A SOFTWARE ARCHITECTURE FOR INTEGRATED LOGISTIC MANAGEMENT SYSTEM

Fabrizio Simeoni*, Srecko Maksimovic*, Walter Geretto* George Georgoulas, Chrysostomos Stylios****

*Teorema Engineering Srl, Area Science Park Basovizza, Trieste, Italy
**Technological Educational Institute of Epirus, GR-47100, Arta, Greece

Fabrizio.Simeoni@teorema.net; Srecko.Maksimovic@teorema.net; Walter.Geretto@teorema georgoul@gmail.com; stylios@teiep.gr

ABSTRACT
This paper presents an integrated software architecture for the management of transfer goods (trucks and containers) between port and dry-port facilities. This system is a large scale system and it has to deal with huge and continuously updated set of information. The required information is gathered from other information systems managing business activities within the involved areas. This is the reason why there was required the development of software components that are able to manage the processes of acquiring and sending information to the other systems. In this paper the technological choices and the main information flows managed are described.

Keywords: .NET - BizTalk - Marketplace – On Board Computer - RFiD tag – SAIL - SOA – SQL Server-Subscription – Visual Studio 2010 – WCF – Web service – XML.

1. INTRODUCTION

In this paper we are presenting the characteristics of the implementation of the software solution developed within the "ICT System addressed to Integrated Logistic Management and Decision Support for Intermodal port and dray port facilities, SAIL project. The project aims at developing an integrated ICT tool able to support efficiently logistic chain of goods flow and all business operations provided in the port and the dry port areas, mainly related to the transfer between the two facilities (Caris et al., 2008; Turban et al., 2010).

Within the SAIL project, we are designing and implementing the planning and scheduling tools for optimizing the transfer of goods and the sizing of the resource capabilities. To achieve this goal, we have to gather information from other systems in order to know how many transfers are foreseen, requested, when they have to be executed, etc., in other words all the operational data needed by our algorithms to provide reliable results.
The present paper contains a description of the architecture of the implemented solution for the central hub, which is the central SAIL module, aimed to provide the functionalities for data exchange among sub-modules.

We define the development environments chosen, programming languages, and the technological choice of BizTalk Server 2010 as the integration tool due to its flexibility and modularity. Moreover we describe the details of the implementation of web services and the routing logic of messages. Microsoft technology has been used as it is the core of Teorema activities.

The rest of the paper is structured as follows: Section 2 presents the overall software architecture. The communication flows are described in Section 3 and Section 4 concludes the papers and presents future developments.

2. ARCHITECTURE

2.1 Programming Languages

The chosen programming language is C # and the software development is based on version 4 of .NET Framework (Figure 1) (Troelsen 2012). The framework libraries for WCF technology have been very important to the development of web services that allows achieving a high level of interoperability and communication with other software modules implemented with technologies other than .NET. Another important aspect is the Workflow Foundation that is the technology on which the designer offered by the SDK (Software Development KIT) of BizTalk Server is based: it was used to define the data orchestrations that implement the logic for message routing through the central hub (Rosanova, 2011).

2.2 Development environment

The chosen development environment is Visual Studio 2010 Ultimate Edition (Randolph *et al.,* 2010). This is the latest Integrated Development Environment (IDE) developed by Microsoft for programmers who develop for Windows and .NET Framework 4.0. It allows the utilization of multiple programming languages, including VB.NET, C ++, C# and others. It also offers the ability to create applications and ASP.NET Web Services in C# or VB.NET (Pathak, 2011). BizTalk Server provides developers with an SDK as an extension for the Visual Studio projects using visual tools.

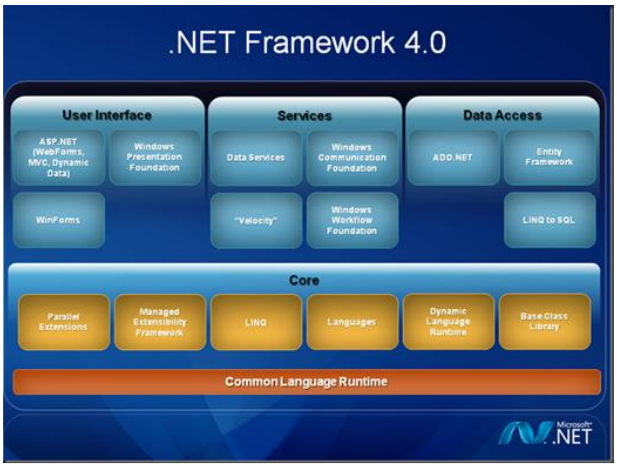

Fig. 1. The .NET Framework 4.0 structure.

