# E-FRAME

Extend FRAMEwork architecture for cooperative systems


WP300

## D15 – FRAME Architecture – Part 5: FRAME Architecture Methodology

FRAME Architecture Version 4.1

Dissemination Level

Public

SEVENTH FRAMEWORK PROGRAMME

E-FRAME is a Support Action funded by the European Commission, DG Information Society and Media in the 7th Framework Programme

European Commission
Information Society and Media


**Contract Number:**

FP7-ICT-2007.6.2 Nr. 224383

**Acronym:**

E-FRAME

**Title:**

Extend FRAMEwork architecture for cooperative systems

**Contractual date of delivery:**

August 2011

**Actual date of delivery:**

September 2011

**Main author(s) or editor(s):**

Richard Bossom (Siemens), Peter Jesty (PJCL)

**Other author(s):**

Angela Spence (MIZAR), Alexander Frötscher and Robert Ebner (ATE)

**List of Beneficiaries of the E-FRAME Project:**

| Beneficiary No. | Short Name | Participant name | Country |
|---|---|---|---|
| 1 | PJCL | Peter Jesty Consulting Limited | UK |
| 2 | Siemens | Siemens plc – Traffic Solutions Division | UK |
| 3 | ATE | AustriaTech - Federal Agency for technological Measures | AT |
| 4 | RWS-DVS | Rijkswaterstaat - Dienst Verkeer en Scheepvaart | NL |
| 5 | CTU | Czech Technical University in Prague | CZ |
| 6 | CERTU | Centre for Studies on Urban Planning Transport Utilities and Public Construction | FR |
| 7 | MIZAR | MIZAR Automazione | IT |


Version History:

| Version | Date | Main author(s) | Summary of changes |
|---|---|---|---|
| 0.1 | 23.02.2011 | Peter Jesty and Richard Bossom | First draft version based on a conference paper. |
| 0.2 | 02.05.2011 | Peter Jesty and Richard Bossom | Minor updates |
| 0.3 | 31.08.2011 | Peter Jesty and Richard Bossom | Version for review with changes for FRAME Architecture Version 4.1 |
| 1.0 | 08.09.2011 | Richard Bossom | Final Version for publication after internal review |

## Approval History:

| | Date | Name of author/reviewer | Version |
|---|---|---|---|
| Draft | 31.08.2011 | Richard Bossom | 0.3 |
| Internal reviewed | 07.09.2011 | Alexander Frötscher | 0.3 |
| Draft II | | | |
| External reviewed | | | |
| Reviewed Version | 08.09.2011 | Richard Bossom | 1.0 |
| Approved Version | 11.09.2011 | Peter Jesty | 1.0 |

## Table of Contents

## List of Figures

# Executive Summary

This document forms part of the FRAME Architecture deliverable (D15) that has been produced by the E-FRAME project. The deliverable consists of the following parts:

Part 1: Overview – provides an overview of the history of the FRAME Architecture, its contents and the uses that have been made of it;

Part 2: FRAME Browsing Tool – enables the contents of the FRAME Architecture to be viewed and is only available for downloading from the FRAME website at www.frame-online.net, in the Folder named The Architecture;

Part 3: FRAME Selection Tool Database – enables sub-set ITS architectures to be created through the use of the FRAME Selection Tool and is only available for downloading from the FRAME website at www.frame-online.net, in the Folder named The Architecture;

Part 4: FRAME Architecture Changes Document – describes the changes made to the FRAME Architecture since its previous version;

Part 5: The FRAME Methodology – this document.

Part 6: Function, Data Flow, Data Store and Terminator Descriptions – the Function, Data Flow, Data Store, plus Terminator and Actor descriptions taken from the Access Database used by the FRAME Selection Tool. (Note the same text will also appear in the FRAME Browsing Tool.

Parts 1, 2, 3, 4 and 6 will be updated every time a new version of the FRAME Architecture is produced. Part 5 should remain constant with each version of the Architecture and therefore not be updated.

Part 5 (this document) provides a description of the methodology that is behind the creation and use of the FRAME Architecture. The potential complexity and size of Intelligent Transport Systems (ITS) requires that they be implemented through a systems engineering approach based on the use of ITS Architectures. These enable a high level set of “views” of the proposed ITS to be obtained early in its lifecycle so that many of the details and implications can be checked and, if necessary, changed at significantly less cost than if the need for a change is only found when some/all of the development work has been completed. The FRAME Architecture has been created for use as the starting point for any ITS deployments, including those involving cooperative systems. A methodology for its use has been developed, which is supported by two FRAME Architecture Tools.

# 1 Introduction

## 1.1 The Aim of this Document

The aim of this deliverable document is to provide a description of the methodology that is behind the creation and use of the FRAME Architecture. It is intended to act as source of information that will help Architecture users to appreciate why its contents are arranged in a particular way and how it should be used. These issues are addressed by other parts of this deliverable – see section 1.4.

## 1.2 Assumptions behind this Document

It is assumed that readers will have some knowledge of the systems engineering principles and their use. Knowledge of what is in the FRAME Architecture would be helpful but is not essential and can be obtained from Part 1 of this deliverable.

## 1.3 Document Plan

This document has been organised into chapters including this one. Each of the subsequent chapters contains the following:

Chapter 2: Background – an introduction to ITS architectures and the ITS Lifecycle;

Chapter 3: Methodology – a description of the FRAME Architecture methodology;

Chapter 4: Tools – a brief description of the FRAME Tools and how they are used;

Chapter 5: Cooperative Systems – extending the FRAME Architecture to include support for the services that can be provided by cooperative systems;

Chapter 6: Conclusions.

References use in these chapters are provided in chapter 7.

