Use case scenarios and preliminary reference model

M. Radimirsch, E.V. Matthiesen, G. Huszerl,
M. Reitenspieß, M. Kaâniche, I.E. Svinnset,
A. Casimiro, L. Falai

DI–FCUL TR–07–18

September 2007

Departamento de Informática
Faculdade de Ciências da Universidade de Lisboa
Campo Grande, 1749–016 Lisboa
Portugal

Technical reports are available at `http://www.di.fc.ul.pt/tech-reports`. The files are stored in PDF, with the report number as filename. Alternatively, reports are available by post from the above address.

Project no.: **IST-FP6-STREP- 26979**

Project full title: **Highly dependable ip-based networks and services**

Project Acronym: **HIDENETS**

Deliverable no.: D1.1

Title of the deliverable: Use case scenarios and preliminary reference model

Contractual Date of Delivery to the CEC:	30th September 2006
Actual Date of Delivery to the CEC:	29th September 2006
Organisation name of lead contractor for this deliverable	Carmeq
Author(s): Markus Radimirsch, Erling V. Matthiesen, Gábor Huszerl, Manfred Reitenspieß, Mohamed Kaâniche, Inge Einar Svinnset, António Casimiro, Lorenzo Falai	
Participant(s): Marc Löbbers, Marc-Olivier Killijian, Tone Ingvaldsen, Audun Fosslie Hansen, Irene de Bruin, Hélène Waeselynck, Nicolas Rivière	
Work package contributing to the deliverable:	WP1
Nature:	R
Version:	1.1
Total number of pages:	89
Start date of project:	1st Jan. 2006 **Duration:** 36 month

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	Public	**X**
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Abstract:

This document provides the starting point for the development of dependability solutions in the HIDENETS project with the following contents:

(1) A conceptual framework is defined that contains the relevant terminology, threats and general requirements. This framework is a HIDENETS relevant subset of existing state-of-the-art views in the scientific dependability community. Furthermore, the dependability framework contains a first list of relevant functionalities in the communication and middleware level, which will act as input for the architectural discussions in HIDENETS work packages (WPs) 2 and 3.

(2) A set of 17 applications with HIDENETS relevance is identified and their corresponding dependability requirements are derived. These applications belong mostly to the class of car-to-car and car-to-infrastructure services and have been selected due to their different types of dependability needs.

(3) The applications have been grouped in six HIDENETS use cases, each consisting of a set of applications. The use cases will be the basis for the development of the dependability solutions in all other WPs. Together with a description of each use-case, application-specific architectural aspects are identified and corresponding failure modes and challenges are listed.

(4) The business impact of dependability solutions for these use cases is analysed.

(5) A preliminary definition of a HIDENETS reference model is provided, which contains high-level architectural assumptions. This HIDENETS reference model will be further developed in the course of the HIDENETS projects in close cooperation with the other WPs, which is the reason why the preliminary version also contains a collection of potential contributions from other WPs that shall be developed and investigated in the course of the HIDENETS project.

In summary, the identified use-cases and their requirements clearly show the large number of dependability related challenges. First steps towards technical solutions have been made in this report in the preliminary reference model, whereas the other work-packages have started in the meanwhile to develop such solutions further based on ‘middleware technology’ (WP2), ‘communication protocols’ (WP3), ‘quantitative analysis methodology’ (WP4), and ‘design and testing methodology’ (WP5).

Table of Contents

Bibliography

[1] CAMP - Vehicle Safety Communications Project, „Task 3: Identify Intelligent Vehicle Safety Applications Enabled by DSRC - Interim Report – Jan. 2003

[2] C2CC APP, List of applications (BMW, DaimlerChrysler, Volkswagen), NOW project (in German)

[3] Enkelmann W. FleetNet Applications for Inter-Vehicle Communication IEEE Intelligent Vehicles Symposium (IV 2003), pp. 162-167, Columbus, OH, USA, June 2003 (to be found at http://www.et2.tu-harburg.de/fleetnet/english/documents.html)

[4] Wischhof L., Ebner A., Rohling H., Lott M., Halfmann R. SOTIS - A Self-Organizing Traffic Information System 57th IEEE Semiannual Vehicular Technology Conference VTC 2003-Spring, Jeju, South Korea, April 2003 (to be found at http://www.et2.tu-harburg.de/fleetnet/english/documents.html)

[5] A. Avizienis, J.C. Laprie, "Dependable computing: from concepts to design diversity", Proceedings of the IEEE, vol. 74, no. 5, May 1986, pp. 629-638.

[6] A. Avizienis, J.C. Laprie, B. Randell, C. Landwer, "Basic Concepts and Taxonomy of Dependable and Secure Computing ", IEEE Transactions on Dependable and Secure Computing, vol. 1, no. 1, January-March 2004, pp. 11-33.

[7] W.C. Carter, "A time for reflection", in Proc. 12th IEEE Int. Symp. on Fault Tolerant Computing (FTCS-12), Santa Monica, California, June 1982, p. 41.

[8] J.C. Laprie, A. Costes, "Dependability: a unifying concept for reliable computing", Proc. 12th IEEE Int. Symp. on Fault Tolerant Computing (FTCS-12), Santa Monica, California, June 1982, pp. 18-21.

[9] J.C. Laprie (Ed.), Dependability: Basic Concepts and Terminology, Springer-Verlag, Vienna, 1992.

[10] IEEE 802.11 WG, “Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specification”, IEEE 1999.

[11] IEEE 802.11 WG, “Draft Supplement to Part 11: Wireless Medium Access Control (MAC) and physical layer (PHY) specifications: Medium Access Control (MAC) Enhancements for Quality of Service (QoS)”, IEEE 802.11e/D13.0, Jan. 2005.

[12] Moy, J. OSPF Version 2. IETF RFC 2328 (STD 54), April 1998.

[13] Callon R W. Use of OSI IS-IS for routing in TCP/IP and dual environments. IETF RFC 1195, December 1990

[14] Rekhter, Y. A Border Gateway Protocol 4 (BGP-4). IETF RFC 4271, January 2006.

