Web based service for embedded devices

Dr. Ulrich Topp
Peter Müller

ABB Corporate Research
Wallstadter Straße 59
68526 Ladenburg, Germany

Summary: Field devices are becoming more and more 'intelligent'. This report describes how ABB can use this intelligence to offer service capabilities via internet standards and to integrate these devices into the Industiral-IT world.

Content

1 Introduction

In industrial automation systems devices like sensors for various physical measurements and actors to perform some action on the process are located in the field of operation. This is why we call them “field devices”.

In a normal plant there are several standards for such devices to communicate with some controlling computer and/or the operator station. Among these standards are Profibus, Foundation Fieldbus, CanBus, and others. These are suitable for control networks with distances below 500m.

In other situations field devices are used in a locally wide spread application, like in power networks or along a pipeline. Thus the user needs a way to operate and maintain his equipment remotely.

In addition, often several devices are grouped together, and these devices are not necessarily from the same manufacturer. Here the urgent need is, to have a common way of communication in order to avoid multiple access infrastructure.

This paper describes the use of standard internet technologies to achieve remote service and operation capability. Two proposed solutions enable the user to survey his device remotely and to discover needed maintenance tasks prior to an urgent need, often signalled by total failure of the device.

Using standard technologies, like dynamical generated html and SOAP, based on TCP/IP as transport protocol we describe the needed supplier independence. In future systems, there will be one local IP-network and one remote connection to the operator and the maintenance staff.

As ABB is introducing its new Industrial-IT platform as basis for all automation software, this paper also includes the description how this work fits into ABB’S software strategy.

This work was done at DECRC/Ladenburg in collaboration with the ‘Institut für Automatisierungs und Softwaretechnik’, IAS, University of Stuttgart.

2 Motivation and scope

Today it is a well known key issue to provide excellent service with every part ABB is going to sell. This paper illuminates the situation especially for field devices and shows how ABB can assure outstanding service capabilities using state of the art internet technologies fully aligned with the strategy of Industrial^{IT}

2.1 The situation today

The classical service for field devices is done by a service engineer going into the field and visiting and maintaining every part of the equipment manually. A step forward is the usage of remote service capabilities offered by connecting to the device using a modem line. This gives the engineer the possibility to examine remotely the status of the device and to be better prepared if a visit is necessary. Often it will be enough to change some parameters or just to reset the device to ensure a proper function until the next maintenance visit. The benefit of this approach is limited in situations where in one place several different devices are located, often from several manufacturers. The limitation occurs due to usage of proprietary protocols to communicate and to the fact that the devices often could not be connected in a coherent network.

Here the usage of standards concerning the networking (e.g. TCP/IP) and the higher level of communication (Internet protocols like http) could bring a major advantage in order to facilitate or even enable remote communication. This will allow a much better service and life cycle management.

2.2 Trends from outside

A recent NEXUS report [NEXUS] on technology roadmaps in industrial instrumentation highlights two technology trends that will dominate the future of process instrumentation

a) Microsystems technology or Micro Electromechanical Systems (MST/MEMS) and

b) Internet technology.

Especially the combination of both will result in **"Smart Systems"** comprising additional functions for monitoring, diagnosis, asset management, (self-) configuration and offering new opportunities and possibilities such as remote sensing, remote access and advanced diagnostics.

Despite the importance of accuracy, performance and cost of devices, user needs are becoming motivated by a new set of drivers that may overshadow the traditional attributes that previously drove the demand for field devices and sensors. These drivers, lead by field networks, include asset management software and the growing need for Web enabled communication, may reverse the trend of slow growth of field devices and continue the prosperity of discrete sensors [ARCFDSS]. In addition more and more requirements arise to have a better integration that spans not only process and alarm data but integrating all aspects of a device starting from nameplate data, maintenance records, documentation, configuration, alarm handling and of course operation and monitoring.

Users believe that information assets from smart devices are key to trouble-shooting instrumentation problems and reducing maintenance costs. By identifying devices that need attention in lieu of the typical preventive maintenance program based on time in service, they save costs of unnecessary maintenance.

Invensys and Endress+Hauser are actively researching device failure modes/analysis to develop sophisticated embedded high-level sensor diagnostics. Invensys in an alliance with the University of Oxford, formed the Invensys University Technology Center (UTC) to research areas such as advanced sensors, condition monitoring and self-validation, known today as SEVA. The goal of Invensys is to incorporate algorithms into their process field devices to validate sensors, improve control loop tuning and overall industrial processes. UTC has patented methods to provide data that will continuously update the health of the instrument and tune instruments back into calibration due to minor faults [UTCAI].

Placing remote I/O and small PLCs with IP/Ethernet connections on the shop floor is currently economical, and is becoming quite popular with systems integrators (FF->FF-HSE, ControlNet->EtherNet/IP, Profibus->PROFINet, Interbus -> IDA …). *ARC predicts that over the next five years, IP/Ethernet will dominate all field connections except for the final connection to the lowest level sensors and actuators [ARCSWT]*.

As devices begin to communicate over the Internet using TCP/IP or UDP/IP protocols allowing users to view and manipulate data using a standard browser, a new device category, Internet Appliances, is emerging.

