Milestone 4 - Common set of upgraded specifications for Hybrid communication

Specifications for IF2 for hybrid communication version 1.0

Version number:	1.0
Main author:	Igor Passchier
Dissemination level:	PU
Lead contractor:	NMIE-R
Due date:	31/08/2018
Delivery date:	1/03/2018
Delivery date updated document:	

Co-financed by the European Union
Connecting Europe Facility

Grant Agreement No:
INEA/CEF/TRAN/M2015/1143833
Action No: 2015-EU-TM-0159-S

CONTROL SHEET

Version history			
Version	**Date**	**Main author**	**Summary of changes**
0.1	25/09/17	Igor Passchier	Initial draft
0.2	06/10/2017	Igor Passchier, Emi Matthews, Marcel van Sambeek	Drafts of sections 2, 3, 4, 5.1 and 5.2
0.3	25/10/2017	Igor Passchier	Review of 4, additions to 4, update of 5
0.4	31/10/2017	Igor Passchier	Updates of 4, 5,6, based on F2F meeting
0.5	02/11/2017	Igor Passchier	Final version before Stakeholder consultation
0.6	20/11/2017	Igor Passchier	Included review comments from NL and UK
0.7	30/11/2017	Marcel van Sambeek	Included review from BE and FR
0.8	01/12/2017	Igor Passchier	Example implementation added
0.95	12/12/2017	Igor Passchier	Final draft version
0.99	17/12/2017	Igor Passchier	Final version discussed with WG
1.0	23/02/2018	Igor Passchier	Final

	Name	Date
Prepared	Igor Passchier	17/12/2017
Reviewed	Core Management Team Advisory Committee & General Assembly	29/12/2017 13/02/2018
Authorised	Ronald Adams (NMIE-R)	28/02/2018

Circulation	
Recipient	**Date of submission**
INEA	01/03/2018
InterCor consortium	01/03/2018

 © InterCor Consortium

Authors (full list):

Igor Passchier (TASS International), Paul Spaanderman (PaulsConsultancy), Marcel van Sambeek (TNO), Emi Matthews (TASS International), Jurgen Latte (MOW Vlaanderen), Marie-Christine Esposito (DGITM), Hacène Fouchal (University of Reims Champagne-Ardenne), Cliff Lunnon (Highways England), Paul Warren (WSP)

Project Coordinator

Ronald Adams

Rijkswaterstaat

Office address: Toekanweg 7, 2035 LC, Haarlem (NL)

Postal address: Postbus 2232, 3500 GE, Utrecht (NL)

Mobile:+31 6 518 480 77

Email: ronald.adams@rws.nl

Legal Disclaimer

The information in this document is provided “as is”, and no guarantee or warranty is given that the information is fit for any particular purpose. The content of this document reflects solely the views of its authors.

The InterCor consortium members, jointly or individually, shall have no liability for damages of any kind including, without limitation, direct, special, indirect, or consequential damages that may result from the use of these materials.

Neither the European Commission nor the Innovation and Networks Executive Agency (INEA) are liable for any use that may be made of the information contained therein.

TABLE OF CONTENTS

 © InterCor Consortium

 © InterCor Consortium

LIST OF FIGURES

 © InterCor Consortium

LIST OF TABLES

 © InterCor Consortium

Terms and abbreviations

Abbreviation	Definition
AC	Advisory Committee
AL	Activity Leader
AMQP	Advanced Message Queuing Protocol
ASN.1	Abstract Syntax Notation One
ASR	Action Status Report
CAM	Cooperative Awareness Message (message type)
CEF	Connecting Europe Facility
CEN	Commission for European Normalization
C-ITS	Cooperative Intelligent Transport System
CMT	Core Management Team
CPU	Central Processing Unit
DAB	Digital Audio Broadcast
DENM	Decentralized Environmental Notification Message
DP	Data Provider
DSRC	Dedicated Short Range Communication
EC	European Commission
ETSI	European Telecommunications Standards Institute
FNTP	Fast Networking & Transport Layer Protocol
GA	Grant Agreement
GLOSA	Green Light Optimal Speed Advisory
GN	GeoNetworking
IEEE	Institute of Electrical and Electronics Engineers
IF	Interface
INEA	Innovation and Networks Executive Agency
INEA	Innovation and Networks Executive Agency
IPR	Intellectual Property Right
ISO	International Standards Organization
ITS	Intelligent Transport System
ITS-G5	OSI layer 1 and 2 technology specified in ETSI EN 302 663
IVI	In-Vehicle Information (message type: IVIM)
IVS	In-Vehicle Signage
LAT	Latitude
LON	Longitude
LTE	Long Term Evolution (4th generation mobile networks, 4G)
LTE-D	LTE-Direct
MAP	Road/lane topology and traffic manoeuvre message (message type: MAPEM)
ML	Milestone Leader
MQ	Message Queue
MQTT	Message Queuing Telemetry Transport

 © InterCor Consortium

MS	Member State
OBU	On-Board Unit
PC	Project Coordinator
PER	Packet encoding rules
PKI	Public Key Infrastructure
PVD	Probe Vehicle Data
RLAN	Radio Local Area Network
RSU	Roadside Unit
RWW	Road Works Warning
SASL	Simple Authentication and Security Layer
SP	Service Provider
SPAT	Signal Phase and Time (message type: SPATEM)
SRM	Signal Request Message (message type)
SSM	Signal Status Message (message type)
TCP	Transmission Control Protocol
TIC	Technical & Interoperability Coordinator
TLC	Traffic Light Controller
TLS	Transport Layer Security
TTL	Time to live
UWB	Ultra-Wide Band
WAS	Wireless Access Systems
WAVE	Wireless Access in Vehicular Environments
WSMP	WAVE Short Message Protocol

 © InterCor Consortium

1 Executive summary

Within InterCor activity 2.1b, a high-level system description has been developed that supports interoperability between hybrid communication solutions that exist or are under development in the participating countries. This system description includes two interfaces that need to be specified to realize cross-border interoperability. The first interface (called IF1) is the ITS-G5 interface between vehicles and road side systems, as defined by sub-activity 2.1a. The second interface (called IF2) is an interface between back-office systems, to support the exchange of the information between back-offices required to support the services (also) via cellular communication.

This document describes the interface definition of IF2 to enable international interoperability in hybrid communication for three of the services of InterCor: Road Works Warning (RWW), In Vehicle Signage (IVS), and Green Light Optimal Speed Advisory (GLOSA). It includes a minimum set of requirements and both functional and technical specifications. These specifications need to be implemented by at least 2 countries, and tested in the Hybrid TESTFEST, carried out by activity 2.2.

Once the final TESTFEST on C-ITS Services will be completed in 2019, this Milestone 4 report will be finalized, including the results of the remaining work of sub-activity 2.1b and results from the hybrid TESTFEST planned for October 2018

 © InterCor Consortium

2 Introduction

Hybrid communication is a generic term interpreted in many ways within discussions on communication technologies. The concept of hybrid communication in the context of Cooperative Intelligent Transport Systems (C-ITS) was "invented" in the EU's CVIS project, and enabled in the ITS architecture standard ISO 21217 (First version adopted by ETSI IN EN 302 665 with editorial changes; latest version is from 2014). "Hybrid communication" includes communication technologies for **network-based** communication (e.g. cellular networks, Internet) and **localized** direct communication (e.g. ITS-G5 based on IEEE 802.11p combined with a single-hop messaging protocol (e.g. ETSI GeoNetworking (GN), ISO FNTP, IEEE 1609.3 WSMP). The term "Hybrid communication" was introduced for C-ITS in the German CONVERGE project. Hybrid communication was further recognized within the ITS community when used in the Phase-1 report[1] from the C-ITS Deployment Platform by the European Commission. There it was stated as:

> *The availability of an increasing amount of ITS data enables the realisation of connected, cooperative and automated ITS services and applications, with highly varying functional and technical communication requirements, require an open hybrid communication approach supporting future new technology adoption, today including a number of access and communication technologies, such as 3G/4G, LTE, LTE-D, 5G, WAS / RLAN versions of IEEE802.11, IEEE802.11p/ITS-G5, Bluetooth, ZigBee, UWB, CEN DSRC and DAB. A hybrid communication concept including mechanisms supporting these highly varying especially traffic safety and efficiency related C-ITS requirements taking advantage of current and upcoming complementary technologies is needed.*

A mature hybrid communication solution is expected to improve the quality of the services offered by combining multiple aspects at the same time. This can include an improved geographical coverage, an improved robustness due to the availability of multiple networks at the same time, and/or an improved performance of the network by offloading or load-balancing of traffic between the various networks. A complete commonly agreed and standardized approach for the management of hybrid communication does not exist yet. There are essential technical standards in support of hybrid communication, i.e. EN ISO 17423:2017 (Intelligent transport systems -- Cooperative systems -- Application requirements

[1] https://ec.europa.eu/transport/sites/transport/files/themes/its/doc/c-its-platform-final-report-january-2016.pdf

 © InterCor Consortium

and objectives), and ISO 24102-6 (Intelligent transport systems -- ITS station management -- Part 6: Path and flow management - currently under revision).

One of the objectives of InterCor is to provide C-ITS services on a broader scale by specifying, using and fostering a hybrid communication approach to utilise a combination of cellular (network-based) communication and direct (localized) ITS-G5 communication. InterCor will focus on creating international interoperability for those services, based on hybrid communication solutions that are under development in the participating countries. In contrast, InterCor does not focus on the improved quality of service itself by combining multiple networks, but extending the geographical availability of services for end-users by connecting hybrid solutions in different countries.

Hybrid communication in InterCor is further reduced in scope by only considering ITS-G5 and currently available cellular technologies: other communication technologies and future extensions of these two technologies are not (explicitly) taken into account.

Sub-activity 2.1a has provided specifications for ITS-G5 communication for a subset of the InterCor services. These already provide international interoperability. Sub-activity 2.1b has the task to develop specifications for a complete hybrid solution, where these specifications serve as a basis and should (preferably) not be changed to extend the support to cellular technology.

2.1 InterCor context

Within InterCor activity 2.1b, a high-level system description has been developed that supports interoperability between hybrid communication solutions that exist or are under development in the participating countries. This system description includes two interfaces that need to be specified to realize cross-border interoperability. The first interface (called IF1) is the ITS-G5 interface between vehicles and road side systems, as defined by sub-activity 2.1a. The second interface (called IF2) is an interface between back-office systems, to support the exchange of the information between back-offices required to support the services (also) via cellular communication[2].

The approach for this IF2 specification is dynamic and should be generally applicable. This IF2 specification will be developed in two stages. Version 1 of interface IF2 is limited to the InterCor services Road Works Warning (RWW), In-Vehicle Signage (IVS), and Green Light Optimal Advisory (GLOSA). This document provides version 1 of the specifications of IF2.

