C-ITS Architecture Framework

Priyanka Karkhanis, Yanja Dajsuren and Mark van den Brand

***Abstract*— C-ITS (Cooperative Intelligent Transport Systems) is an initiative to facilitate cooperative, connected and automated mobility. The C-ITS domain comprises widely spread systems like traffic management systems, traffic light controllers, or vehicle on-board units. Such complex and heterogeneous systems have independent uses but demand a strategy to facilitate their convergence. This initiative is currently demonstrated by C-MobILE project. C-MobILE is a large scale deployment Europe Union project spanning from 2017-2020. The main objective of C-MobILE is to define an integrated architecture based on a number of existing C-ITS projects. The architecture provides a way to standardize and to have a unifying modeling approach by means of a common language that can be reused by other organizations to guide their internal processes.**

In this paper, we describe about adaptation of architecture framework, architecture viewpoints and modelling approach for C-ITS systems. The architecture framework and the same modelling approach can enable common language and will be reused for the next deployment projects. To put architecture framework in context, we extend the conceptual model of the ISO/IEC/IEEE 42010 [1] architecture framework. The architecture framework is intended to be used as a basis for developing C-MobILE pilot site architectures. This will enable C-MobILE deployment at EU defined eight pilot sites and beyond satisfying stakeholders from public and private parties in an EU context.

I. INTRODUCTION

An architecture framework is one of the widely-applied approaches used in software/system architecting of complex systems. The architecture frameworks facilitate communication and cooperation between different stakeholders during architecting and building complex systems such as C-ITS. Many different stakeholders with their interweaving concerns require a systematic approach for addressing complexity and full lifecycle of the system. Example representations of widely applied architecture frameworks include Kruchten?s 4+1 View Model, TOGAF, and Zachman framework. 4+1 is developed by Kruchten as a generic architecture framework for describing the architecture of software-intensive systems based on the use of multiple, concurrent views [2]. TOGAF provides a practical and industry standardized approach for designing an enterprise architecture [3]. Zachman framework is used for modeling an enterprise?s information infrastructure from six perspectives [4].

According to the ISO/IEC/IEEE 42010:2011 standard, an architecture framework establishes a common practice for creating, interpreting, analysing and using architecture descriptions within a particular domain of application or stakeholder community [5]. As a well-defined architecture framework is considered to be an important part of any architecture description [6], we define an architecture framework for the C-ITS domain. To put the architecture framework and architecture description concepts in context, C-MobILE extends the conceptual model of the ISO/IEC/IEEE 42010 international standard for architecture descriptions of systems and software [5], uses architecture viewpoints of the architecture framework for automotive systems [8] and concept of architecture perspective for shaping the architecture views [7]. Without the common C-ITS architecture framework, different categorizations and ad-hoc notations are used in the existing C-ITS architectures. The C-ITS domain covers not only software/system engineering field, but also traffic engineering, civil engineering, information technology etc., which require a unified definition of architecture framework for the C-ITS domain.

II. PROBLEM STATEMENT

The C-MobILE project aims for a large scale demonstration across various deployment sites. These various deployment sites all have their own defined ITS architecture, and their own multidisciplinary approach towards their deployed strategy. These architectures consists of different informal design patterns. There is no standard notation to help in merging these architectures into a standardized architecture. Without a common C-ITS architecture framework, different categorizations and ad-hoc notations have been used in existing C-ITS architectures. These differences creates an inefficient behavior towards a seamless C-ITS implementation across various deployment sites. This impacts in addressing concerns like interoperability. Hence, a standardized approach is required to consolidate and integrate existing architectures, addressing concerns such as interoperability, security, availability, and maintainability. To help reach the project goals, a good architecture, capable of being deployed to whole Europe is needed.

III. METHODOLOGY

As a well-defined architecture framework is considered to be an important part of any architecture description [6], we define an architecture framework for the C-ITS domain. To put architecture framework and architecture description

Funded by European Union Commission , 723311.

