Cooperative Cities extend and validate mobility services

WP3

D3.1 – ITS system specification description and reference platform for validation

Version
2.0

Dissemination level

Public

Co-Cities is a Pilot Type B Project funded by the
European Commission, DG Information Society and Media
in the CIP-ICT-PSP-2010-4 Programme

European Commission
Information Society and Media

Contract Number:

270926

Acronym:

Co-Cities

Title:

Cooperative Cities extend and validate mobility services

Contractual date of delivery:

2012-03-07

Actual date of delivery:

2012-08-30

Main author(s) or editor(s):

Axel Burkert (PTV)

Other author(s):

Lena Reiser (ATE), Alexander Froetscher (ATE), Said Rahma Rodriguez (ATOS), Carlos Maestre Terol (ATOS), Roman Pickl (FLU), Mirjana Artukovic (FLU), Michele Masnata (SOF), Marco Garre (SOF), Tvrzský Tomáš (TMX)

Version History:

Version	Date	Main author(s)	Summary of changes
0.1	06.08.2011	PTV	Document setup
1.0		PTV	Draft 1.0
1.1		All	Update
1.2	15.12.2011	All	Update
1.3	13.2.2012	PTV	Document crash
1.4	8.3.2012	SOF	content update methods, models and core chapters
1.5	13.3.2012	PTV et al	Final Draft for Peer Review
1.6	14.6.2012	ATOS	Added reviewer comments from EU
1.7	05.07.2012	PTV	Update Draft
1.8	20.07.2012	PTV et al	Ready for internal review
1.9	23.08.2012	PTV	Consolidated input of internal review, ready for peer review
2.0	31.08.2012	PTV	Consolidated input after peer review, final version

List of the Co-Cities Project Partners

Partner no.	Partner name	Partner short name	Country
1	AustriaTech Gesellschaft des Bundes für technologiepolitische Maßnahmen GmbH	ATE	AT
2	Softeco Sismat S.P.A	SOF	IT
3	Telematix Software, a.s.	TMX	CZ
4	Fluidtime Data Services GmbH	FLU	AT
5	Brimatech Services GmbH	BRI	AT
6	Left intentionally blank		
7	The Regional Organiser of Prague Integrated Transport	PID	CZ
8	POLIS-Promotion of Operational Links with Integrated Services	POL	BE
9	Atos Origin Sociedad Anonima Espanola	ATO	ES
10	PTV Planung Transport Verkehr AG.	PTV	DE
11	Asociacion Cluster Del Transporte Y La Logistica De EUSKADI	MLC	ES
12	Regione Toscana	FIR	IT
13	Reading Borough Council	RED	UK
14	MemEx S.R.L.	MEM	IT

Table of Contents

List of Tables

List of Figures

1. Executive Summary

Co-Cities is set to define interfaces and methods to establish feedback channels from the end user to the RDSS, via the TISP, for all defined In-Time services. This means, that Co-Cities provides a detailed specification for the interface between RDSS and TISP – as this is the In-Time CAI – but does not prescribe any methodology for the TISP side for the following reason:

The service generation on side of the TISP and the connection between TISP and end user is usually proprietary. In fact these areas are harbouring some of the major distinguishing factors in terms of quality and service delivery which have a huge impact on end user satisfaction and thus success on the market. This means, that only principle statements on the service generation and recommendations on how to transfer the information from the end user to the TISPs´ systems are provided by Co-Cities.

For the very same reason, among others, D3.1 would not provide any prescribed methodologies on data handling and generation on side of the RDSS.

D3.1 is thus describing the extensions (marked in red) to the In-Time CAI (marked in green, below), but the data processing (including the handling of the feedback) within the realm of neither the RDSS nor the TISP is comprised. These systems are of very different technical and organisational nature and in most cases proprietary.

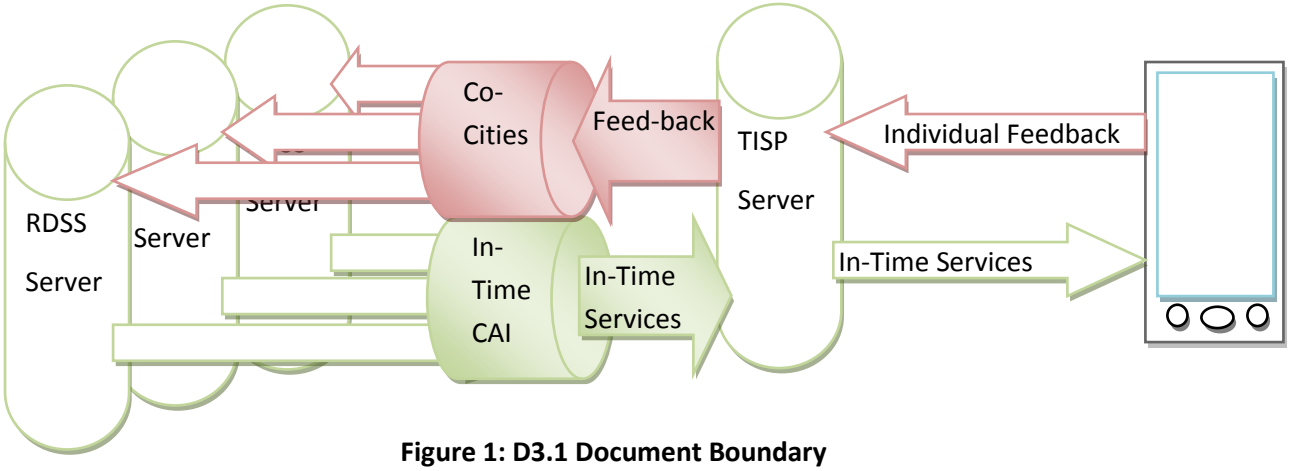


Figure 1: D3.1 Document Boundary

It should be noted, that the term "CAI" in In-Time as well as in Co-Cities is not only describing a standardised interface protocol but also comprises physical gateway servers.

The document at hand describes the data and service model of the Co-Cities CAI in chapter 5 providing details on the system boundary and general agreements used during the modelling process.

Chapter 6 holds the descriptions of the methodology put in place to extend the In-Time CAI, the feedback data definition based on the use cases devised in WP2, the relation of the feedback data

to the In-Time services, the standards used and the interface between handheld device and TISP server.

Within chapter 0, the feedbackservices are specified in detail depicting the foreseen types of services as well as the related technical specifications.

While chapter 8 holds the detailed data models, chapter 9 holds the service models for the feedback services.

Chapter 0 describes the data security aspects and chapter 0 describes the reference platform used for validation of the Co-Cities services detailing on objectives, target groups, testing of the service delivery chain at the CAI level, requirements towards the reference platform and its corresponding features as well as the specification of the reference platform.

Finally, the Annex holds a list of abbreviations as well as theXSDs of the Co-Cities CAI specification.

2. Introduction

2.1 Project summary

The currently existing bottleneck for the dynamic adaptation of traffic management measures according to policy goals is the information distribution to end users in urban areas and the adaptation of the information provided to the needs of the single user group. These two aspects are addressed by Cooperative Cities by providing one standard interface between city traffic management information and the Transport Information Service Providers, the In-Time common interface.

Secondly, the full availability of data enables an end-to-end testing and validation process for the single traffic information service in the cities and elaborates the future expansion steps for cities and service providers.

The objectives of the Cooperative Cities project are:

- To extend the numbers of cities which install the In-Time common interface and connect it to the traffic management centre with a regular feed of data and information.

- To add new service providers as users of these data and allow them to adapt their traffic information services to the available data and information in the participating cities.

- To develop a fast and reliable validation process for cooperative traffic information services with the use of a “reference platform” with a feedback loop to the cities on data quality and the respective user acceptance of the traffic information services.

- To extend the number of traffic information services and provide them in a more integrated and coherent way to the end user and

- To make these services more attractive and appealing to users, which is the basis for the future strength and viability of the business case for transport related personal information services.

The traffic and traveller information services of Cooperative Cities are developed in a partnership between cities, public authorities, transport operators and, which deliver high quality traffic information, regions on the one hand side and the Traffic Information Service Providers on the other hand side, which bring in their relations to the end users, customers and the capacity to extend the service concepts and enhance user acceptance.

The core information services of Cooperative Cities are the following mentioned below:

S1: Static and intermodal incar navigation in urban areas

S2: Parking information in urban areas

S3: Public transport advice

S4: Dynamic and intermodal in-car navigation in urban areas

S5: Walking and bicycle route advice

Table 1 shows the entire set of services as defined in In-Time:

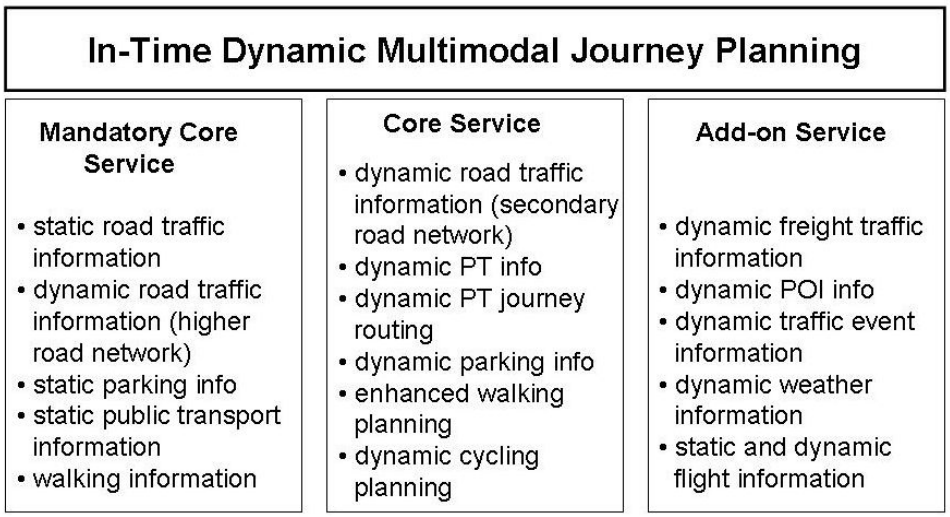


Figure 2: In-Time Services

These services are not new but based on the known needs of mobile users, travellers and proven service concepts. The real challenge for cooperative cities and service providers is the accuracy and quality of the services delivered together with an attractive delivery to the end user in an integrated and consumer friendly way. Because the traffic management centers of Cooperative Cities combine data and information from several sensors before generating the messages to the Traffic Information Service Providers, these services get more accurate and precise in terms of location and traffic impacts, are delivered faster to the traveller and are integrated in an attractive end user device with the option to directly give feedback or submit new traffic related events to the cities' traffic management centers.

With this establishment of an "feedback loop" from the users to the traffic management Cooperative Cities can react to changing traffic conditions and adapt their traffic management and control plans faster than before, communicate changes dynamically to travellers and people on the move and enhance mobility in urban areas.

2.2 Purpose of the Document

This document holds the description of the Co-Cities CAI and the reference platform as basis for the implementation in WP 4. It contains the data and service model for the CAI and the specification of the reference platform used during performance testing and evaluation.

This document shall serve as a basis for the implementation of the regional CAIs in the test sites.

2.3 WP 3 introduction and goals

WP 3 has the goal to specify the extension to the In-Time CAI (see document In-Time D 2.3.1), the end user feedback to the TISP and to define the methods how the implemented interfaces can be tested.

The specification will be used by RDSS and the TISP to implement specific services within the project. To foster sustainability, WP3 also needs to define a backwards compatible method on how to extend the interface in the future in conformity with implemented In-Time CAIs.

2.4 Overview about Co-Cities feedback process

Within WP2000, an abundance of use cases was defined which ranged from quality feedback (e.g. rating scheme utilizing 5 stars for quality assessment or commenting on a piece of information received from a service) to submitting entirely new data sets via the Co-Cities feedback channel. In principle, a given Co-Cities feedback is either linked to a service delivery, e.g. a route, in which case the feedback information also carries the identifier of the related service delivery, or it is not linked to a prior delivered service item which means that a new data set is submitted by the end user, e.g. a new congestion event on the motorway, which is relevant for the service generation on RDSS/TISP side.

For the first ones, the content delivered via a service can (as mentioned above) be ranked in general or updated, modified or ranked in detail. The latter ones comprise all kinds of content ranging from new POIs to new traffic information or schedule data.

According to the In-Time principle, which is the sound basis for Co-Cities, the RDSS is predominantly responsible for the service delivery on regional level and the redistribution of the feedback to the relevant entities and systems. Of course, the TISP can either use services (eg. the intermodal routing service) of an RDSS or build end user services on basis of data and information received by a RDSS. This means, that a TISP can provide an intermodal routing service for a region either by receiving it from the RDSS or by using motorised individual routes as well as public transport and park and ride information, to name just one example.

The TISP, on the other hand, is directly linked to the end user, his client, to which he delivers services on basis of the In-Time services provided by the RDSS.

The principle schema for a feedback generated on basis of an In-Time service item delivery is depicted in the following Figure 3 (numbers in brackets indicate the succeeding steps):

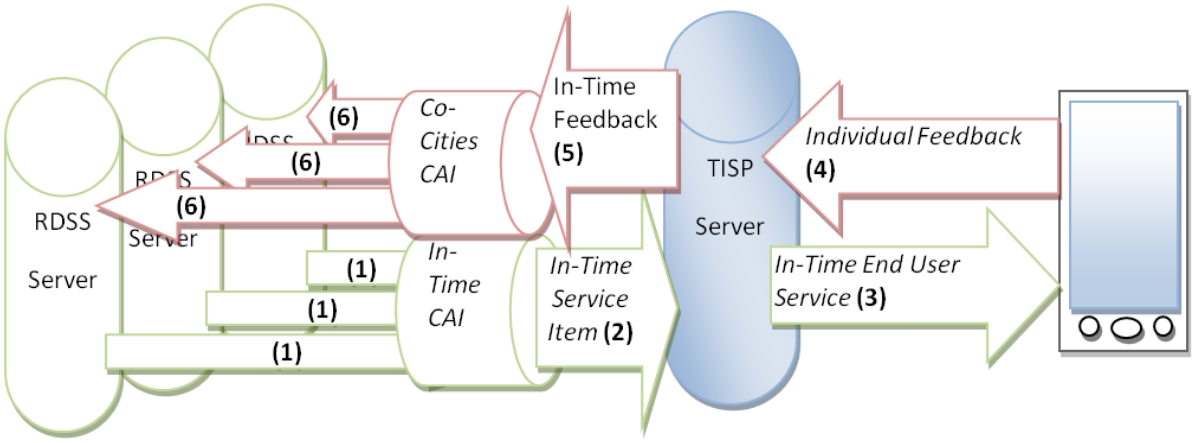


Figure 3: High level view on feedback process

Note: computations in the CAI to assemble the RDSS server deliveries into an In-Time service as well as the computations on TISP server and handheld side are not depicted although executed

The following steps are depicted in the Figure 3 above:

Contained within In-Time delivery Chain:

1. Upon request of the TISP the CAI requests data and information sets from applicable local systems which deliver those to the CAI server.
2. The CAI system assembles/computes the data and information sets into an In-Time service and delivers the resulting service item to the TISP.
3. The TISP delivers the In-Time service or a TISP service built upon the In-Time service item content to the end user.

Contained within Co-Cities delivery Chain:

4. Upon submission of a feedback by the end user the feedback is transferred to the TISP system.
5. The TISP system assembles/computes the feedback into the Co-Cities feedback services and provides it to the CAI.
6. The RDSS, responsible for the CAI and fully aware of the local systems and organisational ancillary conditions, allocates the feedback to the correct local recipients.

The steps 2 and 5 are described within the In-Time documentation, these are the data and exchange models defined for the communication between RDSS and TISP. The steps 1 and 2, however, were not described in In-Time as they are “proprietary” meaning not only, that the variety of data exchange methods in a given region may be extensive but also that the organisational background, so the question of which institution to approach for which feedback data, is potentially unique. Step 3 was not described by In-Time either, for the reasoning on

proprietary data delivery methods mentioned above. Step 4 and its ancillary processes are described, as far as possible, within this document. The document D3.1 holds the description of step 5, the feedback delivery through the Co-Cities CAI.

The following two typical example scenarios illustrate the chain of eventsThe first one describes a use case, where a user requests a route which is calculated by the RDSS. As the destination is incorrect (eg. a POI with a similar name but different cathegory)a subsequent user feedback is generated, the TISP forwards the user feedback to the RDSS. The RDSS must now allocate the correct recipient in his local network, with the challenge to first identify the problem – the example below assumes that it was a clearly identifiable geocoding problem, maybe a wrong coding of the clear name address and allocates the response to the respective recipient. This step, however, may pose a major obstacle in the processing chain as understanding the clear reason for suboptimal information or service delivered is sometimes very difficult. It will thus not always be clear which institutions are concerned by which feedbacks, if services provided by the RDSS involve several institutions within the process chain.

The second example describes a user generated congestion event.

First example:

The End-User requests a route within a region. This request if forwarded (by the TISP) to the CAI and thus the RDSS collects the required information and data sets from the local systems (1) and provides it to the TISP via the CAI (2). The TISP sends the route to the end user (3). The end user finds that the destination he is arriving at is not the desired one and provides his feedback to the TISP (4). The TISP formats the feedback according to the Co-Cities specifications and sends it to the CAI (5). The RDSS distributes the feedback to the relevant local system which is responsible for the Geo-coding of the destination (6).

Second example:

An end user drives on the motorway and encounteres a congestion event which is not yet mentioned by the TISP´s service delivered to him. Hence he inaugurates a new feedback message containing the location as well as other details of the congestion event and sends it to the TISP (4). The TISP reformats the new data set from his proprietary format into the Co-Cities format and provides it to the CAI (5). Again, the RDSS operating the CAI forwards the new item to the relevant local server/service.

3. In-Time as Basis for Co-Cities

The project In-Time defined a range of interfaces for different traffic data and information services.

The actors are regional providers of data and services (RDSS) which may be a city or a regional traffic information centre and, as the other group, service providers (TISP) which deliver services to the end user via e.g. Mobile devices. They utilize mostly proprietary technology for the link between their centre and the handheld devices. In-Time covered the definition and implementation of the so called *Commonly Agreed Interface* or *CAI* which allows for data and information provision from the RDSS to the TISP (the feedback from the TISP to the RDSS is subject of Co-Cities). The priniciple idea is to provide any information available in a given region to a TISP so the TISP can either use the information directly and/or utilise it for the provision of intermodal routing services.

