer you the service “PRINT”.

This automatic configuration is called “Plug and Play”.
In a production environment this would mean that you would **connect a slave device to a PC** (called master cell controller) and a **Software** (called configuration module) **would offer** you on a HMI the **services available** on such devices.

Device **robot** offers the service **MoveTo**

Device **drilling tool** offers the services **StartDrill, StopDrill**
Synergy effect: super device

• Even more advanced, it could mean that the Software (configuration module) would recognize the synergy effect and offer you the service “DrillTo”.

Device / Process synthesis
“robot + drilling toll”

1. MoveToA
2. StartDrill
3. MoveToA+Z
4. MoveToA
5. StopDrill

• The **configuration module** needs to know about the functionality of the connected devices and the process description. Then is able to draw such conclusions.
The ability - provided by the Configuration Module - to add devices to a robot cell and use these devices without the need of configuration is called “Plug and Produce”, according to “Plug and Play” in the office world.
2.
Plug and Produce
Layers

Concept has been presented on ICRA 2007, Rome: “Control Architecture for Robot Cells to Enable Plug’n’Produce”
The functionality of the **configuration module** can be broken down into several **layers**:

- **Application** - Offers services to the user depending on the functionality and possible synergy effects of the connected devices.
- **Configuration** - Configures all the settings the users should not need to take care about, e.g. bandwidth requirements, default values, ...
- **Communication** - Fieldbusses and communication protocols. Automatic setup of a basic means of communication between cell-controller and devices, this includes addressing and discovery of a device.
Generic graphic HMI
Combines processes and
inserts process parameters
according to user description

Process library
Process description

Functionality evaluator
Executable Process

Code generator
Device spec.code

Application generator
Process to application

Device x
Device description

Device control

Executor
Commands & data
1. Automatic discovery and integration of devices

<table>
<thead>
<tr>
<th>Connected since</th>
<th>Driver</th>
<th>Identifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mon Mar 02 16:43:44 CET 2009</td>
<td>UPnPPCP</td>
<td>CLP</td>
</tr>
<tr>
<td>Mon Mar 02 16:43:44 CET 2009</td>
<td>UPnPPCP</td>
<td>Residential Gateway Device</td>
</tr>
<tr>
<td>Mon Mar 02 16:43:44 CET 2009</td>
<td>UPnPPCP</td>
<td>OL9482A3</td>
</tr>
<tr>
<td>Mon Mar 02 16:43:44 CET 2009</td>
<td>UPnPPCP</td>
<td>NETGEAR WNR834B Router</td>
</tr>
<tr>
<td>Mon Mar 02 16:43:44 CET 2009</td>
<td>UPnPPCP</td>
<td>infotec Pro 1106EX</td>
</tr>
<tr>
<td>Mon Mar 02 16:43:44 CET 2009</td>
<td>UPnPPCP</td>
<td>fgfusi</td>
</tr>
<tr>
<td>Mon Mar 02 16:43:44 CET 2009</td>
<td>UPnPPCP</td>
<td>cyclone</td>
</tr>
<tr>
<td>Mon Mar 02 16:43:44 CET 2009</td>
<td>UPnPPCP</td>
<td>pcjls</td>
</tr>
<tr>
<td>Mon Mar 02 16:43:44 CET 2009</td>
<td>UPnPPCP</td>
<td>simu</td>
</tr>
<tr>
<td>Mon Mar 02 16:43:44 CET 2009</td>
<td>XIRP_HAL_Interface</td>
<td>IPA-gripper</td>
</tr>
</tbody>
</table>

⇒ No manual configuration effort required!
2. Task definition by parametrizing processes and ...

