Trains as mobile devices, the Dynamic Insertion Network Object (DINO) concept. The EU Traincom project approach

Andrea Gatti
Trenitalia s.p.a. – Unità Tecnologie Materiale Rotabile - Viale S. Lavagnini 58
50129 Firenze - Italy
e-mail: an.gatti@trenitalia.it

Paper presented to the

World Congress in Railway Research

28 September - 1October 2003

Edinburgh Scotland

Poster session - Satellite Technology

Trains as mobile devices, the Dynamic Insertion Network Object (DINO) concept. The EU Traincom project approach

Andrea Gatti
Trenitalia s.p.a. – Unità Tecnologie Materiale Rotabile - Viale S. Lavagnini 58
50129 Firenze - Italy
e-mail: an.gatti@trenitalia.it

THE TRAINCOM PROJECT

The TRAINCOM [1] is a research project partially funded by the European Community under the "Information Society Technologies" Programme 1998-2002, it involves 14 participants of 7 countries with an effort of 650 Pms, and will end in December 2003. The project aims to fully specify and develop a communication system for telematic applications in the railway field, integrating the on-board network, 2G-3G radio links and Internet technologies. A train, as a cluster of devices connected by a network, is "a mobile object" that is performing some services like passenger information, train identification and fleet management. The train is connected to the ground section with a set of wireless connection and is capable to switch between them "on the fly" according to the requirement of bandwidth requested by the "service".
The following picture sketches the communication components for the TrainCom project as described in [3] and [4]. This document deal with the interaction between the network's elements marked by the dashed bubble: the ground station "RoGs" and the train mobile station "RoGate". The behaviour of the Rogates depends by the train network architecture:

- in case of TCN network, it acts as a proxy, representing the train functional structure to the ground network;
- in case of IP based network, it acts as a router allowing each function/devices to be reachable through its IP address.

Considering the existing network structures and services of the railway operators, key points are the interoperability of the system and the communication protocols of the applications.

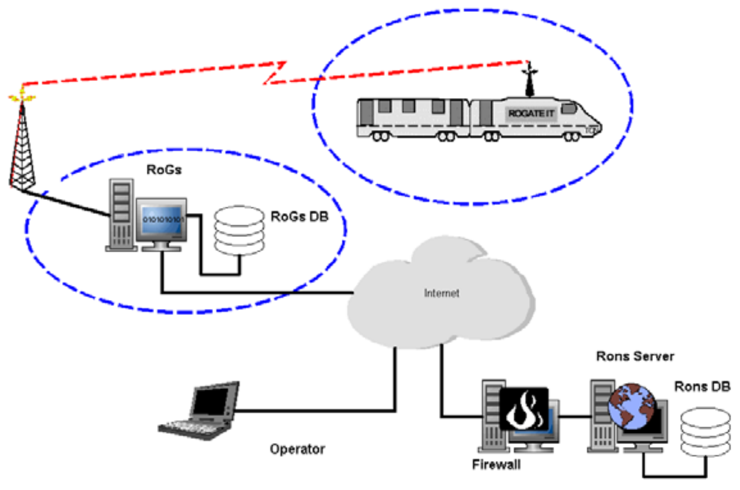

Figure 1 Communication components

THE TRAIN AS MOBILE CLUSTER OF SERVICES

The train is a cluster of devices connected by a network [2]; the devices are mapped in order to perform a set of functions or services. Each performed service needs a “Service Leader” at train/consist level and it is possible to map each service to a communication channel according to bandwidth’s requirement. The RoGate is connected to the ground section using a set of wireless connection and will be capable to switch between them “on the fly” according to the bandwidth’s requirement of the service. This approach allow us to use a separate channel for a group of services. In figure 2 is represented the configuration of a consist of one locomotive and four coaches, linked using two kinds of train network (TCN and an IP based network) with four RoGate, two of them with multiple radio links.

