A Framework for XML-based Workflow Interoperability – The AFRICA Project

Michael zur Muehlen

University of Muenster
Department of Information Systems
Steinfurter Str. 109
48149 Muenster, Germany
ismizu@wi.uni-muenster.de

With the advance of electronic business relationships over the internet, the linking of cross-organizational business processes in virtual supply chains and other scenarios is rapidly increasing. Existing standards for the interoperability of information systems on the business process level are being adapted to suit the needs of the Internet economy. Especially the use of XML as a domain-independent encoding standard for business documents has led to the development of business frameworks such as BizTalk or open/EDI, and interoperability mechanisms that support these standards are being developed. In this paper we describe an architecture for the support of cross-organizational workflows through XML messages. This architecture has been implemented and tested within the AFRICA[1] project at the University of Muenster, Germany. While our work is based upon the emerging Wf-XML standard of the Workflow Management Coalition, it contains a number of significant enhancements that provide a secure, reliable management of global workflow processes.

1 From EDI to Inter-organizational Workflow Management

The interest in automated business-to-business transactions involving the Internet is increasing rapidly. The need for companies to expand the automated enactment of their business processes beyond the boundaries of their own organization is driven by the resulting savings in transmission time (automated exchange of process data over the internet, automated processing of work items upon receipt), gains in data quality (exchange of predefined documents, elimination of media breaks) and improved monitoring capabilities (up-to-date information about process status at the site of business partners). Even though proprietary EDI solutions have been in place since the early 1980s, the current movement towards electronic data interchange is fueled by the relatively inexpensive exchange of business data over the Internet. The development of the eXtensible Markup Language (XML) for the encoding of Internet traffic [1] fosters the development of vendor-independent frameworks that aim to standardize data schemas for commonly used business documents, such as invoices, delivery notes, purchase orders etc..

Workflow management technology supports the execution of business processes through the automated coordination of tasks, data, application logic and workflow participants (resources). While the use of workflow management technology inside the boundaries of single organizations is a well understood concept, and the number of implementations is steadily increasing [2,3,4], the use of workflow management for the coordination of inter-enterprise processes is still at a very early stage. Currently, workflow management systems are shifting from stand-alone applications towards embedded solutions, that are delivered as an inherent component of surrounding application systems, such as enterprise resource planning software (ERP systems) [5]. As a consequence, many organizations that currently have no workflow management system in place will have the option of automating part of their business processes using these embedded workflow applications without the necessity to purchase a separate workflow management system. In order to enable these companies to participate in interorganizational workflow settings, a prototype has been developed, that enhances existing workflow management systems with the capability to send and receive standardized XML messages for workflow interoperability. The system has been successfully tested in a helpdesk scenario and is currently being expanded to accommodate a number of different workflow management systems. After a discussion of related work in section 2, we discuss the design and implementation of the AFRICA prototype in section 3. The following section 4 gives an overview of the demonstration scenario implemented using the AFRICA prototype. The paper closes with a résumé and an outlook to future developments.

[1] A Flexible, Reliable, Intelligent Communication Architecture

2 Workflow Interoperability and XML

The AFRICA project touches two areas of research: Interoperability of workflow management systems on one side and XML-based business frameworks on the other side. In both areas, a number of standardization organizations and related projects can be identified. In section 2.1 we discuss the current developments of workflow interoperability standards, while section 2.2 deals with XML business frameworks. Section 2.3 lists academic research projects that are related to our approach.

2.1 Workflow Interoperability

In the area of workflow interoperability, a number of standardization efforts have been in place, namely by standardization organizations like the Workflow Management Coalition, the Object Management Group and the Internet Engineering Task Force.

2.1.1 Workflow Management Coalition Interface 4

The Workflow Management Coalition (WfMC) was founded in 1993 as a non-profit organization to foster the distribution and standardization of workflow technology. It currently has more than 220 members, that contain workflow vendors, users, consultants and academics. The reference model of the WfMC identifies five functional interfaces, that connect a workflow management system with external application systems [6]. Interface 4 deals with the interoperability of different workflow management systems. The interface definition consists of an abstract specification of the API calls, that can be used in order to instantiate a workflow on a remote workflow management system, to change the execution status of a workflow instance or the query the data processed in a remote workflow instance [7]. Instantions of the abstract Interface 4 specification that relate to a specific message encoding scheme (e. g. HTML, MIME, etc.) are called bindings. So far, only a MIME binding has been published by the WfMC [8]. The applicability of this specification has been demonstrated in an interoperability challenge [9] that was carried out in March 1999. Due to the increasing interest in XML message encoding and the use of HTTP as a transport mechanism, most of the WfMC work is now focused on the Wf-XML specification (cf. section 2.1.4).