Priyanka is PhD candidate with Mathematics and Computer Science Department, Eindhoven University of Technology, Eindhoven, The Netherlands p.d.karkhanis@tue.nl

Yanja Dajsuren is an assistant professor and PDEng Program Director, Eindhoven University of Technology, Eindhoven, The Netherlands Y.Dajsuren@tue.nl

Mark van den Brand is a professor at Mathematics and Computer Science Department, Eindhoven University of Technology, Eindhoven, The Netherlands M.G.J.v.d.Brand@tue.nl

concepts in context, we extend the conceptual model of the ISO/IEC/IEEE 42010 architecture framework as illustrated in 1. The conceptual model of architecture framework is highlighted in blue and the relationships of architecture description concepts are added to the original standard diagram.

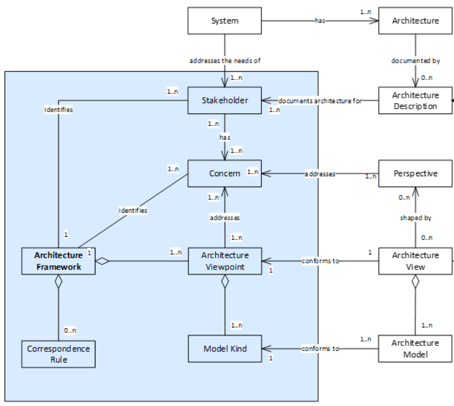

Fig. 1. Architecture Framework in Context

As illustrated, an architecture description documents an architecture for the stakeholders using a set of Architecture Views and models. An Architecture View conforms to a viewpoint which addresses stakeholder concerns and can be shaped by a number of perspectives. The perspective defines concerns that guide architectural decision making to help ensure that the resulting architecture quality characteristics considered by the perspective [7]. This is essential for the C-ITS architectures as the architectural choices are costly to make after the implementation. In the existing C-ITS reference architectures, security is considered broadly albeit from limited aspects e.g. from the information and communication views. As described in the definition of an architecture framework, the C-ITS architecture framework specifies stakeholders, their concerns, viewpoints, model kinds, and correspondence rules.

A. *C-ITS Stakeholders and Concerns*

A stakeholder is an individual, team, or organization holding concerns for the system such as architect, designer, user, and authority [1]. A concern is any interest in the system such as functionality, structure, behaviour, and interoperability [1]. The C-ITS address the needs of some of following stakeholders. The rest of stakeholders description can be referred to C-MobILE Reference Architecture Report [9].

B. *Viewpoints and Views*

We propose a set of six core viewpoints as part of the C-ITS architecture framework: Context, Functional, Information, Implementation, Physical, and Communication. These viewpoints are defined based on the existing literature and ITS reference architectures. We believe that this set of viewpoints enable structured architectural descriptions for the C-ITS. The relationships between views created using these viewpoints are shown in 2. We believe that this set of viewpoints enable structured architectural descriptions for the C-ITS systems.

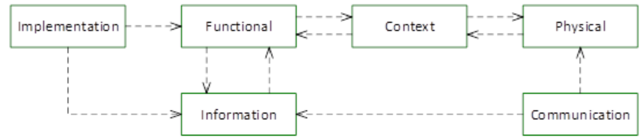

Fig. 2. Relationships between architecture views

Below table 3 briefly describes about viewpoint definitions and its respective views descriptions.

C. *Architecture Perspective*

We identify the key perspectives for large scale demonstration of C-ITS systems. The identified perspectives can be applied to the views. Additional quality characteristics and perspectives may be added in the descriptions of concrete and implementation architectures. Architectural perspectives are used to formalize and guide the process of evaluating and reviewing the architectural models to ensure that the architecture satisfies the required quality characteristics and to select architectural tactics when it does not. We summarize below the definitions of each perspective which are based on the international ISO/IEC 25010 standard [10]:

- Interoperability is degree to which two or more systems, products or components can exchange information and use the information that has been exchanged.
- Security is degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization.
- Performance efficiency is performance relative to the amount of resources used under stated conditions.
- Usability is degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.
- Reliability is degree to which a system, product or component performs specified functions under specified conditions for a specified period.
- Availability is degree to which a system, product or component is operational and accessible when required for use.
- Adaptability is degree to which a product or system can effectively and efficiently be adapted for different or evolving hardware, software or other operational or usage environments.

