VERTICAL INTEGRATION OF TTP/A FIELDBUS SYSTEMS USING WEB SERVICES

Volker Turau, Marcus Venzke, Christoph Weyer and Yesenia Vigil
Technische Universität Hamburg-Harburg
Department of Telematics, Schwarzenbergstraße 95, 21073 Hamburg, Germany

Keywords: Fieldbus system, vertical integration, Web service, TTP/A

Abstract: This paper presents a generic technique to expose data and control of fieldbus systems to applications located at the level of operational management. To provide a high degree of interoperability between the operational level and different fieldbus systems we utilize standardized techniques such as XML, HTTP, and Web services which can be deployed independently of the platform. The proposed service is based on the interface file system for TTP/A smart transducers. The defined interface provides high level abstractions appropriate for the integration into business applications and considers the latency introduced by the internet protocols.

1 INTRODUCTION

In industrial automation systems the network link among different devices such as sensors, actuators and programmable logic controllers is provided by fieldbus systems. Examples of such systems are Profibus, CANbus and BITbus. In practice very often fieldbus systems from different vendors have to be combined to build a complete system, this task is called horizontal integration. In recent years there has been a growing interest in the use of information rooted in automation systems at the level of operational management to enhance production and its control (e.g. for supply chain management, enterprise resource planning and manufacturing execution systems). The provision of data controlled by fieldbus systems at this level is called vertical integration. It enables a large variety of applications outside the realm of the factory shop floor such as condition monitoring, fault diagnosis and predictive maintenance.

Research in this area has been spurred by the success of the internet, XML and related web technologies. We begin to see a tendency that organizations provide functionality over the web, no more employing wire protocols such as those of DCOM or CORBA. XML has become a preferred format for encoding and exchanging data in an open, system-independent way. Web services apply XML for remote software integration. To avoid not being caught in the trap of a hype, a careful analysis of the applicability of Web services for the task of vertical integration is required.

The main contribution of this paper is a requirements analysis for the usage of Web services for vertical integration of fieldbus systems and the concrete design and implementation of such a service for TTP/A fieldbus systems. The proposed service makes use of the recently approved OMG *Smart Transducer Interface* specification and especially of the concept of the *Interface File System* (IFS) (OMG, 2003).

2 INTEGRATION TECHNIQUES

The internet provides the ideal infrastructure to accomplish the vertical integration of fieldbus systems. Currently there is a shift towards web-enabling the industrial Ethernet protocols rather than enhancing them with real-time support. This stems from the perceived need by end users to implement homogeneous vendor environments even when a common higher-level protocol is employed. Companies are concerned that devices from different vendors may not interoperate, even those supporting the same protocol.

Integration of applications on the enterprise level is very often achieved through technologies based on remote procedure calls, examples are DCOM and CORBA. These technologies have also been deployed to the domain of vertical integration (OMG, 2003; OPC, 2003a) and for monitoring and configuration

Turau V., Venzke M., Weyer C. and Vigil Y. (2004).
VERTICAL INTEGRATION OF TTP/A FIELDBUS SYSTEMS USING WEB SERVICES.
In *Proceedings of the First International Conference on Informatics in Control, Automation and Robotics*, pages 390-393
DOI: 10.5220/0001129503900393

(Obermaisser et al., 2001). The problem is, that most higher-level applications do not implement the DCOM or CORBA interfaces necessary to communicate with servers. The success of the World Wide Web opened new opportunities for the vertical integration problem. Some projects exploit traditional Web application technology (Ferrari et al., 2002; Topp et al., 2003). Either through an embedded Web server or a gateway users can access fieldbus systems using a Web browser. While this approach is appropriate for monitoring applications, it is not sufficient for the integration problem where applications and not human users require access to fieldbus systems.

Web services are accessible over the Web and intended to be used by other applications. The output of a Web service is an XML document, a format that is easy to use for the client application and in fact, many vendors of our target applications are in the process of providing interfaces to integrate Web services. The inherent mechanism to handle faults of the underlying protocol SOAP enables an intelligent error handling. Descriptions of the services can be published into registries, which can then form the basis for service discovery tools.

XML has been used in association with fieldbus systems, but mainly to provide interoperability between different fieldbus systems (Eberle and Göhner, 2003). Use cases for Web services to access fieldbus systems are described in (Venzke and Pitzek, 2003). The most advanced usage of Web services in this area up-to-date is the OPC XML-DA specification by the OPC Foundation (OPC, 2003b). The proposed Web service resembles our approach, but there are also significant differences. While XML-DA is built on the OPC-COM interface, our work is based on the fieldbus protocol TTP/A and puts forward a platform independent open architecture. We also include mechanisms that we believe are indispensable for the area of vertical integration such as authentication and authorization.

Summarizing, we believe that Web services are an appropriate technology for the problem of vertically integrating fieldbus systems.

3 WEB SERVICES FOR VERTICAL INTEGRATION

There are many issues to be solved when designing a Web service for vertical integration. In this section we collect requirements and discuss alternatives. The design has to consider the environment in which vertical integration is carried out and the characteristics of the communication technique Web service.