The In-Time services comprise dynamic (individual and public) traffic and transport information inlcuding weather and parking information as well as fully fledged intermodal routes. Within In-Time 17 services were defined to be offered by the RDSS to the TISP via the commonly agreed interface:

- Dynamic Road information on higher network
- Dynamic Road information on secondary network
- Static parking information
- Dynamic parking information
- Static Public Transport information
- Dynamic Public Transport Information
- Dynamic Public Transport Journey routing
- Dynamic POI information
- Dynamic traffic Event information
- Dynamic Weather Information
- Static and dynamic flight information
- Static walking information
- Enhanced walking planning
- Dynamic cycling planning
- Dynamic freight traffic information
- Comparative Dynamic Multi Modal Journey Planning
- Static Road Traffic Information

In practise, only a part of those services will be available in a given region, so the mix of services per region might be slightly different (one might offer dynamic intermodal routes while another region will provide dynamic road traffic data and time tables). This would mean, that a TISP might rely on the intermodal routes delivered by a given RDSS while, in another region, the TISP will compute the intermodal routes based on time tables and dynamic road traffic data.

3.1 The Commonly Agreed Interface (CAI) of In-Time

The commonly agreed interface is the layer which enables Service providers to access and 'understand' services and data offerings originally provided in a heterogeneous format, in a 'common language'. This scenario fits in a ***service orientated architecture (SOA)*** where RDSSs and TISPs are nodes inter-connected through the Commonly Agreed In-Time Interface (C.A.I).

SOA is an architectural style of building software applications that promotes loose coupling of components so that developers can reuse them or build complex services by aggregating simple services taken from a distributed network of suppliers. SOA's characteristics are as follows:

- Services are software components that have published contracts/interfaces; these contracts are platform-, language-, and operating-system-independent. XML and the Simple Object Access Protocol (SOAP) are the enabling technologies for SOA, since they're platform-independent standards.
- Customers can dynamically discover services supplied by third parties using directories (e. g., with UDDI).
- Services are interoperable, and they are designed to be used by third parties within the organisation as well; thus SOA is not just an architecture of services seen from a technology perspective, but the policies, practices, and frameworks by which it is ensured that the right services are being provided and used.
- The basic building block of SOA is the service. A service is a self-contained software module that performs a predetermined task, eg., "verify a customer's credit history". Services are software components that don't require the involvement of developers using a specific underlying technology.

In the simplest scenario, as illustrated in the picture below, the SOA approach is composed of:

- a “service provider” (the business logic implementation) and a software module exporting the service through the network (the “service”);
- a customer accessing the remote service as a “client”;
- an optional service broker or directory where services are published by suppliers and where customers can browse through them.

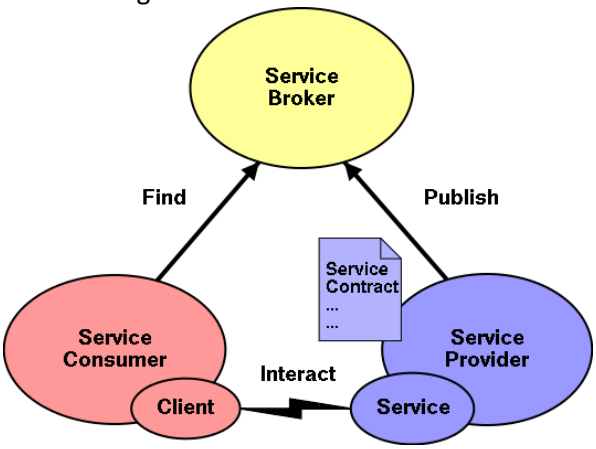


Figure 4: Minimal Service-Oriented Architecture

The aforementioned notions of a SOA are specifically addressed for the context of In-Time by the eMOTION deliverable D6 “eMOTION System – Technical Specification” whereas a SOA view more orientated to the so called ‘value chain’ of actors of In-Time. In the value chain, RDSSs get contents from local Content providers and delivers data and services to the TISPs (and then to the Final User) through the Commonly Agreed Interface, which is the general principle In-Time is based on. The following figure illustrates the general architecture of In-Time which supports this value chain.

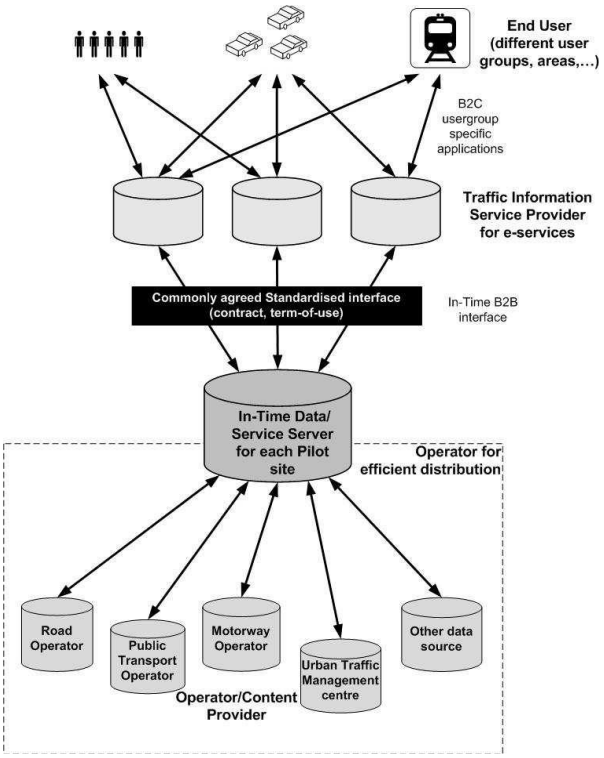


Figure 5: In-Time Architecture

The "Commonly agreed interface" then provides a standard entry-point, for services and data, for the TISPs (Travel Information Service Providers) to each single Pilot Sites RDSS (Regional Data Service Server).

3.2 Scope and Services of In-Time

The scope of In-Time (which is very much related to the concept of the Commonly Agreed Interface) covers the data exchange between the RDSSs and TISPs. They should be seen as 'black boxes' as their internal structure won't change but they will adopt some 'out of the box' adapters in order to implement the features and functionalities of the Commonly Agreed Interface.

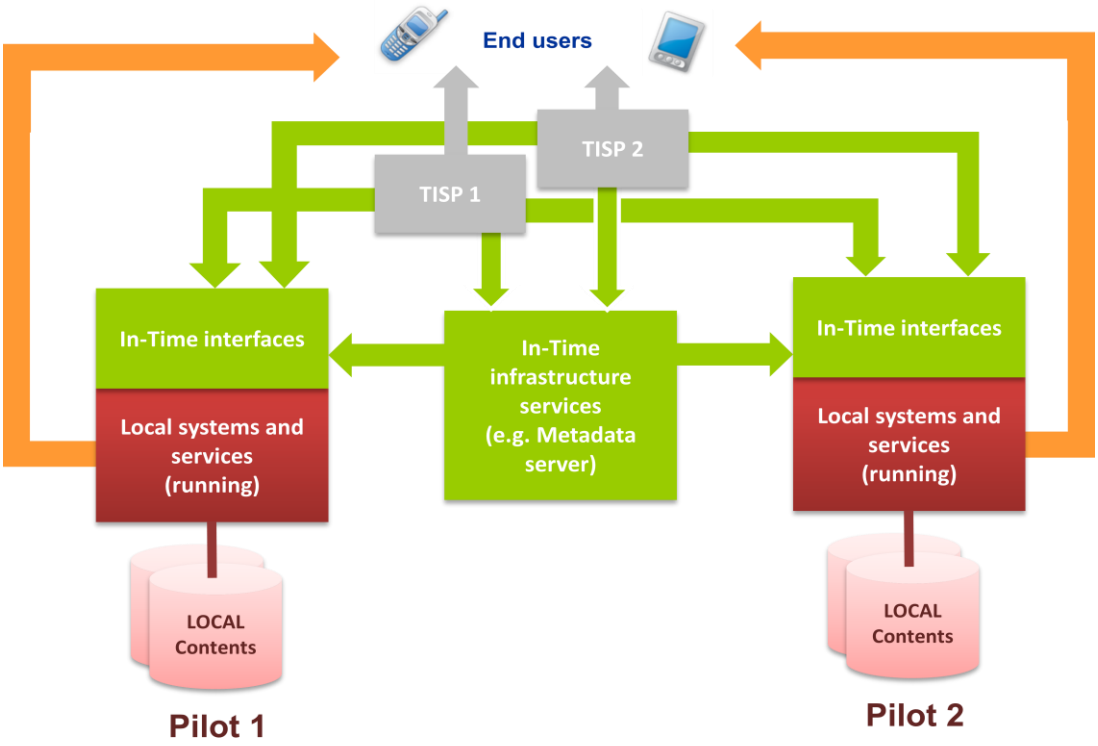


Figure 6: Scope of In-Time for existing Systems

In terms of data and service provision a very straight forward architecture of the interface can be proposed consisting of five main B2B services:

Routing Service: This service provides one or more recommended routes from a starting point to a destination in the traffic network.

Location service: This service relates search strings, names, coordinates or locations on the traffic network to each other and provides additional information to relevant locations.

Message Service: This service provides the traffic messages for the different modes of transport. The messages can be filtered by various criteria.

Map Service: This service provides pixel maps; the maps can be displayed to the end user as a background image. The maps can also provide additional information to the end user. This service could also provide map layers for the data services that are provided at pilot sites.

Meta Service: A generally called “Meta Service” macro category comprehends all the services that will not be exposed to TISPs, but are necessary for the In-Time Architecture to run effectively (e.g. Registry Services, AGORA C encoder/decoder, etc.).

These macro categories (except for the “Meta Service” category which is a particular one) are the aggregation of two or more base services. Each one of these atomic services provides a more specific function in the real time travel and traffic information world.

In In-Time a total of 17 atomic Services is identified. These services are categorized as:

- **Mandatory Core Services:** represent the minimum amount of functionalities that the RDSS of each pilot site should at least provide;
- **Core Services:** cover (together with the Mandatory Core Services) the core functionalities of the In-Time architecture;
- **Add on Services:** cover aspects of the RTTI (Real Time Travel & Traffic Information) that are not essential for the In-Time systems.

The complete list of the services includes:

1. **Static Road Traffic Information (Mandatory):** provides static information (static information like route, distance, specific route characteristics (road type, one ways, restrictions, road junctions);
2. **Dynamic Road Information (Mandatory):** provides dynamic information about current traffic conditions. The service can be provided as pre-trip and on-trip information service to plan a route or to select a destination depending on the current or future traffic conditions;
3. **Static Parking Information (Mandatory):** provides static information like information about parking facilities;
4. **Static Public Information (Mandatory):** provides static information (time table, stops positions, stop related time tables) about all PT modes (Bus, Tram, Metro etc.);
5. **Walking Information (Mandatory):** provides information (static information like map, POI);
6. **Dynamic Road Traffic Routing Information (Core):** provides dynamic information about current conditions for specific connection from origin to destination (and via station) for route planning or turn-by-turn-navigation;
7. **Dynamic Public Transport Information (Core):** provides dynamic information about Public Transport for a specific stop point, service, line, time period or date;
8. **Dynamic Public Transport Journey Routing (Core):** provides dynamic information for specific connection from origin to destination (and via station);
9. **Dynamic Parking Information (Core):** provides dynamic information for parking neighbourhood (occupancy, vacancy, etc.);
10. **Dynamic Walking Planning (Core):** provides dynamic information for walking planning;
11. **Dynamic Cycling Planning (Core):** provides dynamic information for specific connection from origin to destination;

12. **Dynamic Freight Traffic Information (Add on):** provides dynamic information for specialised traffic and transport information like waiting time at border crossings, ferry time tables or weather information for freight traffic;

13. **Dynamic POI Information (Add on):** provides dynamic information for a corridor along a route or for an area or near an address with thematic filter;

14. **Dynamic Traffic Event Information (Add on):** provides dynamic information like information about temporary parking facilities, public transport lines for temporary rerouting in the neighbourhood of the event venue;

15. **Dynamic Weather Information (Add on):** provides dynamic weather information for specific road, route or administrative area with filter for specific message type and specific validity period;

16. **Static and dynamic flight information (Add on):** Provide static and dynamic information on flights, e.g.: companies, destinations, time schedules, prognosis of real-time departures/arrivals;

17. **Comparative pre-trip Dynamic Multi Modal Journey Planning (Add on or Core):** provides dynamic information to allow the end-user a comparison between the different transport modes to generate an ideal (fastest / shortest / cheapest) travel route;

These services are considered during the design of the Commonly Agreed Interface.

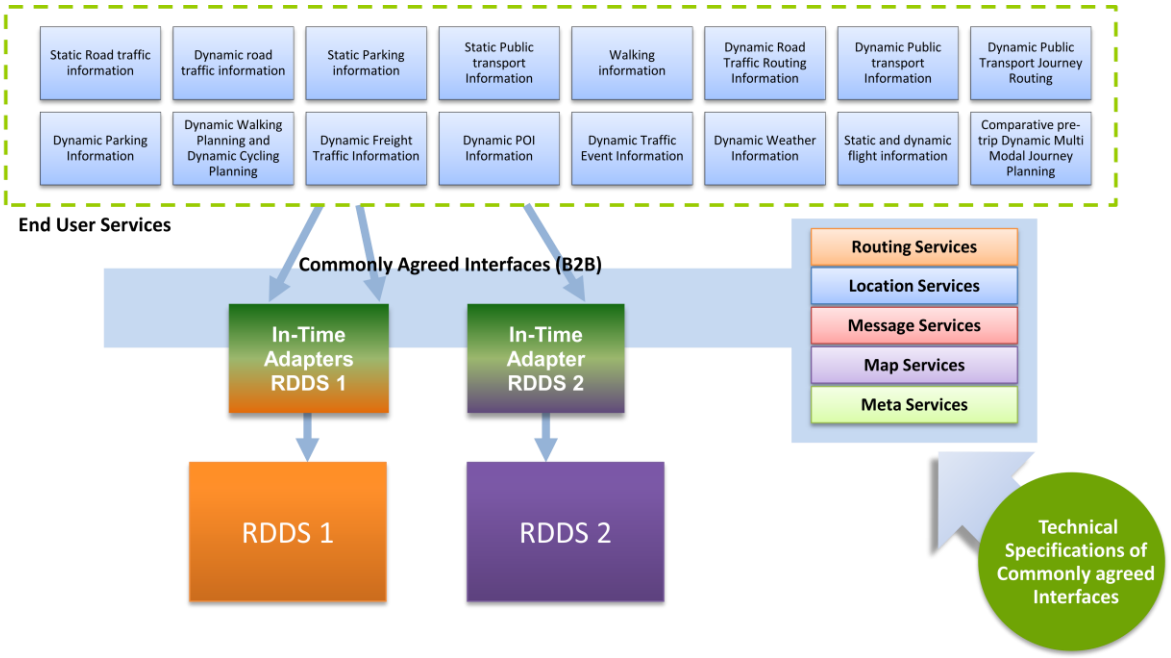


Figure 7: Services of In-Time

It should be noted, that the two separate In-Time services “Dynamic walking planning” and “dynamic cycling planning” are summarised under one box (carrying both designations) in the figure above. Thus, the figure presents all 17 In-Time services.

3.3 Standards used in In-Time

The conceptual reference data model at the basis of the In-Time service oriented architecture is defined selecting a number of international and European ITS standards harmonised into a single, comprehensive and coherent data model. The most relevant standards involved in the definition of the In-Time conceptual data model originate from the eMOTION project (which was the predecessor project to In-Time) and include:

- DATEX 2 for individual traffic and a general situation message model
- Transmodel for public transport base information
- SIRI to describe public transport schedule information
- IFOPT describing fixed transport infrastructures resources and objects, [CEN-IFOPT, 2007] and
- TPEG for descriptive location referencing, road traffic messages, public transport information messages and parking facilities; [ISO 18234-5-6-7, 2006], [ISO 24530-2-3-4-5, 2006]).

Currently many new standard are coming up in the ITS world. Some of them can be based, in turn, on other existing standard. This is the case of Network Exchange (NeTEx) which is a pre-CEN/Technical Standard under development and still in a draft version when this document was written. The NeTEx specification is based on Transmodel, IFOPT and SIRI which have been already integrated in the In-Time/eMotion data model. Consequently NeTEx is already supported, to some extent, by the existing Co-Cities schema. Similar operations of selection and optimization of base standards have been carried out in the first stages of the process of design of the eMOTION model.

Further information can be drawn from the In-Time and eMotion documentation (www.in-time-project.eu and www.emotion-project.eu).

4. Co-Cities Use Cases

The use cases in Co-Cities are based on the services defined by the In-Time project and are described in detail in WP2´s deliverable D2.1.

The descriptions hold

- Use case general description
- Concerned processes
- Concerned entities
- Interdependencies among Use Cases

Co-Cities defined three main data collection services based on feedback use cases related to In-Time services. The following figure describes the feedback services on high level.

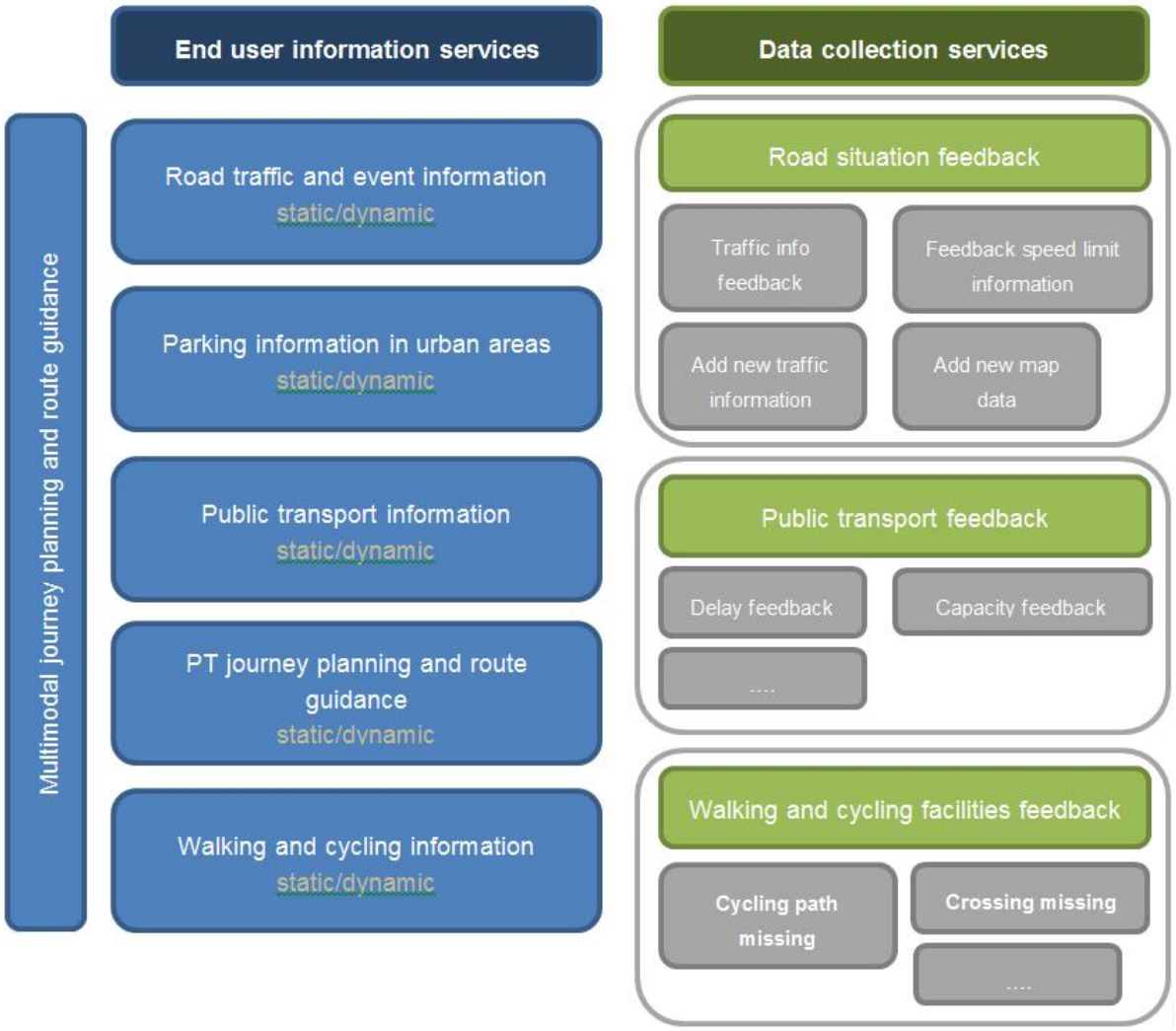


Figure 8: Overview about Use Cases

End user services in this concept would be services which have the primary target to provide information TO the user. But of course most of the end user services will also deliver some feedback information within the context of the end user service. In the best cases the request for feedback is triggered automatically.