From our point of view the need of contribution to these trends seems obvious. ABB has to be an early adopter of advanced technologies especially in the case that the advantage over conventional ones is that evident.

The following table summarizes the pro's and con's of the different approaches.

	Conventional, proprietary protocol	webserver with dynamic html	webserver with SOAP services

Interoperability	—	+	+
User Interface	— (client side generated by hand)	+ (web browser)	O (client side generated with tool support)
Resources	+	O	—
Reuse of components	—	O	+

Table 1: This is an overview over important categories, which could be used to compare the usefulness of different software approaches. Proprietary protocols, as used in the past, clearly have a advantages in resources because they are tailored for one special application. But concerning the reuse of software, interoperability and AIP integration only solutions based on standards will be cost effective and successful.

2.3 IndustrialIT

In 2001 ABB announced a new software platform for all its products in order to enable a global integration of all information, be it static or dynamic. The following gives a (very) short overview over some base principles of IndustrialIT.

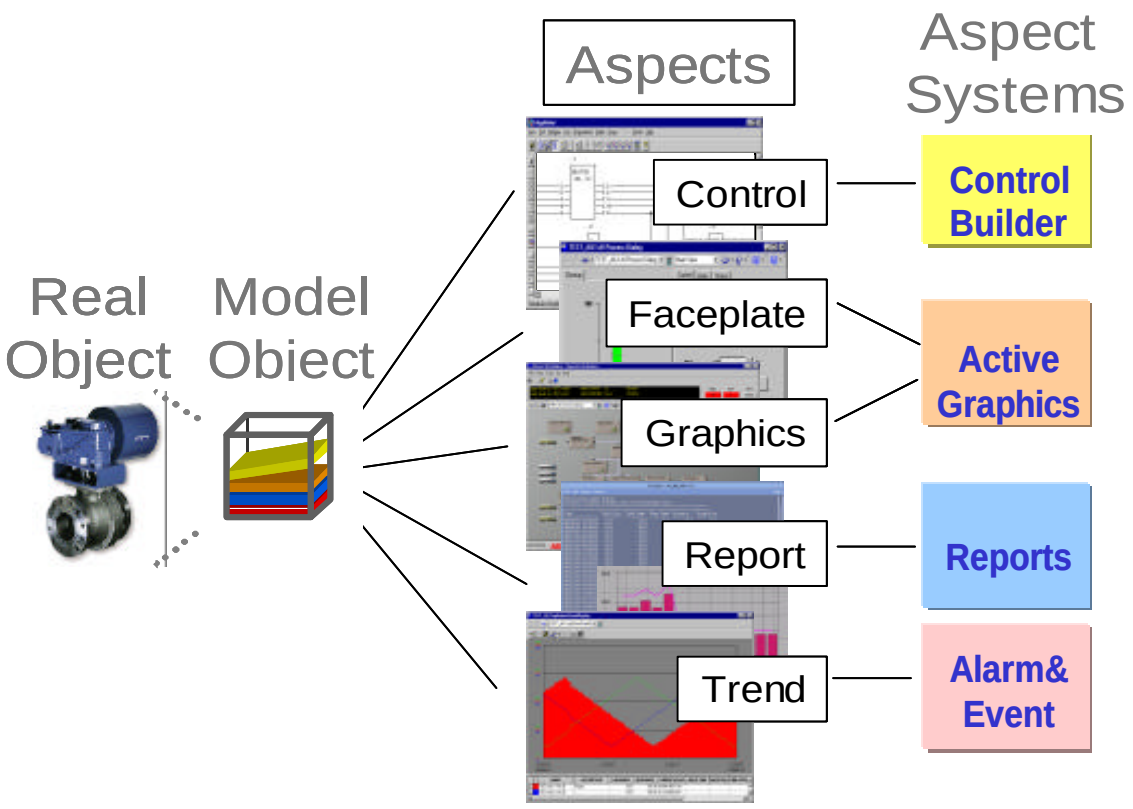

Figure 1: The principles of Industrial-IT: The real world objects are modelled through aspect objects, to which functionality is added by means of Aspects. These Aspects are interfaces to Aspect Systems, which in turn implement the appropriate function

The Idea of IndustrialIT is to model the real world objects in a manner similar to how we as human beings look at them: We see the object and depending on the context and on our role respective to this object we recognise and use different characteristics of it. This approach leads to a software platform, the Aspect Integrator Platform (AIP), where all real objects are modelled through abstract objects which initially has only some generic and basic functionality (name, type information and so on). Special characteristics are added through adding 'Aspects'. These are made up of pieces of software implementing some functions or implementing an interface to an external data source or software package. For example, structural information such as the physical location of the (real) object in the plant or the functional placement in the process or the logical arrangement in a production order is added through 'Structure Aspects', forming a multidimensional grid, where the system and the user could freely navigate. Figure 1 shows several other possibilities to add functionality using Aspects.

In addition, this platform is designed in a way, that it is fully network transparent, allowing to access the information of the objects independently of the users location.

Goal of this approach is to integrate all information sources and make them available in realtime to every user whatever role in the plant he plays, wherever he is located, and whenever he needs the information.

This is in short, what ABB calls “the next way of thinking”.