[15] Clausen T, Jacquet P. Optimized Link State Routing Protocol (OLSR). IETF RFC 3626, October 2003

[16] Spagnolo P et al. OSPFv2 Wireless Interface Type. Internet draft ‘draft-spagnolo-manet-ospf-wireless-interface-01, May 2004.

[17] Chandra M. Extensions to OSPF to Support Mobile Ad Hoc Networking. Internet draft ‘draft-chandra-ospf-manet-ext-02’, October 2004.

[18] Perkins C, Belding-Royer E, Das S. Ad hoc On-Demand Distance Vector (AODV) Routing. IETF RFC 3561, July 2003

[19] Mauve M., Widmer J., Hartenstein H. A Survey on Position-Based Routing in Mobile Ad-Hoc Networks IEEE Network, 5(6), pp. 30--39, November 2001

[20] K. Matheus, R. Morich, et al., „Car-to-Car Communication - Market Introduction and Success Factors“, ITS 2005: 5th European Congress and Exhibition on Intelligent Transport Systems and Services, 1 - 3 June 2005, Hannover, Germany

[21] K. Matheus, R. Morich, A. Lübke, „Economic Background of Car-to-Car Communications“, IMA 2004, Informationssysteme für mobile Anwendungen, 20.-21.10.2004, Braunschweig, Germany

[22] A. Autenrieth, A. Kirstädter “Fault Tolerance and Resilience Issues in IP-Based Networks”, Second International Workshop on the Design of Reliable Communication Networks (DRCN2000), Munich, Germany, April 9-12, 2000

[23] T. Chandra, S. Toueg. Unreliable failure detectors for reliable distributed systems. Journal of the ACM, 43(2):225–267, March 1996.

[24] E. Perera, V. Sivaraman, and A. Seneviratne, "Survey on Network Mobility Support", ACM SIGMOBILE Mobile Computing and Communications Review, 8(2):7-19, Apr 2004.

[25] X.Y. Li and I. Stojmenovic,Broadcasting and topology control in wireless ad hoc networks, in Handbook of Algorithms for Mobile and Wireless Networking and Computing, (A. Boukerche and I. Chlamtac, eds.), CRC Press, to appear.

[26] X. Chen and J. Wu, Multicasting techniques in mobile ad hoc networks, in The handbook of ad hoc wireless networks, CRC presss. Pages 25-40, 2003.

[27] I. Stojmenovic, Geocasting in ad hoc and sensor networks, in Theoretical and Algorithmic Aspects of Sensor, Ad Hoc Wireless and Peer-to-Peer Networks (Jie Wu, ed.), Auerbach Publications (Taylor & Francis Group), 2006, 79-97.

[28] ITU-T Recommendation X.200 (1994) | ISO/IEC 7498-1:1994, Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model (and corresponding references therein)

[29] ITU-T Rec. X.901 | ISO/IEC 10746-1: Information technology — Open Distributed Processing — Reference model: Overview (and corresponding references therein)

[30] Boris Beizer. Software Testing Techniques. Van Nostrand Reinhold, 2nd edition, 1990

[31] T.H. Tse, Stephan S. Yau, W.K. Chan, Heng Lu. Testing Context-Sensitive Middleware-Based Software Applications, Proceedings of the 28th Annual International Computer Software and Application Conference (COMPSAC 2004), pp.458-466, IEEE CS Press, 2004.

[32] Satyajit Achrya, Chris George, Hrushikesha Mohanty. Specifying a Mobile Computing Infrastructure and Services, 1st International Conference on Distributed Computing and Internet Technology (ICDCIT 2004), LNCS 3347, pp.244-254, Springer-Verlag Berlin Heidelberg, 2004

[33] Satyajit Acharya, Hrushikesha Mohanty, R.K Shyamasundar. MOBICHARTS: A Notation to Specify Mobile Computing Applications. Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS’03), IEEE CS Press, 2003.

[34] Vincenzo Grassi, Raffaela Mirandola, Antonino Sabetta. A UML Profile to Model Mobile System, UML 2004,

[35] Hubert Baumeister et al. UML for Global Computing. Global Computing: Programming Environments, Languages, Security, and Analyisis of Systems, GC 2003, LNCS 2874, pp. 1-24, Springer-Verlag Berlin Heidelberg, 2003

[36] W.K. Chan, T.Y. Chen, Heng Lu. A Metamorphic Approach to Integration Testing of Context-Sensitive Middleware-Based Applications, Proceedings of the 5th International Conference on Quality Software (QSIC’05), pp.241-249, IEEE CS Press, 2005

[37] Karl R.P.H Leung, Joseph K-Y Ng, W.L. Yeung. Embedded Program Testing in Untestable Mobile Environment: An Experience of Trustworthiness Approach, Proceedings of the 11th Asia-Pacific Software Engineering Conference, pp.430-437, IEEE CS Press, 2004

[38] de Bruin, D.; Kroon, J.; van Klaverem, R.: Nelisse, M.. Design and test of a cooperative adaptive cruize control system, Intelligent Vehicles symposium, pp.392-396, IEEE CS Press, 2004

[39] Christoph Schroth et al. Simulating the traffic effects of vehicle-to-vehicle messaging systems, Proceedings of ITS Telecommunication, 2005

[40] Ricardo Morla, Nigel Davies. Evaluating a Location-Based Application: A Hybrid Test and Simulation Environment, IEEE Pervasive computing, Vol.3, No.2, pp.48-56, July-September 2004

[41] J.Barton, V. Vijayaragharan. Ubiwize: A Simulator for Ubiquitous Computing Systems Design, Technical report HPL-2003-93, Hewlett-Packard Labs, 2003

[42] Kumaresan Sanmiglingam, Geogre Coulouris. A Generic Location Event Simulator, UbiComp 2002, LNCS 2498, pp.308-315, Springer-Verlag Berlin Heidelberg, 2002

[43] P. Thévenod-Fosse, H. Waeselynck and Y. Crouzet, “Software statistical testing”, in Predictably Dependable Computing Systems, Springer Verlag, pp. 253-272, 1995

[44] C. Perkins, “IP Mobility Support for Ipv4”, IETF RFC 3344, August 2002

[45] J. Rosenberg, et al., “Session Initiation Protocol”, IETF RFC 3261, June 2002

[46] R. Stewart, et al., “Stream Control Transmission Protocol”, IETF RFC 2960, Oct. 2000

[47] Flaviu Cristian, Christof Fetzer. The timed asynchronous system model. In Proceedings of the 28th Annual International Symposium on Fault-Tolerant Computing, pp.140-149, Munich, Germany, June 1998. IEEE CS Press.