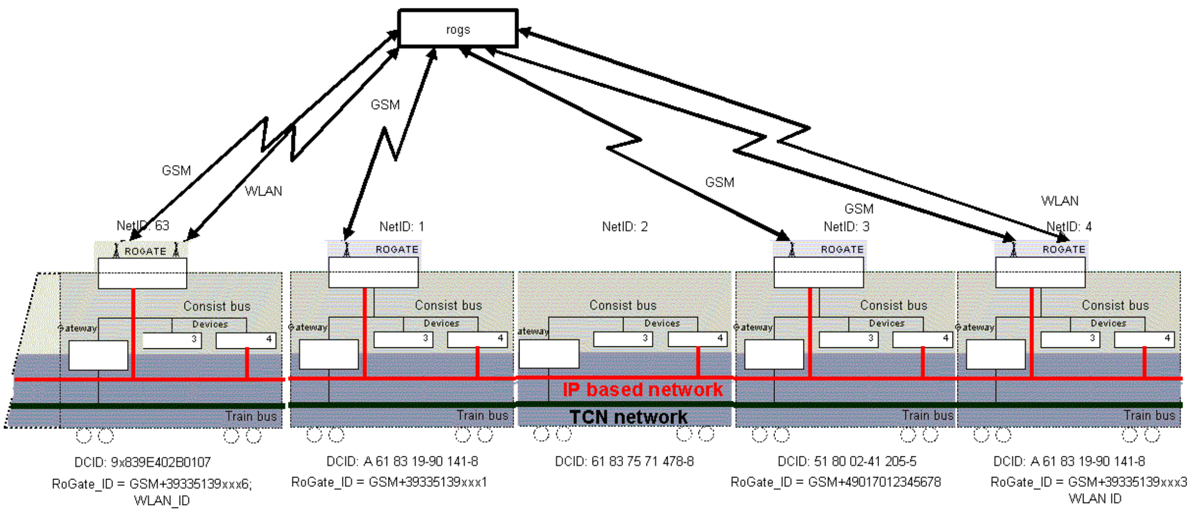

Figure 2 - A consist with multiple RoGate

A Train Object encapsulates all data related to the course, such as the departure and destination cities and times, the current number of available seats, as well as the course’s current status (sitting at a station). Remote objects obtain this information via public, remotely available method calls.

When a new Train Object is created, it registers itself with object registries on the network. These registries allow clients to locate Trains, based on some query criteria, such as the departure and destination cities. An object representing a Seat reservation system might locate all Courses between two cities, and present this list ordered by departure times.

When a client locates a suitable Course, it accesses an object registry for a stub representing that Course on a remote network node, the RoGs in our architecture. The client uses the stub to invoke methods on the remote Train Object. Thus, stubs act as proxies for the Train. There is one copy of this proxy for each remote client node referencing a specific Train Object, see [6] and [7].

THE DYNAMIC OF THE TRAINS

One of the most challenging characteristics of networks in trains are the dynamics in the networks during coupling/decoupling, etc.

We have to distinguish different types of dynamics and different modes of access to a train:

- Train composition changes as run start/end, vehicles coupling/decoupling, add/removal of devices during power up/down;
- functional changes as spare train takes over, change of used/unused driver's cab and redundant device takes over.
- direct access to a specific consist/vehicle regardless where it is.
- access to a specific Running Train, regardless which consists build it. It means the consist is declared ready to scheduled service and linked to a "Running Train ID".

The setting up of a group of vehicles in a depot or in a maintenance facility needs to be traced by the network. Train coupling / decoupling is widely used and has to be supported by dynamic addressing schemes. To meet these requirements we need to have access to:

- each vehicle identified by its "Direct Car Identifier" (DCID);
- a group of vehicles (Consist) with its "Consist Identifier" (CID);
- and the scheduled train, the commercial services, using a "Running Train IDentifier"(RTID).

The access modes linked to the Direct Car IDentifier (DCID) and Running Train IDentifier (RTID) were already analysed by the ROSIN Project [8] in the deliverable D04.2 V2.0. [9].

