TECHNICAL REPORT

ISO/TR 26999

First edition
2012-10-15

Intelligent transport systems — Systems architecture — Use of process-oriented methodology in ITS International Standards and other deliverables

Systèmes intelligents de transport — Architecture de systèmes — Emploi d'une méthodologie orientée processus dans les Normes internationales ITS et autres produits livrables

Reference number
ISO/TR 26999:2012(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

In exceptional circumstances, when a technical committee has collected data of a different kind from that which is normally published as an International Standard ("state of the art", for example), it may decide by a simple majority vote of its participating members to publish a Technical Report. A Technical Report is entirely informative in nature and does not have to be reviewed until the data it provides are considered to be no longer valid or useful.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO/TR 26999 was prepared by Technical Committee ISO/TC 204, *Intelligent transport systems*.

Introduction

The objective of this Technical Report (TR) is to provide guidance on the use of the process-oriented method (POM), also known as data flow modelling, in the development of intelligent transport systems (ITS) International Standards and other deliverables, and in the design and implementation of ITS systems. In particular, it is intended to be used as the basis for the development of high-level system architectures for ITS. These architectures are tools to aid ITS implementations, and a mechanism to identify and promote the creation and use of standards.

The advantages of applying POM to the development of high-level system architectures for ITS include the following:

— POM is easily understood, particularly by non-technical people (e.g. decision-makers) who are often the intended audience for high-level system architectures;

— POM enables a coherent description to be built up from multiple user views;

— training and tool support is available, particularly in Europe and the USA;

— the data descriptions produced by POM are capable of manipulation by a metadata registry for ITS;

— the results of creating a POM system architecture can be easily transferred into requests for quotations (RFQs), expressions of interest (EOIs), tenders and other similar documents;

— the results of POM system architectures can be translated into UML for use by software developers;

— POM is applicable to both hardware and software and does not, therefore, pre-suppose the form in which its functionality will be implemented.

The disadvantages of using POM include the following:

— POM has a bad image, e.g. it is old-fashioned, and is usually not now included in the training of systems analysts and designers;

— parts of a POM system architecture might require conversion to UML before it will be accepted by most software developers.

There are some risks in using POM, but the benefits of its ability to be easily understood by the usual initial audience for high-level system architectures can often help with the initial promotion of ITS implementations. This TR is intended to provide guidance to stakeholders who are considering the use of POM for ITS.

Intelligent transport systems — Systems architecture — Use of process-oriented methodology in ITS International Standards and other deliverables

1 Scope

The scope of this Technical Report is the use of the so-called process-oriented method (POM) in International Standards, Technical Specifications, Technical Reports and related documents.

This Technical Report discusses the use of POM in the development of high-level system architectures for intelligent transport systems (ITS). It is based on the results of the work of the FRAME-S project and the FRAME Forum. Much of the text from Clause 2 through to the end of the document is therefore reproduced by kind permission of the European Commission and the FRAME Forum.

2 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

2.1
actor
sub-element of a terminator

NOTE It is mainly used to enable a particular variant of a terminator to be differentiated from other variants, e.g. to differentiate a public transport vehicle driver from any other type of driver, or all drivers.

2.2
architectural model
model that contributes to the content of an architectural view

2.3
architecture (generic definition)
set of concepts and rules for a system that describes the inter-relationship between entities in the entire system, independent of the hardware and software environment

NOTE Architecture is described through a series of viewpoints that might be at varying levels of generality/specificity, abstraction/conception, totality/component, and so on. See also "communications viewpoint", "functional viewpoint", "organizational viewpoint" and "physical viewpoint" definitions below.

2.4
architectural viewpoint
representation of a system from the perspective of an identified set of architecture-related concerns

2.5
architectural description
collection of information items used to describe an architecture

2.6
aspiration
expression of what a stakeholder wants the ITS implementation to provide, usually written in the language of the stakeholder and thus possibly having little or no formal structure

NOTE There could be many aspirations for each ITS implementation, depending on its scope and the number of stakeholders that are involved.

2.7
communications viewpoint
one of several architectural viewpoints of the system of interest, showing the links between the building blocks in the physical viewpoint that will enable them to communicate with each other, and including details of expected data throughputs and any other constraints that will affect the eventual choices of communications hardware and software

2.8
functional viewpoint
one of several architectural viewpoints of the system of interest, showing the functionality that will be needed to fulfil the requirements expressed in the user needs, this functionality being shown as a series of functions and data stores plus the data flows between them and the data flows between the functions and the terminators

2.9
ITS architecture
specific form of a system architecture for use as a tool in the initial stages of an ITS implementation

2.10
model
representation of an entity from which the important elements have been abstracted by removing certain detail while at the same time retaining the interrelationship between the key elements of the whole

NOTE 1 A model can be made more or less abstract by the successive suppression of detail such that the concepts and relationships come into enhanced focus and become more readily understood. However, the process can be taken too far when the simplification has exceeded the threshold where a necessary understanding can be achieved. Thus the process of modelling is one of going only far enough to achieve the optimum understanding and insight — and no further.

NOTE 2 A model is a way of representing something, other then in its natural state (see "Models of ITS" documents at the web site given in Reference [2] in the Bibliography).

2.11
organizational viewpoint
one of several architectural viewpoints of the system of interest, showing how the building blocks from the physical viewpoint (or the functional viewpoint) can be allocated to the different types of organization (or organizations themselves, if known) that will be involved with the ITS implementation

2.12
physical viewpoint
one of several architectural viewpoints of the system of interest, showing how the functionality from the functional viewpoint can be allocated to different physical locations and combined into different building blocks

2.13
stakeholder
entity that is involved in some way with the ITS implementation

2.14
stakeholder need
formal expression, using "shall" language, to define what the stakeholders expect the ITS implementation to provide, and from which the functional viewpoint is created, also known as "user need"

2.15
system architecture
single, high-level, description of the major elements or objects of a system plus the inter-connections between them

2.16
system of interest
another name used for the system, applied to whatever will be included in the ITS implementation that is created out of the ITS architecture

2.17
terminator
entity that is external to the system but with which the system communicates either to obtain inputs or to which it can send outputs

NOTE 1 Terminators may be split up into actors if necessary.

NOTE 2 In most ITS architectures, the terminators may be the same in both the functional and the physical viewpoints.

NOTE 3 In the US National ITS Architecture, a terminator defines the boundary of the system of interest. Each terminator may represent the people, systems and general environment that interface to ITS. The interfaces between terminators and the sub-systems and processes within the National ITS Architecture are defined, but no functional requirements are allocated to terminators. The logical and physical architecture views of the National ITS Architecture both have exactly the same set of terminators.

2.18
unified modelling language
UML
object-oriented modelling language specified in ISO/IEC 19501

2.19
user need
another name for "stakeholder need"

3 Symbols and abbreviated terms

3.1
E-FRAME
extended framework architecture made for Europe

3.2
FRAME
framework architecture made for Europe

3.3
ITS
intelligent transport system

3.4
KAREN
keystone architecture required for European networks

3.5
POM
process-oriented method

4 Background

4.1 TC 204, working group 1 (WG1)

This Technical Report arose to complement work done by WG 1 on the elaboration of Technical Report ISO/TR 24529 which covers the use of UML in ITS International Standards and other deliverables. WG 1

has never mandated the sole use of UML above other architecture methodologies, and this Technical Report is intended to describe the use of an alternative methodology.

POM should not be seen as a rival to UML, and the two should be seen as complementary rather than competitors. Each methodology should be seen as being most suitable for (parts of) architectures that are created at particular stages in ITS implementations. In general, POM is best suited to high-level architectures that are to be used by decision-makers and others to refine their original concepts for ITS implementation and explore alternatives, without the need for detailed knowledge of architecture modelling techniques. UML is seen by some as more suited to detailed lower-level architectures from which software is to be designed, coded and tested before being included in the implementation.

4.2 Systems and architectures

The topic of system architecture is surrounded by shibboleths and silver bullets which imply that "if you are not using my preferred method of describing them, then you are wrong". The basic flaw in this argument is that system architectures are used for a number of different purposes, and so it is not surprising that different modelling techniques are required. Thus the relevant phrase should be "horses for courses" rather than "one size fits all". Let us consider the two words "system" and "architecture in turn.

Systems of interest are often described in a hierarchical manner and, when this is done, each lower stage contains greater detail than the one above it. Thus each higher stage provides the requirements for the one below it, which is a "design" that conforms to those requirements [1]. In the world of ITS, we can identify at least seven levels of requirements and design, each with their own set of characteristics and needs, as follows [note that (a) some of the terms used are described below and (b) the bottom two bullets are at the same level]:

— The top-level requirements are stakeholder aspirations.

— These are interpreted into a structured design by the system architect in the form of stakeholder needs.

— The stakeholder needs are then used as requirements by the system architect who creates a high-level design — the ITS architecture.

— ITS engineers then use (part of) the ITS architecture to create requirements for procurement.

— The system engineer in the supplier's organization uses these requirements for procurement to create a detailed design for the final equipment.

— Software and hardware engineers use parts of this detailed design to produce requirements for the software and hardware, respectively.