[48] Paulo Veríssimo, António Casimiro. The timely computing base model and architecture. IEEE Transactions on Computers, 51(8):916–930, 2002.

[49] P. Veríssimo and L. Rodrigues. Distributed Systems for System Architects. Kluwer Academic Publishers, 2001.

Abbreviations

AC	Access Control
C2CC	Car-to-Car Communication
CAC	Connection Admission Control
DFCD	Decentralized Floating Car Data
DSRC	Dedicated Short Range Communication
FCD	Floating Car Data
HWI	Hazard warning and information from other vehicles
IAM	Internet access on the move
IP	Internet Protocol
HC	Handover Control
HLR	Home Location Register
LAN	Local Area Network
LC	Load Control
MSU	Maintenance and Software Updates
PC	Power Control
PS	Packet Scheduler
QoS	Quality of Service
RDS	Radio Data System (broadcasts e.g. radio station name for displaying)
RM	Reference Model
RRM	Radio Resource Management
RTC	Interactive real-time communication between cars/terminals
SIP	Session Initiation Protocol (used in 3.2.1.8 when describing session control)
TFC	Traffic Flow Control
TMC	Traffic Message Channel (inside RDS)
TSE	Traffic Sign Extension
VLR	Visiting Location Register
WLAN	Wireless LAN
WP	Work Package (referring to HIDENETS WPs)

1. Executive Summary

Overall objectives of WP1 & context of this deliverable:

This deliverable contributes to the HIDENETS main objectives, which are stated in the Technical Annex:

"HIDENETS addresses the provision of available and resilient distributed applications and mobile services with critical requirements on highly dynamic and possibly unreliable open communication infrastructures."

WP 1 contributes to this overall objective with its activities and takes on an umbrella function. WP1 shall set the framework of the project at its beginning and summarize its results in the end. The WP 1 objectives, as stated in the Technical Annex, are:

"The aim of this WP is to identify a set of use case scenarios that, on the one hand will exemplify those application areas that will benefit from the technology that will be developed in HIDENETS, and on the other hand, serve as a source of requirements and technical challenges to be addressed by the project. This WP will be based on joint work with the contributions of all project partners.

Based on the analysis of the application scenarios, a reference model will be defined to specify the needs and requirements that will be addressed by the design and validation activities to be carried within WP2, WP3, WP4, WP5 and WP6. During the course of the project, the reference model will be refined as a result of feedback from these work packages. At the end of the project, the main lessons learned from all the activities carried out during the project with respect to the deployment and validation of dependable and resilient applications in HIDENETS scenarios will be consolidated and guidelines will be provided to help potential users (telecommunication operators, automotive industry, end-users) to utilize the technology and methods developed in the project."

The WP1 activities have started at the beginning of the HIDENETS project and cover the full 3-year duration of the project. This deliverable documents initial results, namely the applications and use-cases, which are finalized (except for smaller on-demand refinements) as well as intermediate results for work-in progress. The latter refers to the preliminary reference model and the business impact analysis. Both the finalized and intermediate results are important input for all other WPs for the development of the HIDENETS solutions.

Summary of the content of the deliverable:

This deliverable represents the results from the starting phase of HIDENETS. Its task is to identify a relevant set of applications, to summarize them in use cases and to represent a preliminary description of the HIDENETS reference model. Since the overall goal of HIDENETS is to develop end-to-end dependability solutions, Section 2 develops a conceptual framework, starting with a dependability framework. This dependability framework contains the relevant terminology, dependability threats, and general requirements and is built upon a HIDENETS-relevant subset of existing state-of-the-art views in the scientific dependability community. Furthermore, the dependability framework contains a first list of relevant functionalities, both for communication and middleware services, which will act as input for the architectural discussions in WPs 2 and 3.

The dependability solutions that will be developed in HIDENETS will be of general applicability, but in order to guide their development, a set of applications and use-cases has been selected, mostly from the field of car-to-car and car-to-infrastructure communications. This is motivated by the fact that this class of applications optimally represents the overall scenario of HIDENETS, i.e. highly dynamic ad hoc networking together with challenging issues in the fixed network domain. The set of in total 17 applications covers a wide range of dependability requirements, from infotainment-type applications with less strict dependability requirements to safety-critical and very real-time dependent applications like platooning. These applications together with their requirements on service-level middleware and on network communication are listed in Section 3. The requirements are derived from the HIDENETS perspective, i.e. in relation to the dependability framework in Sect. 2.

Certain subsets of these applications have been grouped in six HIDENETS use-cases, which will be the basis for the development of the dependability solutions in all other work-packages. These use-cases are: Platooning, Infotainment, Car Accident, Assisted Transportation, Brigade Communication, and Ad-hoc Service Discovery. After a description of each use-case, application-specific architectural aspects are identified, including the main functional entities (actors) and their roles. As an outcome of the analysis of different failure modes, challenges for the dependability solutions are identified. These use cases represent a framework for more detailed investigations in subsequent HIDENETS WPs. This does not necessarily imply that all use cases with all their applications will be covered by subsequent WPs. Rather, the WPs following WP1 will make a selection of use cases and applications that will be further investigated.

The business analysis part creates a relation between the technical description and the economical effects of the HIDENETS project. It is a starting point, though, and WP1 intends to further refine it in the course of the project.

Finally, this document contains a preliminary reference model in Section 6, by summarizing high-level architectural assumptions for the HIDENETS solutions in addition to the already described dependability framework in Sect 2. This HIDENETS reference model will be further developed subsequently in close cooperation with the other Work-packages. Therefore, the preliminary version also contains a description of envisioned contributions from other WPs to be investigated further in the future. The final reference model will be an important outcome of HIDENETS. It will summarize all results obtained in the project in a generalized and abstract manner, such that subsequent projects and developments in the field can use it as a guideline for their work.