## 1.4 Why is D15 in separate parts?

This E-FRAME project deliverable document (D15) has been divided into five parts, which are as follows:

Part 1: Overview – provides an overview of the history of the FRAME Architecture, its contents and the uses that have been made of it;

Part 2: FRAME Browsing Tool – enables the contents of the FRAME Architecture to be viewed and is only available for downloading from the FRAME website;

Part 3: FRAME Selection Tool Database – enables sub-set ITS architectures to be created through the use of the FRAME Selection Tool and is only available for downloading from the FRAME website;

Part 4: FRAME Architecture Changes Document – describes the changes made to the FRAME Architecture since its previous version.

Part 5: The FRAME Methodology – this document.

Parts 1, 2, 3 and 4 will be updated every time a new version of the FRAME Architecture is produced. Part 5 (this document) should remain constant with each version of the Architecture and therefore not be updated.

The alternative of providing completely separate deliverable documents was rejected because of the close linkage between what is in the Architecture and the methodology behind its use.

# 2 Background

## 2.1 Introduction

There are two main factors that have a profound impact on the success or otherwise of the implementation of Intelligent Transport Systems (ITS) services. The first is that the implementation of ITS services often requires the deployment of many components. These components are often provided by more that one supplier, and in some cases do not actually exist at the time of their specification. This means that some components will need to be developed, which adds a degree of uncertainty to the success of the implementation of the services. In addition, once the ITS services have been implemented the components may be owned and/or operated by different organisations, which usually need to have some level of interaction and co-operation.

The second factor that impacts on the implementation of ITS services is that the components must provide the services that the end users require in a consistent and coherent manner. Thus end users must be able to use the same services in the same way regardless of the part of the geographic area served by the ITS implementation in which they are delivered.

In order to achieve all these aims simultaneously and successfully, it is necessary to have a consistent high level view of how all the components fit together, and how they will all be managed. In addition the design and development of the components themselves will often need expertise from a number of different domain experts. The former is the subject of System Architecture (see reference 8(a)) and the latter of Systems Engineering (see reference 8(b)).

## 2.2 ITS Architectures

The need for Intelligent Transport System (ITS) Architectures was recognised in the early 1990's, when the number of possible applications and services that ITS could provide increased greatly. However, instead of producing unique architectures for each deployment, it was also realised that it would be much more efficient to have a Framework Architecture, from which individual ITS Architectures can be developed. The principal advantages of doing this are:

- It is quicker, and therefore cheaper, to produce a suitable ITS Architecture from a Framework Architecture.
- Each derived ITS Architecture has the same properties as the Framework Architecture from which it has been produced. This facilitates the use of similar components in different deployments, and thus extends their potential market.

The first Framework Architecture to be produced was the National ITS Architecture for the USA, which was first published in 1996 – see reference 8(c). This was followed in 2000 by the European ITS Framework Architecture produced by the European Commission funded project KAREN – see reference 8(d)[1]. Both architectures are under continuous review and development as the scope and content of ITS services, and the expectations of the stakeholders, change.

Although these two Framework Architectures have many similarities there are also a number of differences, the most prominent of which are:

- Certain services, in particular those connected with commercial freight vehicles, have very different objectives.
- The US National ITS Architecture shows the relationship between physical components, and users choose those components that they need to satisfy their requirements. However, the European ITS Framework Architecture shows the relationship between functions only, and users first choose those that they need to satisfy their requirements, and then make their own decisions as to how they will be allocated to physical components.

The reason for the first difference is obvious – some things are done quite differently on each side of the Atlantic! The reason for the second difference is that the same thing is sometimes done in different ways in different European Member States, and one way to ensure that they can all be modelled using the European ITS Framework Architecture is to permit the same functions to be put into different physical locations in different countries.

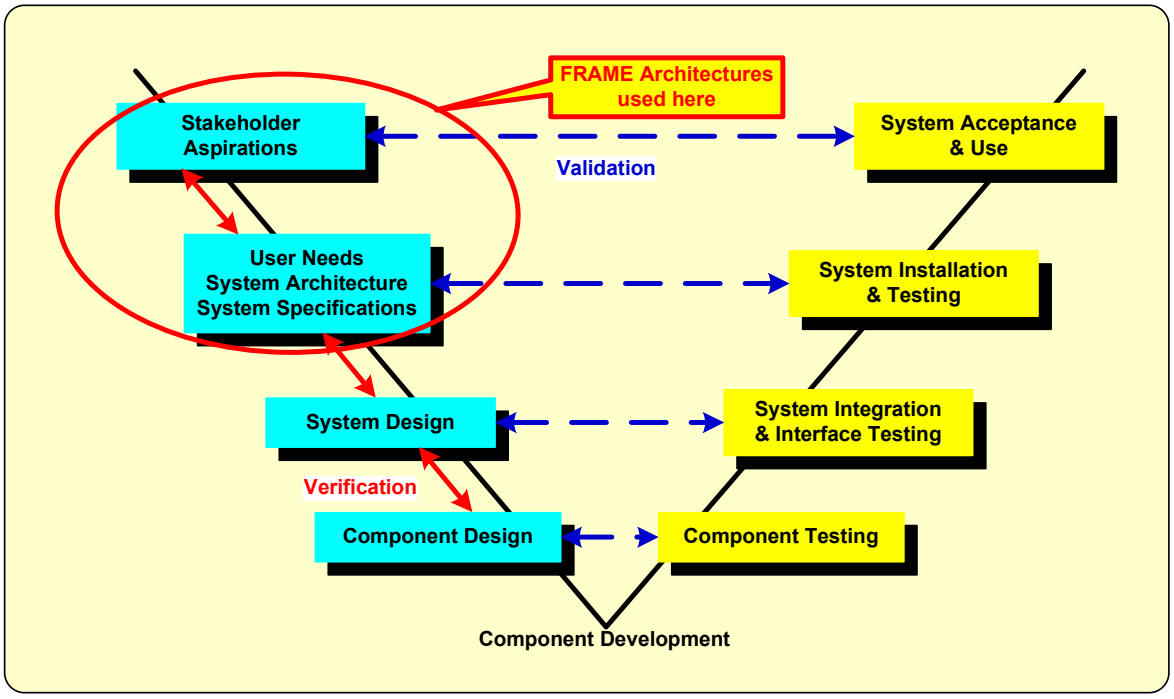
This document describes the use of the European ITS Framework Architecture, now known as the FRAME Architecture after the projects that have maintained it since 2001.