THE "DYNAMIC INSERTION NETWORK OBJECT" (DINO)

From the point of view of a Train Operator Company, the fleet management and the usability of the services offered to internal and final clients are the key points. Each train needs to be reachable in order to obtain information about the status of the on-board equipment and the service available either for professional user (e.g. maintenance staff, service coordinator etc.) or final clients. A Train Operator Company can exposes each train as an object on the network.

In following the conventional approach, the train needs to be logged-in and out in the network, so if we analyse the figures about the quantity of trains involved daily, the maintenance of this process is very resources consuming. Rather than requiring that explicit installation and removal from the network, this kind of network environment calls for components that are able to join and leave the network in a far more ad hoc fashion.

The DINO concept

The RoGate is the device in charge to present the services to the network and in case of multiple RoGate it is necessary to define which role will play the several RoGates in a consist; this task is managed at application level. The ROGATEs convert the TCN data format into XML data according to the TrainCom scheme and joins dynamically the network as a cluster of Jini™ services. Detailed information are in [3]. This scenario became populated by objects that join the network in dynamic way.

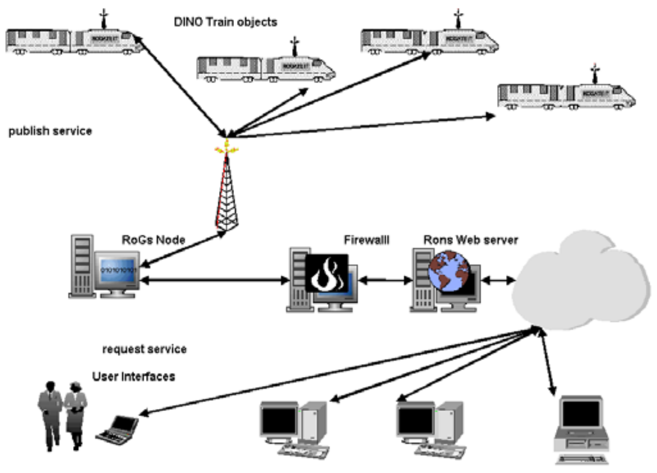

Figure 3 -Dino train objects

The root service of DINO is the Train registration, as we need the access to the consist using the DCID or CID. Without it is not possible to perform the other services. The RTID shall be assigned, for example, either to exchange DPIS and seat reservation information or to monitor the quality of service in terms of comfort.

A DINO train can provide a set of services like Seat Reservation, GPS Localisation, Diagnostic Data for maintenance, Storage of audio files for information and entertainment, etc. The "Function Leaders" on the consist's network need to map the relationship between a specific Services and the "Service Leader RoGate".

For example, starting from the composition shown in Figure 2, Table 1 reports the description of several services, Table 2 maps the vehicles and the RogateID, while Table 3 maps the vehicles and the services in defining the role (Leader "L" or Follower "F") assigned to each device. Each RoGate is associated to a GSM number and/or other radio channel identifier, while network identifier (NetID) is based on TCN numbering scheme [5]. All information are modelled inside the "Train Profile".

Service_ID	**Service description**
Trip_info	Information regarding actual trip condition
DPIS	Passenger Information like time schedule, delay, connection
Seat_res	Seat reservation
GPS_loc	GPS localisation
Diag_data	Diagnostic data, equipment status, quality of service, fleet management
MP3 _Deploy	Deploy entertainment service
MP3_Storage	Management of entertainment data and services

Table 1 - Service description

DCID	NetID (*)	RoGate_ID
9x839E402B0107	63	GSM+39335139xxx6; WLAN_ID
A 61 83 19-90 141-8	1	GSM335139xxx1
61 83 75 71 478-8	2	
51 80 02-41 205-5	3	GSM+49 017012345678
A 61 83 19-90 141-8	4	GSM335139xxx3; WLAN_ID

Table 2- DCID and Rogate ID

Each vehicle supports a set of services, so each supported service needs a "Service Leader" at train/consist level and it is possible to map each service to a communication channel according to bandwidth requirement. This approach allow us to use a separate channel for a group of services. A complete way is to define, for each supported service, a "Service Leader" in order to use a separate channel for a group of services.