➔ No knowledge of device interface(s) required
... defining the order of processes

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>If gripping point calculation was successful, continue</td>
<td>Goto depending on condition</td>
</tr>
<tr>
<td>Otherwise, goto END_OF_TASK</td>
<td>Jump to a process</td>
</tr>
<tr>
<td>Move robot complex above bin</td>
<td>Move robot along a path. All setting...</td>
</tr>
<tr>
<td>Gripper waits for robot</td>
<td>Gripper waits for robot to finish mov...</td>
</tr>
<tr>
<td>Set gripper angle</td>
<td>Set gripper angle</td>
</tr>
<tr>
<td>Robot waits for gripper</td>
<td>Robot waits for gripper</td>
</tr>
<tr>
<td>Move robot near gripping point (NB)</td>
<td>Move robot along a path. Execution ...</td>
</tr>
<tr>
<td>Move robot to gripping point detecting collisions</td>
<td>Move robot along a path detecting c...</td>
</tr>
<tr>
<td>If there was a collision, Goto COLLISION WHILE_GRIPPING</td>
<td>Goto depending on condition</td>
</tr>
<tr>
<td>Gripper waits for robot</td>
<td>Gripper waits for robot to finish mov...</td>
</tr>
</tbody>
</table>

⇒ No writing of source code required!
3. Executing the application

![Execution Control Panel]

<table>
<thead>
<tr>
<th>Device</th>
<th>Process</th>
<th>Current State</th>
<th>Next State</th>
</tr>
</thead>
<tbody>
<tr>
<td>EvaluationAlg</td>
<td>?</td>
<td>EvaluationAlgInit0</td>
<td>EvaluationAlgInit1</td>
</tr>
<tr>
<td>Robot</td>
<td>?</td>
<td>RobotInit0</td>
<td>RobotInit1</td>
</tr>
<tr>
<td>Gripper</td>
<td>?</td>
<td>GripperInit0</td>
<td>GripperInit1</td>
</tr>
</tbody>
</table>

- **Time** | **Level** | **Message** |
- 18:48 | INFO | Device configuration has been changed |
- 18:45 | INFO | Device configuration has been changed |
- 18:43 | INFO | Device configuration has been changed |
Focus on: Communication layer

Devices must be managed by means of different protocols, e.g. UPnP, Ethercat, XIRP, ...

Communication Abstraction Layer for 1&6

- Device description
- Device control
- Commands & data
- Executor

1. Device x
2. Communication Abstraction Layer for 1&6
3. Descriptions
4. Commands
5. Comm. Driver, UPnP
6. Comm. Driver, Ethercat
7. Comm. Driver, XIRP
8. Executor

Stuttgart – May 7, 2009
What are XIRP and UPnP

- **XIRP** – The XML-based *Interface for Robots and Peripherals* is a recommendation that was published by the German standardization institute DIN in 2006. It specifies a standardized mechanism and the corresponding communication protocol for robot device integration.

- **UPnP** – The *Universal Plug and Play* device architecture supports zero-configuring, “invisible” networking and automatic discovery of devices. A device can dynamically join a network, obtain an IP address, convey its capabilities and learn about the presence and capabilities of other devices.
Plug and Produce with XIRP+

XIRP+ Generalizes the device interface

- Common methods for accessing devices
  - Connect / disconnect
  - Execute commands
  - Get / set data
Plug and Produce with XIRP/XIRP+

**XIRP+ unifies configuration mechanism**

- Automatically discover devices
- Automatically retrieve device description
- Automatically configure interface modules

---

**Data & Commands**

```
Network & protocol
```

**Retrieve the description and generate interface configuration**
High Level Programming has been based on Service Oriented Architecture, a version of Plug and Produce.

Generating UPnP services directly from existing code, which is required to take advantage of pre-existing code was a new development initiated in this period.

These and other features were fully demonstrated and reported in papers presented in conferences (ICINCO2008 and CONTROLO2008) and journals.
3.
Plug and Produce in a bin picking application
The actors of “Plug and Produce” are:

- **The Cell Controller**, which acts as the network **Master**.
- **Sensors** and **Actuators devices**, which offer their services over the network.
Comau robot SMART NJ
- C4G-controller runs XIRP-server
- XIRP-interface: SetPath, Move, GetPosition, events, ...

Gripper equipped with a µ-controller
- Microcontroller runs XIRP-server
- Tool changer supplies Ethernet, 24 V, air
- Gripper can be automatically discovered by XIRP-client as soon as it is mounted to a robot flange
- XIRP-interface: Open, Close, SetAngle, GetAngle, events, ...