Examples how to trigger the request for feedback:

- The end of service is detected via the position
- A deviation from the proposed route is detected via the position
- The estimated time of arrival has been reached

Data collection services on the other hand have the primary and only aim to receive information FROM the user. That means the user will actively start the Data collection service to provide information to the TISP.

5. Data and Service Model

5.1 Co-Cities System Boundary and Limitations

For a defined number of the In-Time services, Co-Cities has defined user feedback mechanisms and related interfaces based on the use cases the partners came up with and those which are validated in the test sites.

Co-Cities is defining a clear methodology how to model feedback data for each In-Time service. The set of interfaces defined by Co-Cities is thus, by nature, extendable in terms of In-Time services and data covered. As the interfaces between TISP and end user handheld device are mostly proprietary, Co-Cities makes suggestions on the design of those interfaces to reflect the data structures of the RDSS/TISP interface.

Also not within the system boundaries is the data processing within the TISPs and RDSS. This is a core requirement (as it was already in In-Time) to enable applicability in the existing economical environments.

Finally, due to budget constraints, only those parts of the Co-Cities CAI are modelled in detail, which are really implemented and validated at anytest site.

5.2 General Agreements on Modelling

Co-Cities will amend the In-Time service model if required by the use cases defined within Co-Cities.

An example for this case is the roadside parking application within the parking service feedback. Up to now, In-Time does not provide any service stating the approximate filling grade of parking spaces along a specific road.

Also, the feedback on receipt of the In-Time services is modelled as closely as possible to the existing In-Time model. This means, that existing modelling architectures (egg. for georeferencing, traffic messages etc.) is re-used within the Co-Cities architecture.

Co-Cities will amend the In-Time service model if required by the use cases defined within Co-Cities.

An example for this case is the roadside parking application within the parking service feedback. Up to now, In-Time does not provide any service stating the approximate filling grade of parking spaces along a specific road.

Also, the feedback on receipt of the In-Time services is modelled as closely as possible to the existing In-Time model. This means, that existing modelling architectures (egg. for georeferencing, traffic messages etc.) is re-used within the Co-Cities architecture.

6. Methodology

The Co-Cities project in principle foresees one or more feedback service for each In-Time services. However, for some services this might not be applicable or no use case can be seen at the current point of time. Hence Co-Cities based the definitions on the use cases defined in the frame of WP2 but considered the requirement of full flexibility to add any services required in the future.

As already described in former chapters, Co-Cities describes a common methodology how to design and implement Co-Cities compliant interfaces between a RDSS and a TISP.

Also, Co-Cities defines the interfaces for the use cases to be validated in the project's test sites in detail as they will be implemented and operational during the demonstration phase.

As the Co-Cities system boundaries relate to those of In-Time which defined a set of service interfaces between the RDSS and TISP and explicitly does not those between TISP and enduser (handheld device). The latter are mostly proprietary and it does not make sense to standardise the delivery of information in this direction. Co-Cities acknowledges these ancillary conditions but recommends a specific standard (and thus methodology) for the feedback provision from TISP to RDSS as well. However, compliance to Co-Cities results does not require to follow these recommendations, much in contrary to the RDSS-TISP interface definitions.

6.1 Common Methodology for the extension of the CAI

The project deliverable D2.1 introduces the basic working and design principles of the Co-Cities architecture using the In-Time CAI and its extension.

The Co-Cities infrastructure largely inherits the architectural design of In-Time which is, in turn, based on the eMOTION specification of a system operating in a *Single Information Space*, where all data is known to have commonly-defined semantics and formats.

The system interoperability is achieved on the basis of ISO/TC 211 and OGC (Open Geospatial Consortium) standards and the Technical Specification, first developed in eMOTION, follows the Reference Model for Open, Distributed Processing, RM-ODP [ISO 10746-1, 1998] with the provision of:

- a data model, metadata information model and styling and symbolisation for visualisation *[Information Viewpoint]*
- service interfaces based on a distributed and Internet-based system architecture *[Computational Viewpoint]*
- specifications for communication and interaction of components and other engineering

issues concerning distribution *[Engineering Viewpoint]*

The main pillars of the technical specification are the **data model**, which defines how the common information is structured (e.g. it defines the data structure to describe a traffic message, a parking place etc.) and the **service model** which defines the services through which the information is exchanged between local sites and TISPs.

The methodology applied in the creation of the eMOTION Technical Specification shall be considered as the reference one for any subsequent amendment in In-Time and Co-Cities. Specifically, the amendment of the existing Data and Service model is necessary for the modification of the In-Time infrastructure in order to make it suitable for running the necessary Co-Cities feedback-related services.

In a simplified view the methodology for extending the CAI includes the following activities:

1. **Informal definition of the necessary features**
2. **Conceptual modelling in UML**
3. **Derivation of GML and WSDL definitions from UML**

The following pictures illustrates the three basic steps above.

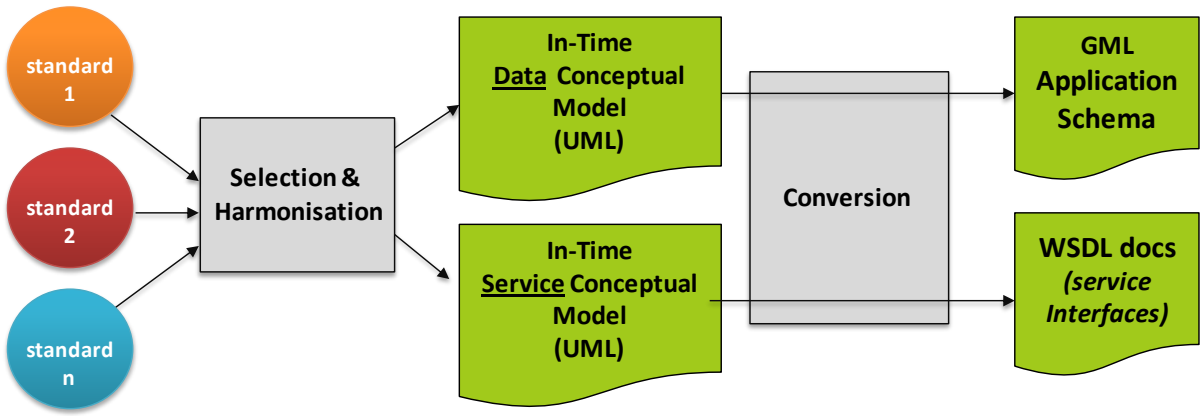


Figure 9: Illustration of steps for the production of the specification

The "Feedback data interface specification" results as a usable system specification which eventually drives the concrete production of the Co-Cities feedback-specific part of the common interface.

It is important to outline that the *feedback* data/service conceptual model, specifically developed for Co-Cities is defined as a *backward-compatible extension* of the existing eMOTION/In-Time

data/service model. It's not intended to be a substitute of the In-Time model but it's the necessary *additional* specification enabling the new feedback data exchange requirements.

6.2 Informal feedback data definition from use cases.

The **Informal definition of features**, which is the first activity of the three-steps process introduced before is strictly tied with the **use case definition** as this informally (yet fully) describes, for all domain of interests, the set of features necessary to ensure that the end user feedback is correctly sent back to the local content/service provider via the Co-Cities common interface.

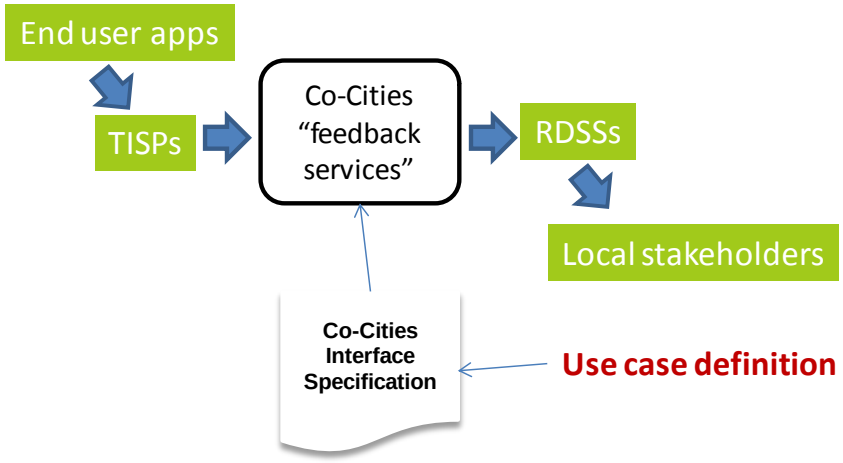


Figure 10: Use Case definition as driver for Co-Cities interface specification

The identification of common feedback data can lead to the definition of the (common) set of features which will enrich the In-Time Common Specification with the feedback loop elements.

In the following paragraphs an introduction on the main methodological elements is given.

Conceptual modelling of data

"Modelling" can be intended as the activity which lead to the definition of the Co-Cities UML conceptual model. The model and application schema definition is based on the ISO 19100 series of standards and is developed following the methodology for the development of an application schema provided in ISO 19109.

For each traffic & transport domain considered, in eMOTION the most appropriate reference international standards has been identified and considered as a starting point to build the model. Usually, no or very few additional features are added to the reference model. Additional useful features or elements may be applied in situations where single domains comprises more sub-domains (example: the "Public Transport" domain includes sub-domains like schedules, services, operators etc.). In these situations different reference standards can be considered and chosen for

the different sub-domains in order to make the best possible choice and to have every aspect covered. Example: Reference standards for eMOTION/In-Time Public Transport model are: Transmodel/TransXChange (Service Description), SIRI (Schedules), TPEG-PTI (Service Information Messages), IFOPT (Equipment and fixed infrastructure).

Whenever different standards are chosen for different aspects of a certain domain, they are integrated in the final data/service model and "harmonized" to avoid inconsistency, broken connections or references, ambiguity etc. In UML modelling this can be done by introducing links between certain objects of the model (for examples definition of associations, superclasses, sub-classes etc. between objects). This is done trying to minimize the impact on the final model.

An example of the last point is depicted in the following figure: In this example an association between a "Stop Place" object and a "Parking" object, not necessarily present in original reference standards, has been defined (as "non-mandatory" – see 0..* UML notation to ensure flexibility). The reason of such modifications/enrichments comes again from point 1. of the specification process (informal requirements). In the previous example an informal requirement of a parking place associated with a stop place may have been defined and this is reflected (more formally) in the above UML association (which is non-mandatory in order to have a minor impact on the model and on the existing reference standards).

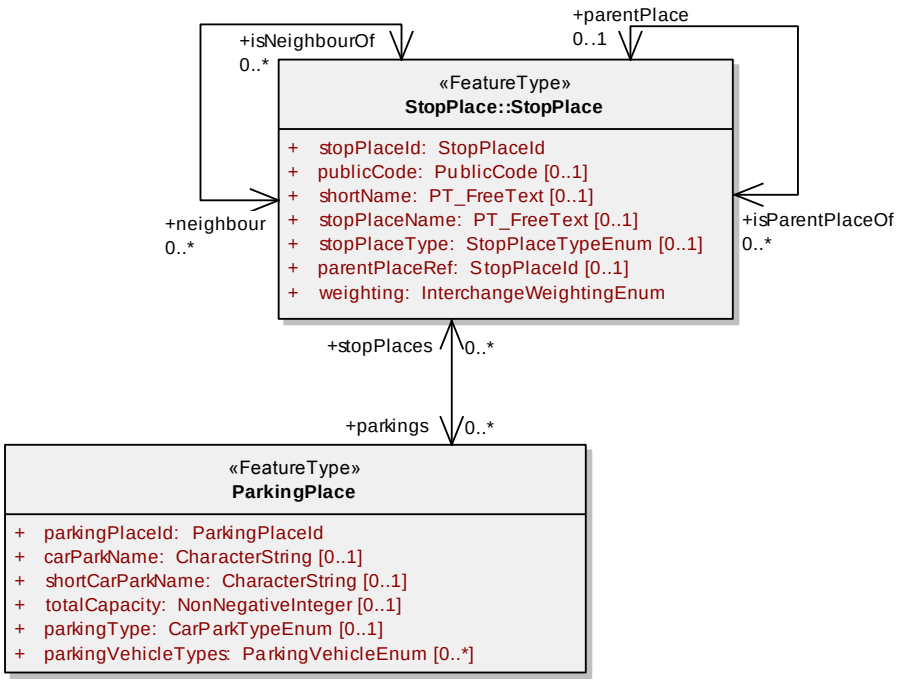


Figure 11: Example of extension of model

More examples of harmonization and extension operations are described in In-Time deliverable D3.2.1.

Feedback datasets identification

As an intermediate step between step 1 and 2 of the modelling process, a more strict identification and classification of all necessary additional "feedback" common information has been done in Co-Cities with the objective of:

- Classify and assign feedback datasets to the domain of interests
- Avoid inconsistency
- Have a real common definition of feedback datasets

For this purpose an excel file has been prepared to collect the following information for each domain of interest:

- Type Of Feedback (example: the correctness of the provided data as part of a quality feedback)
- Description of Data
- Recipient (TISP/RDSS/BOTH) – who is dealing with the feedback data.

The whole set of common feedback datasets, collected in the excel file, is identified from the use case descriptions. In other words, the synthesis made from the use cases resulted in a common identification and definition of feedback datasets which will be reflected in the formal specification necessary to build the Co-Cities common interface.

Automatic derivation of GML Application Schema

An application Schema of Geography Markup Language (GML) is defined as exchange format for data in eMOTION/In-Time and Co-Cities. Using a GML Application Schema enables the OGC Web Feature Service to be used as an all-purpose generic data interface. In eMOTION the GML application schema has been derived from the conceptual Data Model by means of a standardized process documented in ISO 19136 (Appendix D and E) in an automatic way.

In In-Time and Co-Cities this conversion involves a limited number of features and for this reason is carried out manually starting from the existing GML specification.

Service definition

Like for the Data model, the Service definition make use as much as possible of existing standards especially from *OGC* (data provision and mapping)*, OASIS and W3C.* Other *conceptual service models* are defined and modelled in UML where no existing standard-based definition can be found for them. Using the *MDA approach,* service definitions in *WSDL* are obtained. SOAP is used also in order to seamlessly apply security information features (e.g. WS-Security).

Network independence

Multiple location referencing using a large variety of models (such as geometry, geographical identifiers, Alert-C, TPEG-loc, AGORA-C, street addresses, etc.) is the chosen solution for achieving a "network independence" between traffic and mobility data sources which usually refer to various network definitions of local importance.

References

Details on this methodology can be found on the two project deliverables **eMOTION deliverable D6** and the **In-Time deliverable D3.2.1**.

6.3 Interface between Handheld Device and TISP

The interfaces between handheld devices and TISP systems are generally proprietary. One of the reasons is, that a TISP streamlines the data volume to be transmitted to handheld devices according to his specific business cases).

Therefore, Co-Cities does not standardise this link nor the detailed processes on side of the TISP or handheld device. However, there is a semi-standard currently in the procedure of standardisation at CEN, which could be suitable to model the required data objects.

6.4 Data and allocation to In-Time Services

Concerning the data delivery from the TISP to the RDSS we need to think about, where the data foreseen to be generated in the use cases are split up according to the In-Time services definition as well as according to local data holding architectures (e.g. weather data might be delivered by several services as per their local coverage). This can be done, in principle, by the TISP, the RDSS or "the interface", e.g. by foreseeing a "plug" for an external filtering (e.g. mapmatching) and feedback to the interface, which distributes the data accordingly.

In-Time was inaugurated in order to eliminate the data access problem for the TISP, namely having to access a multiple number of heterogeneous organisations to acquire the necessary data coverage for a value added service.

Following this thought, the TISP should not be forced to having to know anything about any local architectures beyond the In-Time interface. Thus, the TISP should not filter services as he could only distinguish between e.g. road classes (on example of In-Time Services 1 and 2), but not between the correct recipients in a specific region. The latter step would have to be performed by the RDSS anyways and should be combined with the implementation of the filter concerning to which In-Time Service a specific data set originating from Co-Cities shall be allocated. This procedure eliminates possible quality sinks by communication lacks between RDSS and TISP, as RDSS physically implements the system and TISP is using standardised features.

6.5 Standards used in Co-Cities

The standards also used as basis for the In-Time model are reused as Co-Cities fully builds upon In-Time. Chapter 3.3 provides a detailed insight into this field.

6.6 Compliance with the ITS Directive

The approach chosen by In-Time and Co-Cities is in line with the ITS Action Plan as it provides the technical means for the interoperability of different sources of information (as In-Time ensured this for the provision of services by the RDSS). The goal of Co-Cities, to standardise user feedback formats and protocols between a regional data provider and the service provider connected to the end user, is thus fully in line with the apporach defined by the action plan. This, of course, fosters the open data access, In-Time for the public/regional and Co-Cities for the business side, which is the core of the directive.

Major data content called for by the directive was already considered in In-Time (which provides an extensable model which can accomodate new data – e.g. flight – in the future), thus Co-Cities, building on In-Time, also considers them also providing an extensible model opening the system for potenial feedback to be amended in the future (as e.g. refined floating car data, feedback on flight schedules etc).