In summary, the identified use-cases and their requirements clearly show the large number of dependability related challenges. First steps towards technical solutions have been made in this report in the preliminary reference model, while the other work-packages have started in the meanwhile to develop such solutions further based on ‘middleware technology’ (WP2), ‘communication protocols’ (WP3), ‘quantitative analysis methodology’ (WP4), and design and testing methodology (WP5).

2. Conceptual framework

This section introduces some basic concepts that will be used in this document to characterize the applications relevant to Hidenets. The concepts cover three main perspectives: 1) the dependability framework defines the properties, the threats, and the fault tolerance related requirements; 2) the communication level services and requirements introduce some relevant concepts covering the communication and networking layers, and 3) the middleware level services and properties introduce potential relevant services to Hidenets applications, that can be deployed at the middleware layer to fulfill the dependability requirements of the applications.

2.1 Dependability Framework

Before describing the main issues to be addressed in the specification of dependability related requirements, it is important to give precise definitions of the concepts and terminology used in this document. The definitions presented in the following are based on the dependability concepts that have been developed and updated since the mid-seventies by the Fault-Tolerant Computing community, and especially the IFIP Working Group 10.4, [5] [6] [7] [8] [9]. It is noteworthy that other concepts similar to dependability exist, such as survivability, trustworthiness and resilience (see e.g. [6] for a definition of some of these concepts and a comparison with dependability).

Dependability is the ability to deliver service that can justifiably be trusted. The *service* delivered by a system (in its role as a service provider) is its behaviour as perceived by its user(s). The *function* of a system is what the system is intended to do and is described by the *functional specification* in terms of functionality and performance. *Correct service* is delivered when the service implements the system function. A *service failure* occurs when the delivered service deviates from correct service. A failure is thus a transition from correct service to *incorrect service*. The period of delivery of incorrect service is a *service outage*. The transition from incorrect service to correct service is a *service restoration*. Based on the definition of failure, an alternate definition of dependability, which complements the initial definition in providing a criterion for deciding if the service is dependable, is as follows: the ability of a system to avoid service failures that are more frequent and more severe than is acceptable.

A systematic exposition of dependability consists of three main parts: the *threats* to, the *attributes* of and the *means* by which dependability is attained. The dependability threats correspond to faults, errors and failures that might affect the service(s) delivered by the system. The dependability attributes define the main facets of dependability that are relevant for the target system and applications. The dependability means correspond to the methods and techniques used to support the production of a dependable system. These means can be classified into four major categories:

- *fault prevention*: to prevent the occurrence or introduction of faults,
- *fault tolerance*: to avoid service failures in the presence of faults,
- *fault removal*: to reduce the number and severity of faults,
- *fault forecasting*: to estimate the present number, the future incidence, and the likely consequences of faults.

Fault prevention and fault tolerance aim to provide the ability to deliver a service that can be trusted, while fault removal and fault forecasting aim to reach confidence in this ability by justifying that the functional and the dependability and security specifications are adequate and that the system is likely to meet them.

Fault prevention is part of general engineering and can be attained through the use of rigorous development techniques, high-level specification and design methodologies, structured programming, information hiding, modularization, etc.

Fault tolerance which is aimed at failure avoidance is generally implemented by error detection and subsequent system recovery. More details about these techniques are provided in section 2.1.3.

Fault removal is performed both during the development phase and the operational life of a system. During the development, it consists of three steps: verification, diagnosis, and correction. Verification is the process of checking whether the system adheres to given properties, termed the verification conditions. If does not, the other two steps are applied. Verification activities are generally implemented using a combination of static analysis, model checking, theorem proving, testing, etc.

Finally, fault forecasting is conducted by performing an evaluation of the system behaviour with respect to fault occurrence or activation. Evaluation has two aspects: a) qualitative, or ordinal evaluation which aims to identify, classify and rank the failure modes or the combinations of event that would lead to system failures, and b) quantitative, or probabilistic, evaluation, which aims to evaluate in terms of probabilities the extent to which some of the attributes of dependability are satisfied; those attributes are then viewed as measures of dependability. Various methods can be used to support these evaluations, including analytical modelling, simulation, experimental measurements as well as judgements.

The solutions investigated in the Hidenets project cover various dimensions of dependability taking into account the four classes of dependability means (fault prevention, fault tolerance, fault removal, and fault forecasting). The development of these solutions will be based on the analysis of the specific requirements and challenges characterizing various applications and use case scenarios presented in the following sections.

Based on the concepts defined above, three main parts should be considered for the specification of *dependability related requirements* characterizing the different use cases and applications that are relevant to Hidenets:

- *Properties*: to specify the desired dependability attributes
- *Threats*: to describe the list of faults and failures that might affect the service(s) delivered by the applications and systems under study in case these are not addressed properly.
- *Fault Tolerance*: to specify the error detection and recovery strategies to be implemented to cope with the dependability threats in order to satisfy the targeted dependability properties.

Besides fault tolerance requirements, it is noteworthy that the other dependability means are equally important and should be addressed carefully. As regards the fault forecasting process, the specification of the dependability threats and dependability properties can be considered as a part of this process. Concerning the fault prevention and fault removal processes, we don’t think that specific requirements need to be specified and described for each application or use case considered in the following. A more general discussion about the challenges and solutions explored with respect to fault prevention and fault removal, considering in particular design methodologies and testing, are discussed in section 7.5 in the context of the Hidenets reference model.

A brief description of the main issues to be addressed for the specification of the dependability properties, threats, and fault tolerance related requirements are described in the following three subsections.

2.1.1 Dependability related properties

Depending on the applications considered, different facets of dependability may be important, i.e., different emphasis may be put on different attributes of dependability. Basic dependability attributes are defined as follows:

- *availability*: readiness for correct service
- *reliability*: continuity for correct service
- *safety*: absence of catastrophic consequences on the user(s) and the environment
- *confidentiality*: absence of unauthorized disclosure of information
- *integrity*: absence of improper system alterations

- *maintainability*: ability to undergo modifications and repairs

Several other dependability attributes can be obtained as combinations or specialization of the primary attributes listed above. In particular, *security* is defined as the concurrent existence of a) availability for authorised users only, b) confidentiality and c) integrity where 'improper' means 'unauthorised'.