## 2.3 Intelligent Transport System Lifecycle

Systems engineering is the activity of specifying, designing, implementing, validating, deploying and maintaining socio-technical systems – see reference 8(e). The systems engineering process is often depicted using the V-model system lifecycle – see Figure 1 on the next page. This model emphasises the need to ensure that the system is both built correctly, and that it satisfies the aspirations of all its stakeholders.

[1] More information about the European ITS Framework Architecture (now called the FRAME Architecture) will be found in Part 1 of this E-FRAME project deliverable.

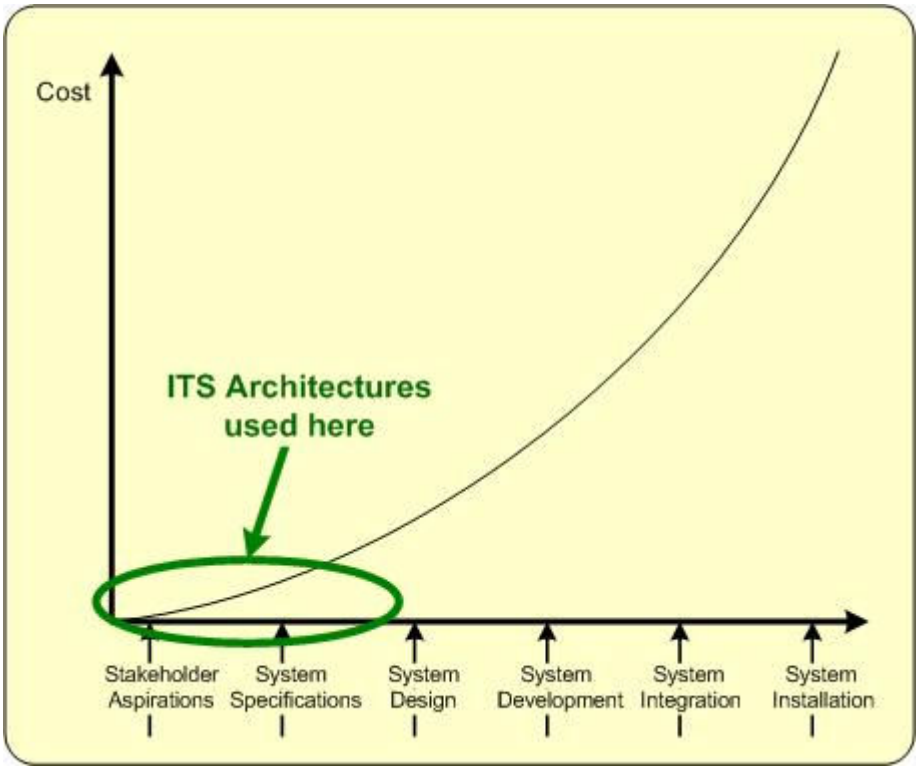
**Figure 1 – V-model system lifecycle**




The FRAME Architecture is used early in the lifecycle both to capture the Stakeholder Aspirations (sometimes called User Requirements for what the ITS implementation should provide), and then to produce the System Architecture that satisfies them, usually called the ITS Architecture.. The ITS Architecture is used to produce detailed system specifications that can be included in Calls for Tender for the design and development of part, or all, of the ITS that will satisfy part, or all, of the Stakeholder Aspirations. Thus the ITS Architecture provides a conceptual, or high-level view of the ITS implementation and thus is often referred to as a "high-level" architecture.

The early part of a system lifecycle is sometimes glossed over quickly so that the “more exciting” stages of design and implementation, and the use of (often new) technology can be reached as quickly as possible. The danger that exists when taking such an action is that the early products (Stakeholder Aspirations, User Needs, System Architecture and System Specifications) will not be complete and/or correct. Thus the resulting System Design, which will be verified against them, will also be incomplete and/or incorrect, and the development may be some way up the right hand side of the V-model lifecycle before the discrepancies begin to manifest themselves, making them much more expensive to rectify. This effect is demonstrated in Figure 2, which is sometimes called “The 10:100:1000 Rule” because the cost of correcting faults in a system increases exponentially (by about a factor of 10) during each successive stage of a lifecycle. There is some supporting evidence for this “Rule” as a result of work carried out in the US during the 1980’s and 1990’s by organisations such as the Software Engineering Institute at Carnegie Mellon University.

**Figure 2 – The 10:100:100 rule**

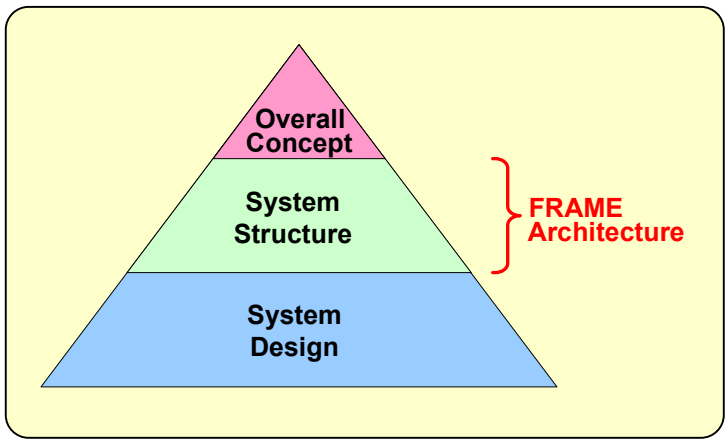


# 3 Different Levels of Architecture

## 3.1 Introduction

The term “architecture” is used by various types of engineer to refer to various type of structure at different levels of abstraction, and hence to describe different issues. Shows the three levels of abstraction that are normally used to describe an ITS deployment.