Also, data wich are not readily avialble to but worthy for the RDSS are now provided which has the potential to help to close gaps of the data coverage for reasons of the extensibility of the model. Of course, also the management of data access is an integral part of In-Time as well of Co-Cities.

Also, the ITS action plan identifies the risk of competition between the different business models. Wtih Co-Cities and In-Time, this risk can be somewhat alleviated as a true cooperation between RDSS (e.g. an authority) and a TISP can be implemented thus providing alternative business cases (give and take) possbily not requiring the involvement of monetary compensation.

Co-Cities and In-Time thus are fully in line with the aspects of the ITS Directive covered by those projects and have the potential to actively contribute to the success of the directive.

7. Feedback Services

7.1 General Architecture and Types of services

To provide the Co-Cities back loop feedback mechanism a number of services usable to send feedback information was planned according to the working principles of the Co-Cities architecture:

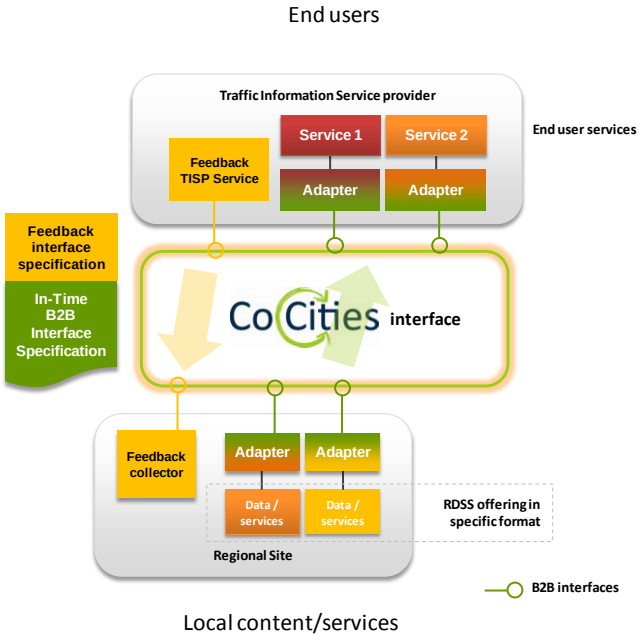


Figure 12: General Co-Cities architectural schema

These services (also called 'feedback services') are defined on the basis of the WP2 outcomes:

- One "feedback service" is used to express the Overall Quality of the end user service provided by the TISP
- More "feedback services" are used to express the Quality of existing data service
- More "feedback services" are used to submit new data

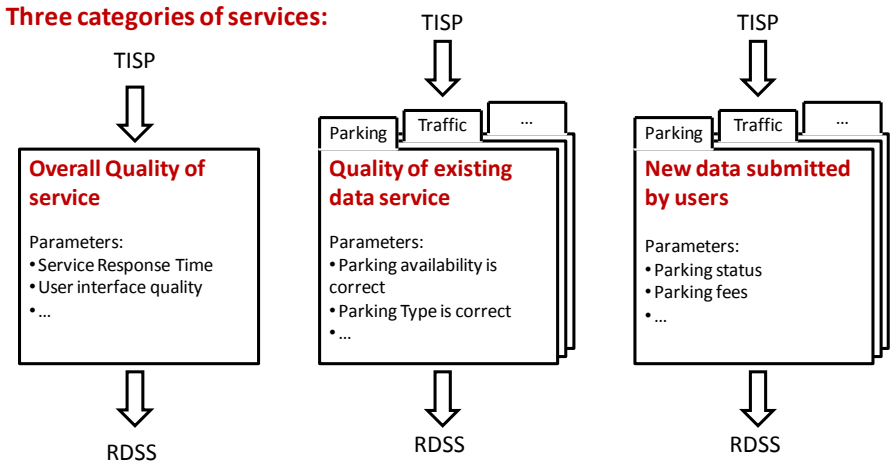


Figure 13: Types of 'feedback services'

The feedback service is defined to be a web service "running" (having a URL) on RDSS side. A TISP is able to access it by a PUSH mechanism.

7.2 Technical specification

The Co-Cities Data Model constitutes the central part of the semantics of Co-Cities and represents the central supporting pillar of the extended CAI, as it was for In-Time project. It ensures a data supply "all of a piece" in a "common language" enabling the actors of the Co-Cities chain (RDSSs and TISPs) to understand and interpret Co-Cities data.

The Co-Cities Data Model is encoded and documented in Unified Modelling Language (UML).

Following the procedures and methodology adopted in others European Projects (In-Time and eMotion), the exchange format for Co-Cities data is defined by an Application Schema of Geography Markup Language (GML) and specified as an UML Schema which has been created using the Enterprise Architect1 Case Tool. The binary model repository or the equivalent XMI exports are available and are part of the Co-Cities Technical Specification.

Using the In-Time Application Schema as starting point, two new Application Schemas were introduced in the Model (Package: Co-Cities).

[1] *By Sparx Systems, http://www.sparxsystems.com.au/*

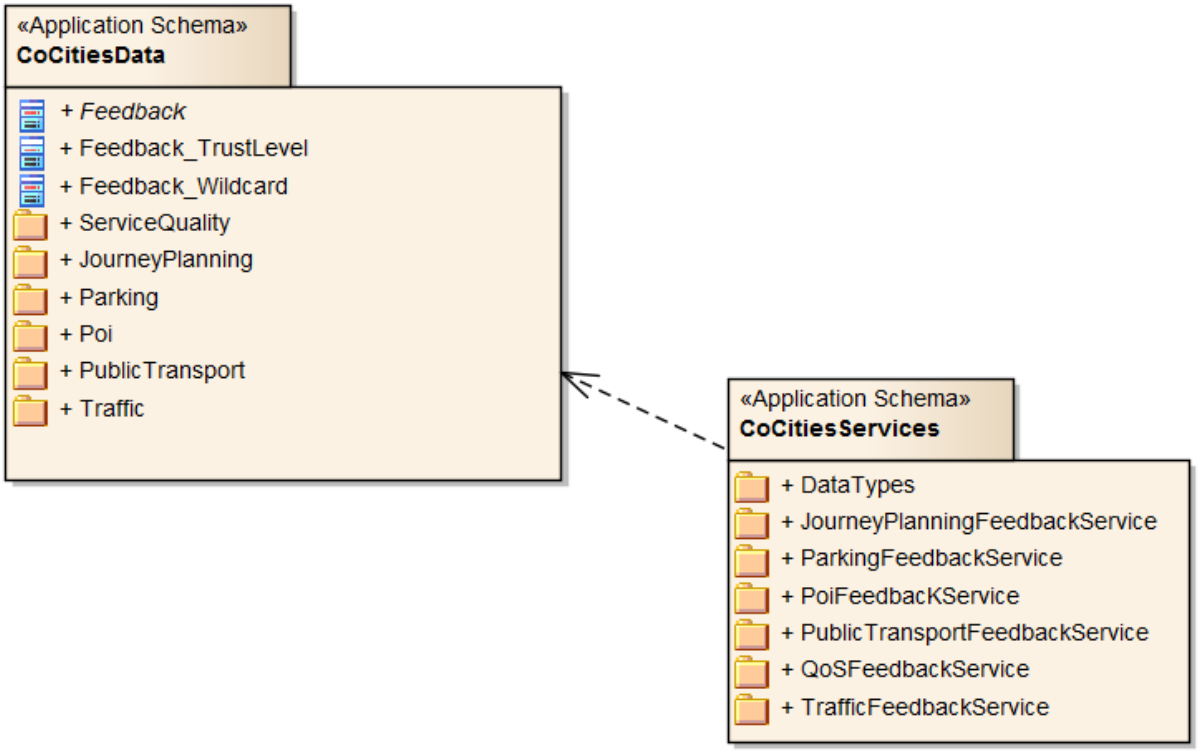


Figure 14: The Co-Cities UML Schema Package

The *Co-Cities Service*s Schema includes all the interfaces of the feedback services reusing the data types and domains defined in the Co-Cities Data Schema (Figure 14).

Also, the *Co-Cities Data* Schema comprehends the definitions of all the new types introduced to support the feedback loop functions. Features and data types are organized using the domains identified in the Co-Cities Deliverable 2.1 – Chapter 6 and briefly described below. These are, in brief:

- Road Traffic
- Parking
- Point of Interest
- Public Transport
- Multimodal Journey Planning

8. Data Model

The *Data Model* Schema (Package: *Co-Cities Data*) contains the new Features and Data types defined to support the information coming from the users and transmitted from the TISPs to the RDSSs.
The schema is structured following the categorization given in Deliverable D2.1 – Chapter 7.

In more detail an Application Schema is defined for each feedback domain (Public Transport, Point Of *Interest, Parking, Traffic Information, Multimodal Journey Planning*). These Packages import the In-Time Application Schema (Package: *EMotion Data*[2]) together with the common features and data types defined directly in the Co-Cities Data Package (Figure 15).
A *Service Quality* Application Schema is also included in the model, defining the QoS feedback common to each domain.

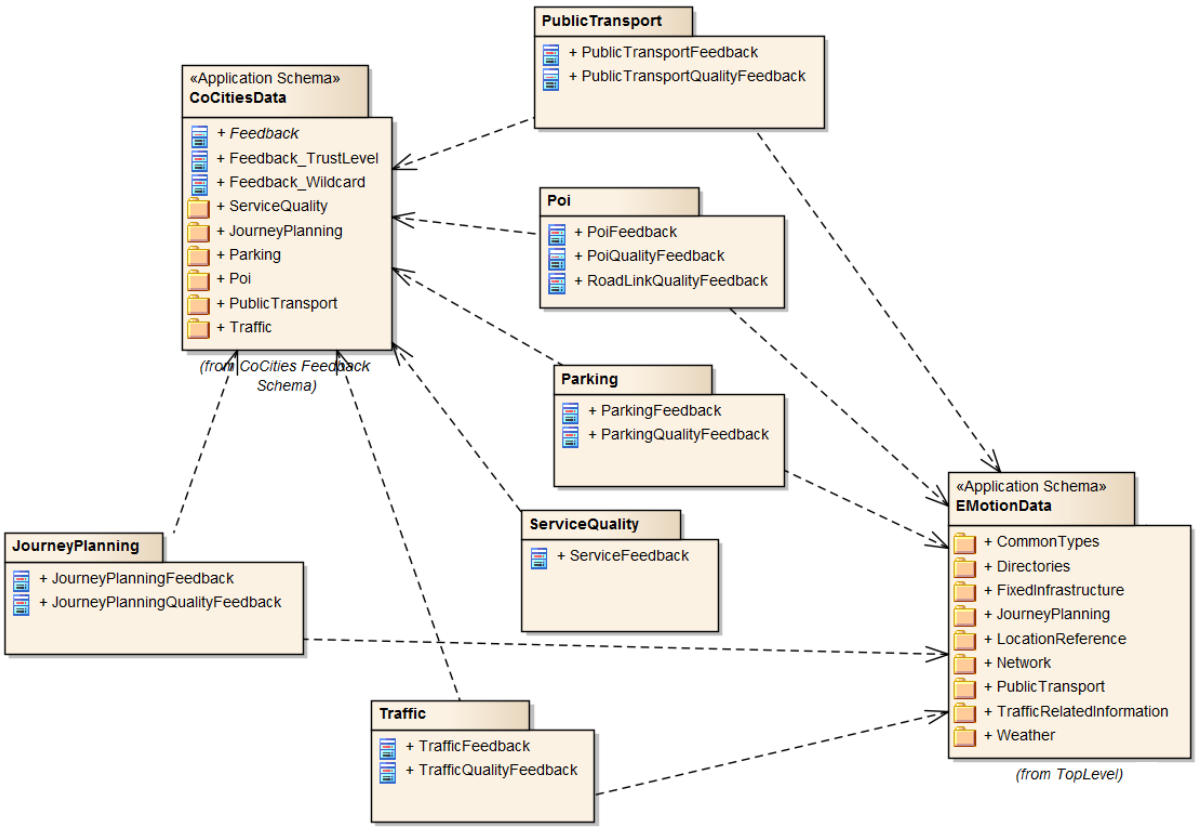


Figure 15: Co-Cities Data Schema Package

[2]The In-Time project extended the eMotion (FP6 European Project) data and service model maintaining unaltered the names of the packages.

8.1 Common Types

While designing the data model for the different domains, it was clear that every type of feedback features holds some common information that could be used to organize and harmonize the schema (Figure 16).

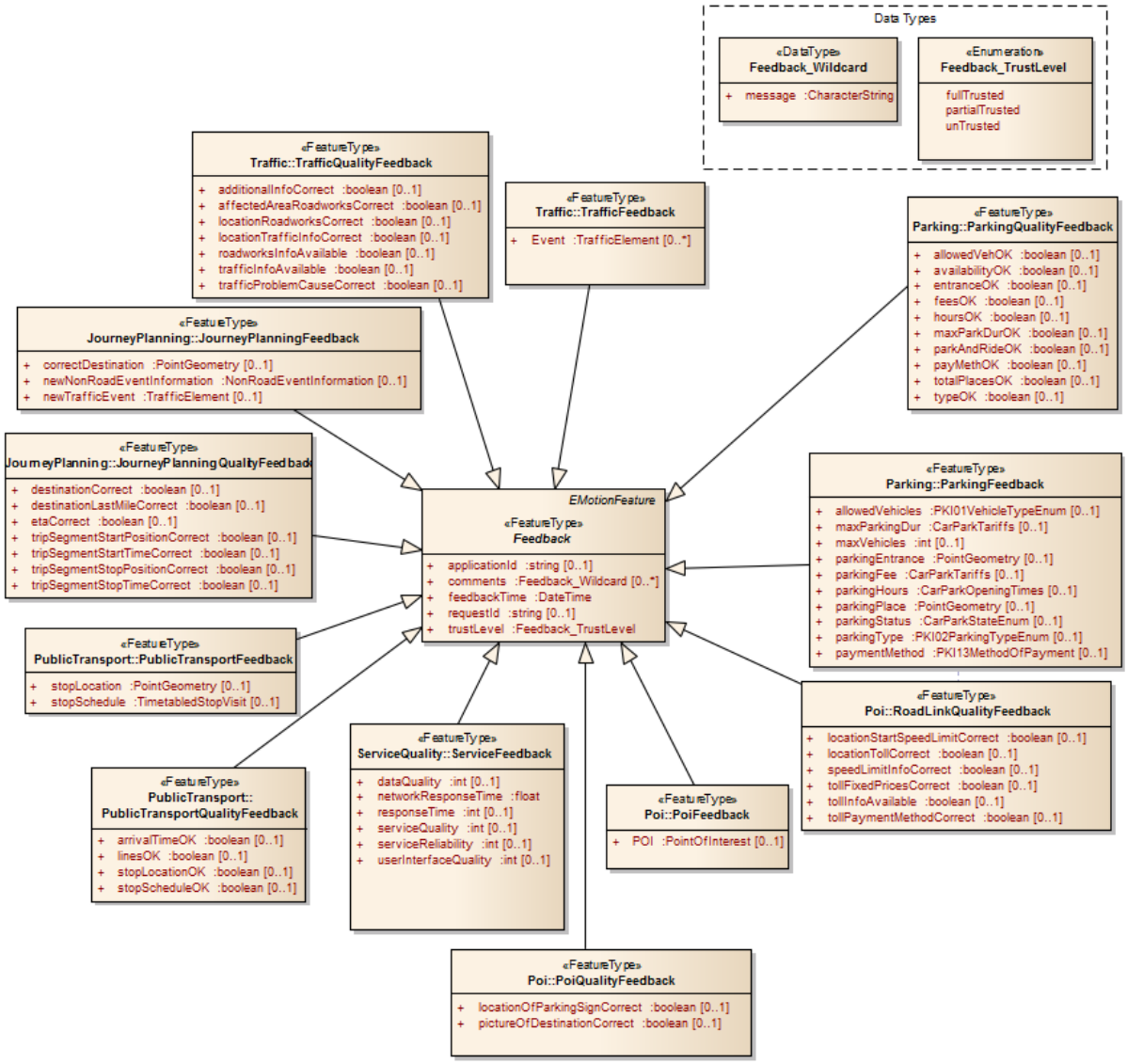


Figure 16: Co-Cities Feedback Data Types

A **Feedback** abstract class, derived from *EMotionFeature*[3], is used as base definition for all the type of feedback feature types.

[3] A detailed documentation of the eMotion Data Model is available in Deliverable D6 Appendix 2 of the eMotion Project.

This base abstract class (Figure 17) includes all the feedback information that are common to all the domains.

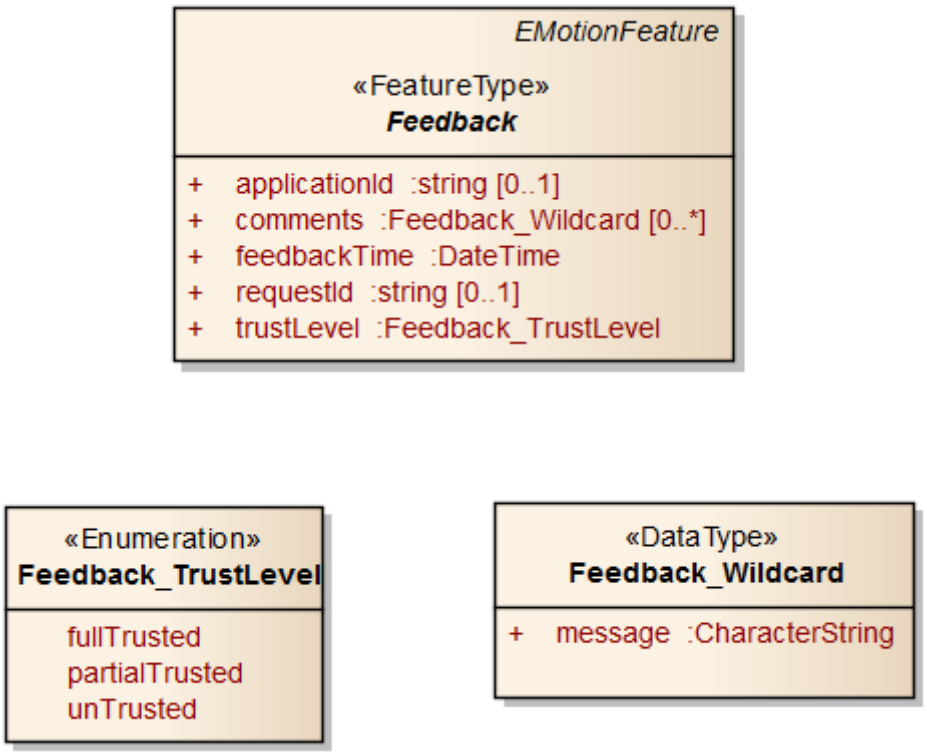


Figure 17: Co-Cities Data Common Types

The *feedbackTime* and *trustLevel* are the only mandatory attributes of the class which can carry information also about the unique identifier of the application, *applicationId*, generating the feedback (for trust management policy on RDSS side) a general free text comment, *comments*, and the original unique request identifier on which the feedback is referring to.