2.4 Conclusion and proposal

Now we are in the situation that we see our demands and we could proceed to the solution, which was studied in the past few months.

Two main paths have been under study. The first consists of an embedded web server, which offers a HTML based user interface and allows the user to do device control, diagnosis and monitoring tasks using a standard internet browser or an Aspect System, integrating the devices functionality into the AIP.

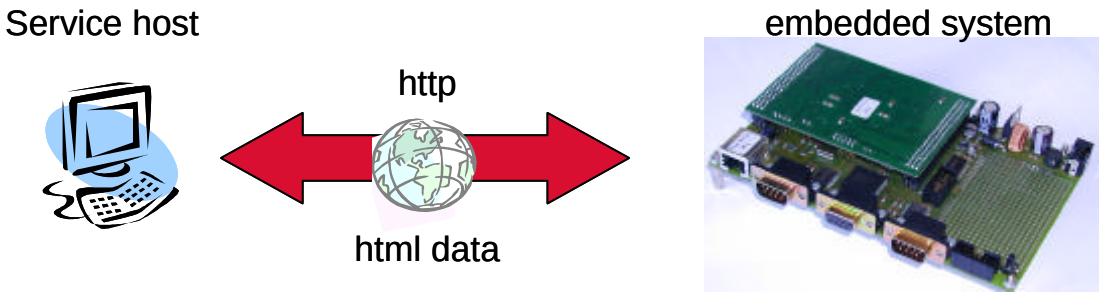

*Figure 2: System topology with the operators PC, the **service host**, and the embedded system, where a web server provides device information and operation capabilities through both static and dynamic generated html pages. On the service host a standard web browser could be used as smple user interface as well as a AIP operator station, integrating the device into a larger system.*

The second development path uses the SOAP protocol to implement several ways of communication and remote procedure call, thus opening the way towards an nearly unlimited communication with and controllability of the device.

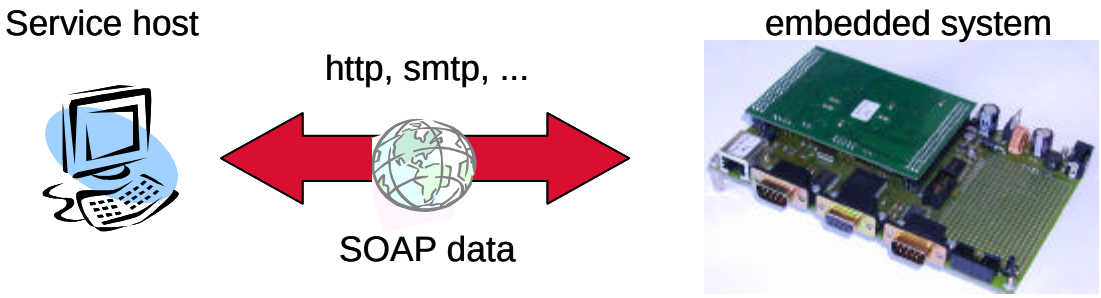

Figure 3: System topology where the embedded system offers its services using SOAP, thus providing a much more powerful and generaic way to access measurement and status data from the

device. This approach s not limited to a http connection but could also be used with the email protocol SMTP.

Obviously, the next logical step would be to combine both paths bringing the enormous flexibility of SOAP communication into the IndustrialIT platform.

3 Proposed solution: Embedded Web service plus SOAP

Before we go into the details of the technologies used, we clarify the scenarios where remote service is needed. After that the test case is discussed, a device for gas chromatography, and the last two sections cover the two main development paths, we worked out.

3.1 Typical remote service scenarios

This subsection presents typical deployment scenarios applicable to all field devices. In detail these are:

- **Operation / Monitoring**: The user is able to perform a particular measurement (in case of a sensor) or control task (in case of an actor) and to capture the resulting values.
- **Diagnosis / Monitoring**: This module takes some measurements concerning the status of the device. For example environmental measures like temperature, humidity, voltage of power supply and so on. The user is able to capture the resulting time series.
- **Configuration**: The user is able to manipulate remotely a wide range of configuration parameters concerning the operation of the device. The parameters are permanently stored locally on non-volatile memory.
- **Alarms**: Inform maintenance staff about errors or send out maintenance triggers (e.g. battery voltage reaches a critical low level). The device is configured to generate alarms when certain conditions occur. The alarm is transmitted to the user by means of e.g. e-mail.

For all four kinds the feasibility has been tested. Figure 4 shows the principal data exchange and messages sent between the service host and the embedded device.

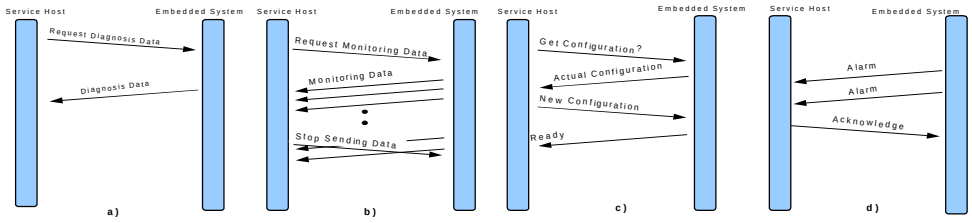

Figure 4: The different scenarios for remote service: a) Remote Diagnosis b) Remote online monitoring c) Remote Configuration and d) Remote alarm handling.