**Figure 3 - ITS Planning and Deployment**




## 3.2 The Overall Concept

The “Overall Concept” contains the organisational structure. This reflect the responsibilities of organizations and their (constitutional) relationship to each other and/or the desired properties of the ITS services, e.g. the need for graceful degradation. It is not always necessary to create an Overall Concept, but when one does exist checks have to be made to ensure that the System Structure adheres to its precepts.

## 3.3 The System Structure

The “System Structure” shows the various key components and their relationship to each other. It shows the functionality that is required, the physical locations where it takes place, and the flow of data/communications between them. The FRAME Architecture is designed to enable System Structures for ITS implementations to be developed in a common manner throughout Europe. Both the Overall Concept, if it exists, and the System Structure are written in a technology independent manner, i.e. they state *what* will happen, and not *how* it will take place.

## 3.4 The System Design

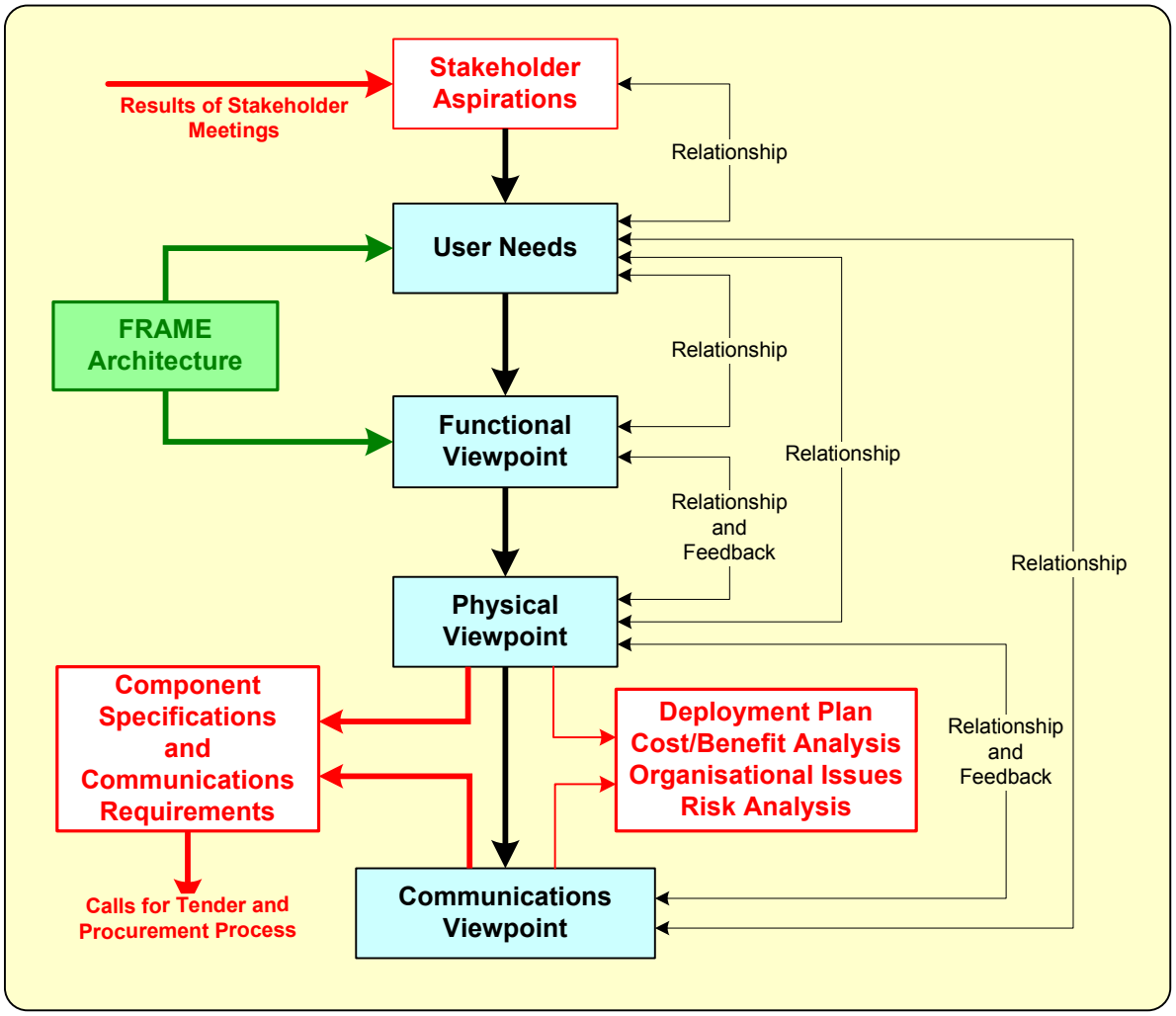
The “System Design” is the prerogative of the system providers and equipment manufacturers. Provided they adhere to the precepts of the System Structure and Overall Concept, together with the use of agreed standards, where necessary, then the resulting equipment/systems should be interoperable with each other. It should be noted that the use of Standards alone may not be sufficient, because some of them do not capture the desired behaviour which is also necessary for full interoperability.

# 4 The FRAME Architecture Methodology

## 4.1 Introduction

Over the last ten years, the FRAME Architecture has been used by several organisations to create ITS Architectures. The process in which the FRAME Architecture can be used contains a series of systematic steps. It was created by the FRAME-S project and begins with the collection of the Stakeholder Aspirations and then continues until the Component and Communications Specifications have been produced. This process is illustrated in Figure 4.

**Figure 4 – The FRAME Methodology**




Note that the FRAME methodology uses the term "Viewpoints" in the names of the parts that make up the FRAME Architecture and its resultant ITS Architectures. In doing this it is following the recommendations of IEEE 1471 – see reference 8(f). The alternative term of

"Architecture" is still used in some publications, but it is considered that an ITS architecture made up of viewpoints is more comprehensible than one that is made up of architectures.