It is commonly agreed that the goal of Web services is to provide high level abstractions. Because of the high latencies of calls to a service, they should have coarse grain interfaces. Querying the status of a register storing few bytes is certainly not a suitable task for a Web service. Also due to the characteristics of the protocols used for web services real time guarantees cannot be provided. This is not a severe restriction, because applications making use of vertical integration usually have no real time demands. We regard the following operations as candidates for a web service accessing a fieldbus system:

- Reading or writing lists of data items
- Reading aggregated or accumulated data
- Executing a sequence of operations that implement a semantically higher concept

Allowing access over the internet demands for a security concept to protect fieldbus systems. Only authenticated and authorized users should be accepted. In most cases authorization schemes will be derived from an enterprise wide security policy. To carry over this scheme to a web service demands for an interface providing an appropriate granularity.

Accessing fieldbus systems over the internet may fail due to a variety of reasons (ill posed requests, network failures, unavailable services etc.). Applications making calls to such services should be given a maximum amount of information in an error case to handle failures gracefully. If an operation has failed entirely, it should be answered with a SOAP fault message. Instead of application data it contains a well-known structure for error information.

For periodically reading the same set of data items, as required for monitoring, callbacks appear advisable. A client subscribes to a set of data items. The Web service then calls a client's operation for passing the data items whenever new values get available or in fixed time intervals. Using callbacks avoids polling by the client. However it makes clients more complex due to the need of server functionality. It cannot be applied if network address translation (NAT) or firewalls only allow connection setup in one direction.

If polling is used it can be optimised using timeouts. Calling a blocking poll operation allows a client to wait until new values get available. A maximum call time may be provided after which the operation is aborted with an error code, to ensure that control is given back to the client after a defined time. Providing a minimum call time makes the poll operation collect new values before returning them in bulk.

Web services can be accessed by many users in parallel, hence implementations of such services have to deal with concurrency problems. To guarantee the atomicity of bulk operations either requests have to be serialized or a more subtle transaction mechanism has to be implemented.

4 TTP/A FIELDBUS SYSTEMS

Fieldbus systems considered in this paper consist of smart transducers connected by a TTP/A bus. A smart transducer (ST) comprises hard- and software, consisting of a small, compact unit containing a sensor or actuator element, a micro-controller, a communication controller and the associated software. The TTP/A bus allows real time and non real time data exchange between STs.

A key feature of TTP/A fieldbus systems is the concept of an *Interface File System* (IFS) (Kopetz et al., 2000). It is a conceptual model for addressing data stored distributedly in the STs. STs have files consisting of 4 byte records. Records are identified with hierarchical names composed of identifiers for the fieldbus system, ST, file and record.

IFS supports three operations. With *Read* and *Write* a record's value can be retrieved or changed. *Execute* performs a node specific operation identified by the record's name.

According to (Kopetz, 2001) STs should support three interfaces. Using the *real time services interface (RS)* time sensitive information is accessed. The *diagnostic and management interface (DM)* is used for diagnostic purposes and to monitor STs. The *configuration and planning interface (CP)* allows configuring STs.

5 WEB SERVICE FOR FIELDBUS SYSTEMS

In this section we describe the core of our research: a Web service for the vertical integration of fieldbus systems based of the IFS specification. The design of the Web service interface follows the categorization of interfaces outlined in section 4.

5.1 General Architecture

The requirements analysis from section 3 and the design of the *Interface File System* led to the overall architecture depicted in figure 1 and presented in more detail in (Vigil, 2004). The core of the architecture consists of a generic implementation of the Web service. The Web service accesses the fieldbus system via a proprietary interface of the gateway.

The operations of the service are implemented in the service processor. The generic SOAP functionality is provided by a SOAP processor. It receives the calls from the Web Server, processes the SOAP messages and forwards the content to the service processor. To decouple the implementation of the web service from a particular gateway implementation, the concept of pluggable adapters is applied.

A cache is introduced for data periodically read by the client. The data can thus be returned immediately without accessing the fieldbus system.

To accommodate the fact that gateways may not support concurrent access to the fieldbus system, requests are serialized. The serializer component queues all requests and handles them in order.

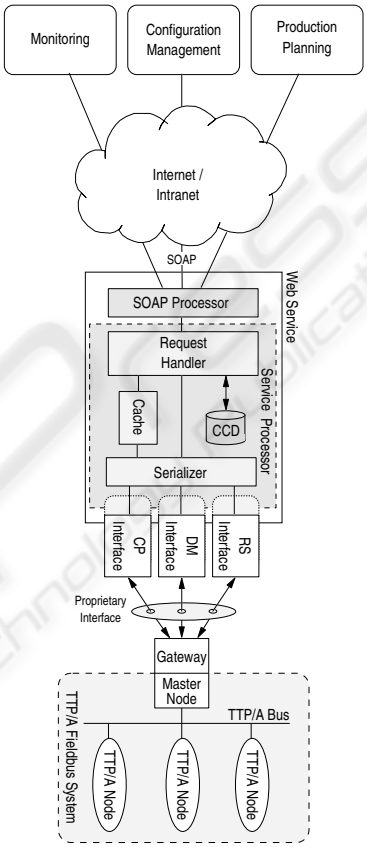

Figure 1: Overall architecture

5.2 Security and Sessions

Since the proposed service will be deployed in an enterprise environment, security is an important requirement. Therefore encryption based on the protocol SSL is used along with authentication combined with state-full sessions.