[2] A third interface, IF3, is also defined, but for hybrid interoperability, this interface does not need to be standardized.

 © InterCor Consortium

These specifications need to be implemented by at least 2 countries, and tested in the Hybrid TESTFEST, carried out by activity 2.2.

Version 2 of the IF2 interface specifications will include all (relevant) InterCor services. Lessons learned from the development, implementation and Hybrid TESTFEST will be included. Furthermore, it will be investigated if and how the PKI solution as developed by sub-activity 2.1c will be integrated.

The IF1 interface has already been finalised in sub-activity 2.1a and can be found in the M3 milestone deliverable [1] . It describes the ITS-G5 interface for Road Works Warning (RWW), In-Vehicle Signage (IVS), Green Light Optimal Speed Advisory (GLOSA) and Probe Vehicle Data (PVD).

2.2 Purpose of this document

The purpose of this document is to provide specifications of IF2 for the services RWW, IVS, and GLOSA. It provides the system overview in chapter 3. Chapter 4 describes the requirements for the interface. The technical design considerations are provided in chapter 5, while the technical specifications are described in chapter 6. The document should contain sufficient detail to allow for implementations of IF2. It is expected that during the implementation of IF2, several technical details need to be filled in. It is advised that the implementing organisations should be brought in contact with each other and with representatives of activity 2.1b to be able to discuss these aspects, e.g. in the context of the TESTFESTs (activity 2.2) and/or pilots (activity 3).

2.3 InterCor Contractual References

InterCor (Interoperable Corridors) links the C-ITS corridor initiatives of the Netherlands (among which the C-ITS Corridor Netherlands-Germany-Austria), the French (among which the one defined in SCOOP@F) and extends to the United Kingdom and Belgium C-ITS initiatives.

InterCor is an action co-financed by the European Union under the Grant Agreement number INEA/CEF/TRAN/M2015/1143833. The Project duration is 36 months, effective from the 1st of September 2016 until the 31st of August 2019. It is a contract with the Innovation and Networks Executive Agency (INEA), under the powers delegated by the European Commission.

Communication details of the Agency:

Any communication addressed to the Agency by post or e-mail shall be sent to the following address:

Innovation and Networks Executive Agency (INEA)

Department C – Connecting Europe Facility (CEF)

Unit C3 Transport

B - 1049 Brussels

Fax: +32 (0)2 297 37 27

E-mail addresses: General communication: inea@ec.europa.eu

For submission of requests for payment, reports (except ASRs) and financial statements: INEA-C3@ec.europa.eu

Any communication addressed to the Agency by registered mail, courier service or hand-delivery shall be sent to the following address:

Innovation and Networks Executive Agency (INEA)

Avenue du Bourget, 1

B-1140 Brussels (Evere)

Belgium

TEN-Tec shall be accessed via the following URL:

https://webgate.ec.europa.eu/tentec/

All communication with the INEA or the European Commission shall be done via the Project Coordinator, Mr. Ronald Adams.

 © InterCor Consortium

3 System overview

The InterCor hybrid communication specifications aim at providing service interoperability between implementations from the different participating countries. In other words, a vehicle from country 1 that is driving in country 2 should at least be able to use all services that are supported in both countries on either communication technology. Since the architectures of the ITS systems in the participating countries are significantly different, service interoperability will be realized by specifying a limited set of interfaces that need to be implemented, on top of (existing) country specific implementations. A high-level diagram describing the hybrid solution for the InterCor services is depicted in Figure 1.

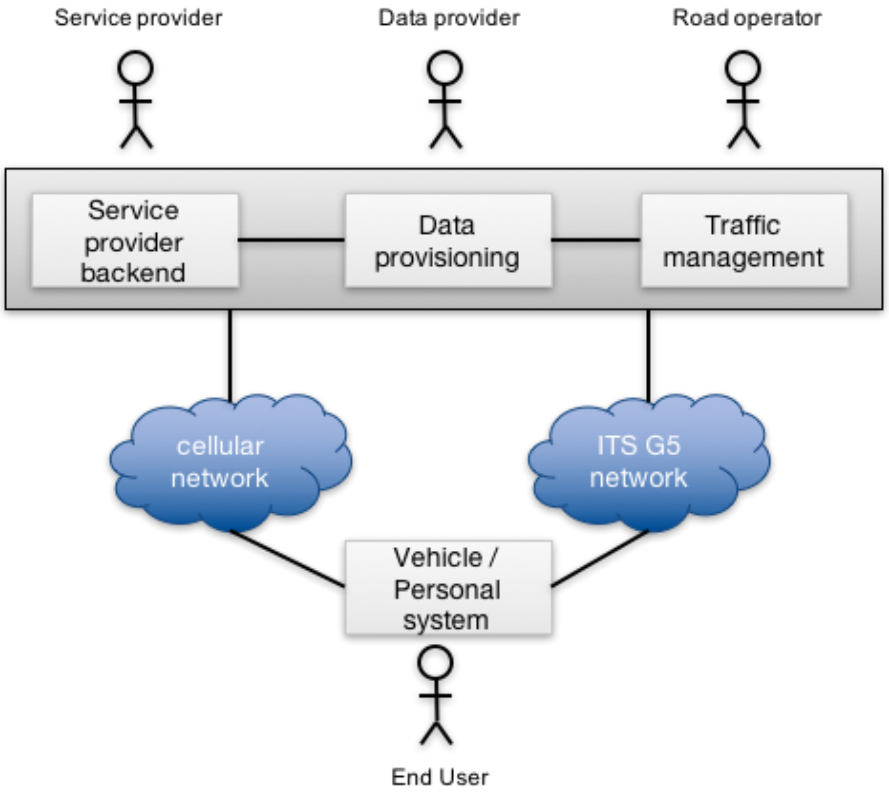


Figure 1: High level diagram for hybrid communication

This diagram should not be interpreted as a formal architecture. It is intended to identify possible subsystems, actors, and networks. Not all subsystems have to be present in every country, nor do all the connections have to be implemented always.

The main actors and subsystems in the diagram are:

- End user: in InterCor, most services are delivered to end users in a vehicle. These services are delivered either on a personal device of the end user (smart-phone, navigation system, etc.), or via systems integrated in the vehicle (on-board unit, etc.).

 © InterCor Consortium

It is also possible that services are delivered via a combination of the above. These are indicated in the diagram with the Vehicle/Personal system.

- Data provider: The data provider is responsible for the data provisioning.
- Service provider: service provider with its back-end systems supports (a part of the) service delivery to the end users.
- Road operator: road operator is the entity in charge of operating a road network and managing traffic. In many services related to traffic management, a road operator is involved with its traffic management systems.

The main subsystems in the diagram are:

- Network: for hybrid communication, at least two independent networks are required. In InterCor, these will be an ITS-G5 based network and a cellular network. These networks connect at least the service provider and traffic manager with the end user, but can also link different end users with each other.
- Traffic management: to enable data exchange between traffic management systems and road side display systems (VMS, panels, traffic signs, etc.) a traffic management system is included in the diagram.
- Service provisioning: to enable data exchange for specific services between end-users and service provider a service provisioning system is included.
- Data provisioning: to enable data exchange between the traffic management systems and the service provider systems, a data provisioning system is included. In practical implementations, the data provisioning could really be an entity on its own (e.g. the NDW in The Netherlands), but could also be integrated e.g. in the traffic management and/or service provider back-end. Furthermore, it could be a rich function including data aggregation, data conversion, etc., or it could only implement a direct forwarding algorithm from e.g. a traffic management system via a cellular network to the relevant end-users.

The box surrounding the traffic management, data provisioning, and service provider backend and the lines connecting from the different networks to this box, are used to indicate that any of the back-office systems could have access to both networks. This does not imply that every back-office system *will* always have access to both networks.

3.1 Interfaces

Based on this high-level diagram a hybrid interoperability diagram is depicted in Figure 2. Note that, similar to Figure 1, this figure should not be interpreted as a formal architecture.

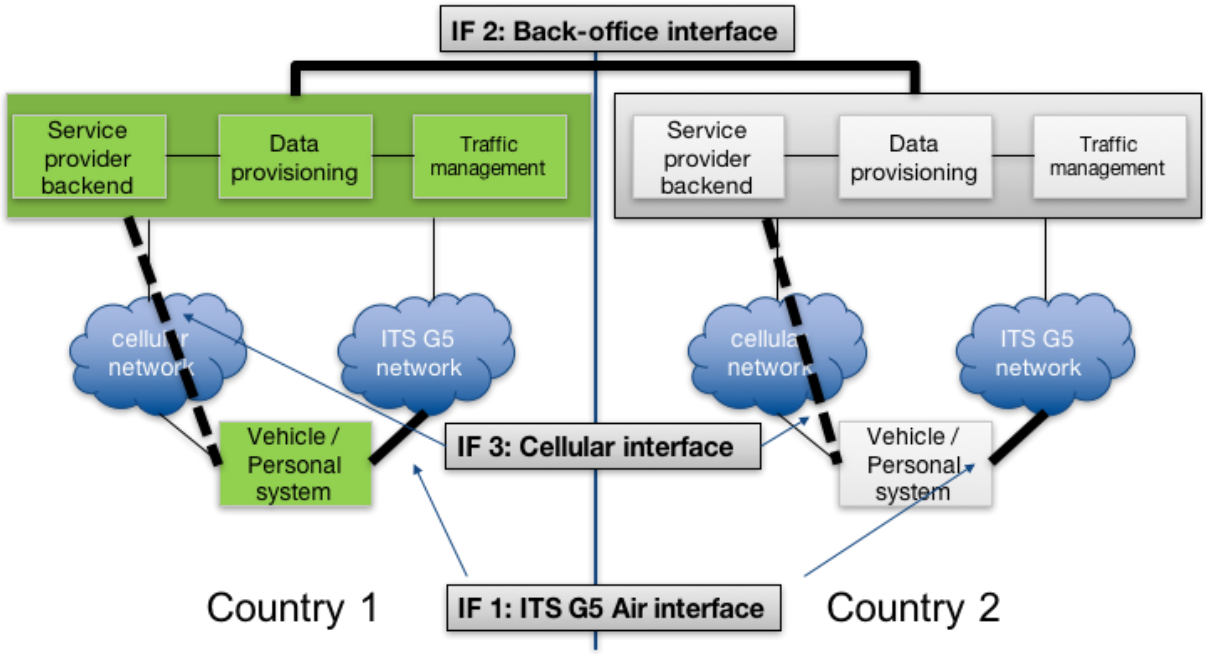


Figure 2: Interoperability diagram for hybrid communication.