It should also be noted that the use of particular technologies or supplier products is not included in the FRAME Architecture, nor do they form any part of the specifications produced by following the FRAME methodology. This is important for two reasons. Firstly the ITS Architectures created using the methodology will not become obsolete through advances in technology, or product development, and secondly it opens up the possibility for the development of new technologies to enable particular functionality to be provided.

## 4.2 Stakeholder Aspirations

Stakeholder Aspirations are statements that express the expectations and desires of the various stakeholders for the services that the final ITS implementation will provide. Although they should be written by the stakeholders, experience has shown that they often need help from the people who will create the ITS architecture – often referred to as the architecture team. There are four classes of stakeholder, as follows:

- Want ITS – this class comprises (local) authorities and road operators who need ITS services to enable their roads to be used safely and efficiently. It also includes public transport operators and freight operators where ITS can enable them to improve the efficiency with which they move people and goods.
- Use ITS – this class comprises the end users who make use of the ITS services and/or operate the equipment. It includes travellers on a multi-modal journey as well as all classes of vehicle driver; freight shippers; pubic transport mangers and specialist system operators.
- Rule ITS – this class represents those who provide regulations and standards. It includes national governments and the various standards making bodies.
- Make ITS – the class comprises the equipment and system manufacturers, communications providers and the system integrators.

Providers of services such as travel information and trip planning, may be in one or more of the Want, Use and Make ITS classes.

## 4.3 User Needs

Usually the Stakeholder Aspirations, when completed, will have been written in a variety of styles. Sometimes they are also obscure and occasionally they are inconsistent. It is therefore necessary to re-write them in a consistent manner that is suitable for the next stage in the process. The result is a set of User Needs (or User Requirements - see reference 8(e)) that express the Stakeholder Aspirations in a consistent and stylised manner whose meaning is clear and whose properties are testable.

The KAREN project produced a set of about 550 User Needs to cover the ITS applications and services being considered for implementation at the end of the 1990's, and the FRAME projects have added to them since then. In its expansion of the FRAME Architecture to include support for cooperative systems, the E-FRAME project has added another 244 new User Needs. Thus, when using the FRAME Architecture the architecture team normally only has to write new User Needs very occasionally, most of the time they can be selected from the existing list that is published as part of the FRAME Architecture – see reference 8(d).

## 4.4 Functional Viewpoint

A Functional Viewpoint (sometimes called a Logical Viewpoint, or even a Logical Architecture by others) shows the functionality that will be required to fulfil the User Needs, and hence the Stakeholder Aspirations. When using the FRAME Architecture the Functional Viewpoint is shown as a series of Data Flow Diagram that contains functions, data stores and terminators, and the data that flows between them. Each of these is provided with its own description which, in the case of functions, includes statements explaining what they do, e.g. collect data from a source outside the Architecture, and then process that data to produce the output Data Flow. Since the FRAME Architecture comprises a Functional Viewpoint that satisfies all of its User Needs, the architecture team only needs to select those parts of it that correspond to the sub-set of User Needs that have been mapped to the Stakeholder Aspirations. New functions etc. are only needed where certain Stakeholder Aspirations have required new User Needs to be added.

Another important part of the Functional Viewpoint is the Context Diagram. This shows the ITS as a single item and the links needed by the functionality within it to communicate with the entities outside it. It is useful for two reasons. Firstly it enables the system boundary to be defined showing what is inside the ITS implementation and what is not. Secondly it enables definitions to be produced of the way in which the functionality inside the ITS expects the outside entities to behave. These outside entities are called Terminators, and either obtain data for the ITS or provide outputs to end users. The same Context Diagram is also part of the Physical Viewpoint.

Some people have a concern that the Functional Viewpoint of the FRAME Architecture has been developed using a process oriented methodology that results in User Needs and Data Flow Diagrams, rather than Use Cases and Object Models (in UML). Although Use Cases and Object Models can be used for architectures at the level of abstraction at which the FRAME Architecture is set, experience has shown that the use of User Needs and Data Flow Diagrams is the correct choice for the following reasons:

- User Needs are concise statements, normally one sentence long, whilst a Use Case tells "a story" in a stylised manner. Use Cases were developed so that System Specifications would contain the detail that was necessary for the system designers, but in this lifecycle these come from the system architecture and are

not inputs to it (see Figure 1). Simple, one sentence, User Needs are all that is needed for this part of the lifecycle.

- At the architectural level, most ITS implementations are best described using a pipeline model of functions, which leads naturally to a Data Flow Diagram – see reference 8(e). Object Models can be used for the detailed design of certain components later in the lifecycle, if required.
- Stakeholders often like to view and pass comment on their ITS Architecture, but few of them are trained to understand Object Models. Experience has shown that Data Flow Models are intuitive to understand, and Stakeholders do not have any difficulty interpreting them.

Thus it can be seen that there is no need for the FRAME and OO methodologies to compete, indeed they address different issues. The FRAME methodology is intended for high-level architectures as it enables non-system designers to understand and make decisions about the form of the ITS implementation at an early stage. If required its results can then be used as input to an OO design methodology so that lower level "design" architectures can be created, which (optionally) include the use of the appropriate technology.

It is also worth noting that the FRAME Architecture shares its use of the process orientated methodology with the US National ITS Architecture – see reference 8(c).