2.1.2 Object Management Group Workflow Facility

The Object Mangement Group (OMG) is a non-profit organization that deals with the with the standardization of object-oriented software development and design concepts. It currently consists of more than 800 members and has been established in 1989. The main work of the OMG is designed around the Object Management Architecture (OMA) that prescribes a way, how object-oriented components can interact in a heterogeneous environment, using a Common Object Request Broker Architecture (CORBA) for message transfer and service invocation. As part of the CORBA framework, the OMG has standardized a facility that provides workflow services through an object request broker [10]. The OMG Workflow Facility describes a set of workflow execution objects and their respective interfaces that can be used for workflow interaction in business object environments. The Workflow Facility standard has been officially released by the OMG, but adjacent components are still awaiting standardization, such as a resource assignment interface for the association of workflow participants to workflow activities [11] and a process modeling standard for the design of workflow processes [12].

2.1.3 Simple Workflow Access Protocol

The Simple Workflow Access Protocol (SWAP) was created in 1998 through an industry consortium under the auspices of Netscape, Oracle, SUN [13, 14]. SWAP deals with the control of asynchronous services over the internet. Since a workflow instance can be perceived as a long lasting sequence of discrete process steps, having a designated starting and ending, SWAP can be used for the control of a workflow instance through a remote control instance. The SWAP-specification, which is still in draft-status, uses the HTTP and XML for the exchange of control information. The original industry consortium has handed over the SWAP specification to the Internet Engineering Task Force (IETF) for standardization. However, progress has been slow with the standardization of SWAP. Since the pressure for an interoperability standard using XML is mounting, the Workflow Management Coalition has adopted the basic ideas of SWAP and merged them with an XML binding of the WfMC Interface 4 specification. The result is the forthcoming Wf-XML standard [15], which is described in the next section.

2.1.4 Wf-XML

Wf-XML [15, 16] is a new standard for workflow interoperability, that is being developed through a WfMC working group. It combines the basic idea of SWAP, namely the interaction of workflow management systems based on the exchange of XML messages, with the abstract commands defined by the WfMC Interface 4 standard. Major workflow vendors have signaled their support for this standard, which is due to be released as Version 1.0 by the middle of the year 2000. Wf-XML defines a set of request/response messages that are exchanged between an observer (which may or may not be a workflow management system) and a workflow management systems to control the execution of a remote workflow instance:

`CreateProcessInstance` instantiates a new workflow instance within a remote workflow management system.

`ChangeProcessInstanceState` manipulates a workflow instance on a remote system (starting, suspending, terminating the remote instance etc.).

`GetProcessInstanceData` requests the status of the remote workflow instance.

`ProcessInstanceStateChanged` signals to the requesting party (i. e. the observer of the workflow process) that the remote process has been completed and passes the result data to the requesting party.

Due to the broad support of vendors and the proximity of standardization, Wf-XML was chosen as the message format for the AFRICA prototype.

2.2 XML Business Frameworks

Recently a number of standardization organizations for XML business frameworks have appeared on the Internet. These organizations aim at the standardization of business documents that are passed from one participant to the next in an interorganizational business process. This development can be seen as the successor to the standardization of the EDIFACT format in the 1970s. The most well-kown organizations of this kind are BizTalk and RosettaNet.

The *BizTalk* forum was created by Microsoft in 1998 and aims at the definition of guidelines for the publication of XML schemas by independent vendors [17]. Furthermore, the use of XML messages for the integration of software systems is propagated by the Simple Object Access Protocol (SOAP), which is an XML/HTTP-based protocol for the platform-independent access to services, objects and servers over the Internet.