DCID	Service_ID, Role
9x839E402B0107	Trip_info, L ;DPIS, F; Seat_res, F; GPS_loc, L; Diag_data, L; MP3 _Deploy, F
A 61 83 19-90 141-8	DPIS, F; Seat_res ,F ; GPS_loc, F; Diag_data, F; MP3 _Deploy, F
61 83 75 71 478-8	Seat_res ,F ; Diag_data, F; MP3 _Deploy, F
51 80 02-41 205-5	DPIS,L ;Seat_res, F; GPS_loc,F; Diag_data, F
A 61 83 19-90 141-8	DPIS, F;Seat_res, F; GPS_loc, F; Diag_data, F; MP3_storage, L; MP3 _Deploy, L

Table 3 - Services and Role mapping

Using the TrainCom approach, we are able to share a communication channel, and we are allowed to subscribe several services over one TCP/IP-HTTP connection. We are also allowed to separate services from the linking technology (GSM, GPRS, GSM-R,, UMTS, WLAN, Satellite etc.).

The use of Jini

Jini technology is useful in developing a prototype distributed system to simulate services based communication. Jini supports serendipitous interactions among services and users of those services. Jini also supports redundant infrastructure to reduce the probability that services will be unavailable if machines within the Jini community crash.

The clustering devices use a shared JVM available on the network. From this point of view, a proxy for the JVM used by the various service devices would exist on the network. Service devices are added to the network, discover the existence of such a proxy device and register with that proxy. Such a registration optionally includes the code written in Java programming language needed by a client of the device (either directly or as an URL to be used to obtain the code) and code needed by the proxy to communicate with the service device.

In our configuration, the RoGate could acts as a proxy or as a router; it implements the interfaces to the Jini discovery and Jini lookup service in the device itself; it includes specialized knowledge of the kind of leases that are handed out by the Jini lookup service and is able to renew those leases directly, and having also sufficient functionality to download and use the stubs for these services.

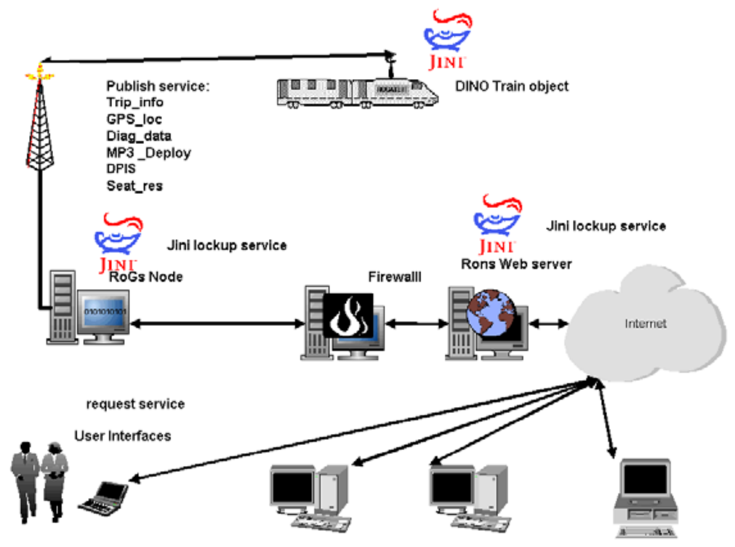

Figure 4 The Jini architecture

This is a particular set of functionalities that is considerably smaller than that required by the whole of the JVM, and should be possible to implement in much less code. For example, such a JVM would not need to contain a security manager, a code verifier, or a number of the other components that are required for a full JVM.