The attributes of dependability may be emphasised to a greater or a lesser extent depending on the application: availability, integrity and maintainability are generally required, although to a varying degree depending on the application, whereas reliability, safety and confidentiality may or may not be required. The extent to which a system possesses the attributes of dependability should be considered in a relative, probabilistic sense, and not in an absolute, deterministic sense. Due to the unavoidable presence or occurrence of faults, systems are never totally available, reliable, safe or secure.

Integrity is a prerequisite for availability, reliability and safety, but may not be so for confidentiality (for instance, attacks via covert channels or passive listening can lead to a loss of confidentiality, without impairing integrity). The definition given above for integrity — absence of improper system alterations extends the usual definition as follows: (a) when a system implements an authorisation policy, 'improper' encompasses 'unauthorised'; (b) 'improper alterations' encompass actions that prevent (correct) upgrades of information; (c) 'system state' encompasses hardware modifications or damages.

Besides the attributes listed above, other secondary attributes can be considered to refine the primary attributes. An example of such a secondary attribute is *robustness*, i.e., dependability with respect to external faults, which characterises a system's reaction to a specific class of faults.

The notion of secondary attributes is especially relevant for security, when we distinguish among various types of information. Examples of such secondary attributes are:

- *accountability*: availability and integrity of the identity of the person who performed an operation
- *authenticity*: integrity of a message content and origin, and possibly of some other information, such as the time of emission.
- *nonrepudiability*: availability and integrity of the identity of the sender of a message (nonrepudiation of the origin), or the receiver (nonrepudidation of reception)

Variations in the emphasis on the different attributes of dependability directly affect the appropriate balance of the techniques (fault prevention, tolerance, removal, forecasting) to be employed in order to make the resulting systems dependable. This problem is all the more difficult as some attributes conflict (e.g., availability and safety, availability and security), necessitating design trade-offs.

2.1.2 Dependability related threats

The dependability threats mainly correspond to the faults, errors, and failures that should be covered by the target applications to satisfy the desired dependability properties.

A service may fail either because it does not comply with the functional specification, or because this specification did not adequately describe the system function. A service failure occurs when at least one or more external state(s) of the system deviate from the correct service state. The deviation is called an *error*. The adjudged or hypothesized cause of an error is called a *fault*.

A system may not, and generally does not, always fail in the same way. The ways a system can fail are its *failure modes*, which may be characterised according to four viewpoints:

- the *failure domain* which leads to the distinction of *content failures* (e.g. incorrect values), *timing failures* (e.g. service delivered too early or too late).
- the *detectability of failures* which addresses the signalling of the service failures to the users,
- the *consistency of failures* when two or more service users are involved leading to the distinction of *consistent failures* (when the incorrect service is perceived identically by all the users) from *inconsistent failures*, usually called byzantine failures, (when some or all users perceive differently incorrect service)

- the *consequences of failures* on the environment which leads to the grading of failure modes according to different *failure severities*.

When designing a dependable system, it is very important to identify which fault classes are to be taken into account because different means are to be used to deal with different fault classes. Thus, fault assumptions influence directly the design choices, and also the level of dependability that can be achieved.

Faults and their sources are very diverse. They can be classified according to different criteria: the phase of creation (development *vs.* operational faults), the system boundaries (internal *vs.* external faults), their phenomenological cause (natural *vs.* human-made faults), the dimension (hardware *vs.* software faults), the persistence (permanent *vs.* transient faults), the objective of the developer or the humans interacting with the system (malicious *vs.* nonmalicious faults), their intent (deliberate *vs.* non-deliberate faults), or their capability (accidental *vs.* incompetence faults).

Malicious faults are human-made faults that are generally introduced with the malicious objective to alter the functioning of the system during use. The goals of such faults are: 1) to disrupt or halt service, causing denials of service; 2) to access confidential information; or 3) to improperly modify the system. They can be grouped into two classes: 1) malicious logic faults that encompass faults introduced during the development phase such as Trojan horses, logic or timing bombs, and trapdoors, as well as operational faults such as viruses, worms or zombies (see e.g. [6] for a precise definition of these terms); and 2) intrusion attempts that are operational external faults. The external character of intrusion attempts does not exclude the possibility that they may be performed by system operators or administrators who are exceeding their rights.

The list of failures and faults assumptions to be addressed in the development process should be completed by the specification of the acceptable degraded operation modes as well as of the constraints imposed on each mode, i.e., the maximal tolerable service interruption duration and the number of consecutive and simultaneous failures to be tolerated, before moving to the next degraded operation mode. The analysis of the impact of the simultaneous loss or degradation of multiple functions and services requires particular attention. Depending on the dependability needs and the system failure consequences on the environment, the need to handle more than one nearly concurrent failure modes could be vital. Such an analysis is particularly useful for the specification of the minimal level of fault tolerance that must be provided by the system to satisfy the dependability objectives. It also provides preliminary information for the minimal separation between critical functions that is needed to limit their interactions and prevent common mode failures.

2.1.3 Fault tolerance related requirements

Fault tolerance is aimed at failure avoidance. It is generally implemented by *error detection* and subsequent *system recovery* (or simply recovery).

There exist two classes of error detection techniques:

- *concurrent error detection* which takes place during service delivery
- *preemptive error detection* which takes place while service delivery is suspended; it checks the system for latent errors (i.e., that are not yet detected) and dormant faults (i.e., that are not yet activated).

Recovery transforms a system state that contains one or more errors (and possibly faults) into a state without detected errors and faults that can be activated again. Recovery consists of error handling and fault handling.

Error handling eliminates errors from the system state. It may take three forms:

- *rollback,* where the state transformation consists of returning the system back to a saved state that existed prior to error detection; that saved state is a *checkpoint*,
- *compensation,* where the erroneous state contains enough redundancy to enable error elimination,
- *rollforward,* where the state without detected errors is a new state.

Fault handling prevents faults from being activated again. It involves four steps:

- *fault diagnosis,* which identifies and records the cause(s) of error(s) in terms of both location and type,
- *fault isolation,* which performs physical or logical exclusion of the faulty components from further participation in service delivery,
- *system reconfiguration,* which either switches in spare components or reassigns tasks among non-failed components,
- *system reinitialization,* which checks, updates and records the new configuration and updates system tables and records,

Usually, fault handling is followed by corrective maintenance that removes faults isolated by fault handling.