The *RosettaNet* consortium was formed by various manufacturers and suppliers of hard- and software in order to standardize supply chain processes with the IT industry domain [18]. The RosettaNet specification defines Partner Interface Processes (PIP) for various supply chain processes such as management of purchase orders, product and technical data interchange, and order status handling. The RosettaNet specification has been successfully implemented by a number of software vendors, e. g. NetFish Inc. [19].

Besides BizTalk and RosettaNet a number of other organizations can be identified, that aim at standardizing XML-based business documents, such as Open/EDI, OBI, CommerceNet and the Open Trading Protocol Consortium.

2.3 Research Projects

The *Interworkflow Project* at the Kanagawa Institute of Technology, Japan, focuses on the definition of a global workflow model for an interorganizational business process [20, 21]. This global workflow model defines the basic interaction between the parties involved and is then transferred into the workflow management systems of the parties involved. Within these systems, the (local) processes are modified to suit the needs of the individual enterprises, while leaving the defined points of interaction intact. During the enactment of the interorganizational workflow, both parties use the WfMC Interface 4 MIME binding for communication.

The ESPRIT *CrossFlow* project [22] is dealing with contract-based workflow interoperability between business partners. In this project, business relationships between a customer and a provider of services are modeled using contracts. The project has been established in 1998 and uses an insurance and a logistics scenario for the demonstration system interoperability.

3 The AFRICA Prototype

The project AFRICA was initiated at the University of Muenster, Germany, in October 1999. The aim of the project was to build a reliable infrastructure for business-to-business workflows, using XML for the encoding of the messages. The focus was on incorporating complex, non-sequential process models involving multiple partners and the integration of a global monitoring service. Since the project team had a number of commercial workflow management systems available for testing and integration purposes, it was decided not to implement a workflow engine with interoperability features, but instead an add-on component, that can be added to existing workflow installations. A reference implementation of this wrapper, written in the C++ language and using the XML format described below, was created, clearly demonstrating the potential of XML-based process communication.[2]

[2] See http://pcwi501.uni-muenster.de/africa/index.html

3.1 Design Rationale

For the design of the AFRICA prototype, a number of design principles were employed, to make the system usable in a large number of contexts. These principles were system independence, reusability, security and support for processes with more than 2 involved parties.

System independence: The AFRICA prototype should enable companies to participate in cross-organizational workflows without modifying the workflow management systems already in place. For this reason, the prototype was implemented as a wrapper that sits on top of an existing application system and encapsulates the Wf-XML message handling from the underlying system. Vendor supplied APIs are used to access the respective systems, leaving the system integrity untouched. Furthermore, this approach fosters the migration of AFRICA to a number of different systems with relatively little effort.

Reusability: The AFRICA prototype was designed with the goal of using as much of the system code as possible in different environments. Therefore, a three tier architecture was developed, separating the transport layer, process logic layer and abstraction layer. For changing transport protocols or security mechanisms, only the transport layer needs to be adjusted, while during the migration to a different workflow management system only the abstraction layer is changed accordingly, leaving the transport and process layer intact. A detailed discussion of the architecture can be found in section 3.2.

Security: Communication between two AFRICA-enabled systems should be secure and reliable. In order to achieve this, additional information have been inserted in the transport section of the Wf-XML messages that are evaluated by the transport layer of the wrapper.

Support for n-party processes: While most interoperability frameworks focus on the peer-to-peer interaction between two business partners, the goal of the AFRICA project was the support for an arbitrary number of involved parties (e. g. a supply chain with several suppliers, a manufacturing enterprise, a transport company and a retailer). In order to maintain the overall consistency of the process as well as provide monitoring information about a global process regardless of the local enactment, a GlobalProcessID was introduced to the Wf-XML messages.

3.2 Message Format

The AFRICA wrapper uses an extended Wf-XML format for the exchange of messages. Each message consists of the four parts Transport, Security, Header and Body.

```
<?xml version="1.0"?>
<WfMessage>
   <WfTransport>
      ...
   </WfTransport>
   <WfSecurity>
      ...
   </WfSecurity>
   <WfMessageHeader>
   ...
   </WfMessageHeader>
   <WfMessageBody>
      ...
   </WfMessageBody>
</WfMessage>
```

Figure 1. Overall message structure

The `WfTransport` section groups those elements that are relevant at transportation time, before the message reaches its eventual recipient, i.e. a local process instance, e. g. the sender and recipient of the message as well as a correlation key for the identification of request/response pairs.