The interfaces required to realize service interoperability as depicted by the thick black lines in Figure 2 are:

- IF1: ITS-G5 air interface - IF1 is the air interface on the ITS-G5 channel which is specified in InterCor activity 2.1a, based on existing European standards and profiles. To support authenticated message exchange, also the PKI solutions needs to be integrated over the different countries, which are being developed in activity 2.1c.
- IF2: Back-office interface: IF2 is an interface between the back-offices to exchange information relevant for the service delivery via the cellular network. It could be implemented by either the service provider backend, the data provisioning, and/or the traffic management. The diagram illustrates that IF2 can be implemented by all central components, but it is not required that this is actually implemented by all components simultaneously: all relevant data needs to be disclosed at least at one location.
- IF3: Cellular interface: IF3 is the interface between the end-device and back-office on the cellular link. Any of the back-office systems could be used to connect via IF3 to the end user systems.

 © InterCor Consortium

3.2 Services

The C-ITS services supported by InterCor are specified in Activity 2.1d [2]. This version of the specifications supports the following services:

- In-Vehicle Signage (IVS): IVS service provides receiving ITS stations the service to inform drivers about static as well as dynamic road signs and variable message signs.
- Road Works Warning (RWW): In this service, the road operator can send information about road works, restrictions, traffic instructions, etc. through the road-side units to the vehicles.
- Green Light Optimal Speed Advisory (GLOSA): In this service, vehicles approaching a traffic light receive information about the topology of the intersection and the phase schedule of each traffic light signal and calculate the optimal approaching speed.

3.3 Implementation examples

An example on how service interoperability should be realized is shown in Figure 3. A vehicle originating from country 1 is in country 2. A single event pops up in a traffic management system in country 2. Immediately this event is (1) forwarded via the ITS-G5 network in this country via IF1. The same event is also forwarded to service providers in other countries using IF2. Service providers can forward the relevant events to a vehicle via the existing cellular network infrastructure of mobile network operators, with their roaming services to support pan-European coverage.

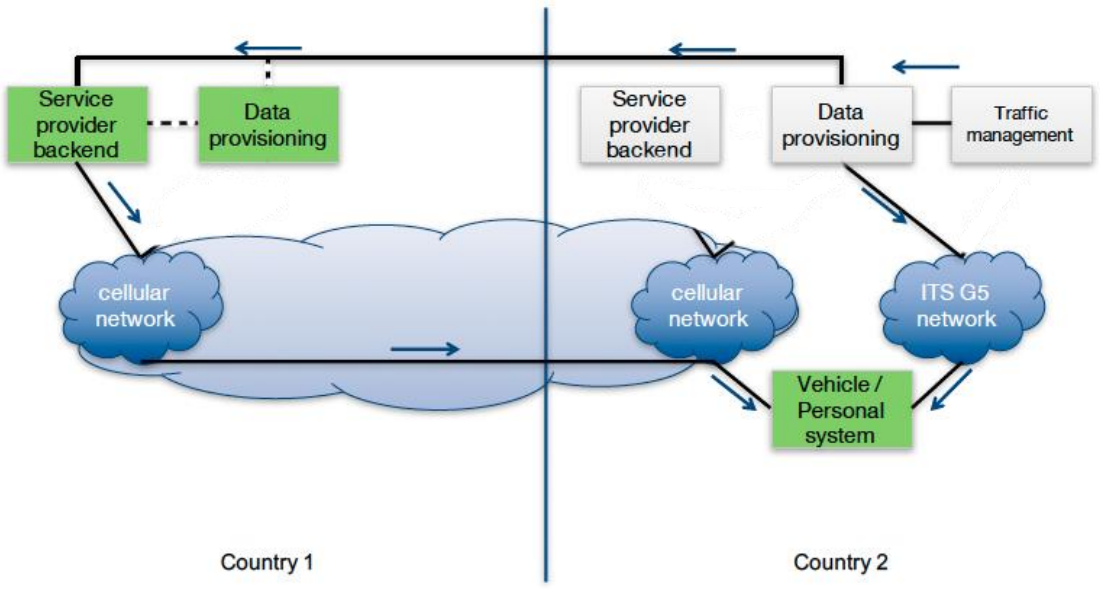


Figure 3: Example of service interoperability for a vehicle originating from country 1 in country 2.

 © InterCor Consortium

3.3.1 Service over IF1

Let us consider the example of a traffic management system sending information, e.g. RWW via data provisioning and the ITS-G5 network to vehicles on its road network. A vehicle from country 1 driving on this network and equipped with the proper systems receives this message via the ITS-G5 network, i.e. via IF1, and will be able to use this information as IF1 is fully specified and interoperable for implementations from country 1 and 2.

3.3.2 Service over IF2

In this example, consider the information provided, e.g. GLOSA, via data provisioning in country 2 to a service provider in country 1 that could be sent to the vehicle in country 2. The service provider in country 1 can use the existing solution to provide this information to the vehicle via a standard roaming cellular connection. To allow this service interoperability, only IF2 needs to be in place and interoperable and the data provisioning in country 2 uses IF2 for communication with service provider in country 1. As a variation, also the option is drawn with dotted lines where data provisioning in Country 1 gets the data from country 2 and forwards the data to the service provider in country 1 via the interface that is already in use for data from country 1.

 © InterCor Consortium

4 Requirements

In the requirements in this document, the following definitions apply: SHALL and SHALL NOT will be used to indicate requirements. SHOULD and SHOULD NOT are used to indicate recommendations. WILL and WILL NOT will be used to describe the behaviour of systems outside the scope of this document. MAY and MAY NOT are used to describe optional behaviour.

4.1 Functional and non-functional requirements

ID	SYS01 Service Support
Requirement	IF2 SHALL implement the data exchange of all operational data required to implement the InterCor services for road users originating from one country and driving in another country.
Note	

ID	SYS02 Time criticality
Requirement	A system that implements IF2 SHOULD adhere to the specific delay requirements posed by the individual services.
Note	Guidelines for target values of maximum end-to-end delay will be provided per service. For a chain of systems from stakeholders, target values will be given for the delay budget per system (national, international).

ID	SYS03 Buffering
Requirement	A system that implements IF2 MAY buffer messages it receives, depending on the timing requirements of the specific service.
Note	

ID	SYS04 Message delivery
Requirement	A system that implements IF2 SHOULD adhere to the specific message delivery reliability requirements posed by the individual services.
Note	When resources are constraint, a balance has to be found between the

 © InterCor Consortium

	reliability of the delivery of messages (to what extend can messages be lost), and the acceptable delay. This will depend on the specifics of a service.

ID	SYS05 Extensibility
Requirement	The protocol used on IF2 SHOULD be extendible, as to support services that require other data to be exchanged.
Note	

ID	SYS07 Message independency
Requirement	The data exchanged via IF2 SHOULD allow the fulfilment of the filtering requirements without the need to inspect the content of the service data being exchanged.
Note	This means that it SHOULD not be needed to decode the MAP, SPAT, DENM and/or IVI messages that are exchanged to implement the geographical and other filters. This does not imply that a system implementing IF2 is not *allowed* to inspect the content.

ID	SYS08 Multiplicity
Requirement	Systems that implement IF2 SHALL support multiple simultaneous connections. Systems that provide data SHALL be able to support multiple simultaneous receivers of the same data type. Consumers of data SHALL be able to receive data from different providers simultaneously.
Note	This requirement is intended to support different geographical regions and/or different services simultaneously by the same system.

ID	FILTER01 Geographical coverage
Requirement	A system that implements IF2 SHALL be able to provide information on the geographical region for which it provides data.
Note	

 © InterCor Consortium

ID	FILTER02 Geographical filtering
Requirement	A system that implements IF2 SHOULD provide a mechanism to limit the geographical coverage of data via IF2 to a specific connection.
Note	This requirement does not specify the level of detail for the geographical filtering, nor the method of specifying the area that need to be filtered. It is foreseen that this can be done via an administrative process before a connection is made, during the connection phase, and/or during the operation of data exchange.

ID	FILTER03 Message rate filtering
Requirement	A system that implements IF2 SHOULD be able to filter the messages based on the message rate it generates or receives, before providing them via IF2.
Note	Provisions SHOULD be made to reduce the message rate transmitted via IF2, to allow for a scalable solution. The detailed requirements will be based on the specific service requirements.

ID	FILTER04 Duplicate filtering
Requirement	A system that implements IF2 SHOULD NOT transmit the same message more than once.
Note	This requirement is intended to prevent the transmission of multiple, identical messages, as is common on the ITS-G5 channel, as IF2 is expected to be implemented on top of a reliable communication stack and thus messages are normally not lost in transmission. Note, that duplicate message removal is not a must, and is therefore not guaranteed, as absolute guarantee might proof to be complex/expensive. If duplicate removal is a must for the receiving party, then it SHALL implement its own duplicate detection.

ID	FILTER05 Message filtering per message type
Requirement	Messages SHALL be filtered on message type.

 © InterCor Consortium

Note	Several services are supported, and filtering per service (RWW, IVS, GLOSA) shall be possible.

ID	MES01 Message support
Requirement	IF2 SHALL support the exchange of ASN.1 encoded messages, like MAP, SPAT, DENM, and IVI messages, as defined and profiled in [1].
Note	

ID	SEC01 Authentication
Requirement	It SHALL be possible to authenticate a system that connects via IF2.
Note	Authentication could be done e.g. via certificates, username/password and/or IP addresses. Note, that this requirement refers to the systems connecting via IF2, which does not mean that the originating source of a message transmitted via IF2 is authenticated.

 © InterCor Consortium

5 Interface specifications

5.1 Communication protocol

Based on the requirements, the IF2 should support flexible, real-time exchange of (potentially) many messages from different sources to multiple destinations. For the services under consideration, the sources are traffic management systems, and the destinations are back-office systems that will provide services to end-users in vehicles. It should be possible to filter messages on several criteria, such as type of message, validity in time and geographical information.

These requirements fit nicely with message queue protocols and messaging systems. In a typical message-queueing implementation, multiple queues are defined in a Message Queue server (MQ server). An application registers with the MQ server, and listens for messages placed into the queue. Other applications connect to the queue on the server and transfer messages onto it. The MQ server stores the messages until a receiving application is available and then sends the messages to the receiving application, which then processes the message in an appropriate manner. Depending on the specific MQ protocol and MQ server software, many configurable options exist per server, per client application, per queue, or per message. These options include policies on message delivery (e.g. every message to a single listener, or to all listeners), security (who is allowed to access which queues or messages), message retention (are delivered messages purged, or retained for future listeners), filtering, etc. A common aspect of MQ servers is that they separate the actual message being transmitted from the protocol and logic of handling the messages.

Two MQ protocols are being considered:

- AMQP is used in the EU project “NordicWay[3]”. NordicWay combines AMQP with simplified and extended DATEX-II based messages
- MQTT is used in the Dutch project “Talking Traffic” by some partners, with ASN.1 encoded, standardised messages (CAM, DENM, IVI, MAP, SPAT, etc.)

[3] NordicWay has defined a complete architecture for end-end cross border service delivery of several connected traffic management and traffic information services. An AMQP based solution is used for the distribution of messages between different systems within this architecture.

