TOWARDS A STANDARDIZED AND EXTENSIBLE MECHANISM FOR ROBOT DEVICE INTEGRATION
A XIRP-based Approach and Test Bed Implementation

Fan Dai and Joachim Unger
ABB Corporate Research, Wallstadter Str. 59, D-68526 Ladenburg, Germany
{fan.dai, joachim.unger}@de.abb.com

Keywords: Robot device integration, plug and produce, XML, XIRP.

Abstract: In industrial robot automation, the integration of intelligent peripheral devices becomes more and more important. But there is no standardized mechanism to setup the communication interface between them and to configure the usage of the device information for the robot applications. This makes the integration task often very tedious and time consuming. 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. Our experiences with XIRP-based implementations have shown its big potential to support robotics PnP on the communication and configuration level. As one of the topics within the SMErobot™ project, we are working on further developments or the concept. This paper introduces our approach for a revised XIRP concept, and discusses our experiences made on test-bed implementations.

1 INTRODUCTION

Integration of peripheral devices like sensors to an industrial robot becomes more and more important because of the demands on more intelligent solutions. However, there is no standardized mechanism to achieve the integration. In most cases, proprietary interfaces and protocols are used that require a profound knowledge about the communication level and often requires an implementation of the interface on one of the devices. This often causes significant costs and time efforts and is one of the burdens especially for small and medium-sized enterprises for using industrial robots in their environments. There is a high demand to support "Plug and Produce" (PnP) integration. It should be as easy as plug-in a USB device to a personal computer, so that the devices are recognized, configured and ready to use.

Within the SMErobot™ project, robotics PnP is one of the major research topics (Nilsson et.al., 2005, Naumann et.al., 2006). In this project, several levels and aspects for Plug'n'Produce were identified as depicted in Figure 1.

On the application level, we can consider each device as a provider of certain functionalities, which we also could call as services. These services have to be combined to compose an application level service that can be used to carry out the desired task. In order to obtain the appropriate services, the devices, including any control units, must be configured accordingly. The single services must be assigned with the right parameters. Finally, on the communication level, the parties have to agree on which protocols are to be used. To achieve PnP, there must be a standardized mechanism for the configuration and for the establishment of the communication. The communication protocol must be also standardized. In this way, there will be no need to implement the communication interface for different devices each time when you get a new device, or connect an existing device to another control unit.

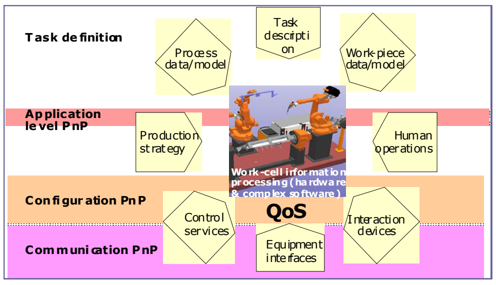

Figure 1: Aspects and Levels of Robotics PnP according to SMErobot™.

In this paper, we will focus on the lower levels, thus the communication and configuration levels.

Our approach is based on the general concept of XIRP – the XML-based Interface for Robots and Peripherals (VDMA 2006, Gauss et. al. 2006). XIRP 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. Our experiences with XIRP-based implementations have shown its big potential to support robotics PnP on the communication and configuration levels.

Within the European project SMErobot™ (Plug and Produce work package), the project partners are working corporately on a revised XIRP version, also defined as XIRP version 1.1 or XIRP+. This paper introduces our approach that was partly our contribution to XIRP+, and discusses our experiences made on test-bed implementations. Finally, we will describe our view on the integration of different communication concepts and standards to support robotics PnP.

2 THE APPROACH

2.1 Application Model

In an automation environment, there are basically two roles: control units and peripheral devices. This can be compared with the control points and the services in the context of UPnP. Depending on the application scenario and how the user configures the whole system, each device can theoretically take both of these roles. However, in a robot work cell, the control unit is usually the robot or the cell controller.