— A programmer uses the software requirements to produce a program, which is a form of design.

— A hardware engineer uses the hardware requirements to produce a design for the hardware.

Consequently, a modelling technique that was originally created to aid, say, the development of (low-level) software sub-systems might not be suitable for describing the higher levels of the system of interest, which include features from other engineering disciplines. Also, the intended audiences for the highest levels of abstraction might not always be from an engineering background, and will therefore need something that is easy and intuitive to understand and that contains few "hidden" semantics.

Architectures describe the fixed parts of the structure of the system of interest and its sub-systems. They range from a high-level structure of a system of interest that all users need to understand to some degree, to the low-level structure of a (complex) piece of software which only needs to be understood by its developers and maintainers. In fact, during the development of a large and complex system, architectures might be used at a number of different levels, each containing different levels of detail [1].

The situations described above are extended when a system architecture is not used to describe a single system of interest, but a class of systems from which any particular deployment might take some, but not all, of the components. This is called a system framework architecture.

This Technical Report describes a process for the first four stages stated above, using a sequence of models that have been developed specifically to aid in the creation of large complex systems of interest [1] and is proven in use. It is also the process and models that underlie the European ITS Framework Architecture [2], which has already formed the basis of a number of regional, national and trans-national ITS architectures, as well as architectures for individual projects.

4.3 ITS architecture development approaches

Several approaches to architecture development have been tried during the last 20-30 years. In essence, all of them are trying to capture the mental model that system designers create when asked to design a system. As these systems have become more and more complex, they have often needed to be partitioned into sub-systems, or for the required functionality to be provided through the creation of several separate systems with what is hoped to be fully agreed communications between them.

Once communications have been agreed between systems, it usually means that they are said to be "interoperable". This is required to enable systems to deliver the same services across national and other geographical boundaries and to enable individual systems, sub-systems and components to be replaced without adversely affecting system operation or performance. The probability of systems communicating with each other successfully can be improved through the use of system architectures, as these can be used to ensure that communicating systems are not based on conflicting assumptions.

ITS has not escaped this move towards the need for high degrees of complexity and interoperability. Indeed, ITS, by their very nature, are highly complex and, in many cases, interdependent. This has resulted in the adoption of formalized approaches to the development of ITS system architectures, or "ITS architectures" as they are usually called.

At first the ITS architecture development approaches were manual, but in time and as a result of the added complexity already mentioned, they have become computer based, with some levels of computer assistance also being introduced. At the moment, there are two main approaches in use: the object-oriented (OO) approach, usually using UML, and the process-oriented method (POM), also known as data flow modelling. Both have their advocates and detractors and both have computer-based tools to aid in their use.

Proponents of UML will quite rightly claim their differences and in some circumstances advantages over POM, and it is not the objective of this Technical Report to claim preference for POM, but simply to say that in many cases it is a suitable technique and to consider the ways of using it most effectively.

5 The process-oriented model

5.1 General

Before beginning to understand the process-oriented model, is important to understand its objectives and the limitations of the claims that are being made for it. They are as follows:

a) It is an effective model to describe a high-level ITS framework architecture, and the ITS architectures that are derived from it.

b) It is anticipated that it will be used in the early part of a hierarchy of requirements and designs. In particular, it is anticipated that (parts of) a resulting ITS architecture will be used to produce component and infrastructure specifications for suppliers to design and build (using their own in-house methods and modelling techniques).

c) An ITS framework architecture might not be complete. It might be necessary to add items to create a defined ITS architecture to satisfy its stakeholders.

d) The model is technology-independent, i.e. the decision as to how to implement the components and infrastructure is left to the suppliers (subject to any restrictions imposed by the purchaser).

The model comprises the following elements:

— Stakeholder aspirations

— Stakeholder needs

— Functional viewpoint

  — Context diagram

  — Data flow diagrams

  — Control model

— Physical viewpoint

  — Context diagram (same as that for the functional viewpoint)

  — Sub-system specifications

  — Module specifications (optional — depending upon system complexity)

— Communications viewpoint

5.2 Stakeholder aspirations

These are unstructured statements that express the problems that need to be addressed and the desires of the various stakeholders. Ideally, they should be written by the stakeholders, but in practice they normally need guidance from their ITS architects. A stakeholder is any person that affects, or is affected by, either directly or indirectly, the system under consideration. They can be categorized into four generic groups, as follows:

— *Want ITS* — These are the problem owners. Their problems may be concerned with traffic and/or transport, and they might be authorities that need to improve the transport environment of their political masters, network owners or operators, or travellers who wish to improve their travelling experience.

— *Make ITS* — These are solution providers. They comprise the component manufacturers and the system integrators.

— *Use ITS* — These are the travellers, system operators and service providers that will come into direct contact with the systems and use them to solve their problems.

— *Rule ITS* — These are the authorities that provide the legislative framework within which the solutions will be created and used. They also include the creators of standards.

Some stakeholders, such as service providers, can be part of more than one category.

Aspirations are written in the stakeholders' own words. Thus they are likely to be unstructured, as can be seen in the examples given in Figure 1.

There is a need for the TMC to be able to include bus information (e.g. bus service delays), especially during inclement weather (e.g. service suspended)

There is a need to develop pre-trip and on-trip traveller information systems to assist the traveller in making journey planning decisions, including the use of VMS, Internet and CCTV.

There is a need to improve the integration between the different modes of transport by providing improved cycle and pedestrian networks, through ticketing, and developing transport interchanges.

Figure 1 — Examples of stakeholder aspirations
(Reproduced by kind permission of the FRAME Forum)

When working with stakeholders, it is often useful to give them a starting point. This can be provided by using some or all of the high-level service descriptions provided in ISO 14813-1 [3] as the basis for a discussion of the services that the stakeholders could consider providing.

5.3 Stakeholder needs

Sometimes called user needs, stakeholder needs are structured statements that form the system requirements of the ITS architecture. They should be written using simple formal statements that specify a function or a feature that should be provided by the final ITS implementation. Stakeholder needs should have a formal structure, usually provided by a numbering system.

This numbering system should reflect the range of services that the ITS is expected to support, with stakeholder needs for the same or similar services being put within the same numerical sequences. If a new set of stakeholder needs is being created, then useful starting points for this structure are again provided by the list of services in ISO 14813-1 [3], but also by the list of European ITS user needs available from the FRAME website [2], or the US National ITS Architecture user services available from its website [4].

Stakeholder needs are usually written in what is called "shall" language because each sentence starts with the words "The system shall …" for each feature that is required for a given application or service. If a feature is optional in the sense that a valid ITS architecture for that application or service can be created without it, then the stakeholder need starts with the phrase: "The system shall be able to …" or "The system shall enable …" All features mentioned in one stakeholder need are to be implemented at the same time. Those features that can be omitted if not required by the ITS architecture for a particular ITS implementation must be in separate stakeholder needs.

As part of the ITS architecture creation process, the ITS architect translates the stakeholder aspirations into stakeholder needs using the format and structure just described. This is illustrated by the example in Figure 2.

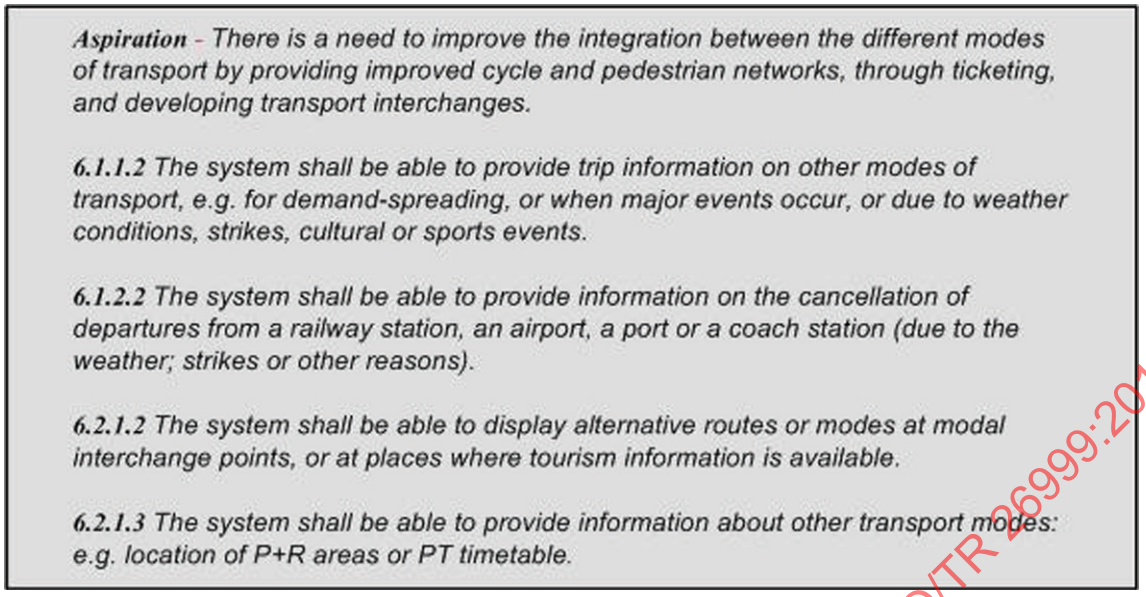


Figure 2 — Examples of stakeholder needs mapped to a stakeholder aspiration
(Reproduced by kind permission of the FRAME Forum)

In order to save time and resources in the ITS architecture creation process, it is better to try and map the stakeholder aspirations to an existing set of stakeholder needs. These can provided either by the list of European ITS user needs that is included in the European ITS Framework Architecture [2], or by the user services that have been created for the US National ITS Architecture [4]. Both of these are written in "shall" language, but have slightly different styles of writing, in particular the sets of sequences of stakeholder needs in the list of European ITS user needs are called "groups", while those in the US National ITS Architecture are called "user service bundles".