 © InterCor Consortium

The design focus of AMQP is to provide messaging solutions in a wide range of scenario's, and has therefore many options and a great flexibility. This of course comes with a price in complexity. MQTT has originally been designed as a simple, small footprint solution for embedded and other relatively dumb devices. MQTT therefore has fewer features, but could also be simpler to implement. MQTT is therefore more focussed on supporting solutions with a small memory and CPU footprint, whereas AMQP allows for better optimization of high-performance systems.

On a high level, AMQP seems more suitable as communication protocol for IF2, although it will be possible to make a workable solution based on either technology. Furthermore, an AMQP solution might be easier to align in C-Roads on a Europe wide scale (because of the usage in NordicWay). An advantage of using MQTT based solution is that an implementation is already available from one of the InterCor related projects (Dutch project Talking Traffic), so experience is at hand, and implementations might be quicker to realize.

Several mature open source and commercial MQ servers are available implementing either of the protocols, and some of them implementing both messaging standards. This means that switching at a later stage (or supporting both protocols) might have only limited impact. It is expected that the realization of IF2 based on either of the MQ protocol mainly consists of deploying and configuring an off-the shelf solution, with no, or limited, software development required. Depending on the existing systems used at the moment, it might be necessary to develop adapters to connect those systems to the MQ server.

Based on this assessment, it has been decided that the AMQP protocol SHALL BE used as the message protocol for version 1 of the IF2 specifications.

5.2 Message formats

It should be possible to handle the messages being transferred without the need to read the actual content of the message, i.e. IF2 should be implemented as a wire protocol[4]. Therefore, all information that is relevant for the transport of the message should be (also) available without the need to decode the actual message. This allows for easier extension to other services and messages, to be able to handle different versions of the same message simultaneously, and would also facilitate the transmission of messages that are not encoded as standardized ASN.1 messages. Most likely, this will be required to be able to support all seven InterCor services in the future. Note, that this is comparable to how information

[4] See e.g. https://en.wikipedia.org/wiki/Wire_protocol for a definition of a wire protocol.

 © InterCor Consortium

contained in the messages is repeated in the GeoNetworking header in ITS-G5 based communication.

Table 1 gives an overview of required information, including the message data itself.

Table 1: Relevant information for routing and filtering of messages.

Field	Required	Description
Message	Mandatory	The actual message/payload. Currently foreseen DENM/IVI/MAP/SPAT. The content of the messages should follow the profile, as defined in [1]. Messages should not be transcoded, e.g. to DATEX II messages. The same ASN.1 encoding rules should be followed as for IF1 (i.e. as used on the ITS-G5 channel)
Message type	Optional	Message type name (DENM/IVI/MAP/SPAT)
Message version	Optional	Version of the message type
Originator	Optional	The source of the information, typically a (short) name of the road operator. This can be relevant for the trustworthiness of the information, and for business aspects.
Location	Optional	Relevant target location of this message. The intent is to be able to use it for filtering. The detailed location information is contained inside the message. These should be consistent
Time validity	Optional	Messages have a limited validity. “Old” messages do not have to be forwarded. The time validity can be specified based on an absolute timestamp, or on a generation timestamp and a validity time.

For a MQ based implementation, the information can be provided in the message queue and/or message properties, including the message topic. Similar properties have been defined in NordicWay as well.

5.3 Filtering

The following filtering requirements need to be filled in:

1. Message filtering per message type and version
2. Message filtering based on originator
3. Geographical filtering (optional function): at least per country, optionally per geographical area per country (e.g. city area or highway numbers, latitude/longitude bounding box).
4. Message rate filtering (optional function)

 © InterCor Consortium

Filter 1 and 2 can be easily implemented in a MQ based solution. The geographical filtering is a bit more complex, as it intrinsically is a 2-dimensional filter. Filtering based on logical names is not a scalable solution, as it requires both sides of the interface to know in advance what the available logical names are. Therefore, filtering will be based on absolute coordinates and bounding boxes.

Geographical filtering means mapping the area of relevance of a message to the area of interest of a receiver. Two options are considered:

1. Filtering based on latitude/longitude positions of the relevance area, and squared, ellipsoid, ore polygonal areas of interest. The receiver should in this case register its area of interest, and a dedicated filter should inspect the latitude/longitude position of every message to determine whether it falls inside this area of interest. This approach could be extended by also allowing an area of relevance (instead of a point), but that would require matching areas with areas, which is more complex than areas with points. This can be avoided by extending the area of interest somewhat, and limiting the relevance area to a point location. Such a filter is not commonly available in MQ servers, but could be added if the server supports filtering extensions.
2. Filtering based on quadtree paths for both the area of relevance and the area of interest. A quadtree is a tree data structure in which each internal node has exactly four children and are used to partition a two-dimensional space by recursively subdividing it into four quadrants or regions. The number of recursive steps (“zoom level”) determines the size of the area, whereas the index of the quadrants (“tiles”) determines the exact location. It can be seen as a way of filtering based on squared bounding boxes with predetermined locations and sizes. Because of this, no matching areas on areas does not generate any additional complexity.

Option 1 is more generic, but would also require more implementation effort and processing resources. The latter might become important for large scale deployments. Option 2 results in less complex implementations and better performance. Note, that for messages on events at fixed locations (e.g. IVI messages on speed limits for RWW), the quadtree is also fixed and could even be determined at forehand.

In version 1 of these IF2 specifications, option 2, quadtree based filtering, SHALL BE used.

5.4 Security

The implementations of IF2 should be made secure, in a sense that the providers of the MQ server should make sure only authorized systems can connect. AMQP uses SASL (Simple Authentication and Security Layer) mechanisms. The exact details are currently out of scope, but these should be made available by the organizations that implement IF2.

 © InterCor Consortium

5.5 Concluding remarks

The IF2 specifications will be based on the AMQP protocol because of its reliability and interoperability. The detailed technical specifications are provided in the next chapter, and focus on specifying how this protocol will be applied for IF2.

 © InterCor Consortium

6 Technical specifications

IF2 SHALL BE based on AMQP 0.9.1[5] [3].This specification will be described in terms of clients and brokers. The broker is the central node that implements the message distribution. A client connects to the broker to send (publish) or receive (consume) messages. In the limited services scope of this version of the specification and because only IF2 is in scope of the specifications, all clients only consume messages. In the complete implementation, however, other systems need to be implemented that act as publisher of messages. It is expected that every member state will implement a broker to provide the messages originating from their geographical area.

A high-level overview of a AMQP system is shown in Figure 4

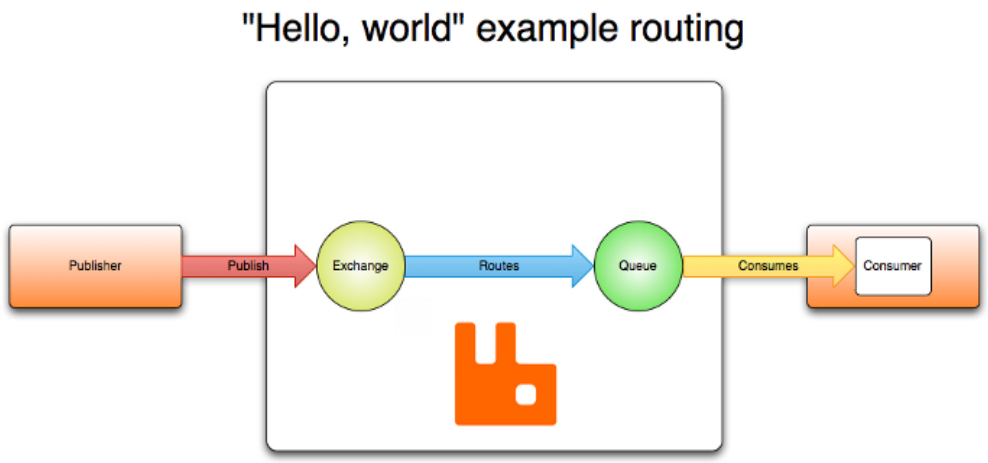


Figure 4 High level overview of an AMQP system.

These specifications focus on the operational aspects of IF2. Various aspects will be left open for decision of the implementing organisation. To be able to have clients and brokers interact with each other, these aspects need to be documented for every broker implementation. In the following, these documents are referred to as the *deployment documentation*.

6.1 Server and virtual host

A single broker implementation can serve multiple AMQP ecosystems at the same time by means of virtual hosts. Virtual hosts SHOULD be used to separate development, test, and operational environments. The deployment documentation should describe which virtual

[5] http://www.amqp.org/specification/0-9-1/amqp-org-download

hosts are available and for what purpose. A client SHOULD be able to connect to the correct virtual host.

6.2 Exchanges

Messages are published on a specific exchange. The name of the exchange SHOULD match the name of the message that is being exchanged. A broker does not have to serve all message types. The deployment documentation SHOULD specify which exchanges are supported. For the current services, these include DENM, IVI, MAP and SPAT.

The exchanges SHALL be implemented as topic exchange. This means that every message published to the exchange is routed to 0 or more queues, depending on the value of the topic and possible other filters. The exchange SHOULD be durable (i.e. should survive), and not auto-deleted (i.e. should not be deleted if no queues are available to deliver messages to). Table 2 gives an overview of the required settings.

Table 2: Exchange parameters.

Property	Value	Note
Virtual host	Implementation dependent	Provided by the deployment documentation
Name	Message type name	Currently, DENM, IVI, MAP, SPAT
Type	Topic	Allows routing based on topic
Durable	True	Exchanges need to survive broker restart.
Auto-delete	False	Exchanges should not be deleted when the last queue is deleted.

Note, that for the current services, in total 4 exchanges, named DENM, IVI, MAP, and SPAT SHOULD be made available.

6.3 Message queues

To facilitate low management overhead, no queues are predefined for clients. Instead, every client requests the automatic generation of a queue, bound to a specific exchange, and with specific parameters. The queue parameters of Table 3 SHOULD be used when creating queues by a client.

 © InterCor Consortium

Table 3: Message queue parameters.

Property	Value	Note
Name	empty	The broker will provide an automatic queue name
Exclusive	True	Queues should not be shared by multiple clients
Durable	False	Queues will be recreated every time a client connects
Auto-delete	True	Queues should be automatically deleted when the client disconnects

Message queues MUST BE defined with a maximum queue length, and SHOULD be defined with a maximum time-to-live. If no maximum queue length is specified, the queue can fill up indefinitely (until resources run out on the server), if the client does not take the messages from the queue. The maximum time-to-live ensures that "old" messages are discarded, instead of delivered.

A message queue is bound to an exchange. When binding, a routing queue used for filtering messages is provided. See details in the next section on the definition of the routing key.