8.2 Service Quality Feedback Data

In Co-Cities enables the end user to provide a "generic" feedback directly related to the perceived quality of service.
This kind of information is modelled in the schema using a **ServiceFeedback**, deriving from **Feedback** base abstract class (Figure 18).

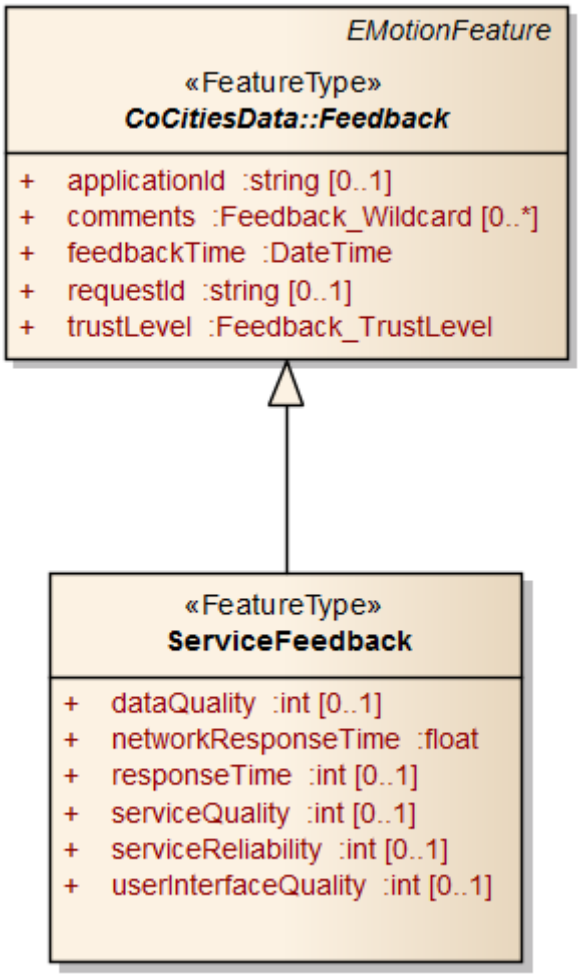


Figure 18: QoS Feedback

RDSSs can receive an overview including the real response time automatically measured at TISP level, carried in the *networkResponseTime* attribute, and the end user rating for service and data quality, service speed, reliability.

8.3 Parking Information Feedback Data

The Parking Information Feedback model is composed by the following types (Figure 19):

- *ParkingFeedback*
- *ParkingQualityFeedback*

ParkingQualityFeedback comprises feedback data related to the quality of the parking information received by the user upon a service call to RDSS. Quality information, in particular correctness using Boolean values, can be send for the most common chunk of data related to parking: type of allowed vehicles (*allowedVehOK* attribute), availability of free parking spaces (*availabilityOK* attribute), position of parking entrances (*entranceOK*), fees (*feesOK*), opening hours information (*hoursOK*), maximum allowed parking duration (*maxParkDurOK*), availability of Park&Ride

(*parkAndRideOK*), accepted methods of payment (*payMethOK*), total capacity of parking spaces (*totalPlacesOK*) and the type of parking (*typeOK*).

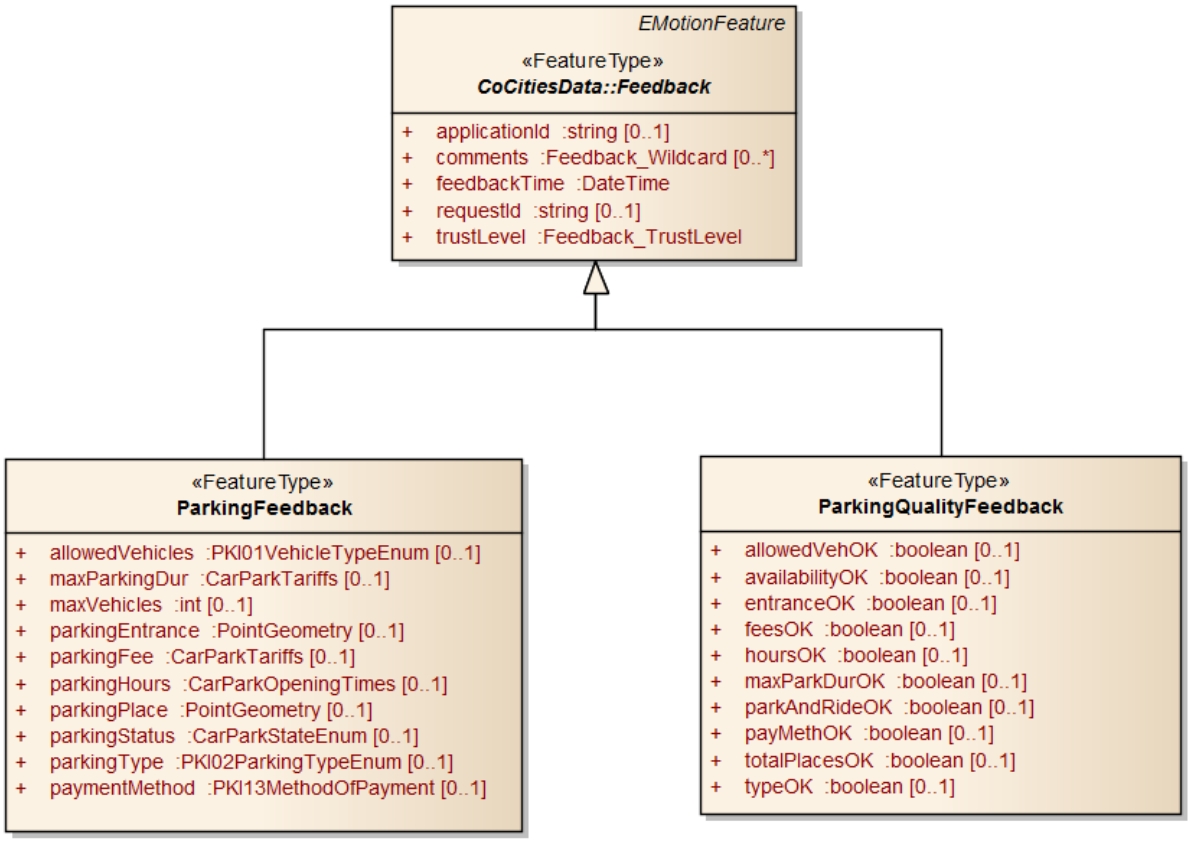


Figure 19: Parking Information Feedback

Together with quality feedback, the TISP, through the *ParkingFeedback* Feature Type, can send to the RDSS new data related to parking domain, collected directly from the end user. This kind of data can be related to correct/complete information for parking already present in the RDSS data source, but also a completely new bunch of information not yet included at the RDSS.

All the types of the attributes of the *ParkingFeedback* are imported by the **eMotion/In-Time** Data Model which is, in turn, derived from *IFO-PT*, *UTMC Car Park* and *Datex II* standards.

8.4 Traffic Information Feedback Data

The Traffic Information Feedback model is composed by the following types (Figure 20):

- *TrafficFeedback*
- *TrafficQualityFeedback*

TrafficQualityFeedback comprises feedback data related to the quality of the traffic information received by the user upon a service call to RDSS. Quality information, in particular correctness using Boolean values, can be send for the most common chunk of data related to traffic event: traffic event location (*locationRoadworksCorrect*, *locationTrafficInfoCorrect*), area affected by Road Works

(*affectedAreaRoadworksCorrect*), detailed information about the event (*additionalInfoCorrect*, *roadworksInfoAvailable*, *trafficInfoAvailable*, *trafficProblemCauseCorrect*).

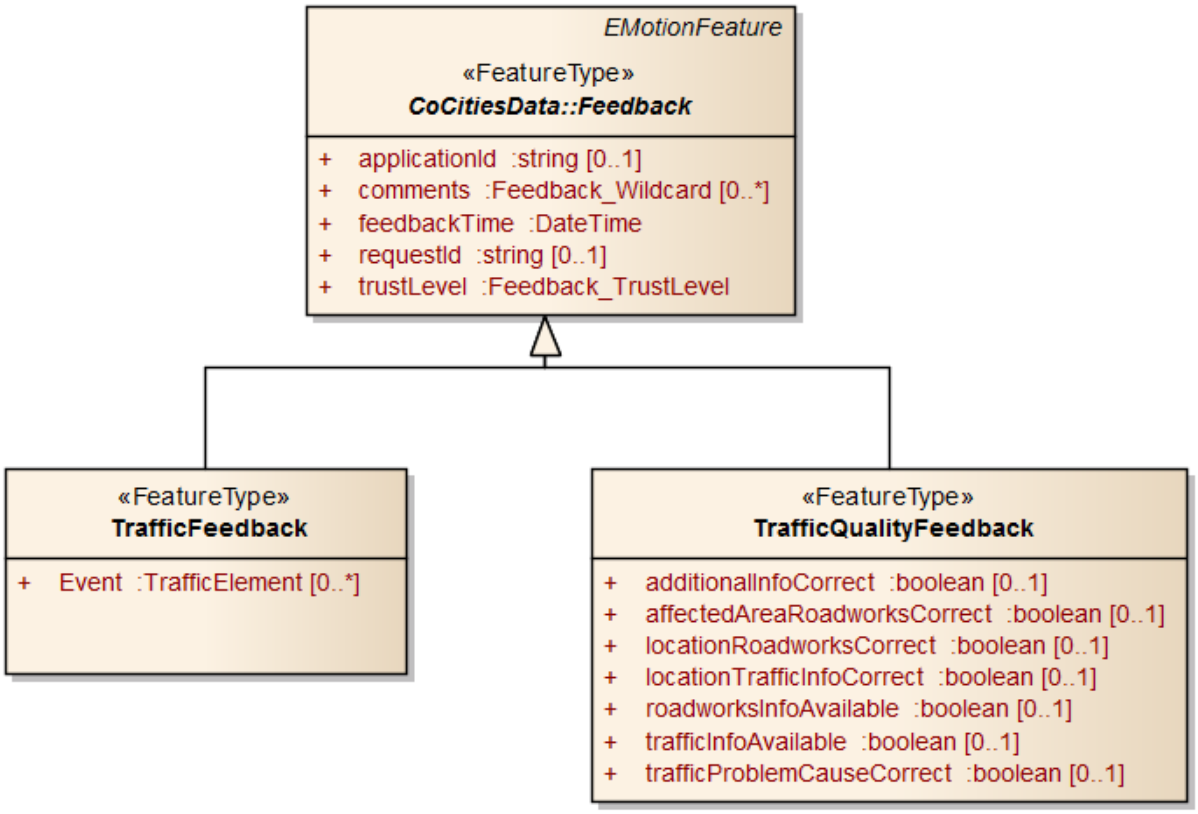


Figure 20: Traffic Information Feedback

Together with quality feedback, the TISP, through the *TrafficFeedback* Feature Type, can send to the RDSS new data related to traffic events, collected directly from the end user (e.g. notify an accident or a congestion on a road link).
The *TrafficFeedback* Feature Type is composed by a unique attribute which is an array of TrafficElement type. This type is defined within the eMotion/In-Time Application Schema and is specialized by different classes like *Accident*, *AbnormalTraffic*, *Obstruction*, *RoadWeatherAndEnvironmentEvent*. Traffic information in this model is based on Datex II.

A complete overview can be found in the eMotion Deliverable 6 – Appendix 1.

8.5 Public Transport Information Feedback Data

The Public Transport Information Feedback model is composed by the following types (Figure 21):

- *PublicTransportFeedback*
- *PublicTransportQualityFeedback*

The *PublicTransportQualityFeedback* comprises feedback data related to the quality of the public transport information received by the user upon a service call to RDSS. Quality information, in particular correctness using Boolean values, can be send for the most common chunk of data

related to public transport information: estimated time of arrival (*arrivalTimeOK*), information about lines (*linesOK*), location of stop point (*stopLocationOK*), information about schedule (*stopScheduleOK*).

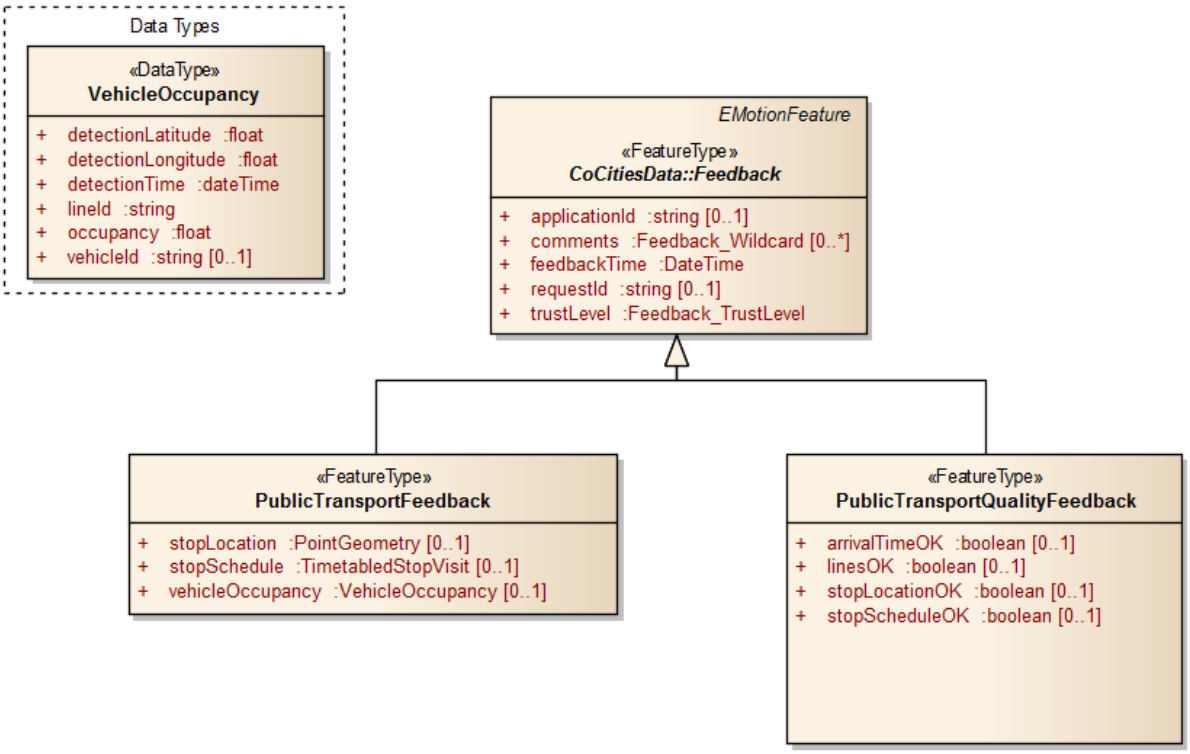


Figure 21: Public Transport Information Feedback

The *PublicTransportFeedback* Feature Type is used to send correction about existing data to RDSS. Its attributes carry data about the location of the StopPoint (*stopLocation*), the schedule (*stopSchedule*) and vehicle occupancy (vehicleOccupancy). The VehicleOccupancy model is defined as a simple class containing the position of the user (*detectionLatitude*, *detectionLongitude*), date and time of the measurement (*detectionTime*), the unique identifier of the public transport line assigned to the vehicle (*lineId*), the perceived occupancy of the vehicle (*occupancy*) and optionally the unique identifier of the vehicle (*vehicleId*). The other two types, PointGeometry and TimetabledStopVisit, are defined within the eMotion/In-Time Application Schema. Public Transport domain in this model is based on *Transmodel*, *SIRI*.

A complete overview can be found in the eMotion Deliverable 6 – Appendix 1.

8.6 Point of Interest Information Feedback Data

The Point Of Interest Information Feedback model is composed by the following types (Figure 22):

- *PoiFeedback*
- *PoiQualityFeedback*
- *RoadLinkQualityFeedback*

In the point of interest information case, the feedback related to quality of received data is divided into two different classes: *PoiQualityFeedback* and *RoadQualityFeedback* which is more related to objects of road link.

PoiQualityFeedback comprises feedback data related to the quality of the point of interest information received by the user upon a service call to RDSS. Quality information, in particular correctness using Boolean values, can be send for the most common chunk of data related to point of interest: location of Parking Sign (*locationOfParkingSignCorrrect*) and the picture of the point of interest (*pictureOfDestinationCorrect*).

RoadLinkQualityFeedback comprises feedback data related to the quality of the road link (map) data information received by the user upon a service call to RDSS. Quality information, in particular correctness using Boolean values, can be send for the most common chunk of data related to this kind of data: location of Speed Limit Start (*locationOfStartSpeedLimitCorrrect*), information about speed limits (*speedLimitInfoCorrect*), tolls and related information (*locationOfTollCorrect, tollFixedPricesCorrect, tollInfoAvailable, tollPaymentMethodCorrect*).