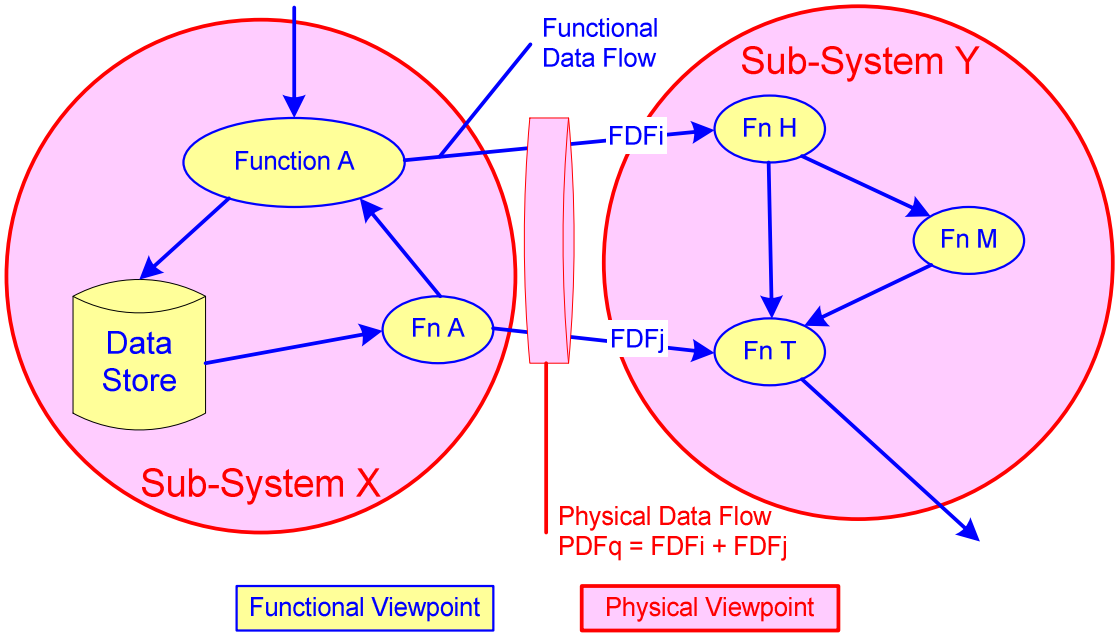
## 4.5 Physical Viewpoint

Once the Functional Viewpoint is complete, the architecture team allocates each function and data store to a sub-system, or to a module that is part of a sub-system – see Figure 5 on the next page. The location and contents of the sub-systems and modules within them is up to the architecture team and will vary from one Physical Viewpoint to another, even though each of them has been created from the same Functional Viewpoint. This makes it possible for the architecture team to investigate the use of different physical configurations so that the best solution can be found and presented to the Stakeholders.

Once the sub-systems and modules have been created and the functions and data stores allocated to them, the component (sub-system or module) specifications can be created from the descriptions of the functions and data stores obtained from the Functional Viewpoint. These specifications are intended to be included in the Calls for Tenders that will enable the components to be procured. As noted in the previous section, they can also provide the input to any OO methodologies that might be used for the detailed design part of the systems engineering lifecycle.

The Context Diagram produced as part of the Functional Viewpoint also applies to the Physical Viewpoint. It again shows the ITS as a single item and the links needed by the functionality within it to communicate with the entities outside it.

**Figure 5 – Example Functional and Physical Viewpoints**




The component (sub-system or module) specifications can be included in the tenders for their procurement – see Figure 7 in chapter 7. They can also provide the input to any OO methodologies that might be used for the detailed design part of the systems engineering lifecycle.

## 4.6 Communications Viewpoint

As can be seen from Figure 5, a consequence of allocating functions and data stores to sub-systems (and modules), is that it is immediately clear which Functional Data Flows lie within a sub-system (or module), and which Functional Data Flows pass between one sub-system and another, or between one module and another. Those that pass between sub-systems or modules make up the Physical Data Flows, and represent a communication channel between sub-systems, and/or between modules.

Since sub-systems are, by definition, located in different places (e.g. in a traffic management centre, at the road side, in a vehicle) it is possible to produce communications specifications by analysing the contents of each Physical Data Flow. The results of this analysis can be compared with a list of relevant and available communications standards to find the one that is most suitable. However if this is not possible then the results can be used as the basis for defining a new standard if the need for it can be agreed.

Analysis of the Physical Data Flows that pass between the ITS and the Terminators can also lead to “standard” interfaces for end users, which can play an important part in making sure that the ITS implementation can be used in the same way, everywhere that it is deployed.

## 4.7 Traceability

An important feature of the FRAME Architecture methodology is the ability to provide traceability all the way through the process – see Figure 4. It should be noted that the services contained in most ITS Architectures cannot normally be deployed all at the same time, both for reasons of cost, as well for reasons of dependability (i.e. one service may have to be established before another can be introduced). Thus those planning the implementation of the components and communications links identified by the ITS Architecture need to take account of any financial and dependability constraints that the proposed deployment may have.

Traceability can provide a relationship between the Stakeholder Aspirations and the sub-systems and modules in the Physical Viewpoint. This enables the owners of the ITS Architecture to quickly identify those components that are needed to satisfy a given set of Aspirations, and thus meet their immediate political goals. A traceability matrix of Stakeholder Needs against sub-systems (and modules) can also show whether certain Aspirations can be satisfied "for free", i.e. having identified the sub-systems and/or modules needed to satisfy a given set of Aspirations, it may be found that it is possible to satisfy some other Aspirations without the need for extra sub-systems and/or modules.

## 4.8 Additional Studies

Once an ITS Architecture has been created, it can be used as the basis for certain analyses of the impact created by the implementation of its contents.

- Organisational Issues – most integrated ITS implementations have components and communications links that are owned and managed by more than one organisation, who must work together in a harmonious manner. Issues such as command and control, ownership of equipment and data, and priority in the use of communications links need to be identified and agreed as early as possible before deployment takes place. The ITS Architecture shows the technical relationship between components, and this can be used as the basis for discussion and agreement about such issues.
- Deployment Plan – as discussed in Section 4.7, the deployment of all the services represented in an ITS Architecture can take some time to complete. In addition to providing the component and communications specifications, by showing their interrelationship the ITS Architecture will help the creation of plans that get a sensible sequence for their deployment and use.
- Cost/Benefit Analysis – the ITS Architecture is a representation of all the components, data and communications links needed to satisfy some or all of the Stakeholders' Aspirations. A cost/benefit analysis can be undertaken using the capital and operating costs of the components and communications links, and the benefits that can be assigned to their use in order to justify, or otherwise, a particular ITS implementation. It is also possible to use the cost data together

with the Deployment Plan to create what is sometimes called a "financial profile" for the ITS implementation. This can be used to give an idea of what expenditure will be required and when, so that an implementation budget can be prepared and integrated into some or all of the Stakeholders' own budgets.