We also provide architectural tactics for the interoperability and security perspectives so that they could be of use as an inspiration for elaborating the relevant tactics for the C-MobILE deployment architectures. Defining the perspectives for C-MobILE architecture definition and eventually large-scale demonstration of C-ITS systems would help avoid expensive changes in the later stages of development.

Viewpoint	Definition	Respective View
Context	Describes the relationships, dependencies, and interactions between the system and its environment (people, systems, external entities interacting with the system).	A *context view* helps the system's stakeholders to understand system's responsibilities and how it relates to its organization.
Functional	Describes the system's runtime functional elements, their responsibilities, interfaces, and primary interactions.	A *functional view* helps the system's stakeholders understand the system structures and has an impact on the system's quality properties.
Communication	Describes the communications (e.g. interfaces, communication protocols) between different subsystems deployed on different hardware environment.	A *communication view* supports stakeholders involved in defining/enabling communication between systems.
Information	Describes how the architecture stores, manages, and distributes data and information.	An *information view* provides high-level view of static data structure and information flow to users, developers, testers, and maintainers.
Implementation	Describes the implementation for realizing functionality into real life software systems.	An *implementation view* supports stakeholders involved in building, testing, maintaining, and enhancing the system.
Physical	Describes the physical environment where the system will be deployed and the dependencies that the system has on elements of it.	A *physical view* supports stakeholders involved in deploying, testing, and maintaining the system by capturing the hardware environment that the system needs, the technical environment requirements for each element, and the mapping of software elements to the runtime environment that will execute them.

Fig. 3. Viewpoints Description

D. Architecture Representation

We propose to use Systems Modelling Language (SysML) diagram types to for architectural notations of the C-ITS architectures. The SysML is a general purpose modelling language for engineering systems and consists of structure diagram, requirement diagram, and behaviour diagram types as shown in 4.

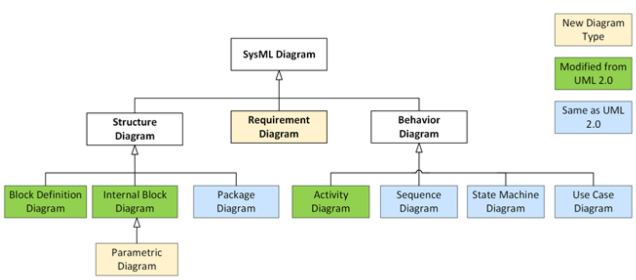

Fig. 4. Sysml Types

IV. Result

By adapting architecture framework and applying architecture descriptions, we defined reference architecure. The reference architecture provides a common vocabulary with which to discuss implementation, often with the aim to stress commonality often based on the generalizations of a set of solutions. The approach and methodology to extract reference architecture is described detail in ”Defining C-ITS Reference Architecture” [9] ¡¡WASA paper namee¿¿.

To develop a common and compatible reference architecture, we analyzed existing C-ITS reference architectures and projects that included the Dutch C-ITS Reference Architecture (DITCM) [11] ¡mention other architectures¿. Besides these C-ITS architectures, we considered ITS implementations of the deployment sites involved with C-MobILE.

As an illustration of the architecture framework and reference architecture, we discuss here in detail the context viewpoint, view, and the context model.

A. Context Viewpoint

The context viewpoint describes the relationships, dependencies, and interactions between the system and its environment (e.g. people, systems and external entities) [7]. The context view conforms to the context viewpoint and helps system?s stakeholders (e.g. system/software architects, designers, developer and users) understand the system context.