The `WfSecurity` section contains a unique identifier for each message and a timestamp. This information is used to identify lost, obsolete or intercepted messages and to acknowledge the receipt of the message by the transport layer (see section 3.3).

The `WfMessageHeader` section contains – different from the original Wf-XML standard – the global process identifier that this message relates to. Each wrapper only needs to know the mapping between the global process and its own local process instances, but does not need to keep track of the local naming schemas of other involved parties. The header section also contains the identifier of the operation to be executed in order to enable the preprocessing of this information.

The `WfMessageBody` section contains the details about the operation to be executed as well as the context data, i. e. the data that gets passed to the local workflow management system for further processing. We assume no predefined structure of the context data, this way, data schemas that have been standardized by other organizations (cf. section 2.2) can be inserted here.

In addition to the operations defined by the Wf-XML standard, a number of additional operations were introduced, in order to facilitate global process management and the handling of monitoring information. These operations are PassProcessInstance, GetHistory and Notify.

`PassProcessInstance` hands the control of the global process over from one party to the next. The sender switches into the state “suspended” and can be activated again, when his wrapper receives another PassProcessInstance command. If a local process instance exists, the wrapper sets the state of this instance to active.running. If no local instance for the global process exists, the wrapper instantiates a new local instance, starts it and updates its lookup table with the local process ID.

`GetHistory` requests monitoring information from a remote party. If the remote party has passed the process control to several other parties, the command is cascaded until the currently active party returns information about its current process status. The parties located in the middle between the sender and the final recipient of the command add their own process status information and pass a merged set of data on until it finally reaches the sender of the GetHistory command.

`Notify` actively “pushes” process status information to the observer of a remote process instance. If a remote party sees the necessity to inform the observer about certain events, the Notify command is used.

3.3 Technical Architecture

In order to facilitate reuse and encapsulation of information from lower levels the AFRICA wrapper was designed using a three-tier architecture and consists of a transport layer, a process logic layer and an abstraction layer. Figure 2 illustrates the architecture of the system, using an incominc Wf-XML message as an example.

The *transport layer* handles the reliable and secure transfer of Wf-XML messages between AFRICA-enabled information systems. The prototype uses TCP/IP port-to-port communication and open SSL encryption for security measures. If other transport protocols or security mechanisms are to be used, the transport layer has to be adjusted accordingly, but the process logic and abstraction layer remain untouched. When a message is received by the TCP Socket Listener, it is forwarded to the Open SSL Decryption unit, where the message is decoded. The plain-text XML message is then handed over to the security validation mechanism, that evaluates the timestamp and the security token of the message. The security token is sent back to the originator of the message to acknowledge the receipt. The wrapper keeps a backlog of the messages that were received during the last 20 minutes. Older messages and messages with an identical security token to an already received message are discarded as potential duplicates. The XML message is then sent to the process logic layer for further processing.