In general there are three kinds of communication services necessary:

- **Request / Response**: This kind of communication can be used to request diagnosis data or the device configuration

- **Subscription**: With this kind of communication callbacks can be realised. This is typically used to retrieve process data without forcing a client to poll. Note that the role of the client and server changes after the subscription was initiated.
- **Spontaneous**: The role of the server and the client has changed to the request / response scenario. Here the client starts communication without request from the service host. This is typically used for sending alarms.

3.2 Test case: the gas chromatograph on a pipeline

Gas Chromatography (GC) is an analytical technology employed within ABB for general process control as well as for dedicated metering and custody transfer applications in the natural gas market. Whereas in the first market, ABB serves customers with the specification and setup of customized and very specialized gas chromatographic instruments, the second is a market, where the application is ever identical, namely BTU-Measurement of natural gas, and strategic high focus is placed on cost, sevicability and diagnostics for remote installations as well as general robustness of the system.

Such devices are located along pipelines for natural gas in order to monitor the quality of the gas. This serves as the described scenario of locally very spread devices which need to be operated and serviced.

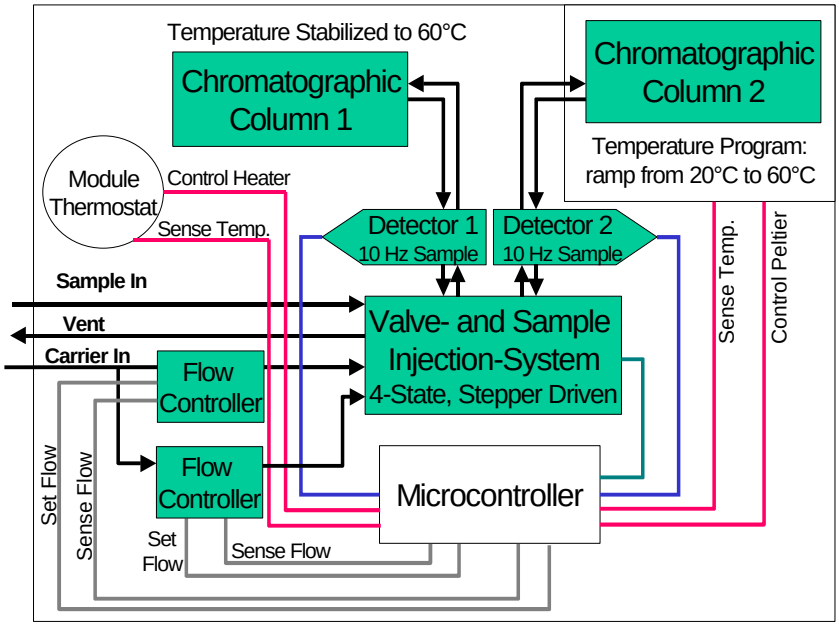

Figure 5: This is a system overview of a gas chomatograph with the main components controlling the measurement. Not shown is the Web interface

In general the GC consists of 4 functional blocks:

1. Temperature stabilization and control: The measurement is valid only if defined climatic environment is guaranteed. Thus PID controllers are employed to stabilize as well as ramp temperature in certain functional regions of the GC.
2. Flow stabilization: The measurement is valid only if defined flow of the gas under test is guaranteed. Thus a PID controller is used to stabilize this gas flow. A defined sample volume is pushed through the system with a carrier gas. Two such carrier gas streams are to be controlled in flow.
3. Valve control: The measurement is controlled by switching a 9-open/close-valve-system through four different states.
4. Main measurement: The uGC splits the gas components in a way that they pass one after the other over the measurement cell, a thermal conductivity detector. Here the deviation of thermal conductivity (measured as voltage) is employed to measure the quantitative admixture of a component to the carrier gas. Different components are identified through their arrival time in the detector.

Every block has its own set of parameters like the two constants defining the linear relationship between the measured voltage and the physical property of a PID control loop.

The microprocessor in use has a twofold task. First, it has to control the operation of the device as described above. Second, it has to provide the 'virtual faceplate' offered as a web service. This means all necessary control options and parameters has to be handled through a web interface, be it a web server with HTML and CGI, see sec. 3.3, or a SOAP server, see sec. 3.4.