B. Context View

The context view conforms to the context viewpoint. SysML Block Definition Diagram (BDD) is identified as suitable modelling notation for capturing the context viewpoint. SysML Use Case diagram can be used to show the usage of a system. The view conforms to viewpoint which satisfies the concerns of stakeholders as described in figure 5.

C. Model Kinds

The context diagram is the key model within a context view, placing the system in its environment by relating it to the external actors that it interacts with via explicit relationships that represent the connections to and from it. A context diagram is usually quite simple and contains elements of the following types:

- System: the system being designed, which is treated as a black box, with its internal structure hidden.
- External Entities: systems, people, groups and other entities that the system interacts with.
- Connections: the interfaces, protocols, and connectors that link the external entities and the system being designed or utilized.

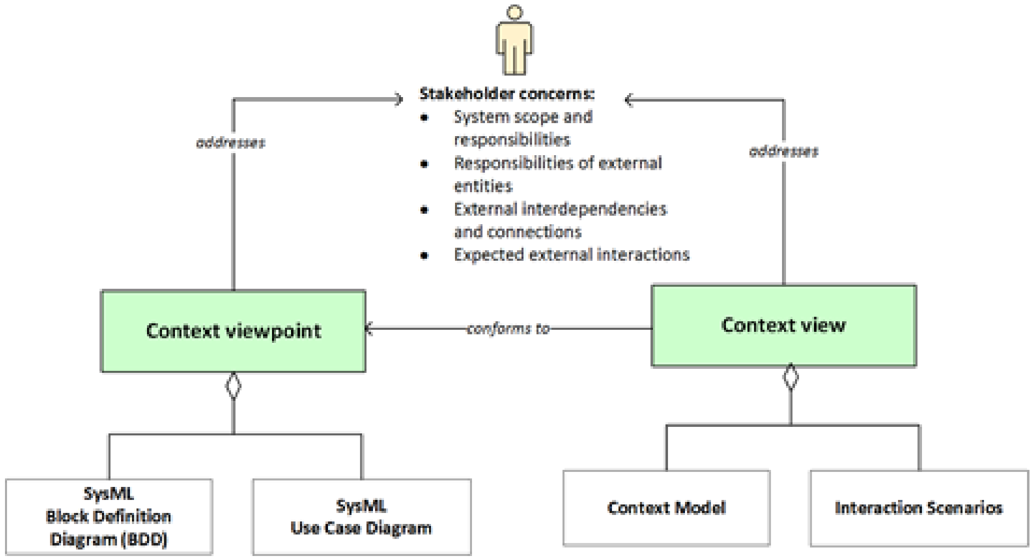

Fig. 5. Context Viewpoint and Views

1) Context Model: The notations that we commonly see used for context diagrams are SysML Block Definition Diagram. C-MobILE consists of five main systems which is depicted as black box and corresponding actors?s connections with those systems is shown in Figure 6. The five main systems of the C-MobILE architecture are aligned with the five layers of the DITCM reference architecture [11]:

- Support System: Comprise of sub-systems performing various tasks e.g.: governance, test and certification management, security and credentials management.
- Central System: Comprise of sub-systems to support connected vehicles, field and mobile devices. The subsystems can be aggregated together or geographical or functionally distributed.
- Roadside System: Comprise of sub-systems which covers the ITS infrastructure on or along physical road infrastructure, e.g.: roadside units, signal/lane control etc.
- Vehicle System: Comprise of sub-systems which are integrated within vehicle such on-board systems (advanced driver assistance / safety systems, navigation, remote data collection or information).
- Traveler/VRU System: Comprise of both personal devices (e.g.: mobile devices, navigations devices) and specific systems connected to vehicles of VRU?s (e.g.: tags).