Such a device would contain a JVM specialized for the application environment for Jini technology, allowing the Jini discovery and Jini lookup services to be accessed and leases of a particular sort to be renewed.

Security issues

Open trains and cross border connections rises some questions about right access to informations available on-board. An example for maintenance data:

- Railway operators must have access to all equipment it bought, but not to the equipment it did not.
- Equipment manufacturers must have access to the equipment it sold, if so allowed by the customer.
- Equipment manufacturers must have no access to data of other manufacturers, except if has got the access rights.
- Troubleshooting procedures need to be shared across several railway operators but the reliability data are surely “private” information.

The only way to keep the security of data and to manage the access rights between all the involved players is to encrypt data using an Application-layer technology such as XML Security. To this effect, it is mainly a question of the level of security and the handling of the cryptographic keys.

The use of X.509 digital certificates and the incorporation of authentication technologies such as Radius, Kerberos and smartcards need to be considered.

In designing distributed systems it is necessary to ensure sufficient flexibility and extendibility to adapt to diverse and evolving requirements. To address this challenge, we can take into account the approach followed by the Pluggable Authentication Module (PAM) framework and a Java version of the PAM: the Java Authentication and Authorization Service package JAAS.

The PAM framework allows applications to remain independent from authentication schemes, because it can integrate multiple authentication mechanisms by plugging them into an application at runtime. Application developers who use the PAM interface with a single high-level application programming interface are decoupled from changes in security policies and authentication mechanisms. System administrators have the flexibility of selecting one or more authentication technologies on the basis of their local security policy and are not required to modify each application.

A common, trusted entity for distributing the keys would be welcome. Some topics related to implementation issues are addressed in [10];

Ground side, services and multiple RoGs

In developing the prototype we use a single ground station, but in modelling the system it is not correct to consider a single RoGs. Open trains and cross border connections are the reference's scenario, so the real environment will present several RoGs, see Figure 5 (where the Train Object leaving the vicinity 1 on the upper left and entering vicinity 2), and our design strives to accommodate heterogeneity of the active objects.

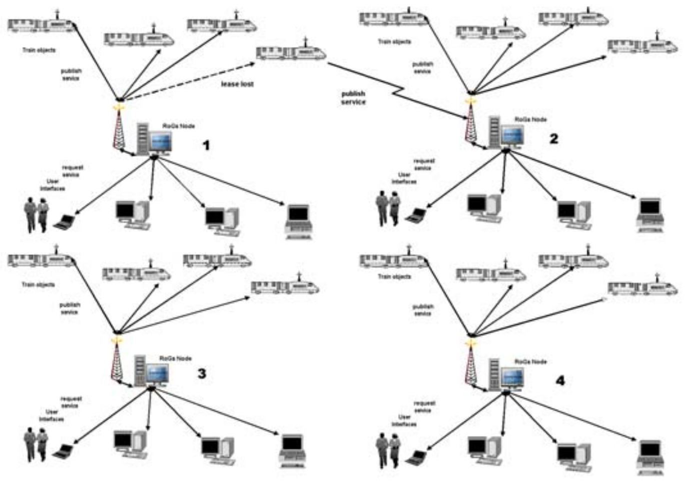

Figure 5 Community interaction, multiple RoGs

In this architecture, the RoGs node is the link for communication among all members of a community. Objects in the system differ in the amount of data they contain, the number of remote method invocations they can service during a given time period, the frequency of incoming method invocations, the amount of data they pass across the network for each method invocation and, especially, the types and numbers of local references an object may hold to other objects.

Each RoGs has the responsibility of a vicinity, constituted by Train Objects, RoGs Node and users inside an area, see Figure 5.

All elements within a given vicinity belong to the same community (Traincom ring). A vicinity may represent a traffic intersection or a neighbourhood of rail links.