- Risk Analysis – a risk analysis can be undertaken into one or more of the above topics to ensure their validity and to plan any required mitigation strategies. This can be very useful for decision makers who can be helped to understand the consequences and constraints arising from the ITS implementation. It is also very useful to be able to assign any mitigation strategies to the most appropriate Stakeholder (or other organisation), as this will speed up their implementation.

# 5 The FRAME Architecture Tools

## 5.1 Introduction

The FRAME Architecture is supported by two tools that facilitate the work that needs to be done by the architecture team to create ITS architectures from it. They are called the FRAME Browsing Tool and the FRAME Selection Tool and can both be downloaded from the FRAME website – see reference 8(d).

## 5.2 The FRAME Browsing Tool

The FRAME Architecture is a large product with many thousands of elements. In addition, the Data Flow Diagrams have had to be organised in a hierarchical manner so that they are understandable. In order to provide a unified front end to the architecture team a FRAME Browsing Tool has been creates that permits all the elements of the FRAME Architecture, and their interrelationships, to be viewed interactively using a standard HTML viewer. Thus, for example, it is possible to follow the passage of data from its collection by a particular functionality, through fusion and processing, to its eventual use in providing a service for the end users.

## 5.3 The FRAME Selection Tool

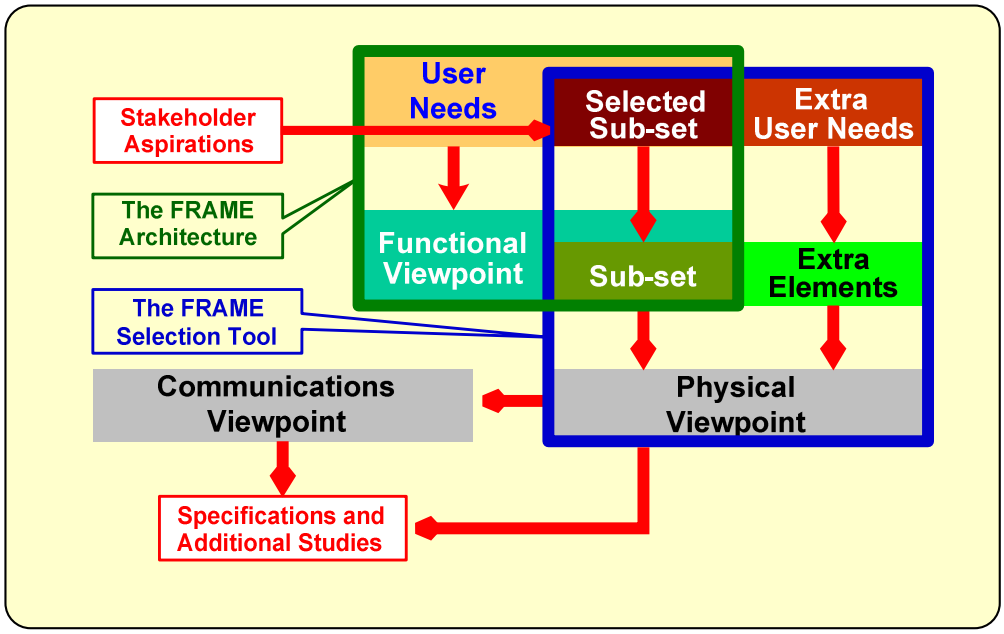
The FRAME methodology described in chapter 3 is supported by the FRAME Selection Tool (see Figure 6 on the next page), which contains a data base with all the elements of the FRAME Architecture. The Tool does not perform any selections automatically, but it does support the architecture team in its use of the methodology in the following ways.

- The team selects those User Needs that reflect the Stakeholder Aspirations.
- The tool will then guide the architecture team to those parts of the Functional Viewpoint that help to satisfy those User Needs.
- The FRAME Architecture does not claim to satisfy every possible ITS User Need, and in some circumstances it may be necessary to add extra User Needs and Functional Viewpoint elements to the Selection Tool data base.
- Since the mapping from User Needs to Functions is not an exact science, the tool will probably report some logical inconsistencies after the first pass (e.g. a data flow with only the function at one end selected). The team can then select further elements, or deselect some of those already selected, until there are no logical consistency errors, and they are satisfied that their selection fully represents the Functional Viewpoint needed to satisfy the Stakeholder Aspirations.
- Once a Functional Viewpoint is considered acceptable, it can be used as the basis for one or more Physical Viewpoints. The architecture team does this by

allocating functions and data stores to individual sub-systems, and to modules within them if required. Modules are used to partition the functionality in sub-systems so that, for example, the functionality for traffic management can be separated from that for parking management.

- Once a Physical Viewpoint has been completed one of the reports available from the Selection Tool can be used to provide the starting point for an analysis of the Physical Data Flows. This leads to the creation of the Communications Viewpoint, which shows the characteristics of the links required between each of the sub-systems and modules, plus those with the Terminators.

**Figure 6 – Use of the FRAME Selection Tool**




Thus, although the Selection Tool does not have any intelligence to make decisions on behalf of the architecture team, it does perform much of the detailed work of recording all the decisions taken by them. Experience has shown that it is not normally necessary to produce a Data Flow Diagram of the Function Viewpoint since all the information required to produce a Physical Viewpoint is held within the data base. The Tool permits more than one Physical Viewpoint to be created from a Functional Viewpoint so that the advantages and disadvantages of different component configurations, physical locations and deployment scenarios can be explored.