2.3 Database server

The engine Database Management System (DBMS) is SQL Server 2008 R2. It is a Relational Database Management System (RDBMS) produced by Microsoft. In early versions it was used for medium-sized databases, but starting from version 2000 it is used also for management of large data bases.

2.4 BizTalk Server

2.4.1 The product

Microsoft BizTalk Server is a middleware (management layer of business logic) that is specializes in the management of business processes and their integration. It is a Business Process Management (BMP) system, i.e. a system that the company uses to create a layer of management of business flows between applications being able to apply existing rules and customizable parameters and can be monitored through a Business Activity Monitoring (BAM) system. In connecting applications across the enterprise, BizTalk Server is an EAI Enterprise Application Integrator (EAI) and also a Message Broker as it converts the native formats of the applications it connects via a system of adapters. BizTalk Server can function as Enterprise Service BUS (ESB) to create Service Oriented Architecture (SOA) infrastructure in which we can grow a forest of services both internal to the company and external - by mechanisms like SaaS (Software As A Service) or Cloud Computing. An ESB manages the flow of information between applications such as exchange of messages, receiving, processing and delivering it, based on metadata associated with the message itself that define the set of operations the message has to go through.

A highly common scenario is that BizTalk is used in companies after having purchased/built systems such as IBM Mainframe, SAP, Navision, web applications, asp.net, jsp/jsf, php, various services, SOAP, etc.. The company faces the need to make these systems coexist, in order to keep data synchronized between the different systems, and even to let a complex operation become feasible on multiple systems simultaneously without writing an application from scratch specifically for such functionality.

One of the requirements of the SAIL system is the interaction between different software modules within the port-dry port system and at the same time to allow integration of both internal and external processes within the business activities to enhance their flexibility and interoperability.

The central hub abstracts the connection between the service and the transport, trying to make it neutral. The solution is actually constituted by a series of processes that can be reused in many other realities, also extremely diversified. Using BizTalk Server provides the infrastructure that enables these processes, and entities related to these data, to be easily inserted into a SOA and they can be used for many other workflows and services within the port.

In the context of SOA, all documents can be transferred safely, while validating the content. Thanks to the reliability of BizTalk, in case of an error, the messages and processes can be completely recovered and the state management is guaranteed.

2.4.2 Operation principles

BizTalk Server architecture is based on the concepts of publication and subscription of messages and is built entirely using XML as a mechanism for data representation. The operation principles of BizTalk are shown in Figure 2 and described in the following paragraphs.

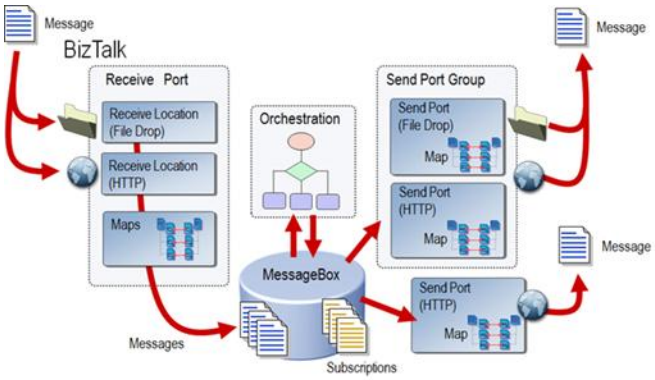

Fig. 2. The operation principle of BizTalk.

Messages are received through the receive port. Each receive port includes more receive locations associated with an adapter that allows communication with a particular type of external entity. The received message goes through a pipeline that takes care of operations like decoding, disassembling, validation and identification of partners. The messages can then be processed in a particular xslt map and then be published in the Message Box. The Message Box is a database for BizTalk internal use: it has a large number of tables many of which are used to store received messages. Each message has associated metadata called Message Context and each element is maintained by a key/value pair called Context Property. Every message that enters in the Message Box is "*published*". There is a rule according to which the posted messages cannot be changed.

The subscription is a mechanism by which doors and orchestrations are able to receive and send messages. Every process running BizTalk Message Agent has a Message Agent that looks for messages responding to subscriptions and occur to subscriptions and routs them to the EPM and sends them where they have to go. EPM is the broker between the Message Box and the pipeline/port/adapter. Subscriptions of orchestration are handled by a different service sub-called XLANG/s. A subscription is a collection of statements of comparison known as predicates that include context property of messages.