Protocol for object migration

Since we assume no central controlling entity among the nodes, object migration requires agreement between any two cluster nodes about the transfer of an object. We can follow the same template presented in [7] where the dynamic of the system is addressed in two ways: first, by allowing the middleware on each RoGs node to accommodate pluggable load balancing and migration algorithms, and, second, by having the middleware build a profile for each object, or a graph of objects. This agreement is facilitated by a protocol consisting of 3 phases. This protocol is outlined in Figure 6.

- Discovery: locate the new node's NodeManager on the network and obtain a reference to it;
- Negotiation: agree on the transfer of the object with the remote node;
- Transfer: perform the object's transfer to the new node.

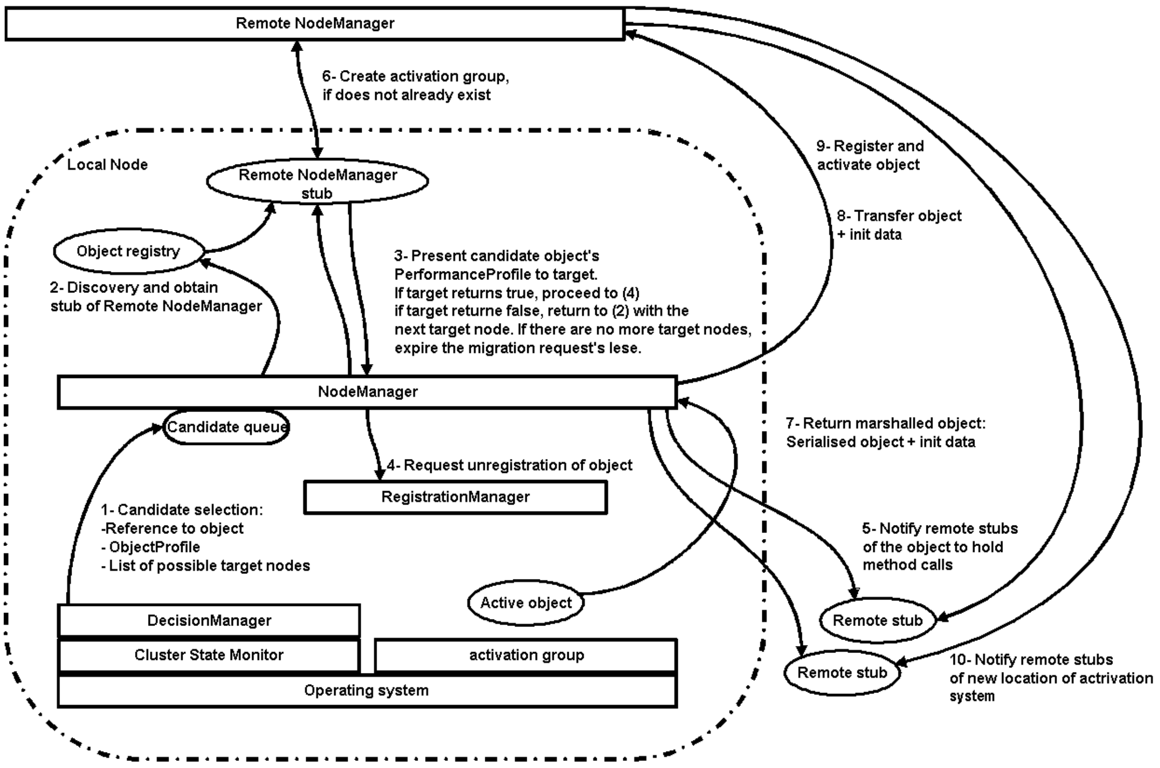
Remote NodeManager
6- Create activation group, if does not already exist
Local Node
Remote NodeManager stub
Object registry
2- Discovery and obtain stub of Remote NodeManager
3- Present candidate object's PerformanceProfile to target. If target returns true, proceed to (4) if target returne false, return to (2) with the next target node. If there are no more target nodes, expire the migration request's lese.
9- Register and activate object
8- Transfer object + init data
NodeManager
Candidate queue
7- Return marshalled object: Serialised object + init data
4- Request unregistration of object
RegistrationManager
1- Candidate selection:
-Reference to object
- ObjectProfile
- List of possible target nodes
5- Notify remote stubs of the object to hold method calls
Active object
DecisionManager
Remote stub
Cluster State Monitor
activation group
Remote stub
10- Notify remote stubs of new location of actrivation system
Operating system

Figure 6 - Object migration Protocol [7]

TRAINCOM PLATFORM AS WEB SERVICES

The implementation of applications over the TrainCom communication architecture could be viewed as the implementation of Web services.