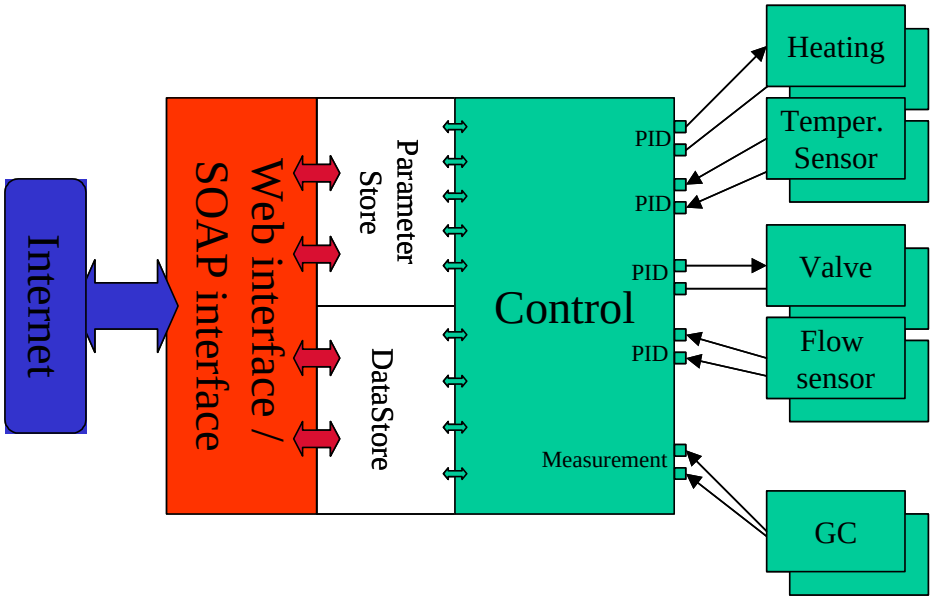

Figure 6: General architecture of the embedded system. The control part covers the sampling of sensor values and updating the actuators according to the set parameters as well as the complete measurement cycle. The web/SOAP interface part handles the connection requests from the service host/operator station. Both parts are connected through two types of 'stores.

The ParameterStore holds all parameter infromatiion, while the DataStore serves as a small database of historical data.

3.3 Development path 1: Embedded web server with CGI interface

A web server responds to a client request always in the same manner: It sends a block of text over the internet, mostly consisting of HTML-code describing the content of the answer and its formatting. The HTML-code could be located statically at the server or could be generated dynamically by the server. The first way is chosen for information which is likely not to change and the second, obviously, for presenting some data from the server, which may change in time. In particular variable data like configuration parameters and, of course, process values fall in this category

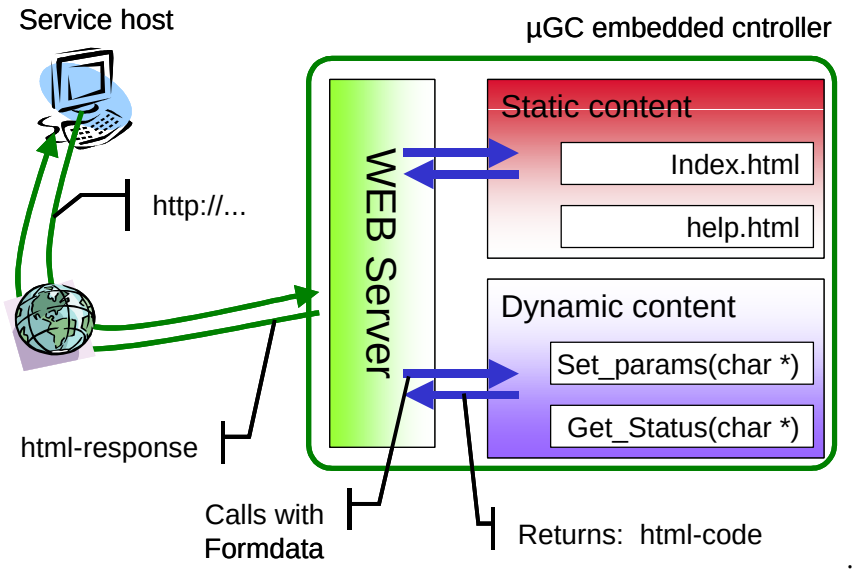

Figure 7: Architecture of the embedded Web-server. The http-request from the service client (top left side) is processed by the web server. Depending on the requested document reference, html-code from the static content pool is returned, or the server calls a registered object method, evtl. with additional form data. This method in turn generates the html-respinse on the fly incorporating some real time values of the system.

This branch of the study implements a web server on the embedded controller, which uses both ways:

The portal to the service and some information and help texts are statically located in the embedded systems non volatile memory.

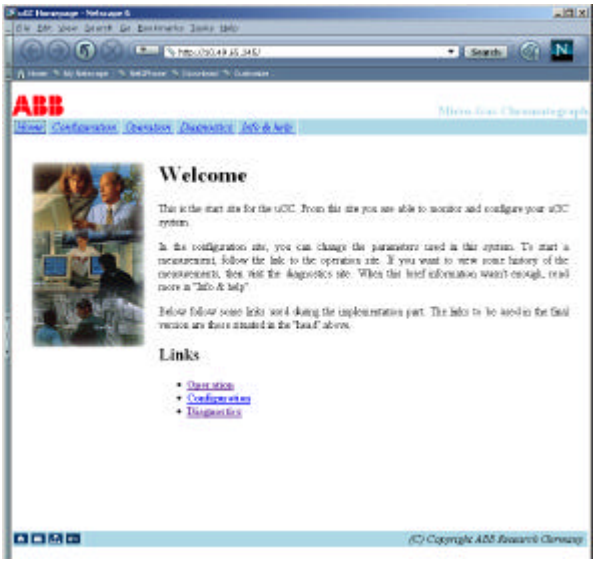

Figure 8: The portal page to the GC embedded controller

From the portal the user is linked to the various services, shown in the following pictures:

- Operational Status (with buttons to initiate some actions)
- Configuration (Overview over different sections and as an example the configuration of a PID controller)
- Diagnosis (As graphical representation as well as textual, in order to export it to mathematical applications)
- Measurement Data is up to now provided as textual information.
- Documentation

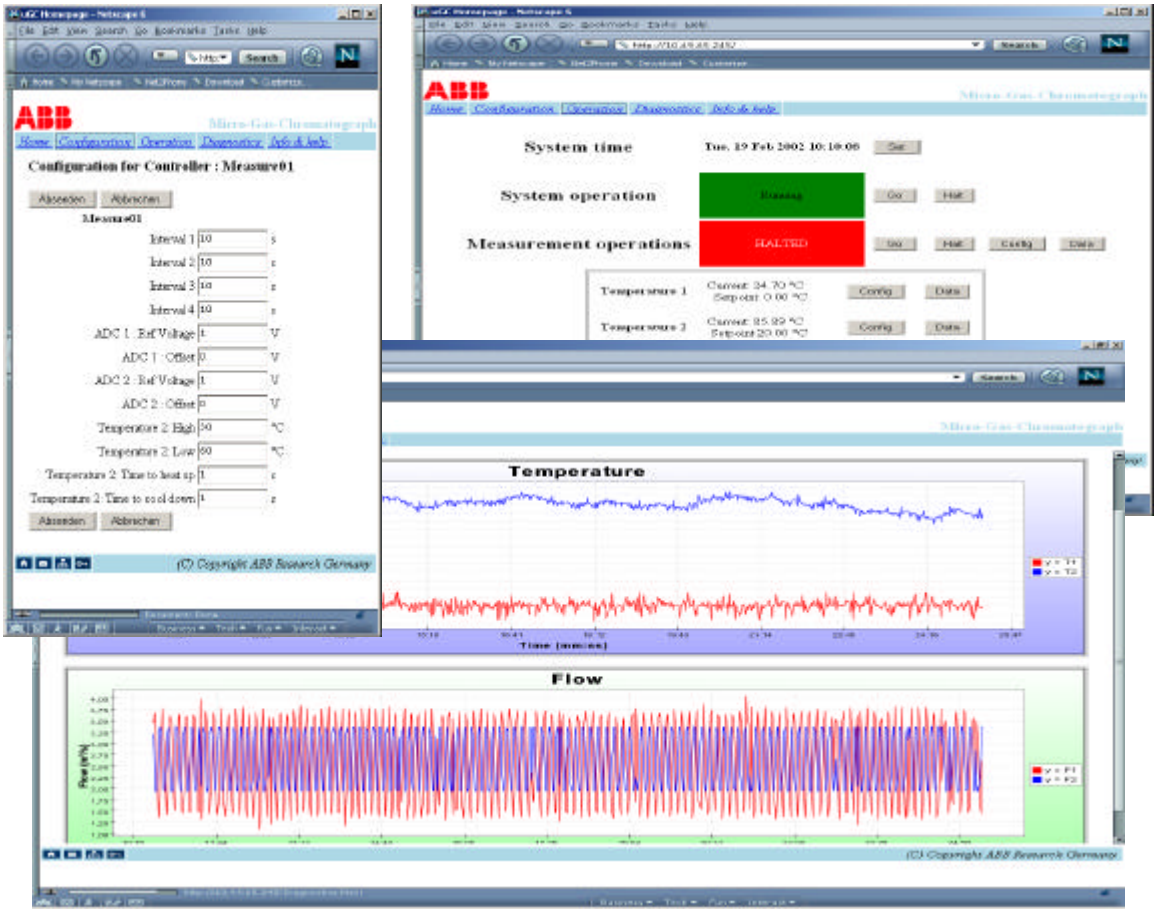

Figure 9: Views of the different services: Top left shown is the configuration page of the measurement; top right the operational status of the device. On the bottom the output of a java applet is shown with the data of the temperature and gas flow controllers.

This implementation make it very easy to integrate the device in an IndustrialIT environment. The AIP provides standard aspect categories to connect to web servers and to show up the HTML information. Figure 10 shows an example scenario for the integration. The aspect object for the GC is equipped with some aspects linked to the appropriate web pages for configuration, diagnosis and so on (see the context menu in the figure). Some basic information needed by the AIP system, like the IP address of the embedded controller, are stored in an standard aspect of type ‘General Properties’.

[FIG deleted]

Figure 10: The GC embedded controller in a AIP system. The functional structure in the top left corner shows a pipeline object and below three GC devices. The pipeline object has a graphical map aspect associated. This map shows the pipeline in Alaska together with four locations of

gas chromatographs. Right-clicking evokes the according context menu, providing the user all available aspects of the device, allowing to navigate e.g. directly to the temperature data of one of the devices.

The next step of integration would be not only to display the process values as text but to fetch them into the system as binary value. The way to achieve that, is through a simple program written in Visual Basic, converting the ASCII representation of values sent by the embedded system into numeric variables. Through the mechanisms of the platform these variables could be published as OPC values, thus opening the way to use the build-in analysis features of the platform, like trend displays, historians, and so on.