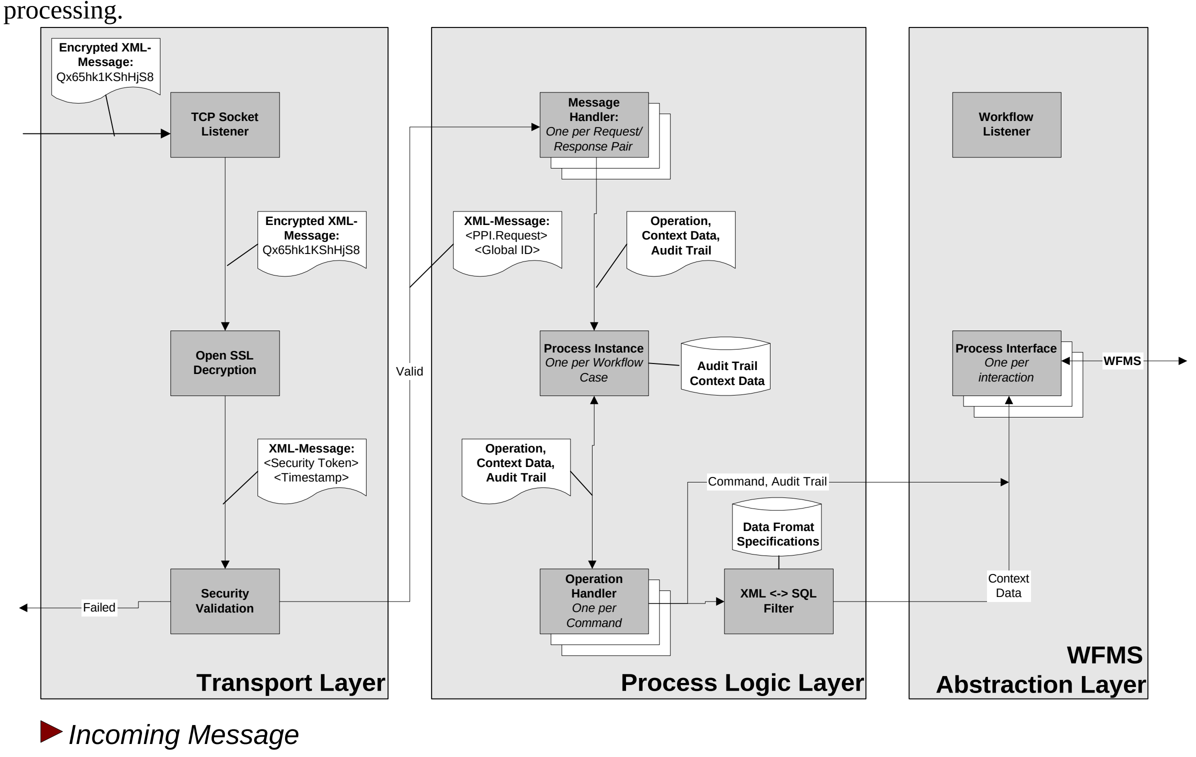

Encrypted XML-Message: Qx65hk1KShHjS8
TCP Socket Listener
Encrypted XML-Message: Qx65hk1KShHjS8
Open SSL Decryption
XML-Message: <Security Token> <Timestamp>
Security Validation
Failed
Valid
Transport Layer
Message Handler: One per Request/ Response Pair
XML-Message: <PPI.Request> <Global ID>
Operation, Context Data, Audit Trail
Process Instance One per Workflow Case
Audit Trail Context Data
Operation, Context Data, Audit Trail
Command, Audit Trail
Data Fromat Specifications
Operation Handler One per Command
XML <-> SQL Filter
Process Logic Layer
Workflow Listener
Process Interface One per interaction
WFMS
Context Data
WFMS Abstraction Layer
Incoming Message

Figure 2. Three tier architecture of the AFRICA prototype (shown for an incoming message)

Within the *process logic layer*, the XML message is parsed, the Wf-XML command is separated from the context data of the message and a standardized call of the abstraction layer API is issued. After the receipt of the message from the transport layer, the Message Handler component determines, whether the message is a response to a request issued earlier (then the appropriate instance of the Message Handler is identified) or if the message is a request itself (in which case a new Message Handler is instantiated). The Message Handler extracts the Wf-XML command from the message and sends it with the context data and audit trail information to the Process Instance handler. This component reads the GlobalProcessID of the message and matches it with the local process instance that exists within the local workflow management system. It translates the global command into a local instance specific command and passes it on to a new instance of the Operation Handler. The Operation Handler then transforms the context data into the format required by the local workflow schema and passes the command and the context data through an API call to the abstraction layer.

The *abstraction layer* encapsulates proprietary API calls to the workflow management system of a specific vendor and exhibits a standardized API to the process logic layer. In the AFRICA protoype, one of the abstraction layers implemented interacts with the Business Workflow component of an SAP R/3 system. Within this scenario, a command and context data received from the process logic layer are translated into SAP specific remote function calls, that invoke the appropriate methods on a workflow business object within the SAP system. In order to accommodate workflow management systems of different vendors, only the abstraction layer needs to be changed, leaving process logic layer and transport layer untouched.

The process of sending out a Wf-XML message through the AFRICA wrapper is exactly reverse to the process described above: The workflow system calls out to the abstraction layer, the process logic layer transforms the command and the context data into a well-formed Wf-XML message and integrates the GlobalProcessID, and the transport layer adds security information, encrypts the message and sends it to the appropriate recipient.