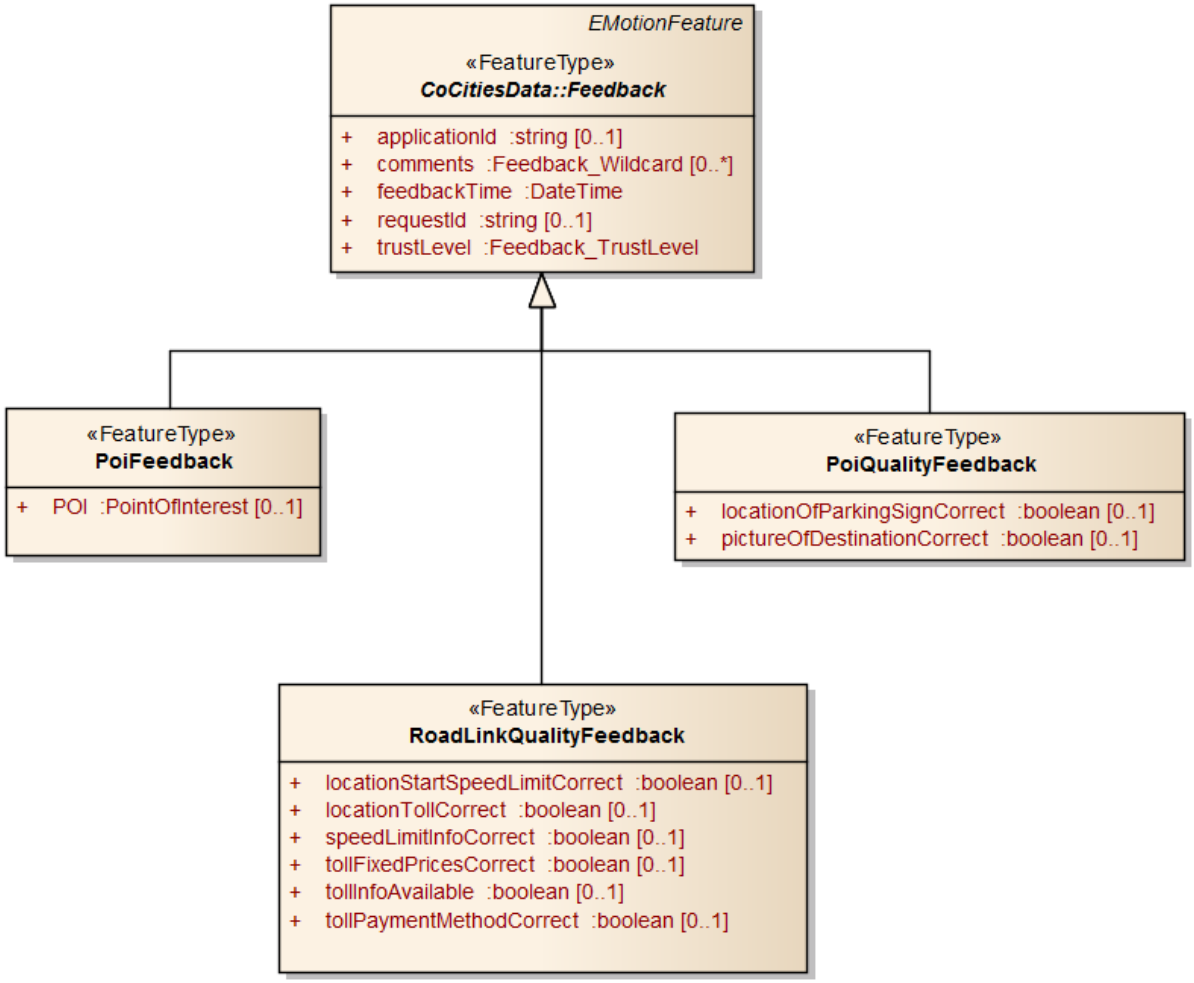


Figure 22: Point Of Interest Feedback Information

Together with quality feedback, the TISP, through the *PoiFeedback* Feature Type, can send to the RDSS new data related to parking domain, collected directly from the end user. This kind of data can be related to correct/complete information for point of interest already present in the RDSS data source, but also a completely new bunch of information not yet included at the RDSS.

The information is transferred using the *PointOfInterest* type defined in the eMotion/In-Time Application Schema. Point Of Interest domain of this model is based on *POIX*, *GDF Model*.

A complete overview can be found in the eMotion Deliverable 6 – Appendix 1.

8.7 Journey Planning Information Feedback Data

The Journey Planning Feedback model is composed by the following types (Figure 23):

- *JourneyPlanningFeedback*
- *JourneyPlanningQualityFeedback*

PoiQualityFeedback comprises feedback data related to the quality of the point of interest information received by the user upon a service call to RDSS. Quality information, in particular correctness using Boolean values, can be send for the most common chunk of data related to journey planning information: estimated time of arrival (*etaCorrect*), destination (*destinationCorrect, destinationLastMileCorrect*), information about a single segment of the trip (*tripSegmentStartPositionCorrect, tripSegmentStartTimeCorrect, tripSegmentStopPositionCorrect, tripSegmentStopTimeCorrect*).

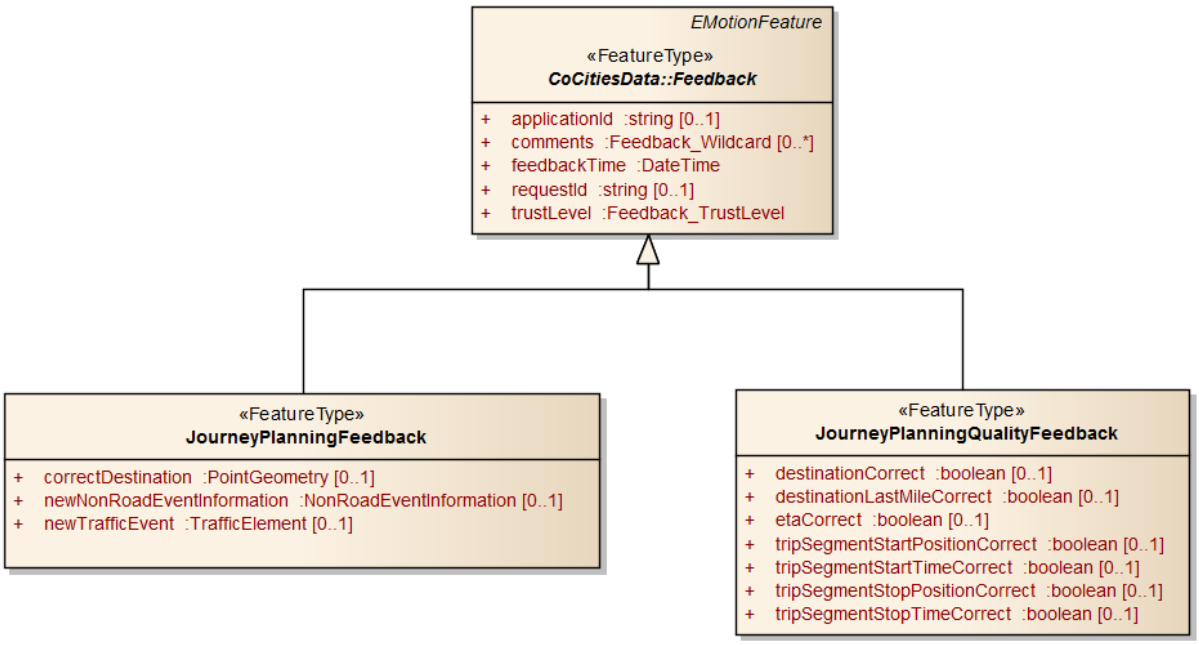


Figure 23: Journey Planning Feedback

The *JourneyPlanningFeedback* Feature Type is used to send feedback regarding the trip. The feedback can be a correction on destination point (*correctDestination*) or events reported by users while on trip (*newNonRoadEventInformation, newTrafficEvent*). These types are defined within the eMotion/In-Time Application Schema.
A complete overview can be found in the eMotion Deliverable 6 – Appendix 1.

It should be noted, that the walking and cycling services are covered by this feedback as well as all other means of transport provided by In-Time services.

9. Service Model

The *Service Model* Schema (Package: *Co-CitiesServices*) contains the interfaces and the data types used upon method invocations.
The schema is structured following the categorization given in Deliverable D2.1 – Chapter 7

In more detail an Application Schema is defined for each feedback domain (*Public Transport, Point Of Interest, Parking, Traffic Information, Multimodal Journey Planning*). These Packages import the Co-Cities Data Application Schema (Package: *Co-Cities Data*) together with the common features and data types defined directly in the In-Time Data Package (Figure 14 and Figure 24).

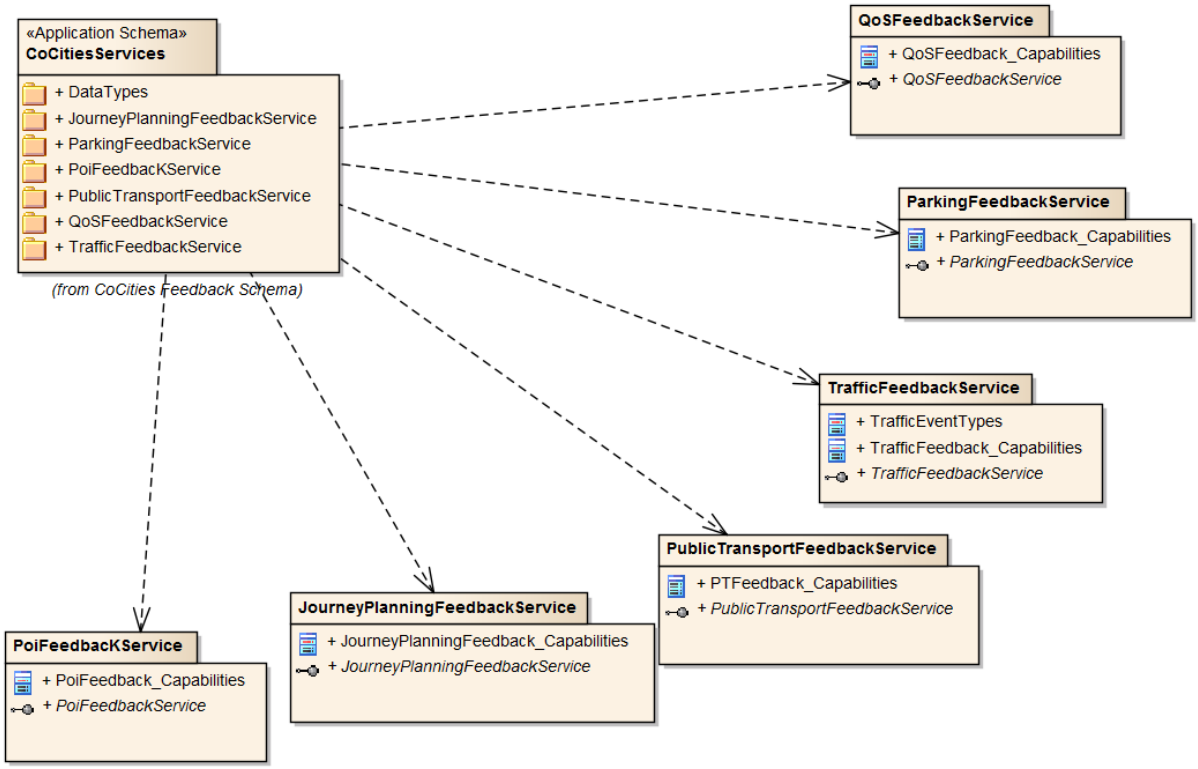


Figure 24: Co-Cities Services Application Schema

A *Service Quality Service* Application Schema is also included in the model, defining the QoS service feedback common to each domain.

9.1 Common Types

All the methods sending feedbacks to the RDSS, send back an acknowledgementin form of an instance of the *Feedback_Result* class (Figure 25).

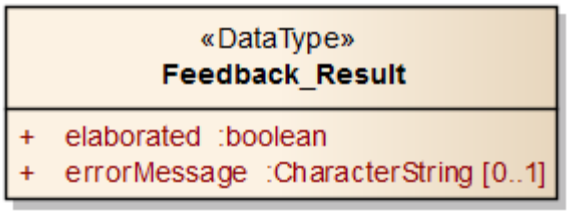
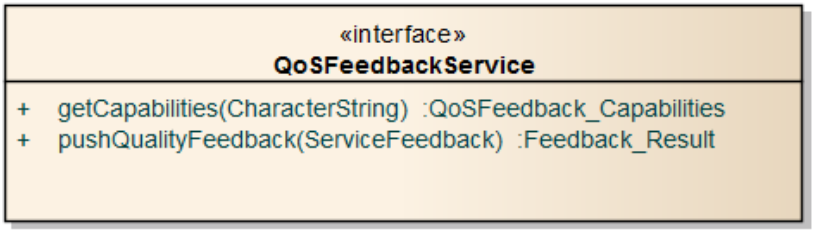


Figure 25: Feedback Result Types

This type holds a Boolean indicating if the data is received and elaborated correctly by the RDSS. If a problem is detected the *errorMessage* field can be used to report to TISP the problem.

9.2 Quality of Service Feedback Service

This section defines the service interface for the delivering of feedbacks regarding parking information.



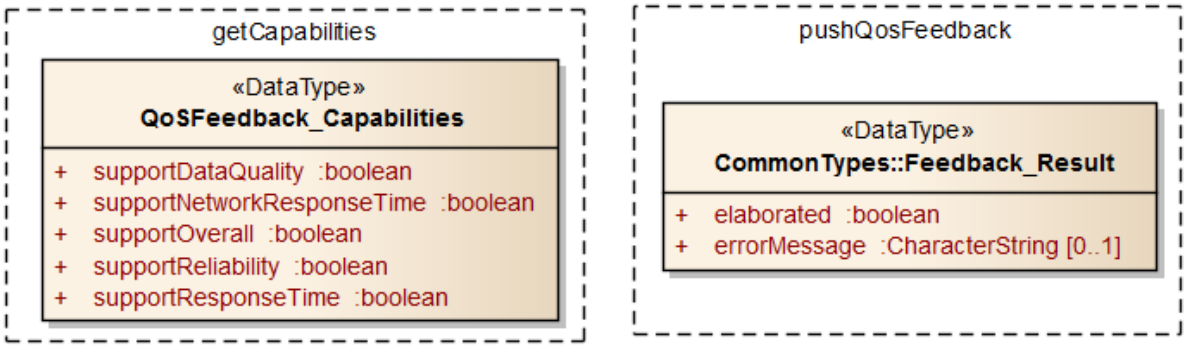


Figure 26: Quality of Service Feedback Service Interface

The service comprehends (Figure 26):

- a *getCapabilities* method. This method, accepting a string parameter indicating the version of the model (for future backwards compatibility), will be invoked by TISP to exactly know what types of feedback is supported on RDSS side (described by *QoSFeedback_Capabilities*).
- A *pushQualityFeedback* method. This method will be invoked by TISP passing a *ServiceQualityFeedback* instance.

9.3 Parking Information Feedback Service

This section defines the service interface for the delivering of feedbacks regarding parking information.

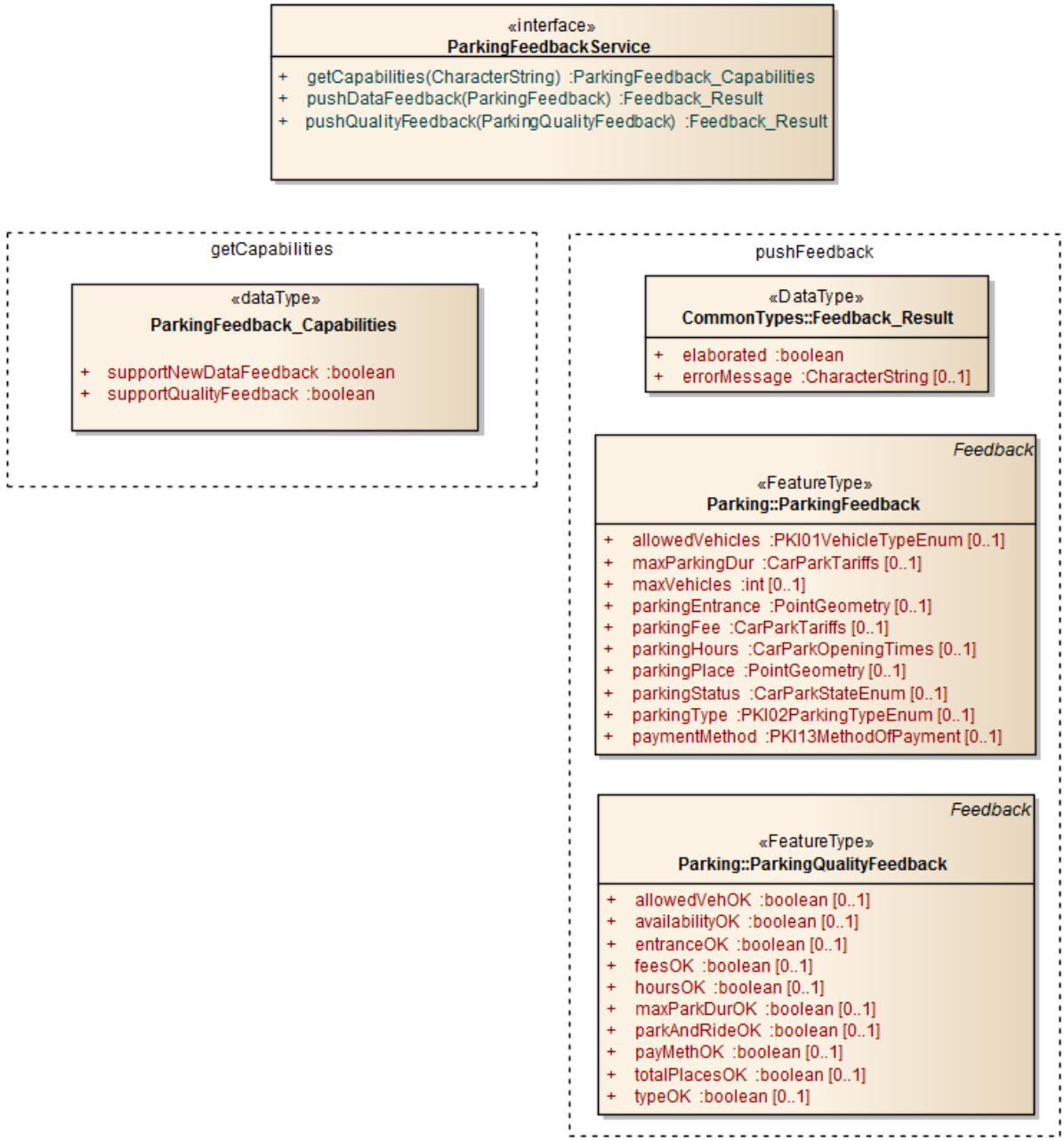


Figure 27: Parking Information Feedback Service Interface

The service comprehends (Figure 27):

- a *getCapabilities* method. This method, accepting a string parameter indicating the version of the model (for future backwards compatibility), will be invoked by TISP to exactly know what types of feedback is supported on RDSS side (described by *ParkingFeedback_Capabilities*).

- A *pushDataFeedback* method. This method will be used by TISP to send a *ParkingFeedback* object. The RDSS will receive the feedback, elaborate it and send back a *Feedback_Result*.
- A *pushQualityFeedback* method. This method will be invoked by TISP passing a *ParkingQualityFeedback* instance.

9.4 Traffic Information Feedback Service

This section defines the service interface for the delivering of feedbacks regarding traffic information.