The selection of the correct starting point (European or US) will depend on which most closely fits the services that the ITS architecture is to support. However, whichever choice is made, it is always possible that no mapping will be found between some of the stakeholder aspirations and either the European or the US set of stakeholder needs. In this case, new stakeholder needs will have to be created, using the appropriate European or US format.

5.4 Functional viewpoint

5.4.1 General

This subclause describes the contents of the functional viewpoint, sometimes called the functional (or logical) architecture. Its contents comprise functions, data stores and terminators, which are linked together by data flows. The relationships between these components are illustrated using data flow diagrams.

5.4.2 Context diagram

One of the most important parts of a functional viewpoint is the context diagram. It defines the system as a single entity but identifies all of the external entities with which the system must communicate. Thus it is used to show the boundary between the system and the outside world. An example of a simple context diagram is shown in Figure 3.

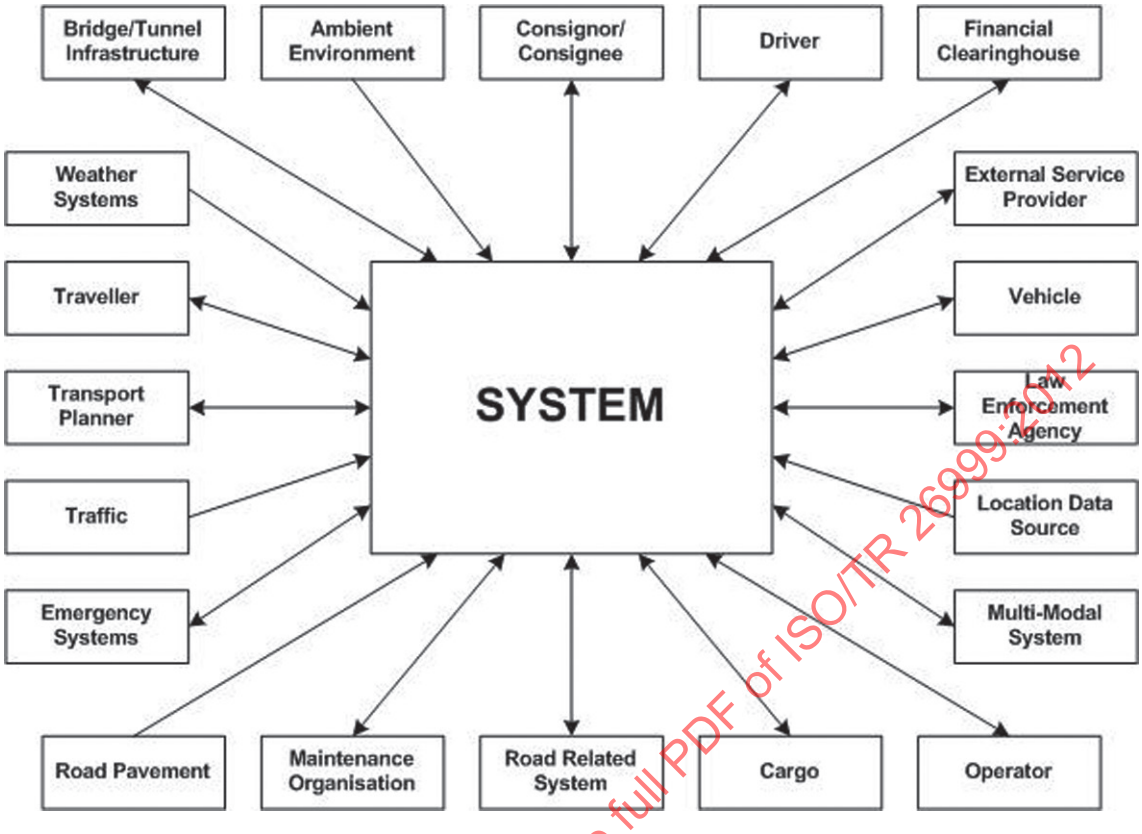


Figure 3 — An example of a context diagram
(Reproduced by kind permission of the FRAME Forum)

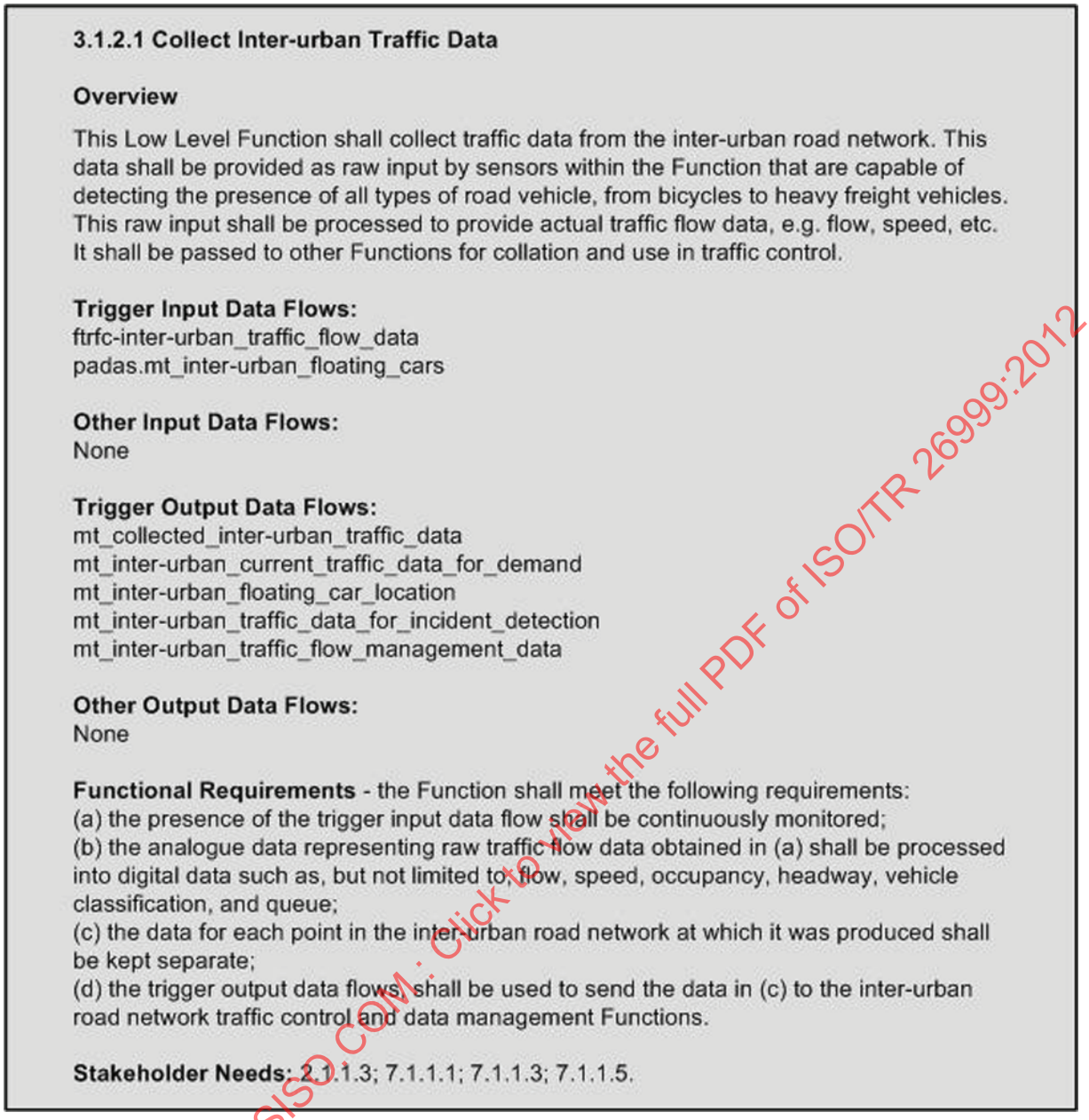
The external entities (other systems or persons) are called "terminators". They can be generic as shown above, or be created as individuals for the various forms of each terminator. This latter concept is illustrated in the more complex context diagram for the US National ITS Architecture — see http://itsarch.iteris.com/itsarch/html/pspec/dfdiagramcontext.htm. In Figure 3, the detail of some of the terminators is provided by the use of actors. For example the "driver" terminator can have actors to represent drivers of different types of vehicle, e.g. private car, heavy goods vehicle, public transport vehicle. In the US National ITS Architecture each of these actors is shown as a separate terminator.