For the control unit, a device providing certain services can be seen as a logical unit with three types of functions:

- connection
- execution of commands
- exchange of data

This is independently from the complexity of a device. It can also consist of several physical devices that in their turn can be connected in any form to provide a more powerful service.

This view is common both for service-oriented or client-server communication models. There is no clear separation between both. However, we don't emphasize the concept of loosely-coupled client-server relationship. We use the client-server model and combine it with the discovery methods of service-oriented concepts.

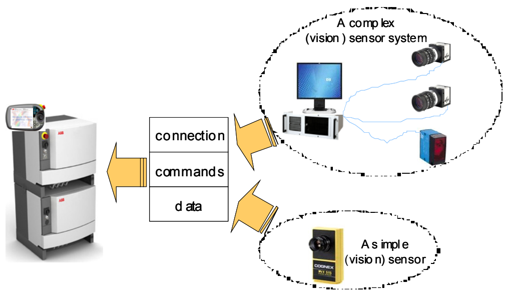

Figure 2: Relation between control units and devices.

In our client-server model, there is a defined relationship between two parties during runtime. One party, typically the robot controller, is a master of the application during the whole session. It provides a better overview of the coherences between services, and allows easier management of the logical structure in an automation world, especially in an industrial robot application. Service-oriented approaches have the potential for flexibility and upwards scalability, which can be beneficial for large scaled, distributed environments. But dealing with robotic cells for small and medium-sized enterprises (SME), it is important to have controllable structure, other than scalability. Furthermore, the real-time efficiency is also essential for robot-device communication.

2.2 Protocol Stack

Figure 3 shows the communication stack of XIRP devices according to the ISO/OSI reference model from ISO/IEC 7498-10.

Although XIRP does not specify Ethernet for the physical layer, it is recommended to use IP over Ethernet. The standardized protocols like TCP, UDP or HTTP can be used on the session layer to handle the reliable or unreliable transport of communication packages.

The message presentation uses XML, whereas XIRP specifies the basic structure of the messages, like SOAP does for Web services. But XIRP messages have less overhead so that it is much more efficient. Metadata as required in the SOAP messages are not required for XIRP, as XIRP presumes, that metadata are exchanged during the device configuration phase, if they are not available before that.

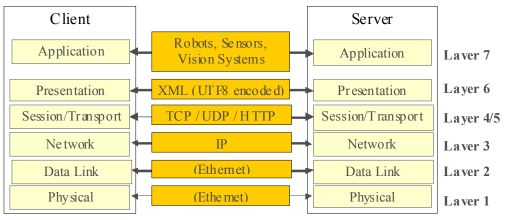

Figure 3: Protocol stack.

There are basically three types of messages that a client device can send to a server device: commands or requests for getting and setting property values. XIRP specifies the structure of these messages and the corresponding reply messages. On top of the basic specification, device type or application domain specific agreements can be defined.

2.3 Communication Channels

When connecting two devices, one device takes the role of the client and the other one the role of the server. The main communication channel (command channel) allows the client to "control" the server by sending command messages, which triggers the execution of an action (command) on the server device, or getting and setting the value of properties on the server.

To enable the PnP integration of devices, automatic discovery via UDP multicast is supported like in UPnP, but XML is used for the message representation without the SOAP overhead. Similarly, device and service descriptions and other files can be retrieved via HTTP. It is furthermore possible to transfer files from a client device to a server device via HTTP.

A client device can subscribe individual events by sending a subscription message to the server over the command channel. If the server device confirms the subscription, the client device will receive these events on the event channel. In addition a XIRP client can also send events to a server device. Before an event can be sent, the client announces it to the server via the command channel, so that the server can be prepared to receive the event on the event channel.

In robotics applications there is also demand for the periodical transmission of data on a defined rate and under some time constraints. This can be done via a periodic channel. Similar to event subscription, periodical data are also subscribed (for receiving) or registered (for sending) using appropriate messages on the command channel.