6.4 Message publishing and consuming

Messages are published with a set of properties and a routing key[6]. It is the responsibility of the organisation implementing the broker to ensure that all properties and the routing key are correct. The routing key encodes the information that is seen as most relevant for filtering, and should be defined as follows:

```
<message type>.<message version>.<provider>.<subtype id>.{quadtree path}
```

Message type is the name of the message (DENM, IVI, MAP, SPAT, in capitals), message version the exact version of the message, where a "." (period) is replaced by a "_" (lower dash). Provider is an (arbitrary) name of the provider of the information, typically a (short) name of the road operator that is responsible for the message. The subtype id can be used to indicate a subtype of the message, e.g. the actionId of a DENM message.

[6] Publishing is not part of the IF2 specification, but the routing key required for the filtering on IF2 is provided during publishing. Therefore, publishing is still described here.

 © InterCor Consortium

Note, that a "." period is used to separate words in the routing key, and so SHALL NOT be used as part of any of the words.

The quadtree path is built from the characters 0, 1, 2 and 3 defining a tile and zoom level, separated by "."; the number of characters is equal to the zoom level (i.e. the size of the relevance area), whereas the actual values determine the location. See http://www.maptiler.org/google-maps-coordinates-tile-bounds-projection/ for a demonstration of the quadtree path definition and a Python implementation of tool that, among others, converts a WGS84 coordinate into a quadtree string.

The zoom level to use is 18. In the Netherlands, this zoom level is equivalent to mini maps of roughly 100 meters square. The following algorithm must be used:

- Determine the relevant location of the message to be sent.
- Find the mini-map at zoom level 18 that contains this point.
- Publish on the quadtree path of the mini-map found.

The relevant location depends on the type of message. For the current message types, these SHOULD be determined as follows:

- For DENM, the event-Position,
- For IVI the reference-Position,
- For MAP the refPoint,
- and for SPAT the refPoint of the corresponding MAP.

The parameters are profiled in [1]. In addition to the routing key, the properties as shown in Table 4 should be set:

Table 4: Message properties.

Property	Description	Note
ttl	Time-to-live	Maximum live time of a message in milliseconds.
lat	Latitude	Latitude of the relevance point.
lon	Longitude	Longitude of the relevance point.

The time-to-live (ttl) SHOULD be used by the broker to prevent messages of being forwarded that are not relevant anymore. The latitude and longitude can be used to filter based on location.

Messages should only be published again:

1. If their content has changed, on bit-level

 © InterCor Consortium

2. If their ttl has expired, but they are still valid
3. At least once per hour

This is different from the republishing strategies on IF1 (ITS-G5 interface), due to the different nature of the interface, i.e. reliable, and not limited by the geographical distance between sender and receiver. The third reason is to allow new consumers of messages to catch up with messages that are still valid, but have been published before they subscribed. This is typically relevant for MAP, which hardly ever change, but are required to be able to interpret the SPAT messages.

A consumer selects the messages it wants to receive when it binds its (automatically created) queue to an exchange. The exchange name selects the type of message. The routing key determines the filter that needs to be applied. In creating the routing key filter, a * can be used as a wildcard for a single word, and a # can be used for a wildcard of zero or more words. If all messages in all versions from all providers and without geographical filter are of interest, then the routing key filter is as simple as a single "#". If only DENM messages of version 1.2.1 coming from RWS in the area of Rotterdam are of interest, the routing key filter would be

```
DENM.1_2_1.RWS.*.1.2.0.2.0.2.1.1.2.#
```

Where the series of numbers at the end is the quadtree path at zoom level 9 of the Rotterdam area. The "*" makes that all subtypes are accepted, the "#" makes that everything at a higher zoom level is accepted.

Because all messages are published at zoom level 18, a filter should always be of a lower number zoomlevel, i.e. 0-18.

6.5 Security

A broker implementation SHOULD at least use identification and authentication to ensure the security of the system. The deployment documentation SHOULD describe how to obtain the required credentials to be able to access the broker. Additional measures, including the use of TLS, VPN tunnels, or IP address filtering, could be put in place as well, and SHOULD also be documented in the deployment documentation.

6.6 Message content

All messages SHOULD be ASN.1 encoded messages (using unaligned PER), as defined in the ITS-G5 profiles of the supported services. See [1] for the detailed specifications. It is the

 © InterCor Consortium

responsibility of the organization that provides the broker to ensure that the messages are consistent with [1].

 © InterCor Consortium

7 Implementation example

7.1 Introduction

This chapter provides a technical description of a basic implementation of IF2. This chapter is only informative, and is not part of the specifications. To create a complete example based on an AMQP message broker, three components are required: a message broker, a publisher of messages, and a consumer of messages. The IF2 specifications cover only the interaction between the message broker and a consumer. In this example implementation, however, also the publisher part is described, as in any implementation also the publisher side is required to have any message exchanged.

This example is based on the RabbitMQ message broker. The broker needs to be configured appropriately to support the IF2 specifications. A basic configuration is described in the next section. Simple publisher and consumer applications are provided in Java in the following 2 sections. These only describe the interaction with the message broker, and do not cover how messages are obtained to be published, or further processed after being received.

7.2 Message broker configuration

The steps below provide a basic RabbitMQ configuration that supports these IF2 specifications. Screenshots are provided from the management interface of RabbitMQ, to provide details of the configuration steps

1. Download and install RabbitMQ. This can be obtained from https://www.rabbitmq.com/download.html. Also install the management plugin, which allows to manage the broker via a web interface. This document is based on RabbitMQ 3.6.14. The management interface can be reached on the installation machine via http://localhost:15672.
2. Create an administrator user, delete the default guest user. Also create users for the consumers and producers of messages.

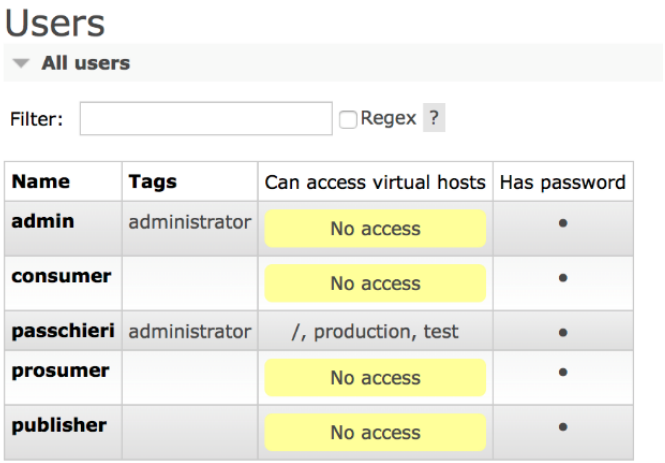


Figure 5: Example user configuration. Please note that the virtual hosts permissions will be changed in a later step, and then this table will also automatically be updated.

3. Create virtual hosts for e.g. test and production. More virtual hosts can be created, depending on how the different development stages are separated. Here, a single test and single production virtual host is created. In the remainder, only the test virtual hosts is configured. The production virtual host should be configured similarly.

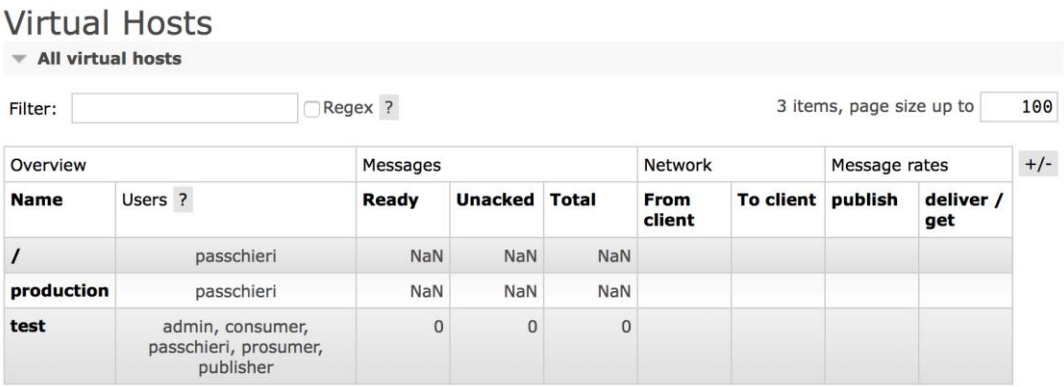


Figure 6: Example virtual host configuration.

4. Configure the appropriate rights for the different users. See figure below for details. The permissions per virtual host can be changed by selecting an individual virtual host in the overview of virtual hosts. All normal users need Configure permissions, otherwise they are not able to create and attach their own queues. Strictly speaking, does the admin user only require Configure permissions, but allow read and write facilitates development more easily.

 © InterCor Consortium

User	Configure regexp	Write regexp	Read regexp	
admin	.*	.*	.*	Clear
consumer	.*		.*	Clear
passchieri	.*	.*	.*	Clear
prosumer	.*	.*	.*	Clear
publisher	.*	.*		Clear

Figure 7: Example user's permission configuration.

5. Configure a policy to ensure that queues will never block the system as a whole. Here, a maximum TTL of 60s and maximum queue length of 1000 messages is enforced via a policy. For the test virtual hosts, these are kept low, which will ensure that possible incorrect data will be flushed automatically quickly, facilitating development and testing. For a production environment, these should be set realistically, based on the expected message rates and acceptable delays due to message handling.

Policies

All policies

Filter: Regex ?

Virtual Host	Name	Pattern	Apply to	Definition	Priority
test	**TTL**	.*	queues	message-ttl: 60000 max-length: 1000	0

Figure 8: Example policy configuration.

6. Create the exchanges for MAP, SPAT, DENM, and IVI messages. All exchanges should be durable, non auto delete, non internal, topic exchanges. Note, that RabbitMQ will make several default exchanges that cannot be removed. You can leave them as is.

 © InterCor Consortium

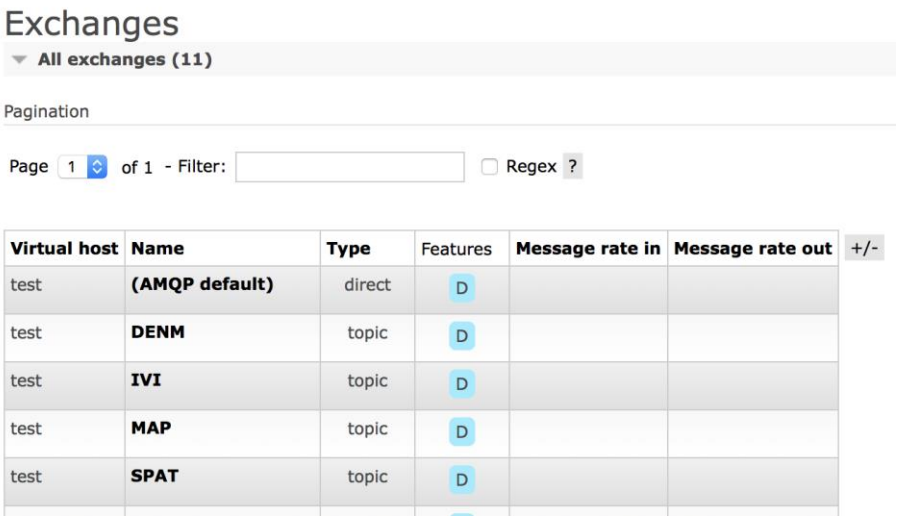


Figure 9: Example exchange configuration.