Subscriptors who have a valid subscription to a message can receive it and send it to the orchestration or send port. The sending process is similar to the receiving process. Messages sent to a send port can be transformed according to a map and then go through a send pipeline in which the steps are executed for validation, assembling and coding. Eventually, the adapter associated with the send port will transform the xml message in the compatible format, based on the type of adapter.

2.4.3 Monitoring

BizTalk activities can be monitored using the Health and Activity Tracking technology (HAT), and directly through the solutions of the Office suite or Share Point Portal Server 2003 with BAM. The central hub can include a BAM portal ready for use, which allows users to easily examine and configure BAM information. Using the BAM portal, users can select a particular instance of the business processes to be monitored, and then choose a specific BAM view into the process to get a different perspective of key performance indicators monitored. Users could be allowed to receive BAM information such as notifications by e-mail or other communication channels, supporting decision-making processes in real time. You can also expose Web services queries on aggregated data and instances, create alerts and retrieval of BAM configurations. The interface for the Web service can then be used to expose the functionality of the BAM portal interface.

2.4.4 Reliability and scalability

The high reliability of the platform is guaranteed by BizTalk Server that provides this capability natively. The central hub is capable of operating in high reliability and load balancing. Balancing the load network guarantees optimal performances, and the ability to add more servers if necessary and perform maintenance on any server without neither consequences for users nor hardware configuration changes. Through the combination of network load balancing and redundant clusters, administrators can add and remove cluster with minimal impact on users.

2.5 Web services and communication

SOA is a software architecture designed to support the use of services (web services) to meet the requirements of users in order to allow the use of single applications as part of the full business process. As part of a SOA architecture, it is possible to modify, in a relatively simple manner the mode of interaction between services with a very flexible design, making dynamic the combination in which the services are used in the process: in this way it is easier to add new services and change processes to meet specific business and process needs. The flow of information is not bound by a specific platform or by an application but can be considered as a component of a larger process, and then reused or changed.

3. *COMMUNICATION FLOWS*

In this section, flows are described, that are managed through the software architecture described in the previous section.

3.1 The overall system

The system consists of five distinct software modules: 1) Central Hub (HC), 2) Gateway3) Emergency Management Component (EMC), 4) Marketplace and 5) Scheduler

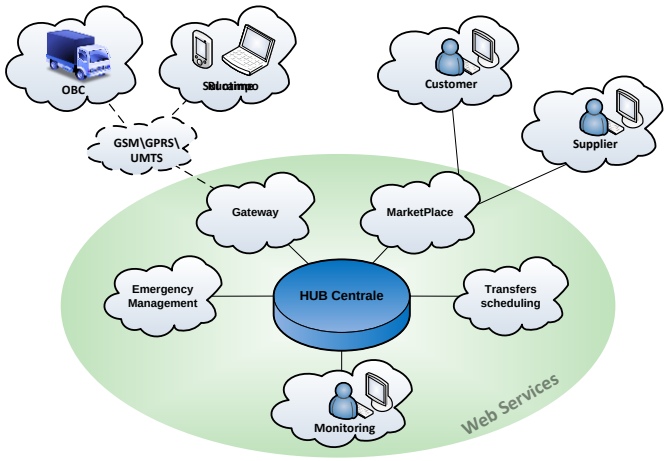

Fig. 3. The overall system.

The communication between the central hub and the other modules is via web service. Web services exposed by the various modules are implemented with different technologies (WCF, asmx, axis2, metro, etc.) and interoperability is guaranteed by compliance with the standard basic profile 1.1.

A global view of the information flows managed within the SAIL project information system is shown in Figure 3, from a logical point of view, and Figure 4, from an architectural point of view. Single scenarios are discussed in the following sections.

3.1 Single scenarios

Central Hub - Gateway

The communication between Central hub and Gateway allows o capture information from the OBC mounted on individual vehicles in transit through the territory and to carry out the functions of group management – a group is formed

as a unit consisted of the travel drive and one or two containers.

All communications originating from HC go to the OBC through the Gateway who sends the messages (Figure 5). The Gateway module is in charge to apply the retry policies to the OBC.