In order to access the TTP/A cluster the client must have authenticated itself at the Web service. The authentication is done via the operation *Login*. The client provides credentials identifying itself and specifies the desired conceptual interface (RS, DM, or CP). A state-full session is started and a session identifier is returned, which must be provided by the client in all future requests. Implementing a session mechanism in this style is independent of protocol specific

techniques such as cookies and thus improves the web service's interoperability.

5.3 Operations

The conceptual interface chosen for a session constrains the operations available for a client. Only those operations are available that are meaningful for the client's viewpoint and for which the user is authorized. All operations allow accessing several records with a single call providing a coarse granularity.

The operations *Read*, *Write* and *Execute* have similar meaning as the respective IFS operations.

The two operations *Subscribe* and *ReadSubscription* allow optimizing periodic reads of the same set of records. A client subscribes a set of records. The Web service returns a subscription identifier. It is provided in subsequent requests for the operation *ReadSubscription* that returns a response similar to the operation *Read*. Subscriptions increase efficiency because record names only need to be checked once and records can be cached and preloaded inside the service processor.

The operations *WriteConfiguration* and *ReadConfiguration* allow configuring a fieldbus system and retrieving its configuration respectively. A new configuration is provided in the request to the operation *WriteConfiguration* as XML data structure explained in (Pitzek, 2002). It is used for configuring the fieldbus system and stored in the Web Service for later retrieval via the operation *ReadConfiguration*.

Operations accessing several records (*Read*, *Write*, *Execute*, *ReadSubscription*) can fail partially. A record, file or fieldbus node might not exist or be inaccessible. In this case the operation still continues with the others. The SOAP response contains a sequence with one data structure per accessed record. To report the failure the data structure contains an error code.

5.4 Implementation

The described Web Service has been implemented as prototype based on Java and Apache Axis (Vigil, 2004). To be self-contained the prototype uses a simulation instead of a physical fieldbus system, implemented as pluggable adapters. The hierarchical addressing of IFS is mapped to a hierarchy of file system directories.

6 CONCLUSION

We have presented a requirements analysis for the usage of Web services for the vertical integration. This results in a loose coupling of all parties concerned and allows to adapt to new requirements gradually. Based on the requirements analysis we discussed a concrete design and implementation of such a service.

Currently we are investigating several ways to extend our research. Available metadata about the fieldbus system can be further exploited to enhance the quality of the service. It could be used to provide detailed information about the data returned by the operations. Furthermore, we plan to implement operations providing higher abstractions dedicated to a particular fieldbus system on this basis.

REFERENCES

Eberle, S. and Göhner, P. (2003). Adaptive Information Exchange with Field Bus Systems. In *Int. Conf. on Comp. Science and its Appl.*, Montreal, Kanada.

Ferrari, P., Flammini, A., Mariol, D., and Taroni, A. (2002). Web Integration of Fieldbus Networks: A Diagnostic Tool. In *ISIE'02, International Symposium on Industrial*, volume 1, pages 84–88. IEEE.

Kopetz, H. (2001). The Three Interfaces of a Smart Transducer. In *FeT'01, 4th IFAC Int. Conf. on Fieldbus Systems and their Applications*, Nancy, France.

Kopetz, H., Holzmann, M., and Elmenreich, W. (2000). A Universal Smart Transducer Interface: TTP/A. In *ISORC'00, 3rd IEEE Int. Symp. on Object-oriented Real-time distributed Computing*.

Obermaisser, R., Peti, P., Elmenreich, W., and Losert, T. (2001). Monitoring and Configuration in a Smart Transducer Network. In *Proc. IEEE Workshop on Real-Time Embedded Systems*, London, UK.

OMG (2003). *Smart Transducers Interface Specification*. Object Management Group (OMG). Version 1.0.

OPC (2003a). *Data Access Custom Interface Standard*. Openness Productivity and Connectivity Foundation (OPC). Version 3.0.

OPC (2003b). *OPC XML Data Access Specification*. Openness Productivity and Connectivity Foundation (OPC). Version 1.0.

Pitzek, S. (2002). Description Mechanisms Supporting the Configuration and Management of TTP/A Fieldbus Systems. Master's thesis, Tech. University Vienna.

Topp, U., Müller, P., Konnertz, J., and Pick, A. (2003). Web Based Service for Embedded Devices. In Chaudhri, A. B., Jeckle, M., Rahm, E., and Unland, R., editors, *Web, Web-Services, and Database Systems*, volume 2593 of *LNCS*, pages 141–153. Springer.

Venzke, M. and Pitzek, S. (2003). Accessing Fieldbus Systems via Web Services. In *WISES'03 – First Workshop on Intelligent Solutions in Embedded Systems*, Vienna University of Technology, Austria.

Vigil, Y. (2004). Web Service Interface and Architecture for Accessing Fieldbus Systems. Master's thesis, Technische Universität Hamburg-Harburg.