These steps create a basic setup for a RabbitMQ broker that supports the IF2 specifications. Out of scope of this description is possible firewall or VPN solutions that could be made to increase the security of the solution. In this example, only a few users have been created. In an actual implementation, it is expected that every client of the system will get its own login credentials.

If messages should be available via multiple virtual hosts, e.g. real operational data from the virtual host operation in the virtual host test, then a plugin named shovel can be used to bulk transport messages from 1 virtual host to another,

7.3 Java consumer and publisher code

This section describes the basic code required to interact via IF2 with the message broker, as configured in the previous section. This code is only intended to demonstrate the basics of connecting, publishing, and consuming messages. It is not intended as a complete, operational implementation of IF2. Error checking is not implemented, and no proper multithreading is implemented. The focus is on explaining how to interact with the message broker in line with these IF2 specifications.

Part of the interaction with the broker, is the creation of the proper routing key, based on the quadtree path. The basic java code provided to calculate the quadtree paths, has been based on the python code from http://www.klokan.cz/projects/gdal2tiles/. Quadtree paths and other tiling schemes can be visualized via http://www.maptiler.org/google-maps-coordinates-tile-bounds-projection/.

The example code can be obtained as a complete Maven project from https://github.com/passchieri/Hybrid-IF2. The project uses the java RabbitMQ amqp-client

libraries for the interaction with the broker. The spring-framework is used to be able to generate an executable jar easily, but is not required to run the code. The code can be loaded e.g. as an Eclipse maven project, and run from within the development environment. If you do not want to use maven, it is still possible to compile the code, as long as the amqp-client v5.0 library from rabbitmq.com is made available in another way. The code assumes java compiler version 1.8 or higher.

All code is included in a single class intercor.if2.sample.IF2Client.java, except for the Quadtree path related code. All code snippets below come from this class, unless otherwise stated.

7.3.1 Connecting to a broker

The connection to the broker is done in 2 steps: First, setup a TCP/IP connection, and then open a channel, an additional abstraction layer provided by amqp. In this example, they are always used together, so a single method connects on both TCP/IP level and channel level.

```
/**
 * Connect to the broker, and generate a channel.
 */
public void connect() {
   ConnectionFactory factory = new ConnectionFactory();
   factory.setHost(HOST);
   factory.setUsername(USER);
   factory.setPassword(PASSWORD);
   factory.setVirtualHost(VIRTUALHOST);

   try {
      connection = factory.newConnection();
      connection.addShutdownListener((ShutdownSignalException cause) -> {
         System.out.println("Connection closed");
      });
      System.out.println("Connection opened");
      channel = connection.createChannel();
      channel.addShutdownListener((ShutdownSignalException cause) -> {
         System.out.println("Channel closed");
      });
      System.out.println("Channel opened");

   } catch (Exception e) {
      e.printStackTrace();
      disconnect();

   }
}
```

The host, virtual host, and credentials are stored in constants, and can be changed in line with the deployed RabbitMQ broker.

The amqp-client library is based on event listeners. Here, only listeners to the shutdown Events are implemented. For a complete implementation, see what other listeners might or must be implemented.

At the end of the interaction with the broker, the connection and channel are closed:

```
/**
 * Close the channel, and disconnect
```

```
 */
public void disconnect() {
   try {
      if (channel != null)
         channel.abort();
      if (connection != null)
         connection.abort();
      System.out.println("Disconnected");
   } catch (IOException e) {
      e.printStackTrace();
   } finally {
      connection = null;
      channel = null;
   }
}
```

7.3.2 Publishing data

In a normal implementation, real data messages are available and should be published to the broker. However, for this example class, a simple helper class has been created to simulate fake data. Every instance of this class constitutes a single message and its meta data. Instead of an ASN.1 representation of a real message, here a string representation of the fake data will be published, so it is easy to see what messages are delivered by the message broker.

```
static private class FakeData {
   double lat; // latitude of the data
   double lon; // longitude of the data
   int zoom; // zoom level at which the data should be published
   String messageType; // message type, In this example always DENM
   String messageVersion; // message version. The current DENM version is
            //1.2.1, encoded as 1_2_1
   String provider;  //identifier of the organisation that publishes
            //the message
   String subtype;   // subtype of the message. In case of DENMs, we use
            //the causecode

   public FakeData(double lat, double lon, int zoom, String messageType, String
messageVersion, String provider,
         String subtype) {
      super();
      this.lat = lat;
      this.lon = lon;
      this.zoom = zoom;
      this.messageType = messageType;
      this.messageVersion = messageVersion;
      this.provider = provider;
      this.subtype = subtype;
   }

   public String getRoutingKey() {
      // none of the strings in the routing key should contain ".".
      //This should be checked/ No extra "." after the subtype, as the
      //quadtree starts with a "."
      return messageType + "." + messageVersion + "." + provider + "."
            + subtype
            + QuadTreeConverter.getQuadTree(zoom, lat, lon, ".");
   }

   @Override
   public String toString() {
      return "FakeData [lat=" + lat + ", lon=" + lon + ", zoom=" + zoom +
               ", key=" + getRoutingKey() + "]";
   }
}
```

 © InterCor Consortium

To publish a single message to a specific exchange, the following code is used. The routing key is obtained from the fake data (could also have been implemented in this code fragment, based on the lat/lon obtained from the Fakedata), and the prescribed headers are filled: lat, lon, and expiration. As data, the string representation of the fake data is taken. In a real implementation, this should be the ASN.1 encoded message.

```
/**
 * Publish a message to an previously opened connection and channel.
 *
 * @param data
 */
public void publishMessage(FakeData data) {
   // Get the routing key
   String key = data.getRoutingKey();

   // As message, we transmit a string representation of the fake data. Normally,
   // this would be the ASN.1 encoded message
   String message = data.toString();

   // Use the message type as exchange name to publish the message to.
   String exchange = data.messageType;
   try {
      if (channel != null && channel.isOpen()) {
         // fill the properties that go along with the message
         BasicProperties.Builder builder = new Builder();
         HashMap<String, Object> headers = new HashMap<>();
         headers.put("lat", data.lat);
         headers.put("lon", data.lon);
         BasicProperties props =
            builder.headers(headers).expiration("10000").build();
         channel.basicPublish(exchange, key, props, message.getBytes());
         System.out.println("published message " + message);
      } else {
         System.out.println("Cannot publish, no channel available");
      }
   } catch (IOException e) {
      e.printStackTrace();
      disconnect();
   }
}
```

These are the basic methods required to publish messages to the broker. These functions are used in the main method to implement several test cases. The publisher and consumer methods have been implemented in a single class, which makes it possible to keep the example code compact. Normally, this would of course be implemented in separate code.

7.3.3 Consuming data

For the consumer, the same connection and channel are used. Only now an automatic queue needs to be defined, and a listener for incoming messages needs to be registered. 2 properties are defined on the queue: a max-length and a message-ttl. This ensures that the queue will never grow indefinitely, even if the messages sent to the queue are not consumed by a listener. Although a policy in the broker enforces this as well, it is good practise to define your own limits also.

 © InterCor Consortium

The listener registered simply prints out the message to standard out, as here string representations of the fake data are distributed. In a normal implementation, the received messages would e.g. be stored in a java.util.Queue and further processed in another thread.

```
/**
 * Based on the connection and channel opened earlier, create a temporary queue,
 * and bind the queue to the correct exchange with the prescribed key. A simple
 * handler is connected to the key, that prints every message received.
 *
 * @param key
 *            The routing key to use
 */
public void startListening(String key) {
   Map<String, Object> args = new HashMap<String, Object>();
   // Just to be sure that we do not block the broker, put some limits on queue
   // length and lifetime of the messages
   args.put("x-max-length", 1000); // limit queue length to 1000 elements.
   args.put("x-message-ttl", 60 * 10 * 1000); // limit max time to 10 minutes

   try {
      DeclareOk queueDeclare = channel.queueDeclare("", false, true, true, args);
      final String queue = queueDeclare.getQueue();
      channel.queueBind(queue, EXCHANGE, key);
      Consumer consumer = new DefaultConsumer(channel) {

         @Override
         public void handleDelivery(String consumerTag, Envelope envelope,
BasicProperties properties,
               byte[] body) throws IOException {
            // If real ASN.1 messages would be transmitted, the message could
            //not be converted to a string like done here.
            System.out.println("Message received from exchange "
                  + EXCHANGE + ":" + new String(body));
         }
      };
      String consumertag = channel.basicConsume(queue, true, consumer);
      System.out.println("Waiting for incoming messages...");
   } catch (Exception ex) {
      ex.printStackTrace();
      disconnect();
   }
}
```

7.3.4 Quadtree paths

The code to calculate quadtrees is distributed over multiple classes, all implementing part of the conversion from lat/lon to quadtree path at a specific zoom level. Lat/lon→ Mercator point → ImagePoint → Tile. The main class is quadtree.QuadTreeConverter.java. This Class provides 2 functions to get a quadtree path of a specific lat/lon location at a specific zoom level, where the separator between the elements of the path can be chosen. Note, that also this code is very basic, and also here no error checking is done. For example, if a lat/lon location is requested that cannot be represented in the Mercator projection, then the code will generate an unhandled exception.

Several test cases are implemented in the main method of the QuadTreeConverter class.

8 Future work

This document describes the v1.0 of the hybrid specifications of InterCor. These specifications will be implemented in the participating countries, based on which international interoperability will be tested in a Hybrid TESTFEST. It is anticipated that an update of these specifications will be made based on the implementations in the participating countries, resulting in v1.1. This update will be the basis of the Hybrid TESTFEST. After the TESTFEST, an update will be made, resulting in v1.2. These updates will address solving ambiguities that might exist in these specifications, provide additional clarifications, and/or will address omissions that might be found during implementation and testing. No extensions or changes of the interface specifications will be added in v1.1 and v1.2.

At the same time, the development of v2 of the specifications will be started. In v2 the remaining InterCor services will be taken into account, and security and authentication based on the PKI used for ITS-G5 communication will be investigated. The latter will be done in cooperation with InterCor sub activity 2.1c, responsible for the PKI. Also for v2 of the specifications several iterations are foreseen, based on implementations and testing. V1.2 of the specifications will be fully integrated in v2. Overall, the updates to v1.x can be seen as technical improvements of the specifications within the current scope, and v2 as the version with the complete scope of the InterCor project.

 © InterCor Consortium

9 References

[1] Intercor, “M3 - Common set of upgraded specifications for ITS-G5 v1.1,” 2017.