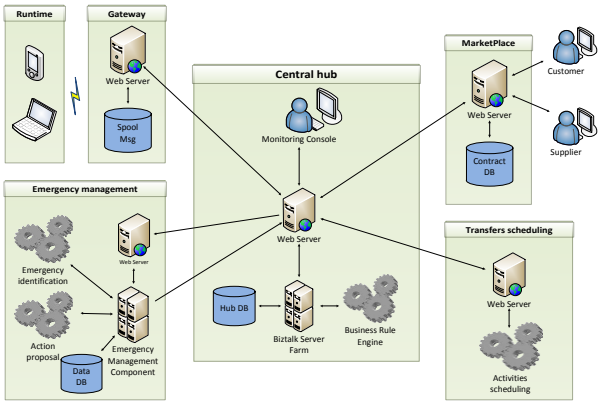

Fig. 4. The modules of the system.

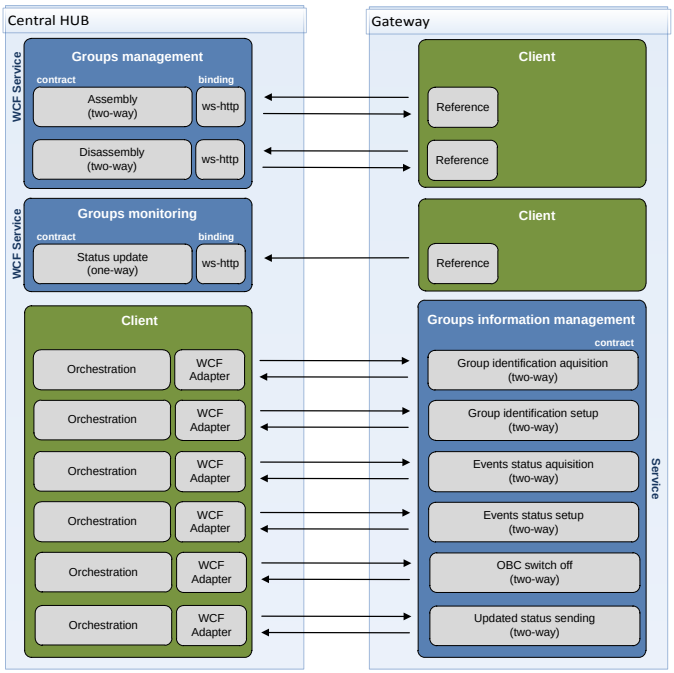

Fig. 5. Exchange of messages between Central Hub and Gateway.

The process of assembly consists of the following steps:

1. the operator installs the RFID tags on containers and on-board computer on the tractor
2. the operator - using a palm device - acquires the identifiers of RFID tags and of the on-board computer. Then it transmits via GSM/GPRS/UMTS to the Gateway module all related information to the equipped vehicle (for example license plate, identifier of the OBC, the identifier of the tag of the container, the container identification, etc)
3. the Gateway, through WCF call, sends this information to HC in synchronous mode
4. the HC generates a new Group ID (id OBC, drive plate, id tags (MAC address), container serial) and responds to the previous call
5. the Gateway responds to the palm with the outcome of the operation and simultaneously sends to the OBC the assembly
6. the operator checks the status LEDs on the OBC and gives the OK to drive, or checks where the procedure failed

The process of disassembly consists of the following steps:

1. EMC sends through a WCF door - exposed in Basic-HTTP – the piece of information that some ID group arrived at the end of its transfer
2. through the Gateway, HC changes the configuration parameters of the OBC in order to turn off periodic sending of messages, but leaves it working on events (e.g. opening of container, etc.): in this way no power is wasted and there are no useless information sent around
3. with the palm device, the operator generates the request of disassembly, reading tags of OBC and containers. He sends this information and waits for the response from the HC that the procedure is successful. Possibly something may be wrong as these tags have never been used for assembly, etc.
4. in the case of a positive outcome of the previous step, HC sends the shutdown message to the OBC

During the transfer, every 30 seconds (configurable parameter), the OBC sends all the information about the state of RFID tags and alarms, through the gateway. Furthermore, for some alarms, like the pressure of the button signalling the presence of an unusual event, the message starts immediately. For this information, HC has a WCF port listening for messages arriving through the gateway. Once arrived, information is written into the database and forwarded to the EMC by a web service exposed by the module itself.

The services offered by Gateway to HC include:

Acquisition of group identification: the HC sends a request that contains the identifier of an OBC. HC receives from gateway the identifier of the group that is currently associated with, or an error message if it no active group is identified

Configuration of Group ID: it is used in the process of disassembly to change the identifier of a group which must be broken and update its value setting it equal to 0

Acquisition of Event Status: given the identifier of an OBC, it returns the status of the events associated with it (eg: battery, GPS connection, GSM connection, etc.)