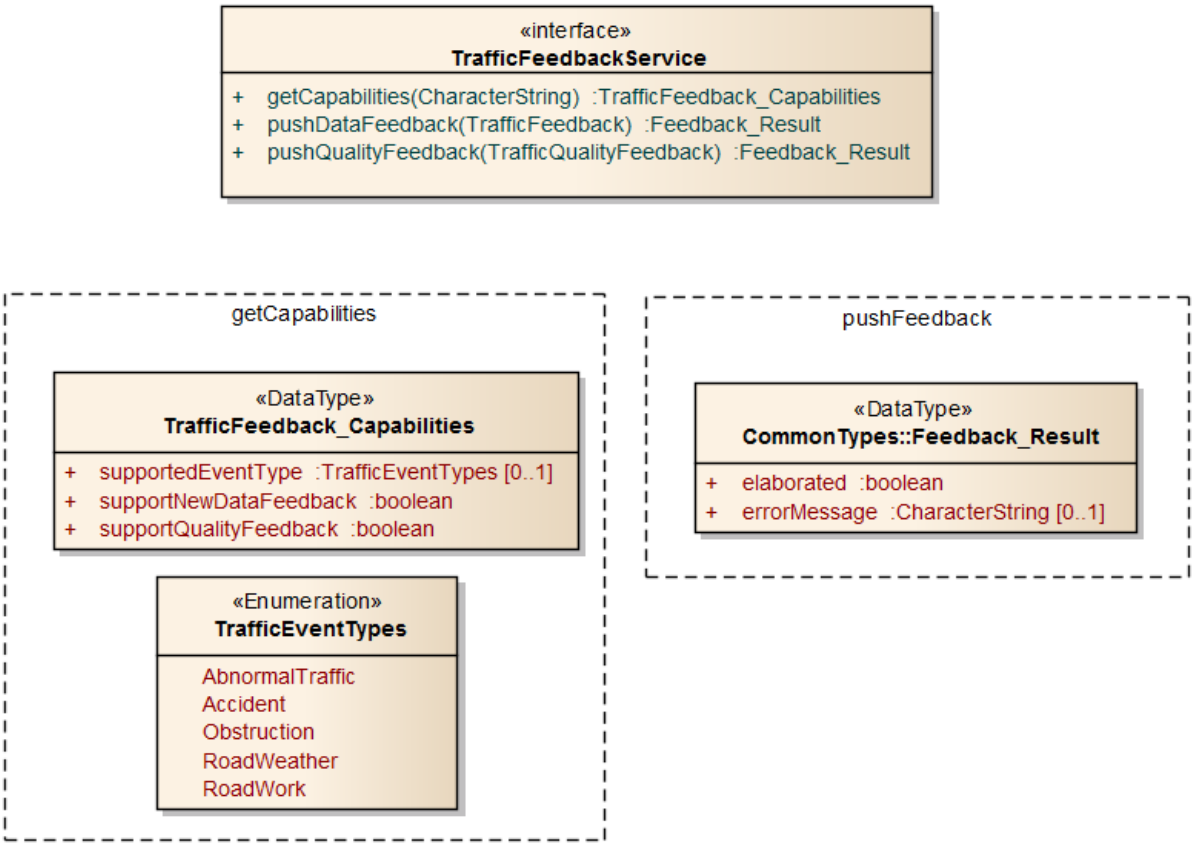


Figure 28: Traffic Information Feedback Service Interface

The service comprehends (Figure 28):

- a *getCapabilities* method. This method, accepting a string parameter indicating the version of the model (for future backwards compatibility), will be invoked by TISP to exactly know what types of feedback, and traffic events, is supported on RDSS side (described by *TrafficFeedback_Capabilities*).
- A *pushDataFeedback* method. This method will be used by TISP to send a *TrafficFeedback* object. The RDSS will receive the feedback, elaborate it and send back a *Feedback_Result*.
- A *pushQualityFeedback* method. This method will be invoked by TISP passing a *trafficQualityFeedback* instance.

9.5 Public Transport Information Feedback Service

This section defines the service interface for the delivering of feedbacks regarding public transport information.

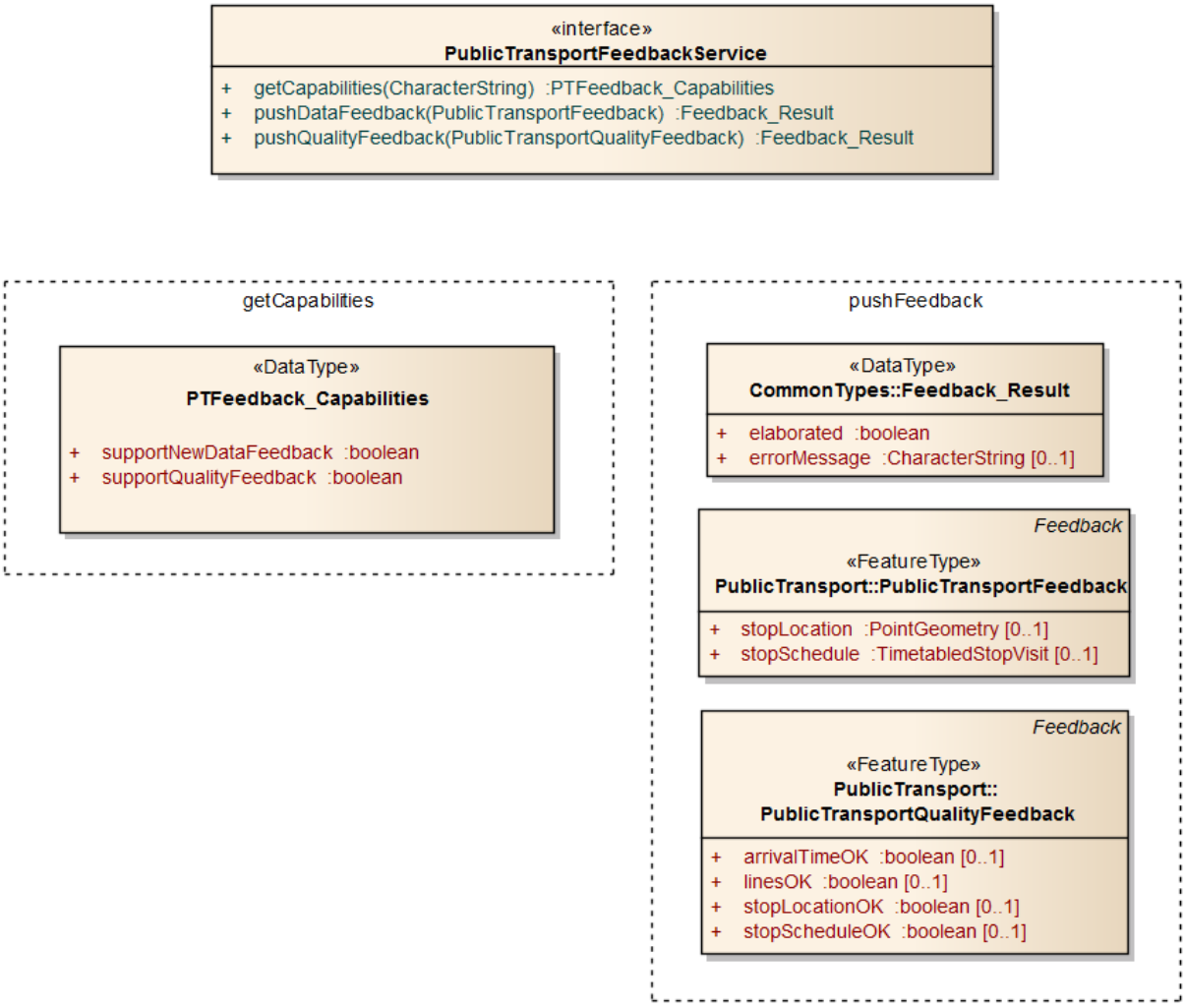


Figure 29: Public Transport Information Feedback Service Interface

The service comprehends (Figure 29):

- a *getCapabilities* method. This method, accepting a string parameter indicating the version of the model (for future backwards compatibility), will be invoked by TISP to exactly know what types of feedback is supported on RDSS side (described by *PTFeedback_Capabilities*).
- A *pushDataFeedback* method. This method will be used by TISP to send a *PublicTransportFeedback* object. The RDSS will receive the feedback, elaborate it and send back a *Feedback_Result*.
- A *pushQualityFeedback* method. This method will be invoked by TISP passing a *PublicTransportQualityFeedback* instance.

9.6 Point of Interest Information Feedback Service

This section defines the service interface for the delivering of feedbacks regarding point of interest information.

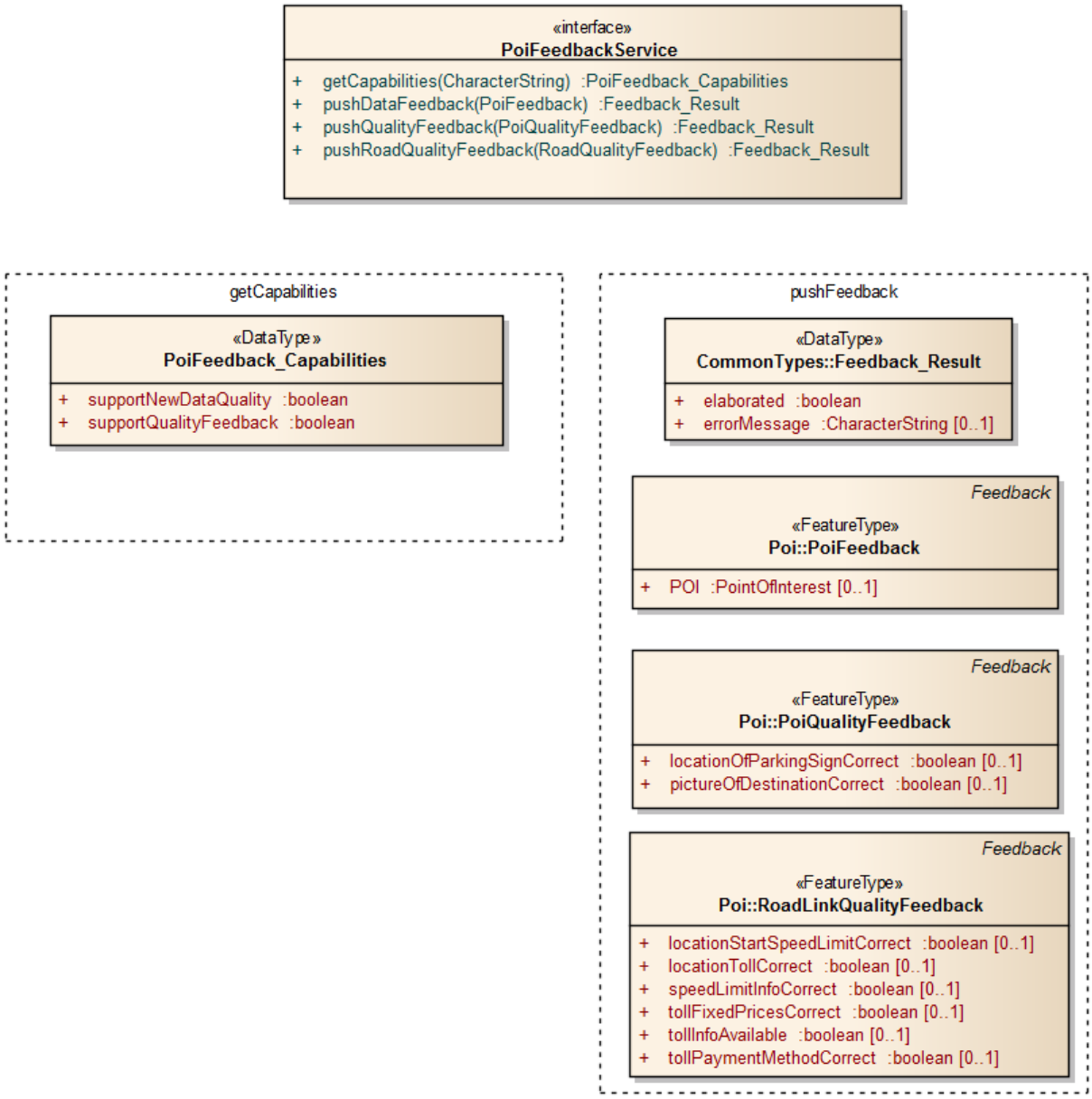


Figure 30: Point Of Interest Information Feedback Service Interface

The service comprehends (Figure 30):

- a *getCapabilities* method. This method, accepting a string parameter indicating the version of the model (for future backwards compatibility), will be invoked by TISP to exactly know what types of feedback is supported on RDSS side (described by *PoiFeedback_Capabilities*).
- A *pushDataFeedback* method. This method will be used by TISP to send a *PoiFeedback* object. The RDSS will receive the feedback, elaborate it and send back a *Feedback_Result*.
- A *pushQualityFeedback* method. This method will be invoked by TISP passing a *PoiQualityFeedback* instance.

- A *pushRoadQualityFeedback* method. This method will be invoked by TISP passing a *RoadLinkQualityFeedback* instance.

9.7 Journey Planning Feedback Service

This section defines the service interface for the delivering of feedbacks regarding journey planning information.

Figure 31: Journey Planning Feedback Service Interface

It should be noted, that walking and cycling routes as delivered by the In-Time intermodal routing services are also covered within the journey planning feedback, together with public and motorised individual transport.

The service comprehends (Figure 31):

- a *getCapabilities* method. This method, accepting a string parameter indicating the version of the model (for future backwards compatibility), will be invoked by TISP to exactly know what types of feedback is supported on RDSS side (described by *JourneyPlanningFeedback_Capabilities*).

- A *pushDataFeedback* method. This method will be used by TISP to send a *JourneyPlanningFeedback* object. The RDSS will receive the feedback, elaborate it and send back a *Feedback_Result*.
- A *pushQualityFeedback* method. This method will be invoked by TISP passing a *JourneyPlanningQualityFeedback* instance.

10. Data Security

The Co-Cities system architecture is based on Web Services (SOAP), XML (particularly GML) and Internet-technology as the information and feedback services are accessed directly via Internet.
As stated in the previous sections, no particular consideration, being proprietary, can be made on the communication channel between the TISP and the end-user devices. In addition to this, in order to minimize privacy issue on the RDSS side no personal information regarding the single end-user is transferred from TISP to RDSS.

Having specified the operative context of the Co-Cities feedback services, it can be stated that the basic requirement for data security should be considered principally at the transport level in the communication channel between the TISP and the RDSS.

To fulfil this requirement, standard and interoperable solution should be applied. In particular the Basic Security Profile guidelines proposed by the Web Service Interoperability Organization (WS-I http://www.ws-i.org) can be followed as it deals with transport security, SOAP messaging security and some other more general security considerations.

The profile is publicly available at: http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html.
Basic Security Profile adopt the most widely used countermeasure against potential threats: HTTP secured (HTTP/S) with either TLS 1.0 or SSL 3.0 (SSL 2.0 is prohibited by the Profile for some well known security issues) as transportation layer mechanism.

HTTP/S is widely regarded as a mature standard for encrypted transport connection to provide a basic level of confidentiality and forms the simplest means of achieving some basic security features that are probably required by some RDSS which provides the Co-Cities feedback services.
Following this guidelines, in the Co-Cities architecture a basic secured connection is made by the use of HTTP/S with TLS 1.0 or SSL 3.0 as the underlying protocol.

Both TLS and SSL protocols defines some basic steps:

- Negotiation by the two parties on the algorithm to use
- Exchange of private keys with a public-key based encryption method and identification with certificates

To increase the level of security Co-Cities services may also follow this other recommendation of the Basic Security Profile:

- TLS-capable implementations may implement TLS_RSA_WITH_AES_-128_CBC_SHA, or FIPS equivalent, as encoding algorithm (called also ciphersuite);
- SSL-capable implementations may implement SSL_RSA_WITH_AES_-128_CBC_SHA, or FIPS equivalent, as encoding algorithm;
- Mutual authentication of HTTPS may be adopted when the authentication of the Web
- Service instance by the consumer is insufficient.

The above considerations permit a basic layer of security for the Co-Cities feedback services.

11. Reference Platform

While implementations and technologies used at TISP and RDSS side differ, are generally proprietary and as a result have to be understood as black boxes in the scope of Co-Cities, data provided at the CAI is agreed upon, standardized and comparable. Consequently, the scope of detailed testing and validation in Co-Cities is bound to the CAI level. A large amount of testing, validation and reporting at this level can be automated with the help of a centralized Reference Platform that allows assessing the availability, conformity, as well as other key performance indicators of the data provision of each service/in each city. After a description of objectives, target groups and a short overview of Co-Cities service delivery chain, the requirements towards the platform and necessary features are analysed. Based on the results the Reference Platform and its components are described and specified.

11.1 Objectives and target groups

The Reference Platform is a tool for testing and validating the Co-Cities services' complete delivery chain at the Common Agreed Standardised Interface (CAI) in each city. It simplifies error analyses, as it allows quickly assessing which data and in which quality was available at the interface. Furthermore comparability between cities and services is ensured.
Project partners, Traffic Information Service Providers (TISP) and Local Data Providers (RDSS) can use it to monitor the interface, run automated tests and access reports/statistics depending on their permissions.

11.2 General Requirements

Based on Co-Cities' architecture, lessons from In-Time, the validation strategy and several discussions with project partners the following requirements for the Reference Platform were identified:

- Relay all traffic between the TISPs and the RDSS while affecting the performance of the service delivery chain as little as possible
- Log incoming and outgoing messages in a standardized way
- Aggregate and prepare these messages for further analysis and to get an up to date status.
- Support a hierarchy of different user roles (Administrator/User) with different permissions (e.g. read only) at each level
- Offer monitoring and analysis capabilities in a Web-GUI to users, depending on their permissions
- Login with Username/Password
- Allow mobile access in the field

11.3 Reference Platform Specification

11.3.1 Overview

Based on the identified requirements the Reference Platform is set up as a central, rule based, configurable communication platform, serving as a reference system for all involved partners. Its functionality is offered via two components:

- **Reference Server:** Traffic between the TISPs and the CAI is relayed through the Reference Server (Proxy) to allow enhanced logging of received and sent data as well as test case monitoring.

- **Reference OBU** (On-Board Unit; Tablet): The Reference OBU acts as a mobile interface to the Reference Server (**via a Browser / Webview)** and other available web services. As a result, the Reference OBU simplifies integration and validation tests in the field. The Reference OBU is not used as a HMI design reference and does not offer end-user services.

The following figure provides an overview of the Reference Platform components in the context of the Co-Cities architecture:

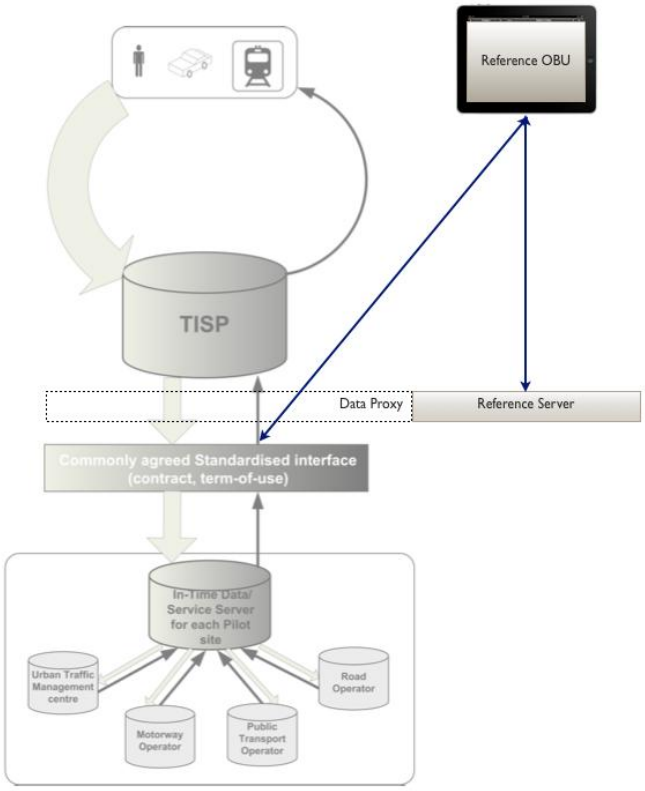


Figure 32: Reference Platform – System Context

Data relayed through the Reference Server's Proxy component is logged and stored to a database/data-warehouse for asynchronous analysis. This data is not only made available to the server's monitoring and reporting components, but can also be accessed by the Reference OBU.