[2] InterCor, “InterCor 2.1d - Service Descriptions,” 2017 (under development).

[3] amqp.org, “AMQP Advanced Message Queuing Protocol, Protocol Specification, version 0.9.1,” 2008.

[4] Project Talking Traffic, “RFP Talking Traffic 1.1_Bijlage 9 Latency Tabel_Beter Benutten_ 2016.07.01.pdf,” 2016.

[5] InterCor, “InterCor_A2.1_a_002 Use Case Comparision,” 2017.

[6] ISO, “EN ISO 17423:2017 Intelligent transport systems -- Cooperative systems -- Application requirements and objectives,” 2017.

 © InterCor Consortium

Annex A Country specific remarks

The sections below are **not part of the normative specifications**, but are added to provide insight in (national) aspects of the IF2 implementations that will be realised based on these specifications.

A.1 Implementation remarks from the UK related to IF2

As this is for the 1st version of the IF2 specification, the following requirements are for the UK TESTFEST and not necessarily for the Pilot. It should also be recognised that as the planning stages for the project is at a very early stage, they may well change.

The TESTFEST will endeavour to create an environment that can demonstrate in a live situation use case GLOSA, RWW and IVS over IF2 (cellular) and also services over ITS-G5. The UK project also can demonstrate a dynamic changeover of service suppliers over a specified geographic area and so simulate inter-country operability. The IF2 of each of the participating countries will need to implement the geographical filter (FILTER01) so as the correct service provider connected to that nation's datahub is implemented. As the vehicle connected to the aforementioned service provider is collecting the correct information from the correct nation's datahub for the configured geographic area. How this is achieved is yet to be defined at the moment it is described in a specification, this may not be via IF2. The UK project will not be able to simulate international roaming as we will be using the existing cellular network. We would also want to support the development of the final specification which will be implemented in the pilots.

 © InterCor Consortium

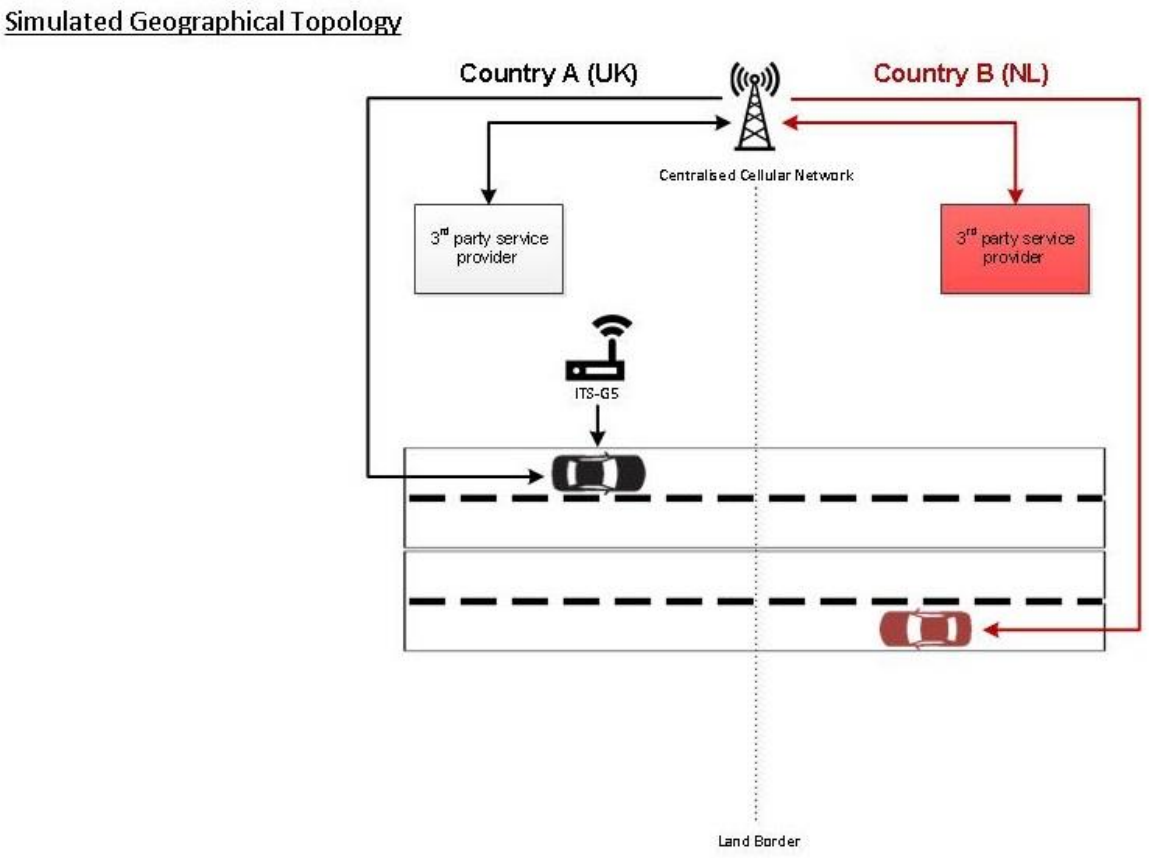


Figure 10: Simulated Geographical Topology.

The UK project does have a concern with the latencies and reliability involved with international roaming and it may become apparent that some services reliant on near real-time connectivity may not be suitable as the networks stand at the moment. It is recognised that improvements can be made with current network management and further work may need to be done to ensure the necessary connectivity.

 © InterCor Consortium

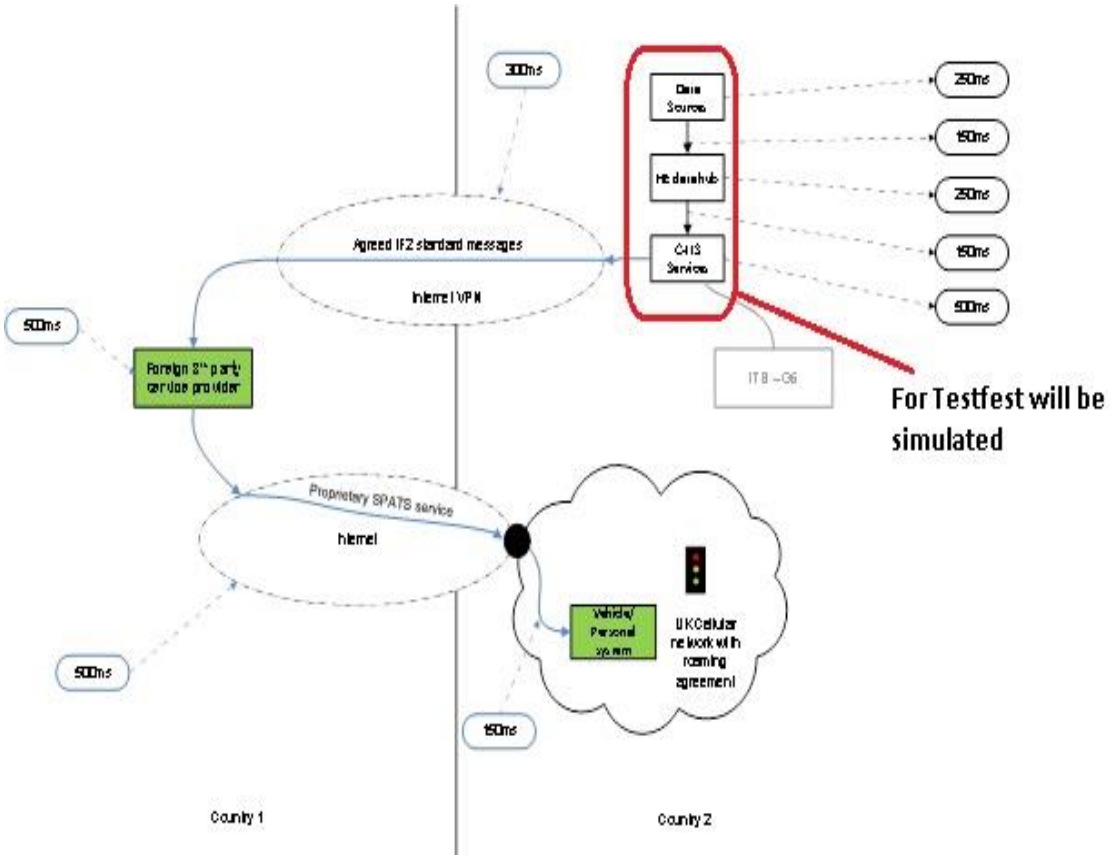


Figure 11: Information flow with typical delay from Data Service to Vehicle via a 3rd party service provider of C-ITS services.

The UK project will implement:

- Push service Only (SYS03)
- Service to support multiple connections (SYS08)
- Use cases GLOSA, IVS and RWW
- ASN1 support (MES01)
- Authentication security using Certificates, Address Protocol Access lists passwords only (SEC01) (Between SP and IF2 Only)

 © InterCor Consortium

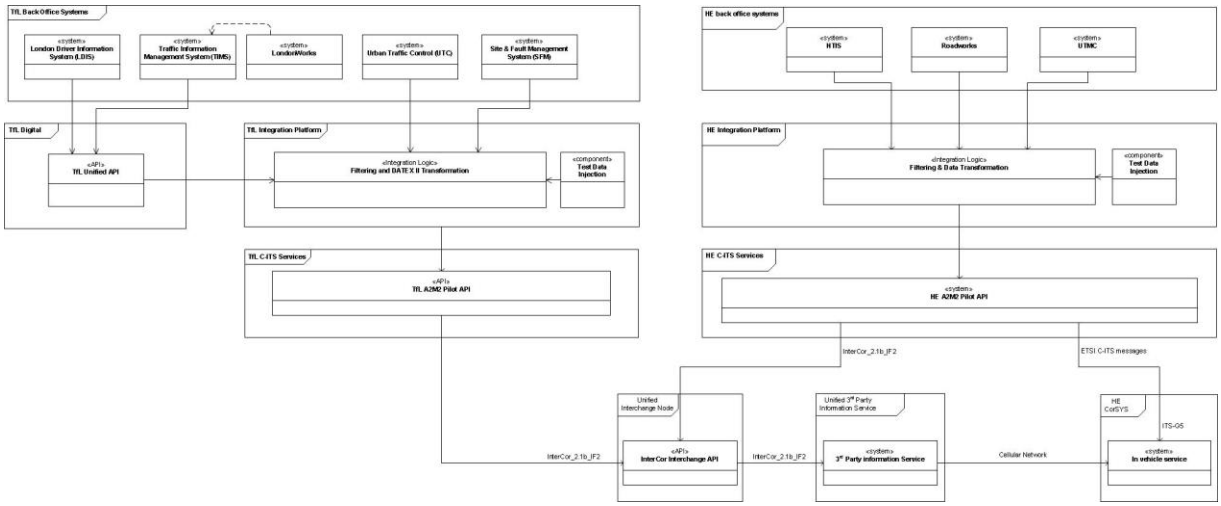


Figure 12: UK InterCor Architecture diagram.