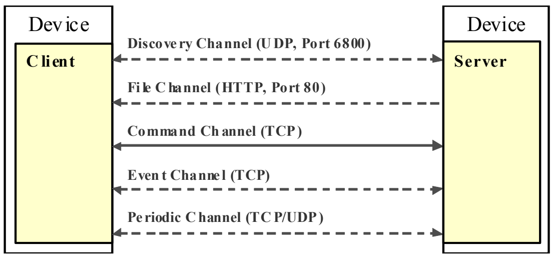
Device
Client
Device
Server
Discovery Channel (UDP, Port 6800)
File Channel (HTTP, Port 80)
Command Channel (TCP)
Event Channel (TCP)
Periodic Channel (TCP/UDP)

Figure 4: Communication channels.

Most robotics applications nowadays can be supported using these communication channels. For applications that have special requirements, we also allow the creation of additional channels via the command channel. These channels would use other standardized protocols. One example is the time synchronization channel that can use the PTP (Precision Time Protocol) protocol of IEEE 1588, to adjust the clocks in a distributed system and create a common time basis. A binary channel using a standardized binary transfer protocols could be also created, even when binary data can be embedded into XML messages using Base64 coding and transferred via the above mentioned standard channels too. These additional channels and the corresponding protocols have to be specified in the description of the devices, and can only be used, if both sides of a communication session support them.

2.4 Device Profiles

The definitions described in the XIRP specification are specified as mandatory or optional agreement. Mandatory definitions have to be implemented as specified. Optional agreements don't have to be supported, but if such a definition is implemented, it has to be done according to the XIRP specification. Furthermore, the definitions are grouped into so-called device profiles and organized into Basic Profile, Class Profiles and Custom Profiles as depicted in Figure 5.

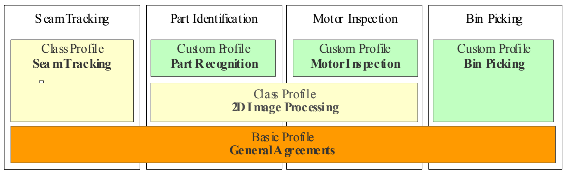

Figure 5: Device profiles.

The Basic Profile contains general agreements that are mandatory for all devices that claim conformity to the XIRP specification. It specifies the basic commands for the connection establishment and parameters for the control of common functions of XIRP-confirm devices. Also specified as general agreements are aspects like

- Schema and rules for description of devices
- Supported communication channels
- Basic structure of communication messages
- Message synchronization
- Termination of communication messages
- Language selection for comments
- Parameters for connection management

- Connection establishment
- Primitive data types
- Rules for extensions

To run applications, additional agreements on commands and parameters are needed. Those are specified in the class or custom profiles. Definitions in a class profile are only mandatory for devices that claim conformity to this device class, which could be smart cameras, laser trackers, complex vision systems, etc. Devices that conform to a certain class profile should be exchangeable without affecting the execution of an existing application, if no custom profile is used in addition. Otherwise, a smooth replacement of devices is not ensured.

Custom profiles can extend the general agreements and class profiles with additional commands and parameters that are needed to run a specific application. They do not represent a standardized specification, but have to conform to the specifications in the General Agreements. If an application can be implemented with existing device class profiles, no custom profile should be defined and used in addition.

Preliminary, a communication profile for a new device, or a set of device functions, which has not been defined in a class profile yet, can be defined as a custom profile and submitted to the XIRP working group as a proposal for standardization. For a certain time slot the preliminary profile will be put up for discussion and then released as a standardized class profile.

2.5 Protocol Schemas

Commands and data type definitions, as well as definitions for message structure are specified as XML schemas, which can be stored in one or several files. A device can refer to multiple schema files as long as they do not contain conflicting definitions. The benefit of using the standardized XML Schema format for these definitions is that the files can be interpreted with validating XML parsers and created with common XML tools.

2.6 Device Description

Each device must be supplied with a device description. It contains the general information about a device and is stored in a XML file. The contents are:

- The XIRP version
- the unique device identifier (or name)
- the type classification for the device
- the category classification for the device

Optionally, it can also contain:

- a textual description of the device
- the user interface URL of the device
- custom or device class schemes associated with the device
- the selection of languages that are supported by the device for the comments on commands and parameters in the configuration files
- The collection of communication channels that are supported by the device.