(Human) actors are treated as external entities that interact with the systems:

- Vehicle Driver: An actor driving in a vehicle. The vehicle is a motorized vehicle (car, bus, truck) and not a vehicle of a vulnerable road user (bike, moped, motor). An actor in this category is directly concerned with Vehicle System as shown in Figure 24 through various vehicle related interfaces like : OBU , HMI etc.
- Vulnerable Road User: A VRU is a human actor like a pedestrian, cyclist or PTW driver; A motorcyclist is also an example of a PTW and is treated as a vulnerable road user in specific road hazard situations with other cars. An actor in this category is directly concerned with Traveler/VRU System as shown in Figure 24 through various interfaces like HMI, tablet, mobile, etc.
- Road Operator: An actor responsible for the traffic management of a road network. An actor under this category is directly concerned with Roadside System through various communication channels and is responsible for collecting and evaluating data related to roadside information.
- Service Provider: An actor (organization) supplying services to one or more customers. Customers are either other organizations, including government (B2B / B2G / G2B / G2G) or end users (B2C / G2C). An actor under this category is directly concerned with Central System which also is responsible providing various services. Typical examples of a Service Provider are a Navigation Provider as a Service Provider providing navigation services to end users or organizations or a Traffic Information Provider as a Service Provider that provides road traffic related information, like traffic jams, incidents, road works warning etc.

D. Correspondence Rules

The context viewpoint has correspondences with functional and physical viewpoints as highlighted in Figure 7. Functional and physical viewpoints have conformance correspondence to the context viewpoint.

V. CONCLUSIONS

The main objective is to provide an architecture for the C-MobILE C-ITS environment. The purpose of this deliverable is to create a C-ITS reference architecture that enables pan-European interoperability of C-ITS architectures based on