Systematic usage of compensation may allow recovery without error detection. This form of recovery is called fault masking. However, such simple masking will conceal a possibly progressive and eventually fatal loss of protective redundancy; thus practical implementations of masking generally involve error detection (and possibly fault handling), leading to *masking and recovery*.

The choice of error detection, error handling and fault handling techniques, and of their implementation is directly related to and strongly dependent upon the fault assumptions. The classes of faults that can actually be tolerated depend on the fault assumptions considered in the development process. Various techniques for achieving fault tolerance can be used such as performing multiple computations in multiple channels, either sequentially or concurrently, where the channels may be of identical design (if the objective is to tolerate independent physical faults or elusive design faults) or may implement the same function via separate designs and implementations, i.e. through *design diversity*, (if the objective is to tolerate solide design faults). Other techniques include the use of self-checking components which provide the ability to define error confinement areas.

Fault tolerance is a recursive concept: it is essential that the mechanisms that implement fault tolerance should be protected against the faults that might affect them. Examples of such protection are voter replication, self-checking checkers, stable memory for recovery programs and data.

Systematic introduction of fault tolerance is often facilitated by the addition of support systems specialized for fault tolerance (e.g., software monitors, service processors, dedicated communication links).

Fault tolerance is not restricted to accidental faults. Some mechanisms of error detection are directed towards both malicious and non-malicious faults (e.g., memory access protection techniques) and schemes have been proposed for the tolerance of both intrusions and physical faults, via information fragmentation and dispersal, as well as for tolerance of malicious logic, and more specifically of viruses, either via control flow checking, or via design diversity. It is noteworthy that the extension and adaptation to security of traditional techniques for tolerating accidental faults, led to the emergence of the *intrusion tolerance* concept. The focus of intrusion tolerance is on ensuring that systems will remain operational (possibly in a degraded mode) and continue to provide core services despite faults due to intrusions.

2.2 Communication level services and properties

The HIDENETS solutions will include functionality both on communication level as well as service-middleware. As indicated in the reference model in section 6, these two blocks will be architecturally separated but still closely coupled via well-defined interfaces, which will be developed in detail in WPs 2 and 3.

As a starting point for the development of the reference model and for the development of the detailed HIDENETS architecture in WPs 2 and 3, this section provides a first list of relevant functionalities for the communication level; the corresponding middleware functionalities are described in Sect 2.3.

The communication functions as defined here consist of the OSI layers 2-4 (link, network, and transport layers), as well as partly layer 5 (session layer). It is assumed that IP is used at the network layer, while several protocols, including TCP and UDP, may be used at the OSI transport layer. At the underlying link

layer, IEEE WLANs in infrastructure and ad-hoc modes, as well as traditional mobile networks like UMTS and GPRS will be used.

The communication-level requirements depend on requirements from the applications and middleware. Actual quantification of communication-level requirements will depend on the use case as well since the same application may pose different requirements based on the setting where it is used. In general, the communication functions transport chunks of information from one network node to another. Basically what the middleware does is to abstract some details of the underlying layers for the application running on top.

This chapter describes a set of common communication-level services that may help to provide important dependability properties. It also lists some communication-level properties. These may represent a measure for the behaviour of the services.

2.2.1 Network protocols and services

2.2.1.1 Wireless link layer protocols

In general, link layer protocols in wireless networks control access to the shared radio medium. There is however great differences between the goal and strategy of such protocols in licence-based mobile networks like UMTS/GPRS/GSM, and licence-free wireless networks like WLANs.

UMTS/GSM/GPRS networks exercise strict, centralized control over the radio resource. The goal is to ensure the QoS agreed upon during session setup and reject sessions when resources can no longer be guaranteed. Radio Resource Management (RRM) is a well-defined concept that aims to guarantee the agreed QoS, maintain the planned coverage area, and offer a high overall system capacity. The following functions are included:

- *Admission control* (AC) handles all new incoming traffic and checks if a new connection can be admitted to the system. AC is based on the resource and QoS requirements of the individual connections, and the available resources of the system. AC is also involved during handovers and connection modifications.
- *Load control* (LC) is used to avoid that the system load gets too high with the consequence that the system breaks down. When the load exceeds a given threshold, actions are taken to reduce the load.
- The *Packet Scheduler* (PS) divides the air transmission time between the active connections according to the resources that have been reserved for the individual connections.
- *Handover Control* (HC). When a connection is handed over from one access point to another is handled by the handover control.
- *Power Control* (PC) maintains the radio link quality and controls the power used.

In 802.11 WLANs [10], the goal is to maximise the network utilization regardless of the number of transmitting terminals, while giving each terminal a fair share of the available bandwidth. Radio resource management is inherent in the media access protocol, and only offers loose control over the radio resource. The original WLAN standards provide a best effort service, and when the offered traffic load is too high, the overall network performance drops. The new WLAN standard 802.11e [11] will provide functions for differentiated (not guaranteed) QoS, and some control over the radio resource can be obtained by fine-tuning these WLAN parameters.

802.11 WLANs may operate in infrastructure mode or in ad-hoc mode. In infrastructure mode the terminals connect to an access point and all communication between the terminals are transmitted via the access point. Several WLANs in infrastructure mode may be grouped to construct larger infrastructure networks. WLANs in infrastructure mode are normally connected to wired infrastructure in that the access points often have a wired interface in addition to the wireless one and forward traffic between the two domains.

In ad-hoc mode, the terminals that are within mutual communication range of each other via the wireless medium communicate directly with each other (without using an access point). WLANs in ad-hoc mode can be connected to wired networks using network layer protocols outside of the WLAN standard, like for

instance Internet Protocol (IP). If for instance one of the terminals is configured as a router and has a wired interface in addition to the wireless ad-hoc interface, it may forward network layer packets between the wireless and the wired domains. The term ad-hoc network is normally used to denote wireless networks in ad-hoc mode with all terminals configured as routers. In such cases network layer routing protocols are used between the terminals to compute the best paths. These protocols are further described in section 2.2.1.2.