5.4.3 Functions

In its simplest form, a function defines what needs to be done to create the required outputs to a terminator from a given set of terminator inputs. However, it is very unusual for this to be done with one function, and so there might be several functions between the input terminator(s) and the output terminator(s). Each one will perform part of the necessary processing.

All functions should be individually numbered and have a name. The name should summarize what the function does in a simple short statement that usually begins with a verb, e.g. "Provide operator interface", "Collect toll". The scheme used for the function numbering will be important for the construction of functional hierarchies — see later.

The description of each function should be divided into four parts. These comprise an overview, lists of input and output data flows, detailed functional requirements (what the function actually does) and a list of the stakeholder needs that are being served. The stakeholder needs associated with each function are determined when it is created. There is no exact science to define how many stakeholder needs should be related to one function, as it will depend on what is in the stakeholder needs and the complexity of the functionality required to fulfil them. A typical example of a function description is shown in Figure 4 (note that trigger flows are described in 5.4.7).



3.1.2.1 Collect Inter-urban Traffic Data

Overview

This Low Level Function shall collect traffic data from the inter-urban road network. This data shall be provided as raw input by sensors within the Function that are capable of detecting the presence of all types of road vehicle, from bicycles to heavy freight vehicles. This raw input shall be processed to provide actual traffic flow data, e.g. flow, speed, etc. It shall be passed to other Functions for collation and use in traffic control.

Trigger Input Data Flows:
ftrfc-inter-urban_traffic_flow_data
padas.mt_inter-urban_floating_cars

Other Input Data Flows:
None

Trigger Output Data Flows:
mt_collected_inter-urban_traffic_data
mt_inter-urban_current_traffic_data_for_demand
mt_inter-urban_floating_car_location
mt_inter-urban_traffic_data_for_incident_detection
mt_inter-urban_traffic_flow_management_data

Other Output Data Flows:
None

Functional Requirements - the Function shall meet the following requirements:
(a) the presence of the trigger input data flow shall be continuously monitored;
(b) the analogue data representing raw traffic flow data obtained in (a) shall be processed into digital data such as, but not limited to, flow, speed, occupancy, headway, vehicle classification, and queue;
(c) the data for each point in the inter-urban road network at which it was produced shall be kept separate;
(d) the trigger output data flows shall be used to send the data in (c) to the inter-urban road network traffic control and data management Functions.

Stakeholder Needs: 2.1.1.3; 7.1.1.1; 7.1.1.3; 7.1.1.5.

Figure 4 — An example of a function description
(Reproduced by kind permission of the FRAME Forum)

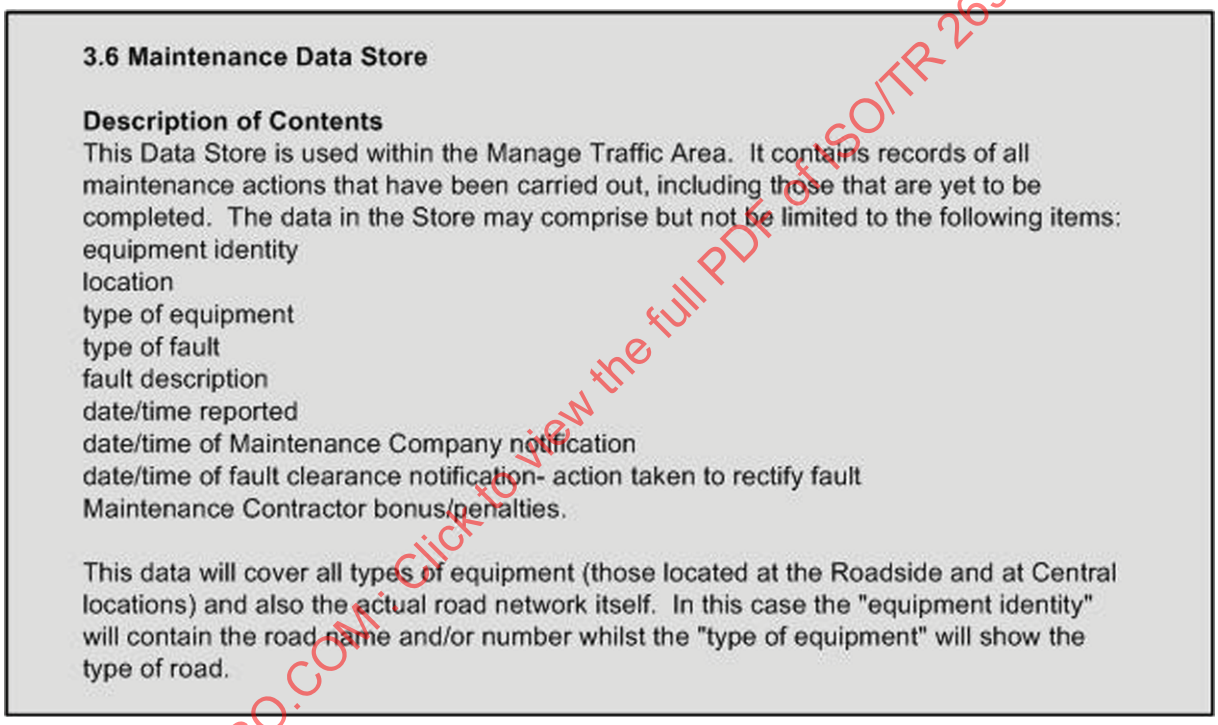
When functions are newly created, it is very easy to have a few of them related to a large number of stakeholder needs. This can make the functions very complex and contain a lot of functionality, which can sometimes cause problems for the creation of the physical viewpoint. Large or complex functions are therefore usually split up into smaller and simpler functions so that each one contains part of the total functionality required to fulfil a collection of stakeholder needs. To help with understanding how these functions relate to each other, they are structured into a functional hierarchy in which the original large or complex function only has an "overview" description and is called a "high-level function". The new smaller and less complex functions are its components, each having a full description as shown in Figure 4 and being called "low-level functions". The functions at the very highest level in the hierarchy are usually referred to as "functional areas" and there might be several of these in each ITS architecture, each containing many functions, some of which might themselves be high-level functions with component low-level functions.

In the example of a function description shown in Figure 4, the function 3.1.2.1 is in functional area 3. It is part of a high-level function, 3.1.2, which is itself a component of high-level function 3.1. Most, if not all, of the ITS architectures using the POM methodology structure their functionality in this way.

5.4.4 Data stores

Data stores are used to hold data that need to be stored for a length of time, and then used by one or more functions. Data stores should be named and numbered, with the name indicating the type of data it contains and provided with a description containing details of what data are in the store.

It is important to note that the term "data store" is not necessarily synonymous with a database, nor is it intended to imply the use of any particular physical design or storage methodology. But it does identify data that either needs to be accessed by several functions, or whose storage must be explicitly identified. Taken together, the data store descriptions will provide the information viewpoint component of the architecture. An example of a data store description is shown in Figure 5.



3.6 Maintenance Data Store

Description of Contents
This Data Store is used within the Manage Traffic Area. It contains records of all maintenance actions that have been carried out, including those that are yet to be completed. The data in the Store may comprise but not be limited to the following items:
equipment identity
location
type of equipment
type of fault
fault description
date/time reported
date/time of Maintenance Company notification
date/time of fault clearance notification- action taken to rectify fault
Maintenance Contractor bonus/penalties.

This data will cover all types of equipment (those located at the Roadside and at Central locations) and also the actual road network itself. In this case the "equipment identity" will contain the road name and/or number whilst the "type of equipment" will show the type of road.

Figure 5 — An example of a data store description
(Reproduced by kind permission of the FRAME Forum)

5.4.5 Functional data flows

Functional data flows are used to provide data links between functions, data stores and terminators. They should be uni-directional only in a functional viewpoint, though they are sometimes shown as being bi-directional in a context diagram — see the example in Figure 3.

Each functional data flow should be provided with a description, which provides a high-level description of the data it contains. An example of a data flow description is shown in Figure 6.

mt_inter-urban_to_urban_data **Trigger Flow**

It contains data that is being transferred from the traffic management functions for Inter-urban roads to those for Urban roads. The data flow consists of the following constituent data flows each of which has its own definition:

mt_inter-urban_to_urban_pred&strat
+ mt_inter-urban_to_urban_traffic_commands
+ mt_inter-urban_to_urban_traffic_data_transfers

Figure 6 — An example of a functional data flow description
(Reproduced by kind permission of the FRAME Forum)

5.4.6 Data flow diagrams

Data flow diagrams (DFDs) are used to illustrate the way that the functions and data stores used to satisfy the stakeholder needs communicate with each other and with the terminators/actors using the functional data flows. Each DFD actually illustrates what is in a high-level function, and at the highest level shows what is in each functional area.

The structure of the DFDs follows the hierarchical structure of the functions, but with the addition of a top-level diagram. This is called a "system diagram" and it shows the data flows that link the functional areas together. An example of this type of diagram is shown in Figure 7.