Following is an example for a device description:

```
<?xml version="1.0" encoding="UTF-8" ?>
<Description Version="1.1">
   <Name>TestDevice</Name>
   <Vendor>MyVendor</Vendor>
   <Model>MyModel</Model>
   <Class>SmartCamera</Class>
   <Url>127.0.0.1</Url>
   <Schemes>
     <Schema Path="Camera.xsd" />
     <Schema Path="Calibration.xsd" />
     <Schema Path="Contour.xsd" />
   </Schemes>
   <Languages>
     <Language Code="en" />
     <Language Code="de" />
   </Languages>
   <Channels Url="127.0.0.1">
     <Discovery />
     <File Port="80" />
     <Command Port="3002" />
     <Event Port="3003" />
     <Cyclic Protocol="UDP" Port="3004"/>
   </Channels>
</Description>
```

The device description is an important element for PnP, and can be used to automatically establish connection and configure the communication and programming setting.

2.7 Plug’n’Produce

When talking about PnP, we mean the ability to use a device after plugged it onto the network, without the need of manually setting communication parameters or configure the interfaces. As indicated in previous sections, the key elements for PnP are: the device description, the corresponding schema definitions and the mechanism of connection establishment.

The following figure shows how PnP works in our approach.

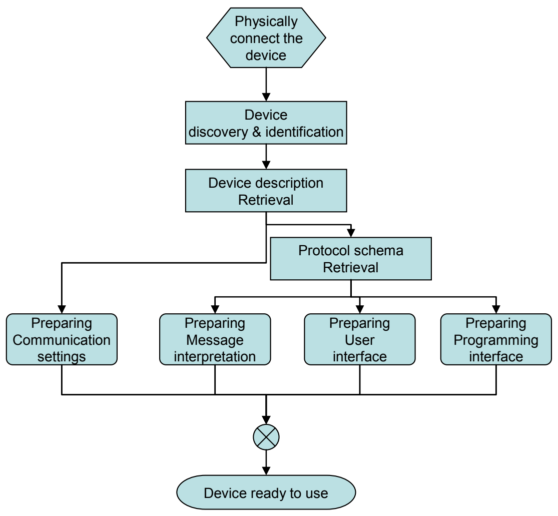
Physically connect the device
Device discovery & identification
Device description Retrieval
Protocol schema Retrieval
Preparing Communication settings
Preparing Message interpretation
Preparing User interface
Preparing Programming interface
Device ready to use

Figure 6: The PnP mechanism.

The discovery channel is the first mandatory functionality of a device that supports PnP. After physically connected the device to a control unit or to the network of the work cell, the device will announce it's presence with a broadcast message via the discovery channel. A control unit has to listen to this broadcast channel, if it wants to be aware of devices that are plugged-in. The other option is to actively detect new devices by sending a discovery message. Every PnP device would respond with a unicast message to this control unit. In this way, the amount of broadcast messages can be reduced to a minimum during runtime of the robot cell, because they are only sent as needed.

After device discovery, a control unit would be able to identify the server devices and check, which additional information are needed to establish the connection to these devices. First of all, the device descriptions must be made available to the control unit. A PnP enabled device should have the device description accessible via the file channel. Alternatively, there could be also a repository of device descriptions elsewhere. But the safest way to provide compatible and up-to-date device description is to provide it on the device itself.

With the device description, the control unit can be prepared to establish the connection to the server device. This means, the communication level PnP would be done. The device can be connected whenever it is needed for the application.

Configuration level PnP needs more information than directly provided in the device description file. But with the references specified in the device description, the control unit can further retrieve the schema files. Information provided with the schema allows the control unit to configure the interpretation of the communication messages. Also possible is to create or update the user interface for application programming based on the commands and data type definitions in a schema.