11.3.2 Business Case Engine

The Reference Server receives messages via input channels (TISP/RDSS) and synchronously forwards these messages to the corresponding output channels. Between input and output the messages can be processed (e.g. validated and stored). These process chains or business cases can be reused and combined in a flexible way. All processing steps are documented and hence can be

easily analysed ex post. While the business logic itself is agnostic to the Co-Cities domain, the domain specific knowledge is implemented in the individual segments of each business case.

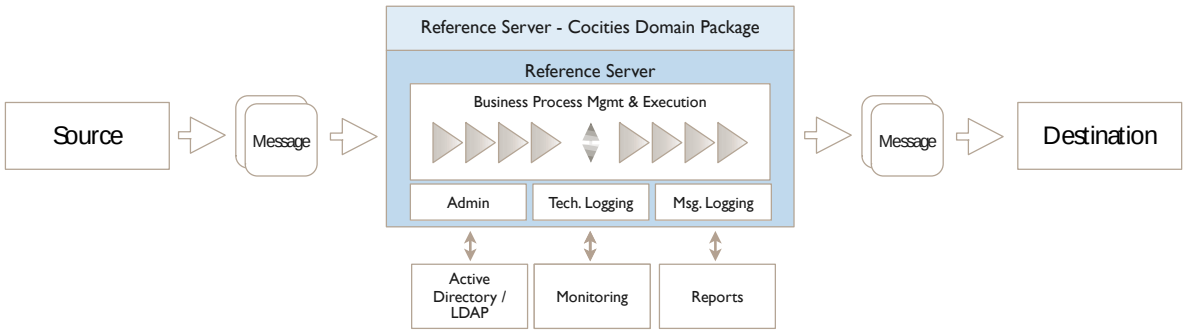


Figure 33: Reference Server – Business Case Engine

A request (e.g. routing request) received can therefore be validated using a validation component, stored in the domain specific data structure and forwarded to its destination. The response generated by the destination server (e.g. routing result) can again be validated, stored and forwarded to the actual source:

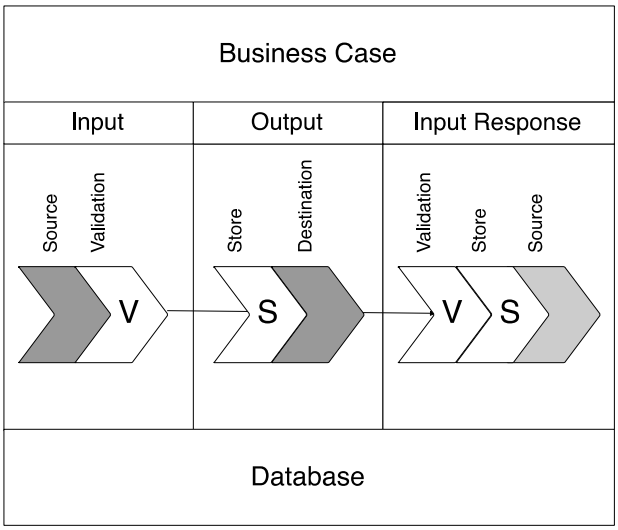


Figure 34: Reference Server – Business Case Example

A central instance of the Reference Server will be set up and dispatch the messages according to the defined business cases for each service/city:

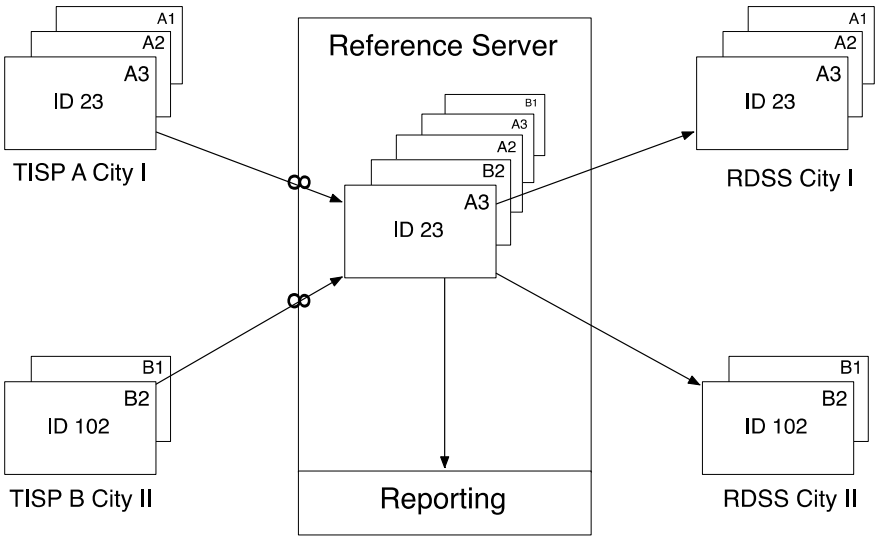


Figure 35 – Reference Server – Data Dispatching

11.3.3 Message handling

Concerning the handling of messages the following features are planned:

- All incoming and outgoing messages are logged and forwarded based on predefined rules
- All messages / transactions can be identified with an external (Request ID) or internal ID
- The messages can be validated against predefined .xsd schemas per service
- Statistics for each message are calculated and stored (e.g. response time, validity)
- Erroneous messages are distinguished and flagged (technical errors, invalid requests, security,…)

11.3.4 Web GUI

The system supports a hierarchy of different user roles, which have access to the Reference Platform's functionalities based on their permissions. Users will be administered with the help of a LDAP directory. After logging in the Web GUI offers access to the following monitoring and analysis capabilities.

A Dashboard is planned to show the current status of all business cases.

Figure 36: Reference Server – Dashboard Mockup

Further analysis is possible with the help of a detailed Message Log. This message log lists all executed business cases and their results at each processing step and can be filtered by parameters such as status, time period etc.

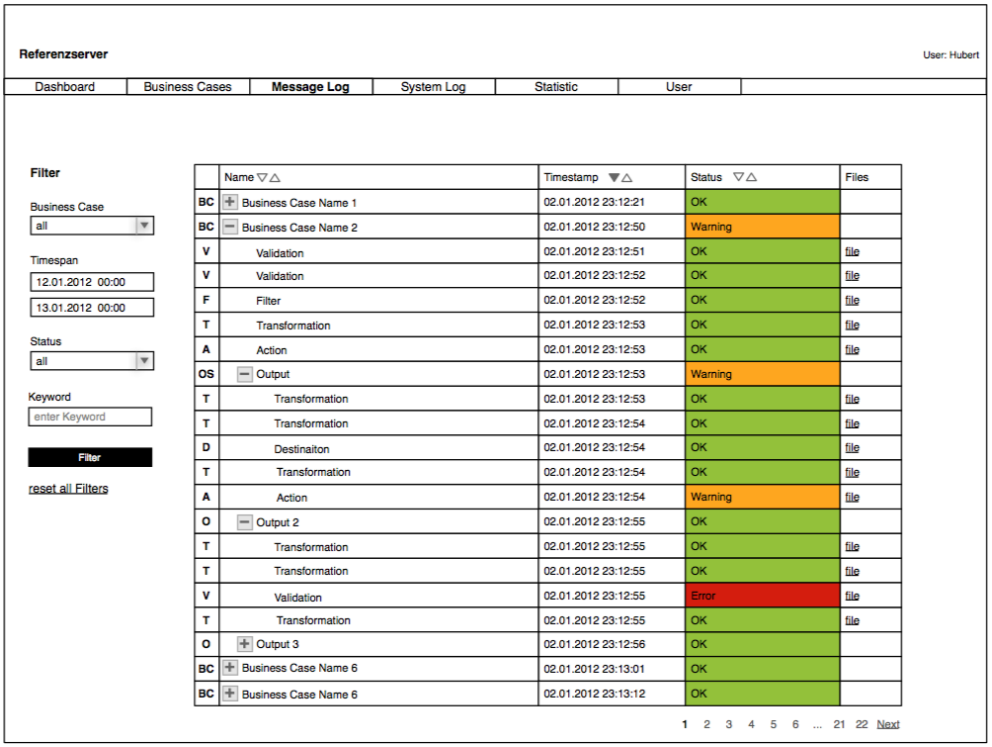


Figure 37: Reference Server – Dashboard Mockup

Technical background information will be logged in a technical/system log. This log will comprise error messages and exceptions thrown by the Reference Server and is meant for debugging purposes.

Furthermore, Accessing the Web GUI with the Reference OBU allows mobile access to relevant data for testing and validation in the field.

Additionally, due to the fact that data about each business case execution and processing step are stored, statistics necessary for validation (e.g. Availability, Conformity, Level of Service, Response Time; see D2.3 for further details) can be provided in csv format and test messages generated by TISPs / RDSS can be easily traced.

11.3.5 Architecture

The Reference Server's modular architecture allows stepwise implementation, flexible configuration and good performance:

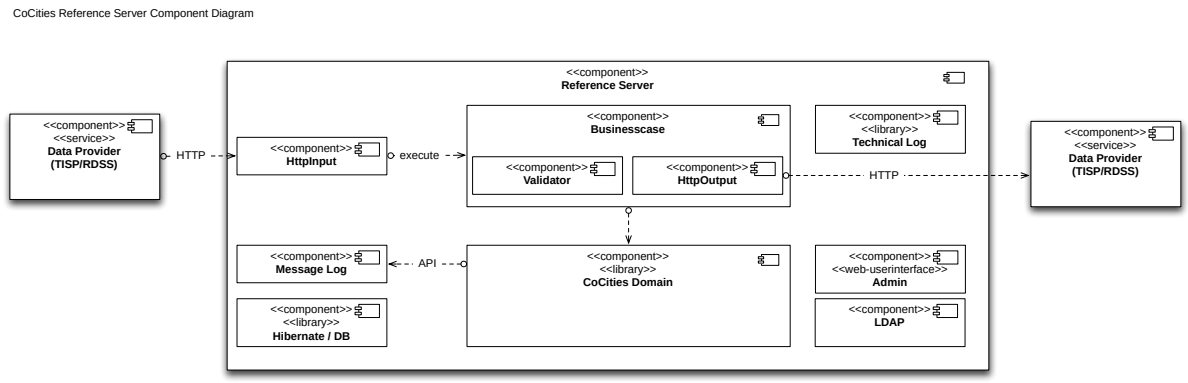


Figure 38: Reference Server – Component Diagram

As a consequence, the Reference Server is an efficient tool for an end-to-end service assessment in terms of data and service quality. It can be used for automated CAI testing, monitoring and analyses of all incoming and outgoing data at the CAI level. Additionally, predefined statistics can be generated to get an up-to-date status and compare results between services and/or cities.

11.4 Testing the complete service delivery chain at the CAI level

Each service / use case consists of a maximum of 5 steps (note that some use cases are no direct responses to service requests and as a result only cover the feedback loop):

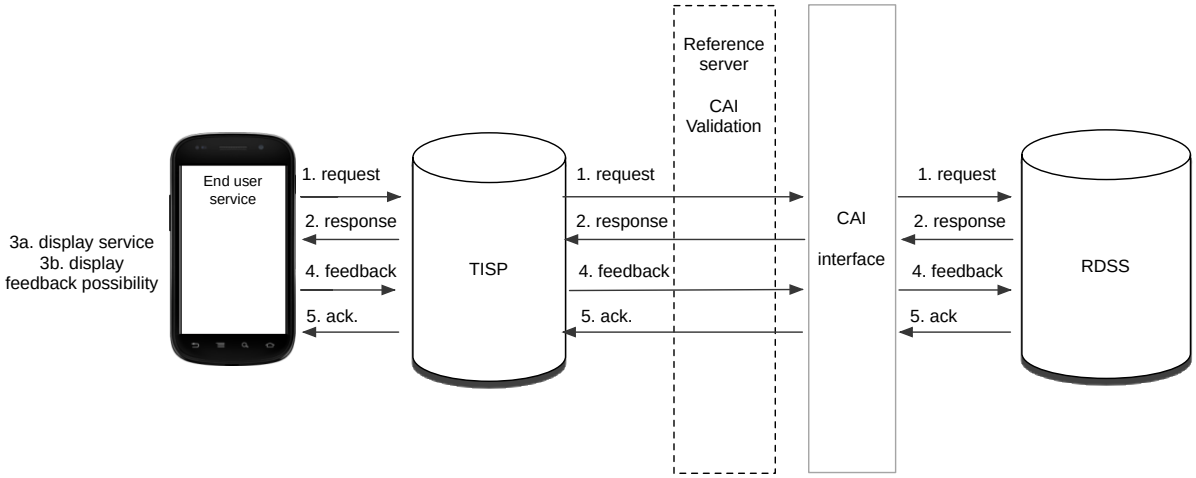


Figure 39: Service delivery chain, including display at the end-user device

Blocks	Steps	Description
In-Time Service	1. Request:	The end-user (client) sends a request to the TISP service provider for a specific service. The TISP service provider forwards the request to the local RDSS in the CAI format, which is responsible for the provision of the corresponding service data.
	2. Response:	The RDSS provides the requested service to the TISP and forwards it to the end-user.
	3. Display:	End-user devices visualize the service (3a) and feedback capabilities (3b). This step is not visible at the CAI level.
	3a. Service:	The result of the called service is displayed in the mobile device or terminal of the end-user depending on the TISP application and associated services installed on the user mobile or terminal
Feedback Extension	3b. Feedback Possibility	Depending on the kind of the current service, the end-user will have the possibility to provide feedback on the service itself. Based on the client interface design there will be three different ways of providing feedback. The end-user can provide feedback specifically related to the received service, general service feedback, or new data. The latter two cases do not require the before mentioned steps (1-3a).
	4. Feedback	The feedback information is send to the TISP and forwarded to the RDSS via the CAI interface.
	5. Acknowledgement:	The feedback reception is acknowledged by the RDSS.

Table 1: Service delivery chain , including display at the end-user device

The Reference Platform supports validation and testing efforts at the CAI level. Even though Step 3 (Display) is not visible at the CAI, using the Reference Platform eases error analysis as request/response can be traced back. For a detailed description of testing and validation

methodology as well as relevant criteria see deliverable D2.3 and the internal reports of SWP3300 and SWP3400.

12. Annex 1 - Abbreviations

API	Application Programming Interface
Ataf	Azienda Trasporti dell'Area Fiorentina – this is the public transport company of Florence
Atl	Azienda Trasporti Livorno – This is the public transport company of Livorno
AVM	Automatic Vehicle Monitoring
CAI	Commonly Agreed Interface (used by In-Time and extended by feedback channel in Co-Cities)
CBA	Cost Benefit Analysis
CIMUBISA	Bilbao Council's computing centre
DAS	Data Acquisition System
DoW	Description of Work
DSS	Decision Support System
ETA	Estimated Time of Arrival
ESB	Enterprise Service Bus
FCD	Floating Car Data
FOT	Field Operational Test
FOTIP	Field Operational Test Implementation Plan
FPP	Full Project Proposal
GPS	Global Positioning System
GQM	Goal-Question-Metric
GUI	Graphical User Interface
HMI	Human-Machine Interaction
ICM	Integrated City Management
ICT	Information and Communications Technology
IT	Information Technology
KPI	Key Performance Indicators
ITS	Intelligent Transportation System

LoS	Level of Service
MIIC	Mobility Integrated Information Centre
MSS	Measurement Support System
OBU	On-Board Unit
OGC	Open Geospatial Consortium
PI	Performance Indicators (PIs)
PND	Personal Navigation devices
PT	Public Transport
QoS	Quality of Service
RDS-TMC	Radio-Data-System - Traffic Message Channel
RDS	Regional Data Services
RDSS	Regional Data / Service Server
ROI	Return On Investment
RTPI	Real Time Passenger Information
RTTI	Real Time Traffic Information
SOA	Service Oriented Architecture
SVVP	Software Verification and Validation Plan
SWP	Sub Work Package
TDE	Test Descriptor. Prefix and nomenclature used in the formalized test description tables, to reference the test descriptor identifier.
TISP	Traffic Information Service Providers
TSP	Test Specification. Prefix and nomenclature used in the formalized test specification tables, to reference the test specification identifier.
UML	Unified Modeling Language
VIB	Verkehrsinformationsagentur Bayern GmbH
VPN	Virtual Private Network
W3C	World Wide Web Consortium
WFS	Web Feature Service
WMS	Web Map Service
WP	Work Package

WSDL	Web Services Description Language
XML	Extensible Markup Language
XSD	XML Schema Definition

13. Annex 2 - References

CoCities D2.1 - Service definition and use cases

Version 1.0; December 2011; Tomáš Tvrzský et al; Source: www.co-cities.eu

In-Time D2.3.1 – Report on In-Time Interfaces and Protocols;

Version 2.0; October 2010; Samson Tsegay et al; Source: www.in-time-project.eu

DATEX II Documents;

Source: www.datex2.eu

eMOTION deliverable D6: eMOTION System – Technical Specification;

Version 1.0; October 2008, Reinhard Erstling, Stefan Olk et al; Source: www.emotion-project.eu

In-Time deliverable D3.2.1 - Specification document of In-Time server, interfaces and protocols ;

Version 1.0; April 2010; Michele Masnata, Marco Garrè et al; Source: www.in-time-project.eu

14. Annex 3 - XSDs

Following the methodology described in chapter 7 the UML Model is used to produce a set of XSDs, for the data model, and a set of WSDLs, for the service model, which are the concrete components to be used to implement interfaces which are Co-Cities compliant.
The resulting XSDs and WSLDs files are compiled and distributed in two different version:

- The first version inherits the file structure and organization from the previous projects (In-Time and eMotion): a set of XSDs is made available for the data model and a set of XSDs + WSDLs are made available for the service model.

- The second version is composed by the WSDL files for the Co-Cities feedback services. These files are self-consistent and each WSDL comprehends also the definition of the data schema without including external XSD files. This version is made available to speed up the implementation simplifying the generation of both Server and Client.