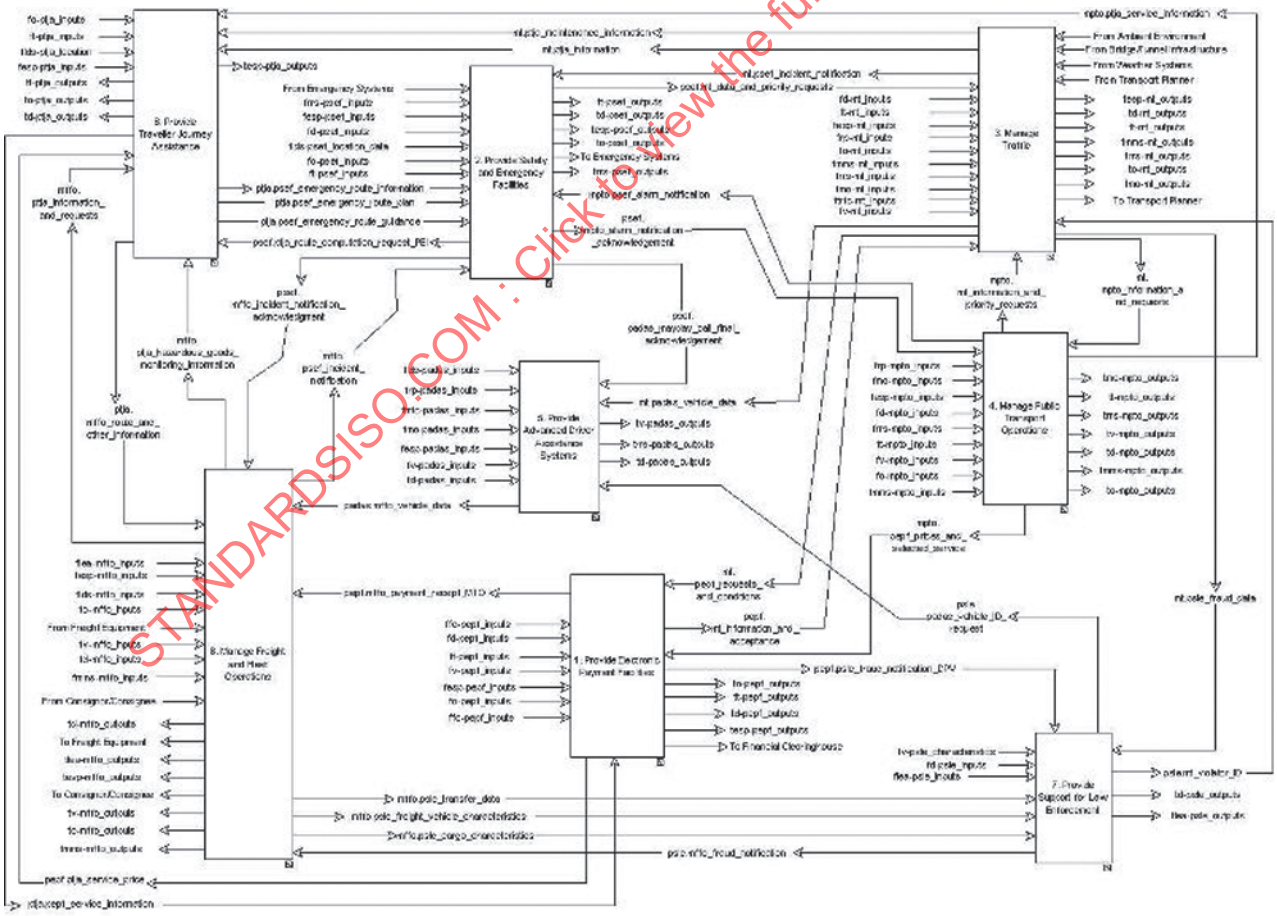


Figure 7 — An example of a system diagram
(Reproduced by kind permission of the FRAME Forum)

NOTE The diagram shown in Figure 7 is a reproduction of a full system diagram which is normally printed on A3 size paper. A full-sized example of such a diagram can be seen by opening the "Functional Area Diagram" in the FRAME Browsing Tool. This tool can be downloaded, by following the instructions in the "Installation" section, from the FRAME website: http://www.frame-online.net/the-architecture/browsing-tool.html.

In the example of a system diagram shown in Figure 7, each of the functional areas is illustrated as a single box. Each of these is also represented by its own DFD, which in turn might contain high-level functions that are also illustrated by their own DFDs. Thus each DFD shows an increasing level of detail, which makes it easier for users of the ITS architecture to understand its contents. An example of the DFD for a high-level function is shown in Figure 8.

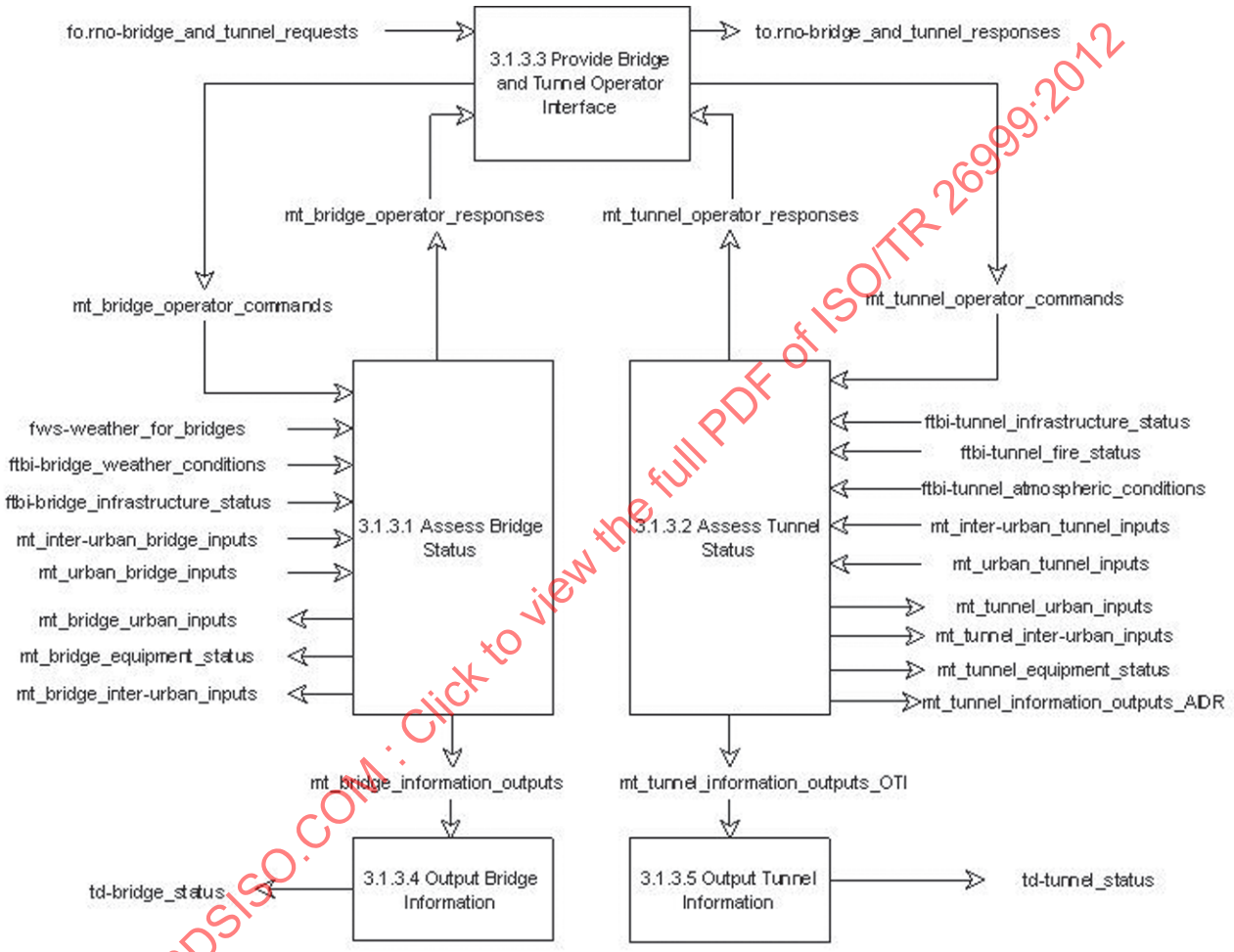


Figure 8 — An example of a (lowest-level) data flow diagram
(Reproduced by kind permission of the FRAME Forum)

The DFD is one of the parts of the architecture creation process that has benefited greatly from the introduction of computer-based tools. The actual appearance of the DFDs provided by these tools will depend upon the facilities they have to show functions, data stores, data flows and terminators/actors. Thus the DFD shown in Figure 8 should be treated as an example of how a DFD *can* appear, rather than the way that all DFDs *should* appear. An alternative representation of the way that a DFD can appear is provided by the system diagram for the US National ITS Architecture which can be found at: http://www.iteris.com/itsarch/html/pspec/dfdiagram0.htm.