Web services, in the general meaning of the term, are services offered via the Web. In a typical Web services scenario, a business application sends a request to a service at a given URL using the SOAP protocol over HTTP protocol. The service receives the request, processes it, and returns a response. The time it takes to return the response depends on the complexity of the routing. In some cases, the request and response will be parts of the same method call, otherwise, the response will probably be sent as an operation that is separate from the request.

All the services listed in Table 1 need to be modelled and standardised and the implementation could be realized using several approaches, either using directory mechanisms or java technologies.

Web services depend on the ability of enterprises using different computing platforms to communicate each other. This requirement makes the Java™ platform, which makes code portable, the natural choice for developing Web services. This choice is even more attractive as the new Java APIs for XML become available, making it easier and easier to use XML from the Java programming language.

Perhaps the most important feature of the Java APIs for XML is that they all support industry standards, thus ensuring interoperability. Various network interoperability standards groups, such as the World Wide Web Consortium (W3C) and the Organization for the Advancement of Structured Information Standards (OASIS) with UDDI and ebXML, have been defining standard ways of doing things so that businesses that follow these standards can make their data and applications work together.

ACKNOWLEDGEMENT

The author offers his thanks to all the TrainCom partners, in special way to the Team of Workpackages 3 and 6 “Architecture of the TrainCom Communication system”.

ACRONIM

DCID	Direct Car IDentifier
DINO	Dynamic Insertion Network Object
DPIS	Dynamic Passenger Information System
RTID	Running Train IDentifier
CID	Consist IDentifier
GPS	Global Positioning Service
JVM	Java Virtual Machine
JAAS	Java Authentication and Authorization Service package
OASIS	Organization for the Advancement of Structured Information Standards
PAM	Pluggable Authentication Module
DPIS	Dynamic Passenger Information System
Pms	Person months
SOAP	Simple Object Access Protocol
TCN	Train Communication Network (IEC 61375)
UIC	Union International des Chemin de Fer
UDDI	Universal Description, Discovery Integration
XML	eXtensible Mark up Language
W3C	World Wide Web Consortium

REFERENCES

[1] The TrainCom project -http://www.traincom.org

[2] Andrea Gatti, Trains as Mobile devices: the TrainCom project - Wireless Design Conference 2002 London.

[3] Traincom Deliverable No. 7, TC1-D-SIE-020-05 “Architecture of the TrainCom communication system”.

[4] Traincom Deliverable No.15, TC1-D-CAF-019-05 Prototype of the TrainCom Communication System.

[5] IEC 61375 Train Communication Network

[6] Architecture Using Jini Technology For Simulation Of An Agent-Based Transportation System - Lisa A. Schaefer - The MITRE Corporation McLean, VA 22102, U.S.A.

[7] Cluster-Based Computing with Active, Persistent Objects on the Web Frank Sommers – Autospaces, Shahram Ghandeharizadeh and Shan Gao Department of Computer Science University of Southern California. IEEE 3rd International Conference on Cluster Computing, October 8-10, 2001

[8] The ROSIN Project: http://www.labs.it/rosin

[9] ROSIN "Deliverable 4.02 - Prototype of the Universal Maintenance Tool ROMAIN” - Version 2.0 - December 1998 (DEL04.2/GAT/WP04/V2.0 /Dec98)".

[10] Traincom document TC1-T-TRI-023-01 Contribution on Security Mechanisms.