The 802.11s extension to IEEE 802.11 provides features for mesh networking. Here, the ad hoc functionality with routing (also referred to as Path Selection and Forwarding) is implemented at the link-layer, while all multi-hop features are hidden for the overlying layers. This means that the network layer treats the multi-hop mesh network as a logical link, and uses protocols for address resolution and host configuration over the mesh network as it does over any other link. For the path selection, 802.11s reuses functionality from regular (network-layer) ad-hoc routing protocols, and complement them with radio metrics. 802.11s will have separate mechanisms for intra-mesh congestion and rate control. A particularly interesting feature of 802.11s is its optional multi-channel operation. One radio channel will be used for control, while other channels might be used for the transmission of the data frames. This will probably improve the network performance and scalability.

2.2.1.2 Routing and forwarding

IP Routing is the process of selecting the best paths for forwarding of data in an IP-network. Three steps are normally carried out: neighbour discovery, information dissemination and shortest path calculations.

The goal of **neighbour discovery** is to determine the set of other routers within direct communications range, i.e. within one hop. Neighbour discovery can also include resource and capability discovery. Both bidirectional and unidirectional neighbour associations can be imagined. Neighbour discovery will mainly take place at layer 3, however with support from layer 2. Exchange of layer 3 “hello” messages is the most common approach.

In fixed networks, associations are normally kept with all directly connected routers, while in ad-hoc networks, the increased dynamics, for instance due to node movement, requires extra topology control functions. **Topology control** can be defined as the problem of computing, establishing and maintaining a connected topology among the routers. At each node, this is achieved by associating with all or only a subset of the neighbouring nodes. Optimisation criteria for the topology control could for instance be reducing energy consumption, improving network capacity, or keeping network stability. In the latter case, connections that have a probability of surviving more than a minimum amount of time should be chosen. For advanced topology control, information geographical position, speed and direction of the hosting mobile node plus the nodes in the vicinity is needed.

Once the neighbouring routers have been discovered, routing protocol messages are used to exchange information about available links and other relevant parameters. In this way, each router obtains a complete picture of the network topology and all possible paths to each destination. This information is inserted into the routing table. Shortest path calculations are performed on this topology, and the resulting best path(s) towards each destination is inserted into the forwarding table. Maintaining and calculating backup routes is also an important part of this module.

Many different routing protocols exist. In particular, different routing protocols are used for fixed/wireless networks and ad-hoc networks, the latter being designed to handle the increased topology dynamics for instance due to node motion. Commonly used routing protocols in infrastructure networks include Open Shortest Path First (OSPF) [12] and Intermediate System to Intermediate System (IS-IS) [13] for routing within domains and Border Gateway Protocol [14] for routing between domains. In ad-hoc networks, proactive routing protocols such as Optimised Link State Routing Protocol (OLSR) [15] or Wireless Open Shortest Path First (WOSPF) [16], [17], and reactive protocols such as Ad hoc On-Demand Distance Vector (AODV) [18] are generally used.

Packets are forwarded hop-by-hop based on the next-hop router found in each router’s forwarding table. The forwarding function may also handle rerouting of packets according to pre-planned backup paths/next-hops.

Particular attention must be devoted to routing over the interface between the ad-hoc network and the infrastructure. In particular, gateway functionality must reside at the node interconnecting the two domains.

2.2.1.3 Broadcast/Multicast/GeoCast

When information is broadcasted [25] in the infrastructure mode, it is distributed to all nodes on a particular subnet. For a message in an ad-hoc network to reach all members of the network, the message must be re-broadcasted/flooded by the nodes. This requires special functionality to allow a node to decide whether a message should be re-broadcast further or not.

In general, network-layer multicast [26] delivers a message to only a defined set of receivers. Multicast group addresses define sets of receivers that should receive particular information, and the receivers may signal that they want to be included or excluded from the different groups.

GeoCast [27] denotes multicasting or broadcasting to all hosts within a geographically limited area. The group membership is then implicit and based on the physical location of the node. GeoCast is designed for use in Mobile Ad-Hoc Networks (MANETs).

2.2.1.4 Error detection and performance monitoring

To achieve increased communication reliability, it is important to detect errors, failures and performance degradation. Such detection can be based on the reception of explicit alarm messages from external or internal sources. Alternatively, error detection can be achieved through performance monitoring, detecting for instance loss of signal, loss of ‘hello” messages, degradation in signal quality, packet loss, bit errors etc. Network performance monitoring may collect information on packet loss (ratio), delay, throughput etc.

In addition to detecting failures and serious performance degradation, information on these parameters may be used for e.g. trend analysis for network planning and dimensioning, monitoring Service Level Agreements (SLAs), traffic engineering, and intrusion detection.

Failure detection and performance monitoring can in principal be carried out on any protocol layer, but in this context we are mainly addressing layer 2-4.

There are two main groups of performance measurements: active and passive. With active measurements, traffic is injected into the system (network), and the response is measured. For passive measurements, existing traffic is monitored. Both techniques have their pros and cons, and care must be taken to find scalable and effective solutions for the Hidenets use cases.

Performance monitoring can be used to infer the performance of a link, a path in the network, a sub-graph, overall network performance, or even higher layer information like TCP connection set-up time and information on packet reordering. For the Hidenets project, the prime concerns are fast failure/degradation detection, and making the correct response in case of a failure/degradation situation. Performance monitoring and measurements can assist in detecting failures, and also in choosing the best failover links, network paths, or even higher layer components.

Examples of important parameters at the link layer are capacity, frame loss/retries, and frame delay (among others). At the network layer we have throughput, packet loss (ratio) and packet delay as the most important parameters. Further, at the transport layer, packet reordering can be used to detect anomalies, as well as connection set-up time. At higher layers, the result will depend on the components operating at that layer, as well as all the lower layer components involved.

2.2.1.5 Mobility management

In ad-hoc networks, the routing protocols inherently handle node mobility. When a node operating in infrastructure mode WLAN is handed over from one access point to another which may lead to a change of IP address, separate mobility support functions must ensure that sessions are not broken. The solution may

be based on mobile IP, [44]. Other technologies that might be used are Session Initiation Protocol (SIP), [45], Stream Control Transmission Protocol (SCTP), [46], and possibly Network Mobility (NEMO) [24].