It is not always necessary to draw the data flow diagrams for the functional viewpoint. This particularly applies where the ITS architecture is being created from an existing architecture with the objective of producing a physical viewpoint — see later subclauses.

5.4.7 Control

A functional viewpoint might also contain a control model, which can be explicit, e.g. a finite-state machine, or implicit by assuming a function will execute whenever it has sufficient data to do so properly. Alternatively trigger flows may be used, as shown in Figure 4. These are special forms of input data flow which, when received by a function, will cause it to do something. In this case, the functional requirements will be written to say that the function will not do anything until a trigger flow has been received. This can be either to start the function or to carry out a specific part of its functionality.

5.5 Physical viewpoint

5.5.1 General

This subclause describes the contents of the physical viewpoint, sometimes called the physical architecture. Its contents comprise sub-systems and (optionally) modules, which are linked together with physical data flows. The relationships between these components are illustrated using diagrams.

5.5.2 Context diagram

A physical viewpoint has the same context diagram as the functional viewpoint because it is the same system that is being shown. It will therefore appear identical to the diagram shown in Figure 3. The physical viewpoint provides a "physical" view of the system instead of the "functional" view provided by the functional viewpoint.

5.5.3 Sub-systems and modules

A separate sub-system is created for each physical location that will be used in the ITS implementation. Each sub-system is allocated some of the functions and data stores from the functional viewpoint. The way that the allocation is made depends on the actual physical locations where it is considered desirable to place the various parts of the system functionality. Thus the sub-systems show the distribution of functional units in the different physical locations that are to be used for the ITS implementation.

Sub-systems may be subdivided into modules, if desired. This can be to enable a program of phased deployment for different services being provided by one ITS implementation, or for the identification of the components that deliver a particular service when they are within different sub-systems.

Physical data flows are used to show the communications between the sub-systems and modules and the communications that they have with the outside world through the terminators/actors shown in the context diagram. These physical data flows are combinations of one or more of the functional data flows created in the functional viewpoint.

The sub-system and module descriptions are created from those for their constituent functions and data stores. These descriptions are then used to provide the component specifications that are used for the procurement of the physical items that will be required for the ITS implementation to deliver the services that the stakeholders identified in their stakeholder aspirations.

In the physical viewpoint (physical architecture) for the US National ITS Architecture [4], an alternative structure is used. In this structure, modules are replaced by equipment packages which are "deployment-sized pieces" of functionality within sub-systems. The US architecture also has what it calls "market packages", which include functionality that addresses specific services, such as surface street control, interactive traveller information. The market packages collect together several different sub-systems, equipment packages, terminators and physical data flows that together can provide a particular service.

5.5.4 Physical viewpoint diagrams

The structure of the sub-systems and modules within a physical viewpoint are best illustrated using one or more diagrams. They are used to show the communications (physical data flows) between sub-systems and modules and between these and the terminators/actors. As with the DFDs, the physical

viewpoint diagrams can be structured in a hierarchical manner, starting with a diagram showing the different sub-systems. An example of this type of diagram is shown in Figure 9.

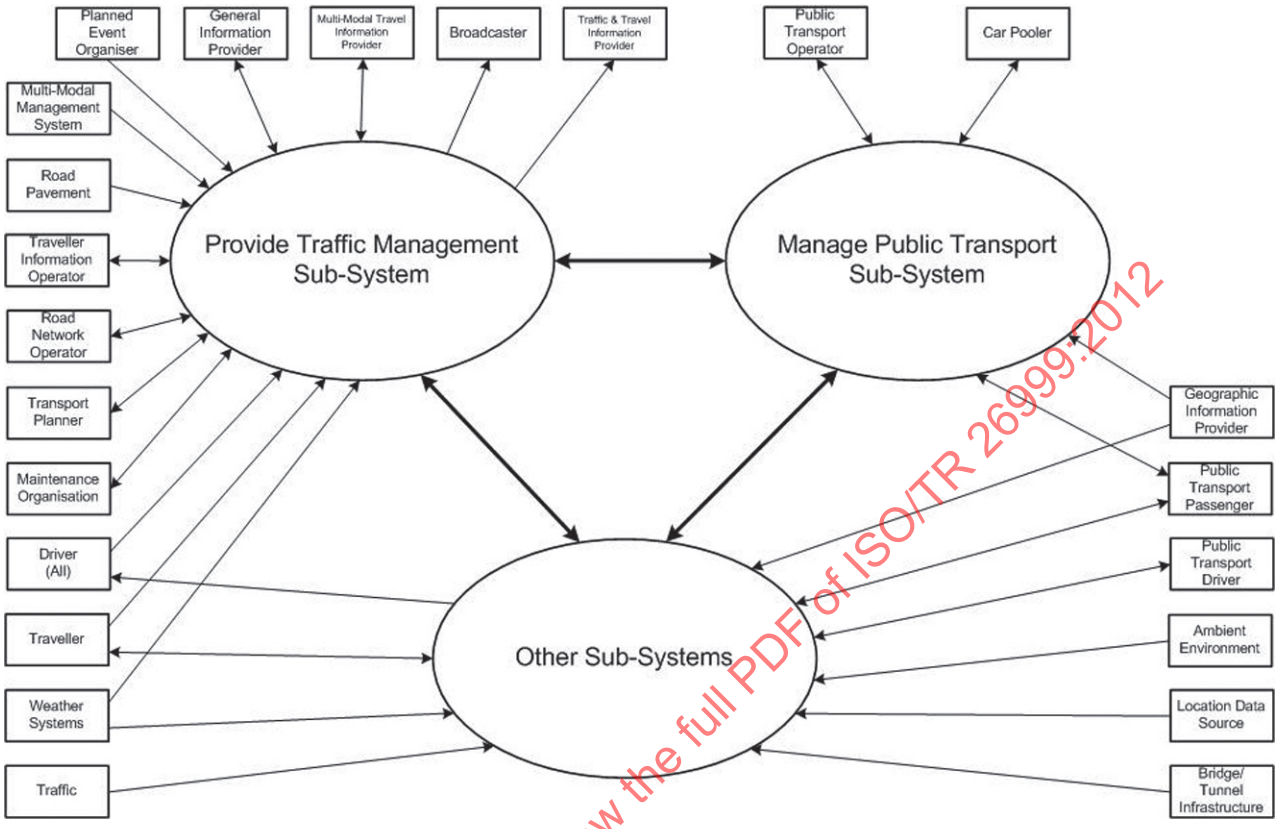


Figure 9 — An example of a top-level physical viewpoint diagram
(Reproduced by kind permission of Kent County Council, UK)

For every sub-system that has constituent modules, separate diagrams are then produced to show any data flows that provide links between the modules. These diagrams also show in more detail how the data flows shown in the top-level diagram link to the individual modules.

The top-level physical viewpoint (physical architecture) diagram produced for the US National ITS Architecture can be found at its website [4]. It is often called the "sausage diagram" due to the way that its sub-systems are illustrated. The US architecture also provides lower-level diagrams that show separate diagrams for each of the sub-systems, the physical data flows that link them to other sub-systems and those that link them to terminators.

The two types of sub-system diagram provided by the US architecture are very useful because they enable the communications for a particular sub-system to be clearly seen. Similar diagrams can also be produced to show the physical data flows used by individual modules. An example of this type of diagram is shown in Figure 10.