4 Implementation Scenario

The AFRICA prototype was implemented using a 2-stage helpdesk scenario as an example. Within this scenario, a client has contracted an external service provider to peform helpdesk tasks, in case an employee of the client encounters problems that relate to soft- or hardware used by the client enterprise. The helpdesk provider then tries to solve the problem using his internal knowledge base. In some cases, additional information from the person that originated the workflow may be required (e. g. for clarification reasons), in this case a request for additional information is sent back to the client enterprise. If the helpdesk is able to provide a solution (with or without the additional information), this solution is sent back to the client, who may either accept the proposed solution or send it back to the helpdesk for further refinement.

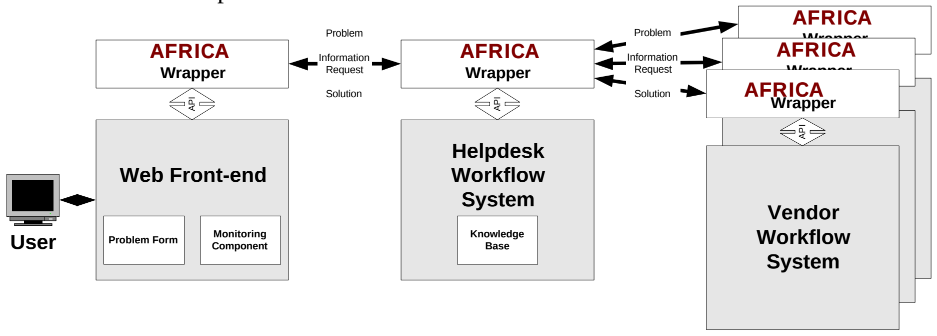


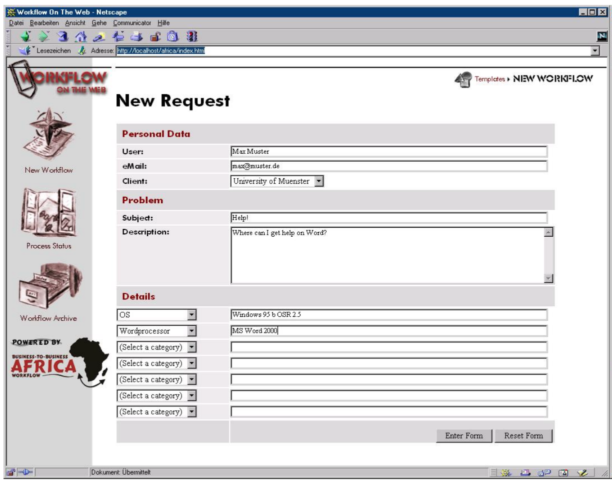
Figure 3. Helpdesk scenario

If the helpdesk is unable to find a solution to the given problem, it may send the problem description (with possible enhancements) to one or more soft- or hardware vendors, that supply second-level support. In some cases, the vendor may require additional information from the client. In this case, he contacts the helpdesk which either provides the information directly or in turn contacts the client. Finally, a solution is sent from the vendor to the helpdesk and is passed on to the client for acceptance or rejection. The helpdesk may choose to collect solutions from all vendors that have been contacted initially, or to notify these vendors that their services are no longer needed and their local workflow instances may be terminated. Figure 3 gives an overview of the implementation scenario.

The web front-end used in the scenario is a small workflow engine designed specifically for the front-end operations of entering problem data and displaying the status of global process instances. This system uses API calls to communicate with the AFRICA wrapper, that translates the HTML form data entered by the user into the Wf-XML message format and adds the appropriate command structure. The workflow systems on the helpdesk and vendor side were simulated using two separate SAP R/3 4.5 B systems. The abstraction layer of the wrapper used proprietary SAP remote function calls to create workflow instances within the embedded Business Workflow component of the SAP systems.

The client application of the helpdesk scenario can be used using only a web browser. Data entry and requests are handled via database-driven web pages. The interaction between the web front-end and the AFRICA wrapper is handled via server-side API calls, transparent for the user. Figure 4 shows the data entry screen for the workflow process. The user has the option to enter a freetext description of the problem she is faced with and may add an arbitrary number of name/value pairs to give technical information about the problem encountered. This information is encoded in XML and inserted into the ContextData section of the Wf-XML-Message.