3 TEST-BED IMPLEMENTATIONS

Three test-bed implementations are described briefly in this section. They represent three scenarios with different requirements on robot-device communication, so that the functionalities of the different communication channels could be demonstrated and evaluated.

3.1 Smart Camera

As an example for relatively simple, but intelligent device, we used a smart camera from DVT (now owned by Cognex) of the type Legend 510. This camera has embedded image processing functions and Ethernet link on-board. It can be configured and programmed using a kind of scripting language via a PC. Afterwards, the camera can be used stand-alone.

The device description file and corresponding schema files are located on the camera, and are automatically downloaded to the robot system via the file channel (using HTTP). The robot system is then automatically configured so that the commands and variables defined in the schema file are accessible for robot programming.

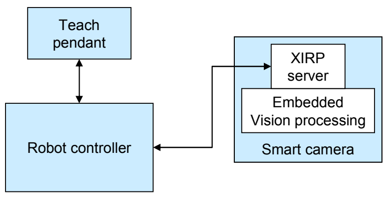

Figure 7: Smart camera test-bed.

In the sample application, a vision method for blob finding and analysis is implemented on the camera. It can be triggered from the robot system. Results like number of objects found, and size and position of each object can be read from the camera using the "GetOnce" command. This data is used to generate robot motion targets to pick up the objects.

The device and its properties as well as the vision object data are automatically mapped to variables in the robot programming language RAPID. They can be used like other RAPID variables on the teach pendant.

3.2 The Chess Robot

The ChessRobot scenario, which has been presented on the Hannover Fair 2007, was used for the evaluation of XIRP-based PnP as well. In this scenario, multiple XIRP server instances were connected to the robot controller, using command and event channels. The test-bed architecture is illustrated in Figure 8. Figure 9 shows the setup on the Hannover exhibit 2007.

The robot controller is the control unit in this scenario. It controls two robot arms, each of them playing two games – one against a visitor and the other one against each other. The boards for the games with visitors are observed by cameras. The moves of the visitors are recognized using a vision system, whereas a game engine checks the validity of the visitor moves and generates counter-moved for the robot. To test the concurrent communication with several server devices, we used three instances of XIRP server device running on three separate PCs.

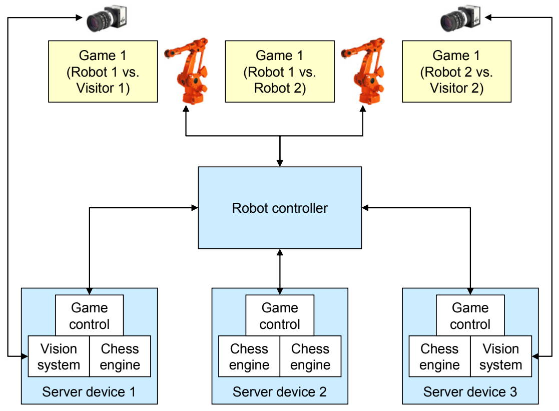

Figure 8: ChessRobot testbed.

Figure 9: ChessRobot on exhibition.

3.3 Safety Zone Supervision

Another test-bed application used cameras to supervise the safety zone in the robot workspace. A modified version of SIMERO (Gecks, Th., Henrich, D., 2004) runs on a Linux PC and analyses the spatial relation between human worker or moving obstacles and the robot arm. Depending on the spatial situation, it sends then adapted velocity data to the robot so that the robot moves always safely.

For this purpose, a periodical channel is used to transmit robot data to the vision system and the updated velocity data to the robot controller. The data exchange could be done in real-time with very small jitters. In addition, the robot data are assigned with timestamps so that correct mapping of spatial data is possible.

4 DISCUSSIONS AND OUTLOOK

We implemented our XIRP-based approach for three scenarios with different requirements. With the smart camera test-bed, it has been shown that even using scripting language, it is possible to implement XIRP server on a low-cost smart device. PnP worked with downloading the device description from the camera. With the ChessRobot test-bed, multiple server devices were connected to the robot controller. Also the functionality of event subscription could be demonstrated. The safety zone supervision test-bed further evaluated the subscription of periodical data for real-time communication.