Configuration of Event Status: it allows changing the status of the configured events on the OBC and in particular the state of sending periodic messages that is disabled when arrival in a safe area is notified by the EMC

OBC switch off: it is used in the process of undressing, to notify the gateway that the OBC must be turned off. The gateway is responsible to send the communication to the OBC

Status Update: it allows the operator to request a status update for a particular group by entering its ID in the request. The request shall be made in case of lack of status for a period larger than expected, or to verify the consistency of the information received from a previous update

Central Hub – Emergency Management Component

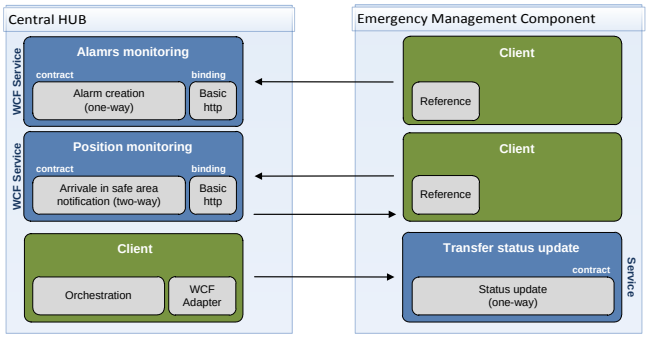

Fig. 6. Interaction between Central Hub and Emergency Management Component.

The services exposed from the HC to EMC include (Figure 6):

Alarm creation: the EMC uses this service to indicate the presence of an alarm identified by an alarm code and accompanied by the description of the action to be taken to manage the emergency

Notification of arrival in safe area: EMC analyzes the status messages and determines when a group reaches the target area, so it is therefore in a safe place. Once the group has arrived at destination, it is no longer needed the transmission of position, state of the tag, etc.. The EMC module sends a message to the HC for notifying the arrival to destination and disable the periodical sending of status messages.

The service exposed by the EMC is (Figure 7):

Status update: HC uses this service to forward to EMC all status messages received by the gateway. All received messages are forwarded without filters. Inspecting the content of individual messages and analyzing their content in relation with archived messages, EMC is able to detect possible alarm situations and notify the HC. The service call does not wait for a response, but it only sends the updated information

Central Hub – Operational level

The communication between the HC and the Operational level is designed to obtain the scheduling of transport activities booked through the Marketplace portal or directly in the traffic center. The scheduler input is a list of activities called units. A unit may represent a container or a vehicle.

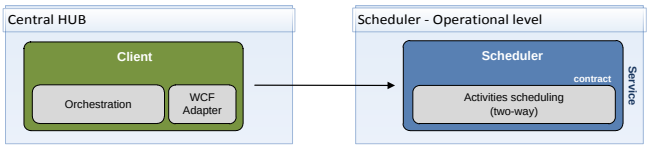

Fig. 7. Connection between Central Hub and Operational level.

Central Hub – Marketplace

Transfer requests are generated within the Marketplace for customers who wish to book the transfer service or from the (dry-)port management system, for vehicles which arrive there without reservation. The (dry-)port staff takes charge of entering into the system orders through an appropriate system outside of SAIL. For the purposes scheduling, both kinds of activities requests are included in the scheduled plan: a) in the first case the description of the activity is obtained from a web service exposed by the marketplace portal, b) in the second case the description of the activity is obtained from the external system that (dry-)port crew uses for order entry. Also this system exposes a web service that the HC can consume to get a list of orders to be taken into account

The Marketplace is considered the master in the management of information on contracts, as it represents the meeting point between supply and demand, where customers meet the suppliers of transport. Only in one case, the Marketplace consumes a service from the HC: this service provides an operation that allows the Marketplace to obtain detailed information on the activities associated with the contracts (Figure 8).

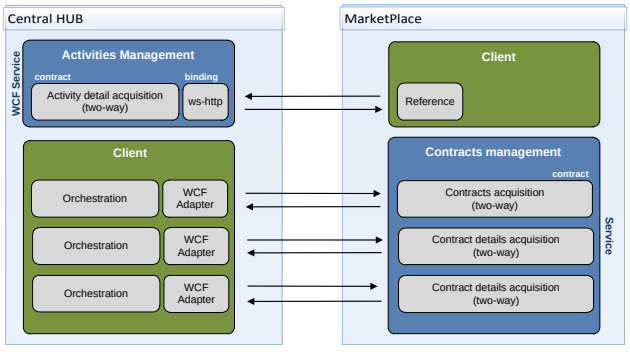

Fig. 8. Central Hub and Marketplace interactions.