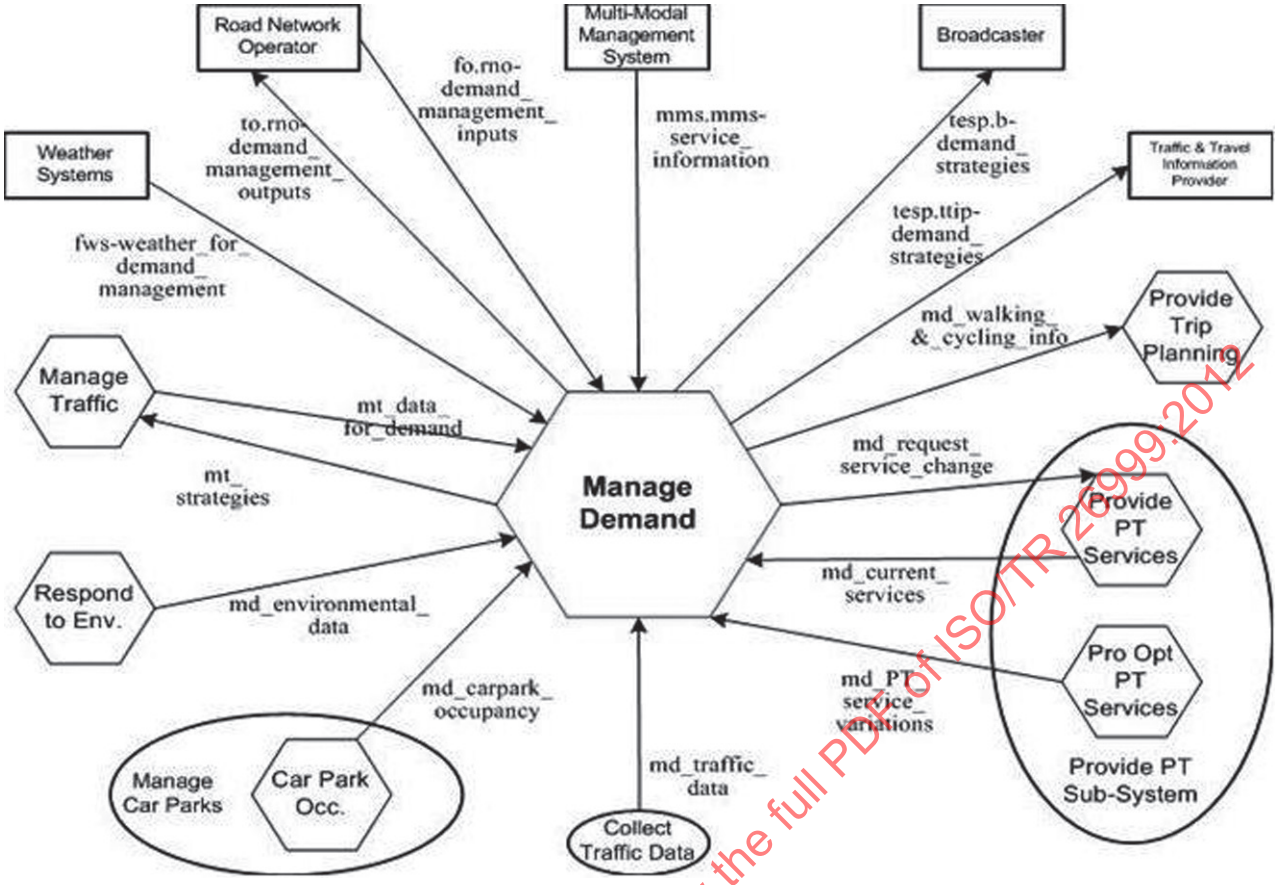


Figure 10 — An example of a physical viewpoint diagram for a module
(Reproduced by kind permission of Kent County Council, UK)

5.6 Communications viewpoint

The content and characteristics of the physical data flows in the physical viewpoint depend on two things. Firstly, how much data are being transferred between the functions in the different sub-systems and modules created in the physical viewpoint and, secondly, the characteristics of the data transfer itself. These characteristics comprise information such as how often the data transfer takes place, how soon it is needed, security considerations, privacy considerations, plus physical attributes such as whether one of the sub-systems or modules is mobile, i.e. mounted on a vehicle or carried by a traveller.

This information provides the top-level requirements for the communications links between the sub-systems and the modules. These requirements are used to form the communications viewpoint, and are utilized to calculate such things as the bandwidth that the communications links will require. All of the information that is thus provided by the communications viewpoint can then be used to produce the infrastructure specifications for the procurement of the communications infrastructure that will be required for the ITS implementation. An example of the way that the content and characteristics of the physical data flows can be defined in the communications viewpoint is shown by the example in Figure 11.

Physical Link

Driver (All) - Provide Traffic Management SS

Direction	
From	**To**
Driver (All)	Provide Traffic Management SS

Physical Data Flow	Consituent Functional Data Flows	Data Type	Max Bytes / Message	Delay Time *(secs)*	Message Interval *(secs)*	Transfer Mode	Security Level	*Data Rate*
mi_incident_warnings	mt_urban_incident_warning_commands	Raw Data	1000	1	15			1000
mt_traffic_commands	mt_urban_lane_commands	Raw Data	100	0.5	15			200
	mt_urban_traffic_management_requests	Raw Data	100	1	5			100
	Minimum Data Rate for link	1300	**Bytes/Second**			Wire Line	None	
	Minimum Message Gap	5	**Seconds**					

Direction	
From	**To**
Provide Traffic Management SS	Driver (All)

Physical Data Flow	Consituent Functional Data Flows	Data Type	Bytes / Message	Delay Time *(secs)*	Message Interval *(secs)*	Transfer Mode	Security Level	*Data Rate*
mem_equipment_status	mt_urban_equipment_status	RawData	100	5	60			20
mt_traffic_responses	mt_urban_traffic_management_responses	RawData	100	0.5	15			200
	Minimum Data Rate for link	220	**Bytes/Second**			Wire Line	None	
	Minimum Message Gap	15	**Seconds**					

Figure 11 — An example of communications link characteristics provided in the communications viewpoint (Reproduced by kind permission of Kent County Council, UK)

The level of abstraction associated with the high-level ITS architectures for which the POM methodology is best suited does not require the definition of the communications technology to be used. In fact, it is often highly desirable not to include requirements to use a particular technology within the communications viewpoint, so that the eventual suppliers have the freedom to offer what is most appropriate. This also has the advantage that it gives the "design authority" to the supplier (where the most competence should lie) rather than to the system architect or system owner who might not have the best and/or most up-to-date knowledge about which communications technology is best suited to particular ITS implementations.

6 Other parts of the description of an ITS architecture

6.1 General

There are several other parts of the description of an ITS architecture created using the process-oriented model apart from those described in the previous clause. Some of them are included in documentation that is separate from that used for the other descriptions.

6.2 Identification and version

The identification and version of an ITS architecture might include such things as its name and a hint at the reason(s) for its creation. So for example the latest version of the European ITS Framework (FRAME) Architecture is "v4 — Cooperative ITS Extended". This tells the user the version number and the reason for its creation, i.e. to support cooperative systems in ITS through an extension to the previous version.

6.3 System stakeholder identification

The stakeholders are often identified in a separate document that might also include a description of what they want (their aspirations) from the ITS implementation that the ITS architecture is to represent. As well as describing each stakeholder and the type of person/organization that they are, such a document

might also describe the part that each stakeholder is either expected, or hoping, to play in the eventual implementation of the ITS services that the functionality in the ITS architecture is to support.

6.4 Viewpoint overviews

If necessary, a document can be created that provides a high-level description of each viewpoint, the reason(s) for its creation and its expected use. The document may also describe the contribution each viewpoint is expected to make to the other viewpoints and to the eventual ITS implementation that is to be described by the ITS architecture.

6.5 Other information

It might be necessary to produce a document that describes the assumptions and choices that have been made during the creation of the ITS architecture. Any known inconsistencies among the architecture constituents may also be recorded, with their impact(s) on the eventual ITS implementation.

7 Types of ITS architecture

7.1 General

Within the process-oriented model, it is possible to create different types of ITS architecture. There are three different types, comprising "framework", "defined" and "specific". The relationship between them is shown in Figure 12.

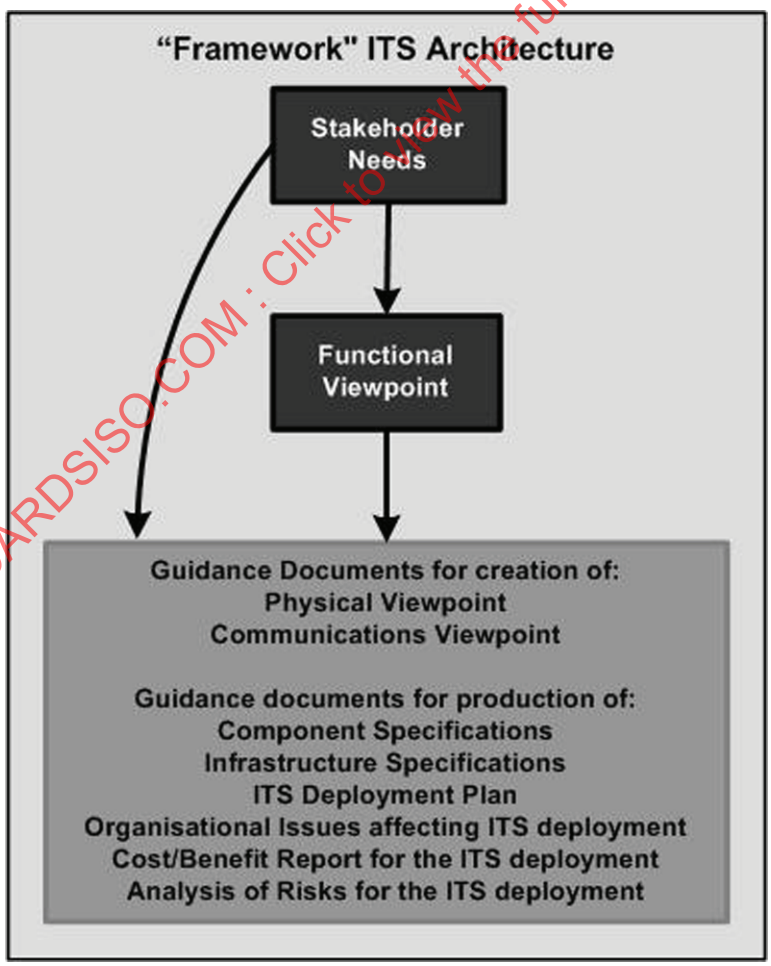


Figure 12 — Overview of a framework ITS architecture
(Reproduced by kind permission of the FRAME Forum)

7.2 Framework ITS architecture

This type of ITS architecture provides the most general and flexible approach to its creation. It is used to capture what needs to be done to satisfy an ITS "vision". This is a set of statements about the services that it is envisaged that ITS should support in the future without mandating how they should be applied. A framework ITS architecture just comprises stakeholder needs that provide a formal description of the ITS "vision" and a functional viewpoint, as shown in Figure 12. It will probably also have some of the supporting documents showing how the ITS architecture can be used and giving examples of its use. This type of ITS architecture cannot therefore be used directly for ITS implementations and needs to be developed into another type of ITS architecture. Thus a framework ITS architecture is the base-level architecture from which all other types of architecture can be derived, usually using an architecture-specific tool.