A.2 Implementation remarks from The Netherlands related to IF2

In the Dutch Talking Traffic project, a GLOSA and RWW service[7] is being implemented via cellular communications, with three 'clusters': (1) Traffic Light Control (TLC) provider, (2) ITS Data Provider (DP), (3) ITS Service Provider (SP) -> Vehicle. The project is based on fixed roles related to the information exchange from TLC -> DP -> SP -> Vehicle (and vice versa). High level requirements for the end-to-end delays have been provided in the request for proposal, and in the technical specifications. These are summarized below in Table 5, and can serve as a basis for the end-to-end requirements for systems that implement IF2.

Table 5: End-to-end latency requirements priority and information of data streams involving IF2 (based on [4])

RFP v 1.1	Requirement	End-to-end latency (ms)
3a	Conditioned priority	
3a1	Conditioned priority for public transport, heavy trucks, group of vehicles	1500
3a2	Conditioned priority for heavy trucks,	1500

[7] The service definitions in Talking Traffic have more functions included than the InterCor definitions, what is reflected in a more extensive list of requirements and message types.

RFP v 1.1	Requirement	End-to-end latency (ms)
3a3	A platoon of vehicles approaching, if possible extend green for passing the crossing	1500
3a4	A group of bikes approaching, if possible extend green for passing the crossing	
3b	Absolute priority	
3b1	Emergency vehicle approaching the TLC is given absolute priority	1500
4	In-Car current information from TLC	
4.1	Informing a waiting or approaching vehicle about time to green for lane or direction of the vehicle. The application (OBU/device) is able to give a current speed advise based on the info.	1500
4.2	Informing a waiting or approaching vehicle about time to green for lane or direction of the vehicle. The application (OBU/device) is able to give a current speed advise based on the info within the boundaries of the speed limits at the location	1500 Excluding the calculation time of OBU/device

The Talking Traffic specifications, related to the TLC domain of a traffic manager, include the following parameters for message update frequencies, see Table 6. These parameters are meant for performance requirements of a system related to message handling of data streams. Information is exchanged via IF2 from TLC to DP and from DP to SP. Table 7 specifies the geographical filtering requirements for Talking Traffic on information exchange.

Note, that these are NOT part of the current IF2 specifications.

Table 6: Message frequency parameters specified in the Talking Traffic project. All requirements relate to data streams going through IF2.

System	Message	Frequency	Comment
GLOSA			
TLC -> Vehicle	SPAT	Max 10 per sec	

 © InterCor Consortium

System	Message	Frequency	Comment
TLC -> Vehicle	MAP	Max. 1 per 60 min	
TLC -> Vehicle	DENM	Max 1 per min	
TLC -> vehicle	SSM	Send triggered by SRM message	May be handled locally, not via IF2
Vehicle -> TLC	CAM and SRM	Send at the moment it is received	May be handled locally, not via IF2
Security	-	-	
RWW			
RSU -> vehicle	MAP	Max. 1 per 60 min or when updated	
RSU -> vehicle	DENM	Max. 1 per min	
RSU -> vehicle	IVI	Send triggered by message	May be handled locally, not via IF2

Table 7: Geographical filtering requirements from Talking Traffic.

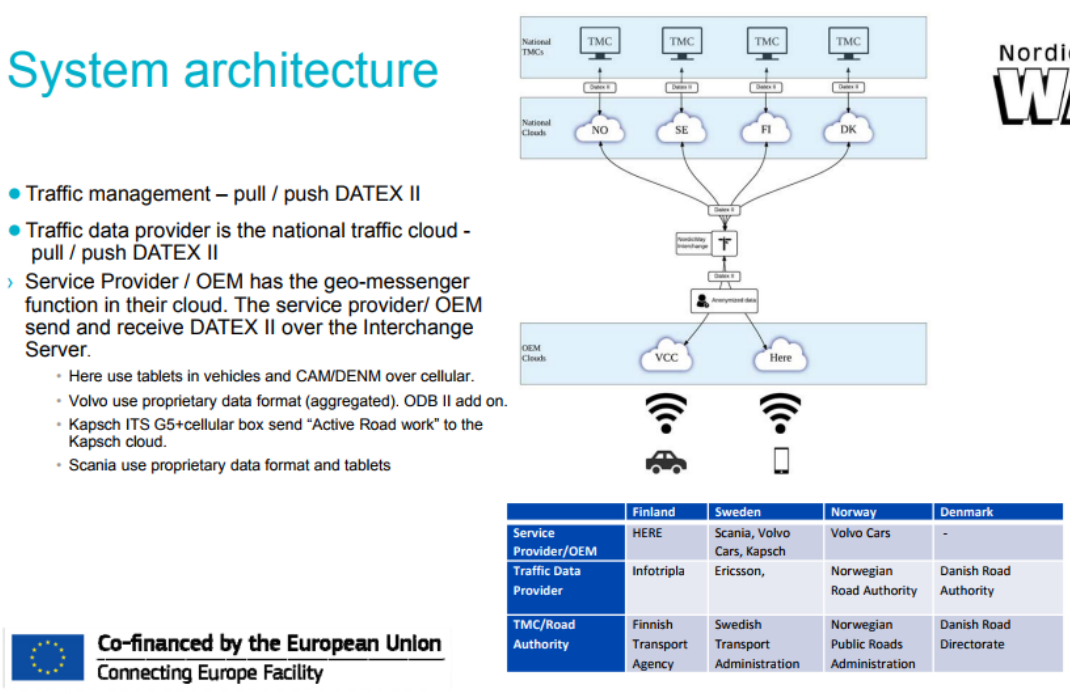
Message	Towards	Range
SPAT	Vehicle	Within 2 km of TLC Only if trajectory of vehicle over the junction is known or lane choice/direction indicator is on the OBU/Device calculates a speed advise Speed advise is only given for a junction if it is the first junction to cross which is signalised
MAP	Vehicle	On route (navigation): within 25 km When in the neighbourhood: 5 km
DENM	Vehicle	On route (navigation): always When in the neighbourhood: 25 km

 © InterCor Consortium

Message	Towards	Range
ASN.1 based high-level traffic information	Vehicle	Within a country and on a boundary around its trip (every minute)
SSM	Vehicle	Always (is on request based on the device and service)
SRM	TLC	Always (is on request based on the device and service)
CAM	TLC/RWW	Within a pre-defined vicinity of the TLC

 © InterCor Consortium

Annex B NordicWay architecture



	Finland	Sweden	Norway	Denmark
Service Provider/OEM	HERE	Scania, Volvo Cars, Kapsch	Volvo Cars	-
Traffic Data Provider	Infotripla	Ericsson,	Norwegian Road Authority	Danish Road Authority
TMC/Road Authority	Finnish Transport Agency	Swedish Transport Administration	Norwegian Public Roads Administration	Danish Road Directorate

Figure 13: NordicWay - system architecture

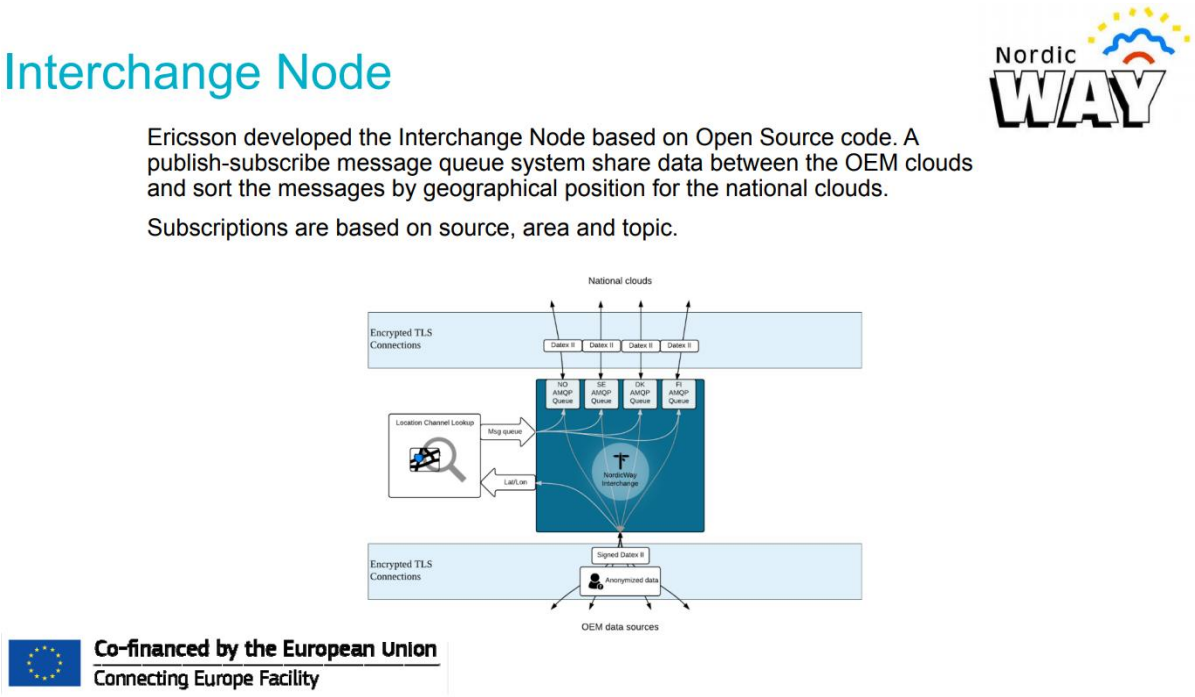
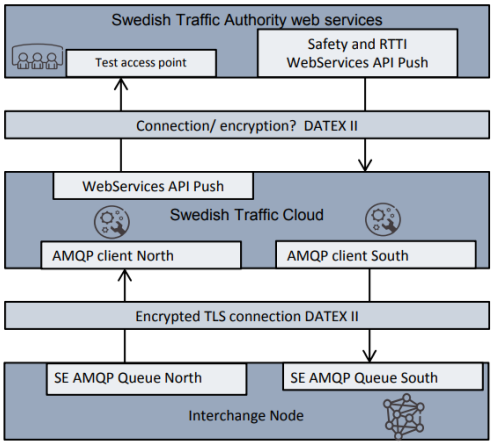


Figure 14: NordicWay – Interchange node

Swedish traffic cloud

- Ericsson developed the Swedish traffic cloud
  - Deliver data from vehicles to the Swedish Traffic Authority
  - Take data from the Swedish Traffic Authority (and other sources), sort and mark it, deliver to the Interchange server.
- All data is in DATEX II (with AMQP meta data header)



Co-financed by the European Union
Connecting Europe Facility

Figure 15: NordicWay – Swedich traffic cloud

 © InterCor Consortium