Figure 5 shows the monitoring of a remote workflow instance through the web interface. The Step ID indicates the sequence of activities that have been performed in the overall process. While in step 1 the problem was entered using HTML form, the data was then sent to the helpdesk (step 2). The responsible workflow participant at the helpdesk site was unable to solve the problem and sent the problem description to the vendor site, where a developer is working on a solution and has given an estimate of the processing time necessary to complete the activity. From this screen, the user can review or update the problem description (which is automatically cascaded through the helpdesk to the vendor site). It is also possible to actively request the audit trail of the process.


Workflow On The Web - Netscape
Datei Bearbeiten Ansicht Gehe Communicator Hilfe
Lesezeichen
Adresse: http://localhost/africa/index.html
WORKFLOW ON THE WEB
New Workflow
Process Status
Workflow Archive
POWERED BY BUSINESS-TO-BUSINESS AFRICA WORKFLOW
Templates ▸ NEW WORKFLOW
Enter Form
Reset Form
Dokument: Übermittelt

New Request

Personal Data

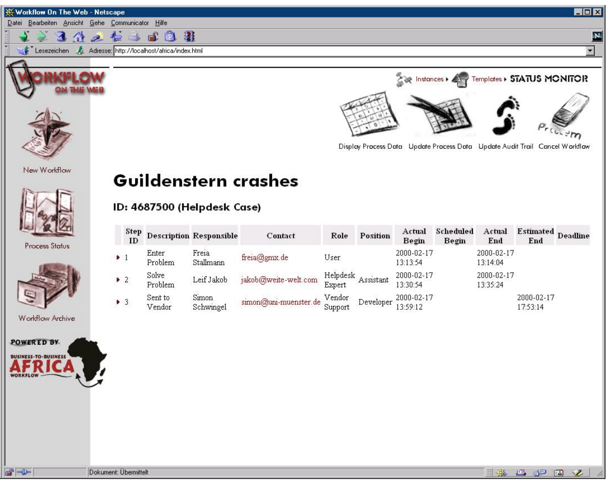
User:	Max Muster
eMail:	max@muster.de
Client:	University of Muenster

Problem

Subject:	Help!
Description:	Where can I get help on Word?

Details

OS	Windows 95 b OSR 2.5
Wordprocessor	MS Word 2000
(Select a category)	
(Select a category)	
(Select a category)	
(Select a category)	
(Select a category)	


Workflow On The Web - Netscape
Datei Bearbeiten Ansicht Gehe Communicator Hilfe
Lesezeichen
Adresse: http://localhost/africa/index.html
WORKFLOW ON THE WEB
Instances ▸ Templates ▸ STATUS MONITOR
Display Process Data
Update Process Data
Update Audit Trail
Cancel Workflow
New Workflow
Process Status
Workflow Archive
POWERED BY BUSINESS-TO-BUSINESS AFRICA WORKFLOW
Dokument: Übermittelt

Guildenstern crashes

ID: 4687500 (Helpdesk Case)

Step ID	Description	Responsible	Contact	Role	Position	Actual Begin	Scheduled Begin	Actual End	Estimated End	Deadline
▸ 1	Enter Problem	Freia Stallmann	freia@gmx.de	User		2000-02-17 13:13:54		2000-02-17 13:14:04		
▸ 2	Solve Problem	Leif Jakob	jakob@weite-welt.com	Helpdesk Expert	Assistant	2000-02-17 13:30:54		2000-02-17 13:35:24		
▸ 3	Sent to Vendor	Simon Schwingel	simon@uni-muenster.de	Vendor Support	Developer	2000-02-17 13:59:12			2000-02-17 17:53:14	

Figure 4. Entry of a helpdesk problem description

Figure 5. Monitoring of a remote process instance

In this case, a GetHistory command is issued, forcing each party of the overall workflow to signal the status of the local workflow instances. Otherwise, only those events that are actively published by the participating parties are displayed on the audit trail page. Another option for the workflow originator is the canceling of the workflow instance, which leads to a cascaded termination of workflow instances, that have been created at vendor sites by request from the helpdesk.

5 Summary and Outlook

In this paper we have presented the AFRICA framework for business-to-business workflow applications. It combines a standardized, extensible message format with a flexible and adaptable technological architecture. The applicability of this framework has been demonstrated by the implementation of a prototype scenario using commercial workflow management systems and a custom-made web-front-end. The extensions to the existing Wf-XML framework have been submitted to the WfMC for consideration in the next version of the Wf-XML standard.