7.3 Defined ITS architecture

A defined type of ITS architecture is created from a framework ITS architecture. It contains stakeholder needs and a functional viewpoint that may be a sub-set of those in the framework ITS architecture. The chosen stakeholder needs are derived from stakeholder aspirations for the services that are to be delivered by the particular ITS implementation that the defined ITS architecture is to support. Clearly the stakeholder aspirations must be based on the ITS "vision" used to create the framework ITS architecture from which the defined ITS architecture is to be created. A defined ITS architecture also contains physical and communications viewpoints, plus those of the ITS architecture outputs which are required and that can be produced from them, as shown in the lower part of Figure 13.

The physical viewpoint is often created using a computer-based "tool" to aid the process and to provide some form of consistency checking. The creation of each of the ITS architecture outputs shown in the lower part of Figure 13 is optional and depends on the ultimate use to which the defined ITS architecture is to be put. It is therefore unlikely that any computer-based tools will be available to help with this part of the work. It should be noted that the ITS risks, costs and benefits covered by the outputs apply to the ITS implementation and not to the ITS architecture itself.

A defined ITS architecture can be used directly as the starting point for the next steps in the ITS implementation, such as the procurement of systems, equipment and communications networks. Careful structuring of the ITS deployment plan will enable users of this type of ITS architecture to select sets of physical entities from the physical viewpoint so that phased deployments can be made, i.e. only particular parts of the physical viewpoint are to be deployed at any one time.

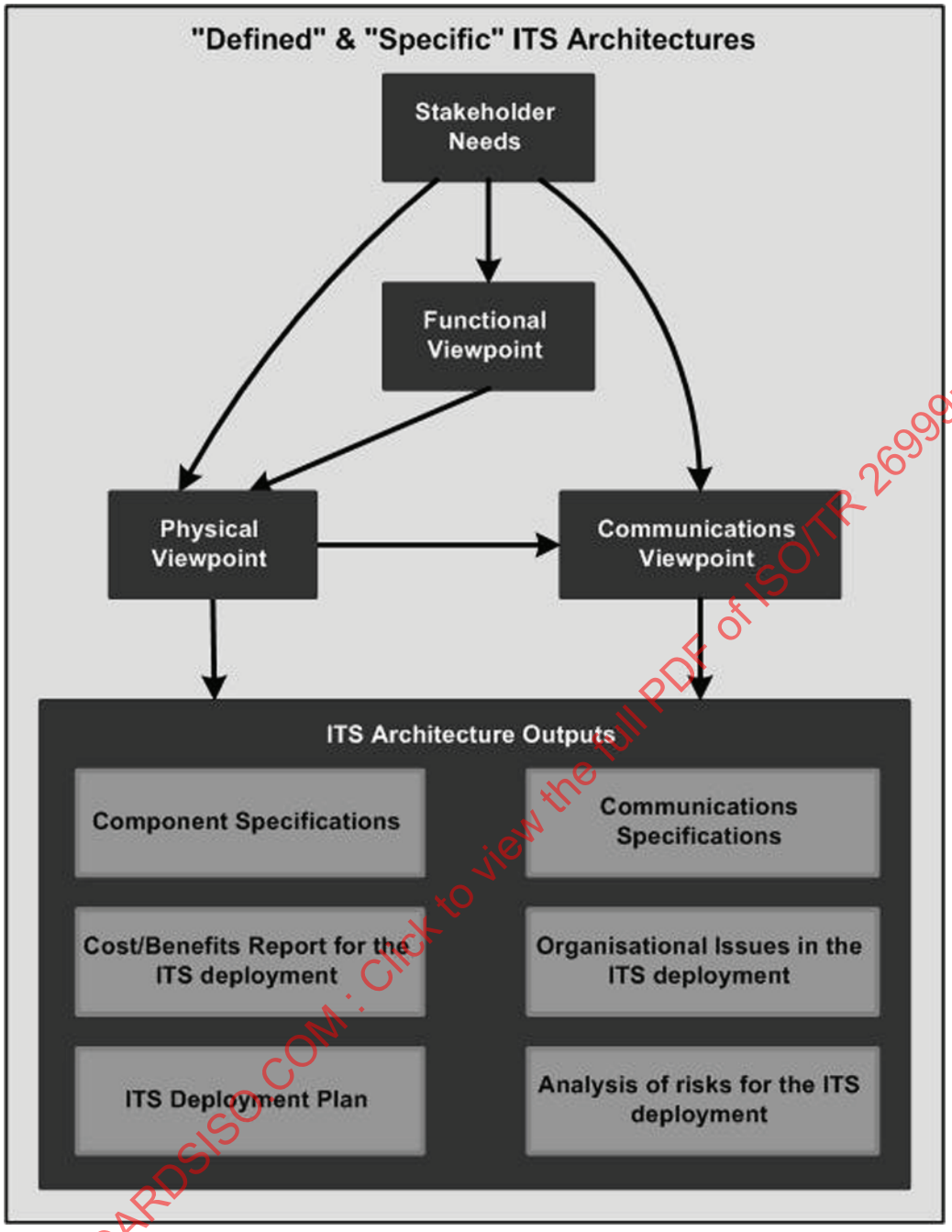


Figure 13 — Overview of a defined ITS architecture
(Reproduced by kind permission of the FRAME Forum)

7.4 Overloaded defined ITS architecture

An overloaded defined ITS architecture may also be created from a framework ITS architecture. It also contains stakeholder needs and a functional viewpoint that are a sub-set of those in the framework ITS architecture. The selection of the stakeholder needs depends on the stakeholder aspirations for the services that are to be delivered by the ITS implementation that the overloaded defined ITS architecture is to support. The stakeholder aspirations must themselves be based on the ITS "vision" used to create the framework ITS architecture. The distinctive feature of an overloaded defined ITS architecture is that options are provided as to how functions are grouped into physical viewpoint sub-systems and modules, i.e. the same function might be in more than one physical entity, though a user will only select one of those physical entities for a given deployment.

Thus an overloaded defined ITS architecture also contains a physical viewpoint and (possibly) a communications viewpoint plus (possibly) some of the ITS architecture outputs that can be produced from them, as shown in the lower half of Figure 13. It is more likely that some of the outputs (and possibly the communications viewpoint) will be replaced by guidance documents showing how they can be produced for the specific ITS architectures that might be created from an overloaded defined ITS architecture.

Users of this type of ITS architecture can select physical entities from the physical viewpoint to provide the services that are to be included for their particular ITS implementations. So, for example, it is possible to select traffic management services with different sets of physical entities dependent on what else the ITS implementation is to include. Thus one combination of physical entities will provide traffic management when it is combined with traveller information, but a slightly different combination will be needed when traffic management is combined with public transport management without any traveller information. The end result of the selection process is to create a specific ITS architecture for the particular ITS implementation.

Although the same flexibility can be achieved by creating specific ITS architectures from a framework ITS architecture, there are two advantages in starting from an overloaded defined ITS architecture. Firstly, because the physical entities already exist, it provides more control by the architecture owners and/or managers over the contents of the specific ITS architectures. Secondly, the architecture creators would have less work to do than if they started from a framework ITS architecture and created a defined ITS architecture.

The selection process used to create a specific ITS architecture from which the ITS implementation starts is usually made available to the users through the use of a computer-based tool. This provides a user interface for the selection process and can also provide a facility that enables some checks on logical consistency to be made.

7.5 Specific ITS architecture

A specific ITS architecture is similar to the defined ITS architecture but its creation process is different. A specific ITS architecture starts with a set of stakeholder aspirations for a given service, region, nation or project. These aspirations are then used to enable the selection of the required entities from the physical viewpoint in an overloaded defined ITS architecture to produce the physical viewpoint for the specific ITS architecture, usually with the use of a computer-based tool.

The communications viewpoint, plus some or all of the ITS architecture outputs that are shown at the bottom of Figure 13, will also be produced. Much of this work might have to be done without the use of a computer-based tool. Again, it should be noted that the ITS risks, costs and benefits covered by the outputs apply to the ITS implementation and not to the ITS architecture itself.

Like the defined ITS architecture, a specific ITS architecture can be used as the starting point for the next steps in the ITS implementation process, such as the procurement of systems, equipment and communications networks. Again, the deployment plan can be used to show the phasing of the component and communications deployment, since it might not be desirable or necessary to deploy everything at once.

7.6 Relationship between the types of ITS architecture

The relationships between the three types of ITS architecture described in the previous subclauses are illustrated by the diagram shown in Figure 14.