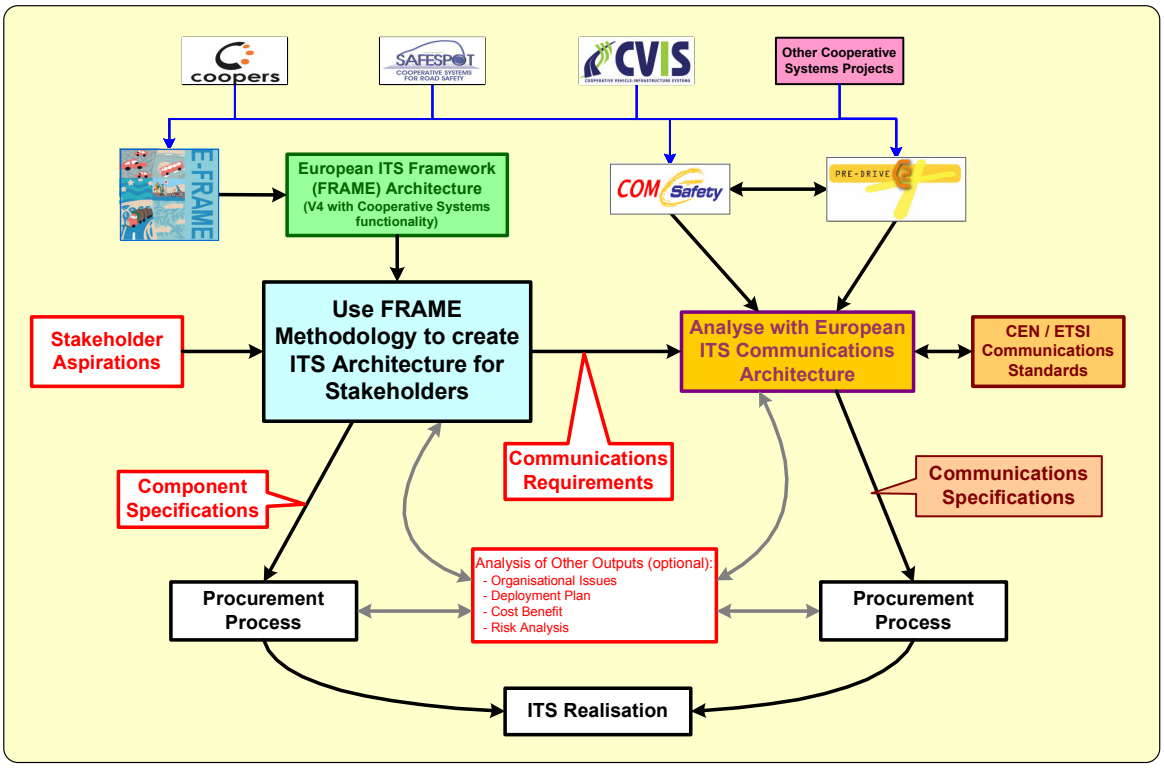
# 6 Extensions to the FRAME Architecture for Cooperative Systems

The E-FRAME project has extended the FRAME Architecture to include the functionality that will be needed for the implementation of cooperative systems. The definition of this functionality has been based initially on the combined requirements produced by the three Cooperative System Integrated Projects, COOPERS, CVIS and SAFESPOT and the work of other project such as COMeSafety and PRE-DRIVE C2X – see reference 8(g). The requirements produced by any other relevant cooperative systems projects have also been included where appropriate, and if they were available before the E-FRAME project finished.

The task of extending the FRAME Architecture with the defined new Functions to support the cooperative systems services was huge and underestimated by all experts involved. In total **over 250 Functions** have been updated or added to create the current version (4.1) of the FRAME Architecture. The total number of Functions in the Architecture has increased by **over 70%**. The figures for **Data Flows** are even larger as the total number in the Architecture has increased by **over 100% to more than 2200**.

The manner in which it is envisaged that the results of this work will fit together is shown in Figure 7. This figure also shows that an important result of the work being carried out by these projects is input to the work of the CEN and ETSI standards organizations.

**Figure 7 – ITS implementation process with cooperative systems project inputs**



# 7 Conclusions

The implementation of ITS is becoming increasing complex, and thus more difficult to achieve successfully. The FRAME Architecture methodology has been developed according to standard system engineering practices and it should be used in the early stages of the system lifecycle.

Although targeted at Europe, the FRAME Architecture can be used anywhere. This is because it comprises only User Needs and a Functional Viewpoint, both of which are independent of any technology or existing product designs. It is also possible to extend the User Needs and Functional Viewpoint to accommodate particular services the Stakeholders want to implement. Several Physical Viewpoints can be created from a single Functional Viewpoint by the architecture team according to the requirements of the Stakeholders, or to enable different system configurations to be explored.

The FRAME Architecture has been extended by the E-FRAME project to include the functionality needed to support the services provided by cooperative systems. It is expected that over time it will be extended and modified further to accommodate changes in the range and scope of services that ITS can provide.

# 8 References

(a) Rechtin, E. 1991 Systems Architecting – Creating & Building Complex Systems. Prentice Hall. ISBN 0 13 880345 5.

(b) Thomé, B. 1993 Systems Engineering – Principles and Practice of Computer-based Systems Engineering. John Wiley & Sons. ISBN 0 471 93552 2.

(c) US DOT. The National ITS Architecture, www.its.dot.gov/arch/.

(d) FRAME. The European ITS Framework Architecture, www.frame-online.net.

(e) Sommerville I. 2007 Software Engineering (8th Ed). Addison-Wesley, ISBN 978 0 321 31379 9.

(f) IEEE 1471-2000. Recommended Practice for Architectural Description of Software-Intensive Systems, http://standards.ieee.org/findstds/standard/1471-2000.html.

(g) COMeSafety/PRE-DRIVE C2X. 2009. European ITS Communication Architecture – Overall Framework – Proof of Concept Implementation.