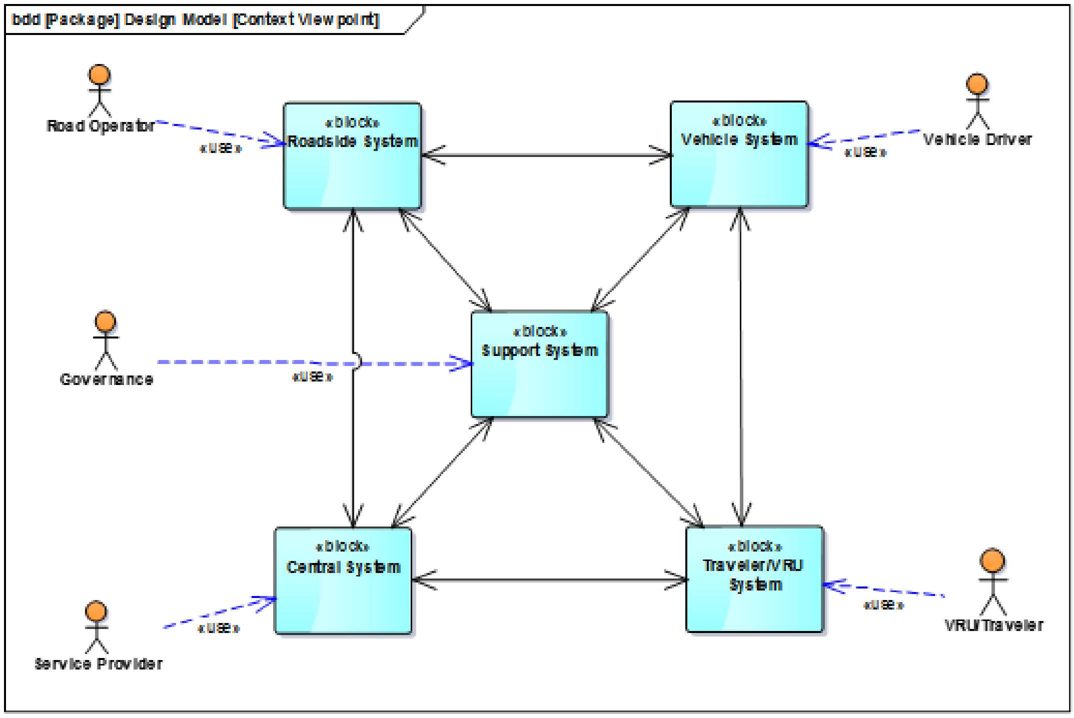

Fig. 6. Context model describing the relationship between systems and actors

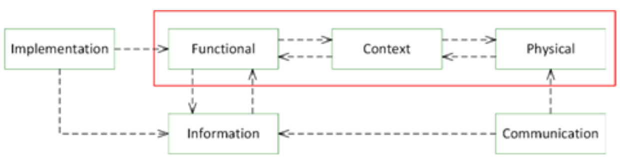

Fig. 7. Correspondence rules

the generalization of existing C-ITS architectures. The C-ITS architecture framework and reference architecture is a first step in an integrated architecture based on a number of projects. We have analysed existing C-ITS architectures, especially CONVERGE, MOBiNET, and Dutch C-ITS Reference Architecture, to define common concepts and vocabulary for the C-MobILE architectures. The different architectures from the pilot sites are used as an input for defining a single homogeneous reference architecture in line with current standards. The C-ITS architecture framework was developed by defining Architecture Viewpoints and their respective views addressing respective stakeholder concerns. This will facilitate the communication between different stakeholders. SysML is selected as a formal modelling language for describing the architectures. The architecture models are developed in Enterprise Architect using SysML profile. SysML block definition diagram is mainly used to create architectural models for different views.

ACKNOWLEDGMENT

The C-MobILE project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 723311.

REFERENCES

[1] ISO/IEC/IEEE Systems and software engineering – Architecture description," in ISO/IEC/IEEE 42010:2011(E) (Revision of ISO/IEC 42010:2007 and IEEE Std 1471-2000) , vol., no., pp.1-46, Dec. 1 2011 doi: 10.1109/IEEESTD.2011.6129467

[2] Kruchten, P.B. "The 4+ 1 view model of architecture." IEEE software 12.6 (1995): 42-50

[3] Haren, Van. TOGAF Version 9.1. Van Haren Publishing, 2011.

[4] Zachman, J. "The zachman framework for enterprise architecture." Zachman International 79 (2002).

[5] ISO/IEC/IEEE 42010:2011. Systems and software engineering ? Architecture description. http://www.iso-architecture.org/

[6] Emery, D., and Rich H. Every architecture description needs a framework: Expressing architecture frameworks using ISO/IEC 42010. Software Architecture, 2009 & European Conference on Software Architecture. WICSA/ECSA 2009. Joint Working IEEE/IFIP Conference on. IEEE, 2009.

[7] Rozanski, N. and Woods, E. Software systems architecture: working with stakeholders using viewpoints and perspectives. Addison-Wesley, 2011.

[8] Dajsuren, Y. "On the design of an architecture framework and quality evaluation for automotive software systems." PhD thesis. Eindhoven University of Technology, 2015.

[9] Dajsuren, Y., Karkhanis, P., Kadiogullary, D. and Fuenfrocken, M. C-MobILE D3.1 Reference Architecture. Available at `http://c-mobile.diviprojects.wpengine.com/wp-content/uploads/sites/19/2018/03/C-MobILE-D3.1-ReferenceArchitecture_v0.1.pdf`

[10] International Organization for Standardization. ISO-IEC 25010: 2011 Systems and Software Engineering-Systems and Software Quality Requirements and Evaluation (SQuaRE)-System and Software Quality Models. ISO, 2011.

[11] van Sambeek, M. Turetken, O., Ophelders, F., Eshuis, R., Bijlsma, T. Traganos, K., Grefen. P (2015) Towards an Architecture for Cooperative ITS Applications in the Netherlands, Version 1.0. DITCM. April 17, 2015. Available at: https://goo.gl/pHSD9D