This step is not taken because of the much more powerful possibilities the usage of a SOAP based service offers, which we will outline in the next section.

3.4 Development path 2: Advanced solution with embedded SOAP

3.4.1 Short walk through SOAP and WSDL technology

If an application aims to provide a broad range of services and interfaces, the implementation must avoid proprietary methods. It assures the maintainability as well as the interoperability, if one uses accepted standards. In the field of services offered via internet connections, the SOAP [SOAP] standard is the way to go.

SOAP is an acronym for ‘Simple Object Access Protocol’ and is a XML based specification for message based communication and for remote procedure call (RPC) communication. It allows to evoke specific function calls on the server from internet connected clients. What SOAP makes it that powerful is the way function parameters are serialized, that means how they are transformed from the client system specific binary format to a completely system independent format. Furthermore SOAP allows the server to respond to the client not only a short status

message but also every kind of data, which could be serialized following the same rules as above.

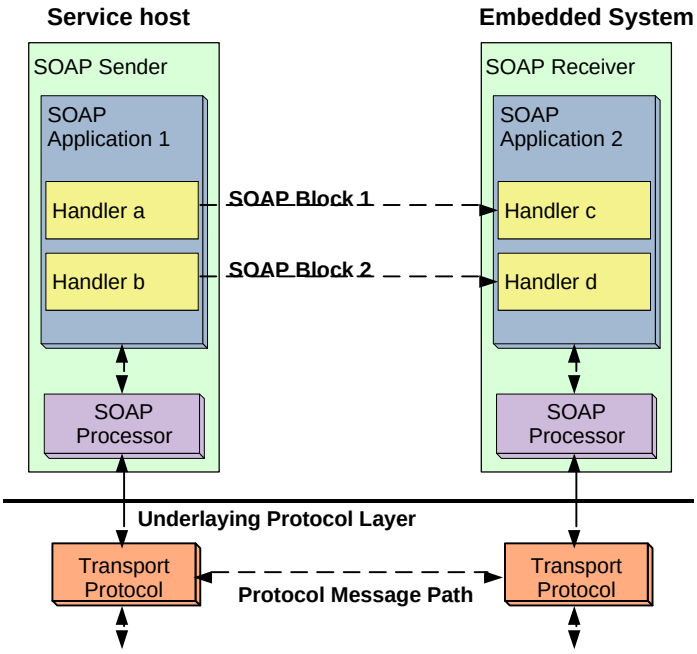

Figure 11: The basic principles SOAP handles remote procedure calls. The service host is able to access functions, the embedded system has opened for remote use. The packing and unpacking of parameters is handled by the SOAP engine and is fully transparent to th

Once you have implemented a SOAP server, the only issue is to make its services, that are the available function calls, public to all possible clients. For this purpose the Web Service Description Language (WSDL) was introduced by the W3 Consortium [WSDL]. This language has exactly the above needed purpose: The standardized description of services, a site offers to clients via the internet. It is not bound only to SOAP services but most SOAP toolkits offer helper programs to generate automatically the WSDL file derived from the source code. Or the other way around: The developer starts with a WSDL file, specifying the service, and the tool generates the skeleton code for a SOAP server/client in the desired programming language (e.g. C++ or Java).

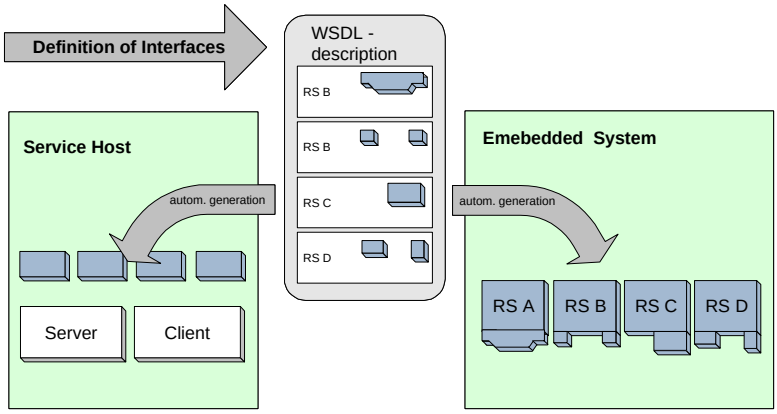

Figure 12: The Web Service Description Language WSDL is used for an abstract definition of remote services (RS). It opens the way for automatic generation of code skeletons for the SOAP client as well as for the SOAP server

Being able to make functions calls on a low level, it will be possible to achieve a deep integration between the software running on the embedded device and the AIP server system. In terms of Industrial^{IT} integration levels level 2 will be quite possible. This includes that the corresponding Aspect System makes use of most of the ABB-Aspect and -Object related operations like navigation, change notification, and so on. All this will be highly maintainable and reusable through the use of the open SOAP standard.

3.4.2 Sample implementation

The given timeframe allowed us to implement two of the above service scenarios (see Sec. 3.1): The remote configuration of a PID controller and an alarm service. The first is the classical case of client-server communication, but the second reverses the roles: The embedded system is now the client, which searches a service host to post an alarm message. Since this is not to be modelled in a normal web-server architecture here the use of SOAP is essential.