In the future our framework will be extended to accommodate a number of different workflow products. We are currently investigating the automated negotiation of communication parameters such as security mechanisms and protocol standards between AFRICA-enabled workflow systems as well as the automatic mapping of XML context data schemes into the proprietary format of the underlying workflow management systems.

Acknowledgements

The author would like to thank all members of the AFRICA project team, Leif Jakob, Florian Klein, Lutz Michaelsen and Simon Schwingel for their effort. Furthermore, we would like to thank the members of the WfMC Working Group 4, especially Rainer Weber and Keith Swenson for many useful comments on the AFRICA XML message format.

References

[1] Walsh, Norman: A Technical Introduction to XML. http://www.xml.com/xml/pub/98/10/guide0.html.
[2] Fischer, Layna; Moore, Connie: Excellence in Practice: Innovation and Excellence in Workflow and Imaging, Lighthouse Point 1997.
[3] Fischer, Layna: Excellence in Practice: Innovation and Excellence in Workflow and Imaging Vol. II, Lighthouse Point 1999.
[4] Fischer, Layna: Excellence in Practice Vol. III: Innovation & Excellence in Workflow Process and Knowledge Management, Lighthouse Point 2000.
[5] zur Muehlen, Michael; Allen, Rob: Autonomous and Embedded Workflow Management. A Workflow Management Coalition Classification. White Paper, Lighthouse Point, March 10, 2000. (available from http://www.wfmc.org)
[6] Workflow Management Coalition: Reference Model. Document Number WFMC-TC-1003, Brussels, 1995.
[7] Workflow Management Coalition: Workflow Standard Interoperability – Abstract Specification. Document Number WFMC-TC-1012. Version 2.0b. Winchester, 1999.
[8] Workflow Management Coalition: Workflow Standard Interoperability – Internet e-mail MIME Binding. Document Number WFMC-TC-1018. Version 2.0b. Winchester, 1999.

[9] Workflow Management Coalition: Interoperability Proving Framework. Document Number WFMC-TC-1021. Version 2.0b. Winchester, 1999.

[10] Object Management Group: Workflow Management Facility, Document Number bom/99-03-01. Framingham 1999. ftp://ftp.omg.org/pub/docs/bom/99-03-01.pdf

[11] Object Management Group: Resource Assignment Interface – Request for Proposals. Document Number bom/00-02-19. Framingham 2000.

[12] Object Management Group: Workflow Process Definition – Request for Proposal. Document Number bom/00-03-NN. Framingham 2000.

[13] Swenson, Keith: Simple Workflow Access Protocol (SWAP) Internet Draft (work in progress). (available from http://www.ics.uci.edu/~ietfswap/SWAP9807.html.

[14] Bolcer, Gregory Alan; Kaiser, Gail E.: Leveraging the Web To Manage Workflow. Collaborative Work: SWAP. IEEE Internet Computing, 3 (1999) 1, pp. 85-88.

[15] Workflow Management Coalition: Workflow Standard Interoperability – Wf-XML Binding. Document Number WFMC-TC-1023. Draft 0.9.2. (available from http://www.wfmc.org)

[16] Hayes, Jim; Perovian, Effat; Sarin, Sunil; Schmidt, Marc-Thomas; Swenson, Keith; Weber, Rainer: Wf-XML: Standards-based workflow interoperability for the Internet. To appear in: IEEE Internet Computing 4 (2000) 2, April 2000.

[17] BizTalk. http://www.biztalk.org

[18] RosettaNet. http://www.rosettanet.org

[19] Netfish, Inc. http://www.netfish.com

[20] Hayami, Haruo: Short Report on the Development and Experimental Proof of an Interworkflow Management System. Presentation at the Workflow Management Coalition Meeting in Vienna, February 1999. (available from http://www.wfmc.org)

[21] Hayami, Haruo: Development and Experimental Proof of an Interworkflow Management System. Presentation at the Workflow Management Coalition Meeting in Tokyo, December 1999. (available from http://www.wfmc.org)

[22] The CrossFlow Project. http://www.crossflow.org