These examples have shown that the implementation of XIRP protocol doesn’t require much computation resources. The approach works both for simple intelligent devices and PC-based systems.

This approach is control unit centric, with clear controller-device relationship. This is also the typical case for SME scenarios. Even though, the XIRP-based architecture is also scalable to support scenarios with multiple clients and servers, we haven’t tested our approach with large-scaled distributed environments. Because of the nature of this kind of client-server model, it has limitations with flexibility and scalability in terms of distributed, large-scaled environments.

When considering distributed environment without real-time constraints, Web services approaches have big potential. We are the opinion that a hybrid approach would better fulfil the different requirements rather than trying to implement one protocol for all cases. Furthermore, we observe progresses in the standardization of communication protocols on the field device level like EtherCAT and OPC UA. Also considering that XIRP might be still too complex for simple I/O devices, it is proposed to develop an integration framework that supports different protocols.

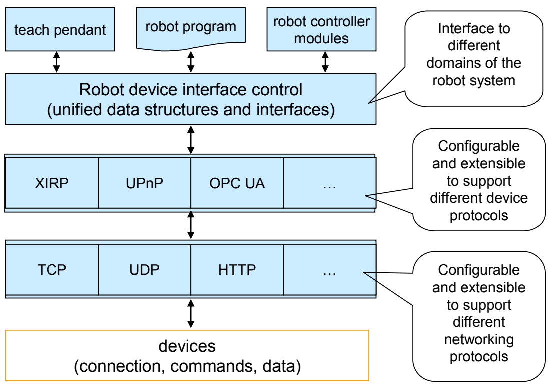
teach pendant
robot program
robot controller modules
Interface to different domains of the robot system
Robot device interface control (unified data structures and interfaces)
XIRP
UPnP
OPC UA
...
Configurable and extensible to support different device protocols
TCP
UDP
HTTP
...
Configurable and extensible to support different networking protocols
devices (connection, commands, data)

Figure 10: Device integration framework.

Certainly, a careful selection of the protocols should be done so that they really complement each other. Figure 10 only shows the principle and doesn't mean any preferences of protocol selection.

To have different kind of devices running different protocols concurrently with minimal implementation and processing overhead, we have to find the common features of the selected approaches. Also to be investigated are the boundaries of each protocol. Based on the investigations, recommendations for the users respectively application engineers can be worked out.

It is very hard, or even impossible to make a standard that fits to everyone's needs. But a selection of standards, a common understanding of the methods and some guidelines or recommendations would help the users, especially SME's to apply robotics PnP.

ACKNOWLEDGEMENTS

This work has been partly funded by the European Commission's Sixth Framework Program under grant no. 011838 as part of the Integrated Project SMErobot™.

REFERENCES

Nilsson, K., Johansson, R., Robertsson, A., Bischoff, R., Brogårdh, T., Hägele, M., 2005: *Productive robots and the SMErobot™ project.* Third Swedish Workshop on Autonomous Robotics, Stockholm, September 1-2, 2005.

Naumann, M., Wegener, K., Schraft, R.D., Lachello, L., 2006: Robot Cell Integration by Means of Application-P'n'P. In: *Proceedings of the Joint Conference on Robotics,* May 15-17, 2006, Munich

Gauss, M., Dai, F., Som, F., Zimmermann, U.E., Wörn, H., 2006: A standard communication interface for industrial robots and processor based peripherals - XIRP, In: *Proceedings of the Joint Conference on Robotics,* May 15-17, 2006, Munich.

VDMA, 2006: *XML-basiertes Kommunikationsprotokoll für Industrieroboter und prozessorgesteuerte Peripheriegeräte (XIRP),* VDMA 66430-1:2006-07, 2006

Gecks, Th., Henrich, D., 2004: SIMERO: Camera Supervised Workspace for Service Robots. In: *Proceedings of ASER 2004, 2nd Workshop on Advances in Service Robotics,* Feldafing, Germany, 20-21 May 2004