Central HUB – Monitoring console

The monitoring console is available to the operator to supervise the movement of vehicles. The console interacts with the central hub in a unidirectional manner using the services offered.

The operations available to the operator are (Figure 9):

Group status request: it allows the operator to ask a group for sending an updated status message. This feature is designed to cover those situations in which a vehicle does not send information about its position and the status of the on-board instrumentation for an interval of time greater than expected. The status update request allows distinguishing situations in which the periodical sending is disabled for some reason arising from situations of potential danger or lack of gsm/gprs signal. The status message obtained has as a sending reason "Request from the center." The call always returns a response, positive or negative, that indicates the assumption by the communication gateway of the communication to the OBC. It may happen that the OBC does not send an updated message, due to lack of gsm/gprs signal or malfunction of the equipment

OBC events acquisition: it allows the user to request the status of the events associated with a group. In response to the request you always get a response that contains the latest information available to the gateway at the moment of the call. The information may not be updated if the on-board computer does not transmit messages for a long time

Group Id acquisition: it obtains the identifier of a group from the identifier of the OBC

Activities re-scheduling: it allows the operator to send a request for a schedule of activities. First, new contracts created within the marketplace portal and new orders entered by the (dry-)port management system are acquired. After updating the list of activities it sends this list to the scheduling module (operational level) that returns a proposal for the execution plan of activities, setting to a higher level of priority those who were already scheduled and confirmed

Contract details acquisition: it allows the operator to obtain detailed information about the customer and the supplier of the shipping service. The information is obtained from the Marketplace that has a master role in the management of this information

Contract cancelling: the operator has the right to cancel a contract if the task cannot be completed on time and as planned. This situation can occur subsequently to the conclusion of the contract. It may be due to network traffic urban or delays in the performance of practical port/customs

4. CONCLUSIONS

This paper presents the architecture of the SAIL project information system and the logical/physical scenarios it is tailored for.

Currently, the system manages the information related to the management of the runtime operations, and their flows. Using an orchestrator allows us to add other information flows just building the adapter to the new information formats and adding the needed maps and orchestrations: the system is open to include any (un)foreseen new set of information.

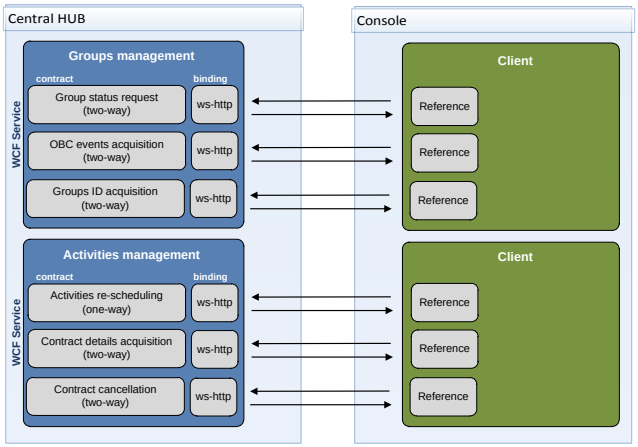

Fig. 9. Central Hub and Console interactions

In an updated version, we will include the modules of tactical and strategic level decisions. Moreover, we are going to include modules for statistical analysis of the gathering data through the connected modules. For the time being, it is not foreseen any connection to other information systems, but in the future the integrations with the (dry-)port information system will be implemented.

ACKNOWLEDGMENT

This work is supported by the E.U FP7-PEOPLE-IAPP-2009, Grant Agreement No 251589, Acronym: SAIL.

REFERENCES

Beckner, M. Goeltz B. and, Gross, B. (2006) BizTalk 2006 Recipes: A Problem-Solution Approach, Apress

Beckner M. (2010), BizTalk 2010 Recipes: A Problem-Solution Approach, Apress

Cibraro, P., Claeys, K., Cozzolino, F., & Grabner, J. (2010). Professional WCF 4: Windows Communication Foundation with. NET 4. Wrox

Dawson, J., Wainwright, J., & Sanders, J. (2009). Pro Mapping in BizTalk Server 2009. Apress

Forum: http://www.biztalkgurus.com/forums/

Klein, S. (2007). Professional WCF programming. John Wiley & Sons.

Lowy, J. (2008). Programming WCF services. O'Reilly Media, Incorporated.

Moukhnitski, S.,Campos, H., Kaufman, S., Kelcey, P., Peterson, D., and, Dunphy G. (2009). Pro BizTalk 2009. Apress, 2009.