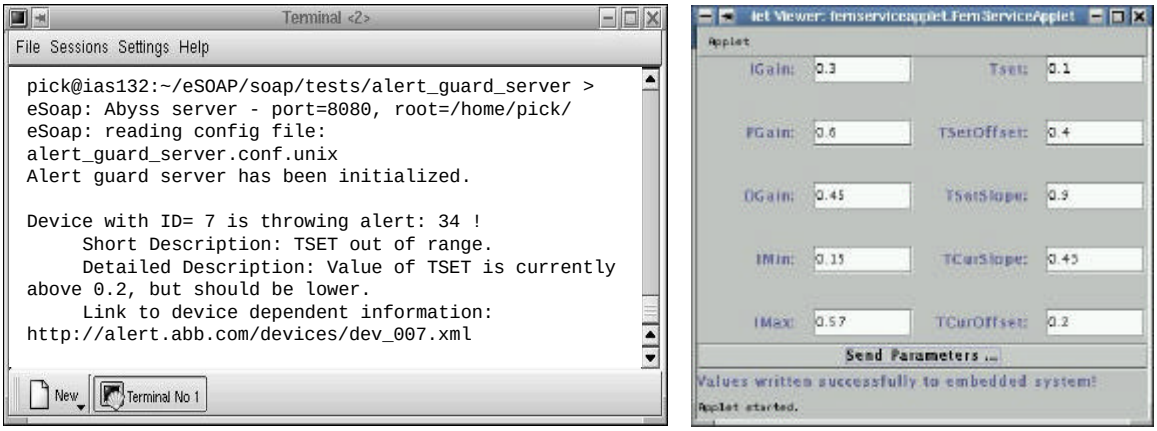

Figure 13: Two simple SOAP applications on the service host: Left hand is the alert guard server, which shows up an error, the embedded system reported. And on the right side a Java applet shows the configuration form for a PID controller and in the bottom line the response from the embedded system indicating the successful transmission.

4 Technical and Cost Comparison

The two approaches show different advantages and, as shown in the Table 1 (page 4), are applicable in several scenarios. This picture becomes clearer, if we consider the costs in terms of memory consumption. (CPU performance becomes more and more a less critical boundary condition.)

The implementation of the two approaches, described in sections 3.3 and 3.4, have a very different resource allocation.

- The embedded web server with CGI interface needs around 24kByte code size compared to ca. 43kByte for the control part of the software.
- The implementation with SOAP is much larger, together with some helper libraries the used eSOAP toolkit uses around 400kByte.

In both cases there are in addition some runtime memory needs depending on the used TCP stack.

As an important remark, we state that we had not the time to apply any optimisation strategy. This means, we used the different toolkits as they are offered in the public domain.

This leaves room for enhancement especially, if we search for commercial solutions for the TCP stack, or the SOAP toolkit, which might be memory and resource optimised in order to fit also in smaller systems. Reducing the memory needs by factor of 2 to 5 seems possible.

Anyway, the above numbers classify the two approaches for different target applications: In cases, where memory is expensive, as in small devices like simple sensors, one would prefer the less powerful solution. This will enhance the product with key features regarding operation and maintenance without making it too expensive.

But there are many applications, where the device consists of more than a sensor, and in order to handle all parts a larger controller or even a small PC is employed. Then memory is not an issue because in such systems RAM is counted in Megabytes rather than in kilobytes.

5 Conclusion and Outlook

The use of state of the art internet technologies will allow a much deeper integration of field devices with the AIP than today possible. In addition, the usage of world approved standards assures interoperability with other third party products and software.

The test case of the gas chromatograph clearly shows how the service capabilities could benefit from the power, which is today offered by modern microprocessors, which are already installed in the device in order to operate it.

The studied scenario is obviously not limited to this GC application. The results are applicable to all devices, which are equipped with microprocessors and a network connection. Since future strategy (e.g. .net framework) is strongly coupled with the use of SOAP, this study could be used as starting point for further projects, targeting the implementation of a generic service description for embedded measurement and control devices, in order to improve significantly remote operation and maintenance capabilities.

Next steps of research include

- Identification other potential devices and systems for the remote service capabilities
- Thorough implementation of the different scenarios together with AIP integration
- Further employment of the SOAP strategy
- Incorporate security issues like authentification or data encryption

6 References

[XML] http://www.w3.org/XML/

[SOAP] http://www.w3.org/TR/soap12-part1/

[WSDL] http://www.w3.org/TR/wsdl

[ARCFDSS] ARC Advisory Group, Field Device and Sensor Strategies for the E-World, February 2001

[ARCSWT] ARC Advisory Group, Software Trends for Automation, December 2000

[NEXUS] http://www.emsto.com/NEXUS/

[UTCAI] Invensys UTC for Advanced Instrumentation, http://seva.eng.ox.ac.uk/

[TCOPC] Functional Specification Thin Client OPC Data Web Services
WE-DOC-03955

[XMLDAOPC] Remote Service of Embedded Devices and Integration in IIT, The OPC Foundation,
Version 0.6 / 14.12.2001, OpcXML_060.doc