PC-CellController runs a XIRP-client
3D Laser Scanner:
- Controlled by a PC with an UPnP-interface
- UPnP-interface: StartScan, SetParameter(s), events, ...

Bin picking Algorithms:
- Software-module with UPnP-interface
- UPnP-interface: StartCalc, SetParameter(s), events, ...
Stuttgart – May 7, 2009

Demonstrator overview

State of the Art: “From order to chaos”

Beyond State of the Art “From chaos to order”

PC - Cell Controller

PLC

From bin to operational unit

1'

full bin IN

empty bin OUT

2

From bin to operational unit

height = 200

height = 800

3

from operational unit to bin

State of the Art: “From order to chaos”

4'

empty bin IN

full bin OUT

4''

empty bin OUT

1''
Thank you for your attention
Other material
Small glossary

- **Connection PnP**, or in the special case of data exchange interfaces, **Communication PnP**. This layer mainly deals with the set up of the basic communication. It is directly related to Layer 1 to Layer 4 respectively Layer 6 of the ISO/OSI reference model. After a successful Communication PnP devices can exchange data. This is the minimal PnP functionality that is necessary to achieve further PnP and this is also what many engineers typically mean by PnP, but at this stage there is still not defined which data that is to be communicated. There is of course a strong dependence on the used bus and the used protocols, but this layer is totally independent on the application.

- The layer of **Configuration PnP** mainly deals with the configuration of all connected devices and controls. It should care about all the things a normal user does not want to care about. Although the deterministic real-time communication is often part of the application in an ideal PnP scenario, this should also be configured automatically. This layer is dependent on both the communication and the application. Therefore it may be necessary to split up this layer in several sub-layers.

- The top layer of **Application PnP** mainly deals with using the functionality of connected devices. From the view of this layer all functionality is available and it is not necessary for the user or the system to know how data is communicated through the lower layers. The main problem in Application PnP is that automatic use of services and functions is not trivial. A simplification is to always require a reboot of the system, but depending on SME needs and type of devices it may also be desirable to support hot plugging.

- **Hot plugging** refers to the ability to add and remove devices to a computer while the computer is running and have the operating system automatically recognize the change. Two external bus standards -- Universal Serial Bus (USB) and IEEE 1394 -- support hot plugging. This is also a feature of PCMCIA. Hot plugging is also called **hot swapping**.
Platforms where to install before the delivery all equipments, included cables and hoses for media supply.
RFID for simple devices

- Conventional manual tools on a robot:
  - Use RFID:
    - to identify a device
    - to store device specific information
    - to configure the robot controller
Mobile Workshop

Drive the cart manually to a new working location

Option 1:
Calibrate robot against workpiece.
Start working.

Option 2:
Detach robot arm from cart, mount it at known working location.
Start working.
Transfer of Results, Examples:

Concept: Off- /On-line integration

**Background:**

- Traditionally only the robot program was used from the offline tool.

- No help in setting up the real cell to run the program.
Concept: Off-/On-line integration

Use the graphical layout to generate a real robot controller system with assistance to help the user to correctly set-up the robot cell.

- Programs
- Configuration

Virtual Controller

Pendant
- Programs
- Configuration
- Process
- Calibration modules
- GUI

Robot Controller

Deployment wizard
The Robot PnP work at LTH has focused on real-time communication issues, and device interfaces that are compatible with such communication.

The new EtherCAT master stack was used for the force feedback of the tabletop PKM robot at Automatica:

PKM robot and force feedback
The Plug-in procedure must guarantee an univocal relationship between the Programming Device and the Robot Control Unit.
The robot is moved by means of a wireless 6 degrees of freedom mouse that is **Plugged** on the robot tool.

**Wireless 6D-mouse:**

**Plug\(^1\)+Plug\(^2\)** and **Produce**

6 degrees of freedom
Intuitive device for robot programming

- The double wireless connections give full freedom of movement around the robot.
- Safeties are managed through the standard enabling device on the wireless programming unit.