An important aspect of mobility in wireless networks is roaming. Roaming allows terminals/subscribers to use a mobile or internet service while outside of their home network. A roaming subscriber is a subscriber that is out of the coverage area of his own service provider, and therefore must connect to the network of another service provider to receive the service.

In order to support node mobility and roaming, cellular networks maintain two types of databases with user and location information. The Home Location Register (HLR) keeps permanent information on an operator's subscribers like profile information and account status, and also the current locations of subscribers that are presently outside of their home network. The Visiting Location Register (VLR) maintains temporary subscriber information for all (own and roaming) subscribers that are currently within the geographical area under control of the VLR. When a mobile terminal appears in a network, location registration is always performed towards the VLR in charge of that particular network, as well as to the HLR in the subscriber's home network. Using the location information in the HLR and VLR together with the nodes that control and switch traffic, roaming can be provided across networks if agreements are in place between the network providers.

Roaming in the internet can be provided by Mobile IP. The same functional split with a Home Agent and a Foreign Agent (optional) is used, but in addition to handling control and location information, these agents are also involved in transporting the actual traffic (since no connections are set up in IP). The Home Agent of a mobile node maintains information on the node's current location. It intercepts traffic for the mobile node and tunnels it to the visited network using a temporary IP address. If there is a Foreign Agent on the visited network, it receives the tunnelled packets, unpacks them and forwards them to the mobile node. If there is no Foreign Agent, the mobile node itself receives the tunnelled packets and unpacks them. Reverse traffic from the mobile node to the originating node can either be sent directly or tunnelled back to the Home Agent, which unpacks and forwards it to the originating node.

The roaming services are relevant in a scenario with several network providers.

2.2.1.6 Transport protocol

A Transport protocol is responsible for handling the end-to-end communication between two communicating terminals/servers. In the car-to-car case, transport layer functionality is present in all cars/terminals, but for a particular connection, it is only in use in the sender and receiver and not in the intermediate nodes. The transport protocol may or may not include functions as:

- Transport connection establishment and release
- Reliable or unreliable data transfer
- Sequence numbering
- Error control - detection and correction
- Flow control
- Congestion control
- Time stamping.

Connection establishment and sequence numbering enable reactive error control mechanisms, which means that the sender keeps track of which data has been correctly received or not, and if necessary asks for a retransmission. Proactive error control mechanisms add redundant information to enable correction of bit/packet errors (e.g. based on checksum). Flow control ensures that the sender does not overwhelm the receiver with data, and congestion control ensures that the senders do not overload the network.

Transport protocols implement one or more of the above mentioned functions, and the choice of transport protocol in effect determines the type of service that is offered to the session layer. Examples of Internet

transport protocols that give quite different services are TCP (Transmission Control Protocol) and UDP (User Datagram Protocol).

2.2.1.7 Cross-layer optimisation

Cross-layer optimisation is the task of reducing the number of redundant functions and overhead in the different protocol layers. For some type of functions, redundancy in different layers is only a matter of resource usage. For instance, having error control in OSI layer 2 and 4 is a trade-off between increased transmission volume and delay on one side and increased bit error rates and packet drop rates on the other side. Such cross-layer redundancy can even be desirable. Adding link layer error control at one particular link that is very prone to bit errors can significantly enhance the overall throughput. However, other types of redundant functions may actually work against each other. For instance, having redundant re-routing functions at different layers may lead to route flapping and routing loops when a failure occurs.

2.2.1.8 Session layer service and lower layer service middleware

Profile Management manages the configuration, updating and dissemination of the profile that describes the capabilities, properties and type of the node. A profile might describe power supply, bandwidth, media types, mobility pattern etc. This profile might be configured automatically based on context and location.

The **gateway and network selection** service selects between different network technologies and/or base stations that are available to a mobile node at a given point in time. This component may be quite simple, and only provide information on general guidelines for selecting network. Then the actual choice of network would be left to the routing module. Alternatively, the gateway/network selection component may take a more active part in the decision about switching to another network technology/base station. It would then need to gather information from other sources like for instance the performance measurement component or resource information services in the network, and generally have more intelligence and functionality.

Resource/Service Discovery is needed to find and locate resources and services available, particularly in the ad-hoc network. The discovery process resolves service names and descriptions to information that can be used to initiate the service, such as IP addresses and port numbers.

QoS and Differentiation Manager has the overall responsibility of the end-to-end QoS management of connections originating in the node and admission and rejection of connections transiting through the node; i.e. the overall QoS policy of the node.

QoS policy regimes for end-to-end QoS management may be based on resource reservation and/or differentiation/prioritization of traffic. With resource reservation, the network seeks to guarantee each traffic stream a certain amount of resources (often bandwidth). In such a case, Connection Admission Control (CAC) must be performed whenever a new traffic stream requests access to the network to check whether there are sufficient free resources to support it. Resource reservation and CAC may be applicable only for some traffic types, i.e. normally ‘long-lived’ connections with hard QoS requirements.

Differentiation, on the other hand, is based on different classes of traffic receiving a differentiation treatment with respect to queue management and scheduling. Each packet is marked according to which service class it belongs to by the QoS and differentiation control module.

Network resilience can be defined as the ability of a system/network to adapt to changes like for instance node and link faults and traffic pattern changes. In contrast to redundancy, where backup systems are installed, resilience mechanisms aim to recover from faults or even to resist being affected by them. Proper resilience mechanisms will enhance network performance and availability and thus the fault tolerance of the system.

Resilience is actually included in the QoS term as defined by [22], even though it is not often used. It may be viewed as orthogonal to other QoS requirements like delay, packet loss, and throughput. Traffic streams may have strict requirements to loss or delay, but loose requirements to resilience. Thus a voice call, which is often prioritised during normal operation because of its strict delay requirements, may be closed down after a

failover to make room for a web session with high requirements to resilience. **Resilience differentiation** is the task of providing different traffic streams with different resilience support during faults.

Session Control manages the sessions set up between applications in two or more different mobile nodes or between mobiles nodes and a server.

SIP is a relevant candidate for controlling media sessions, i.e. creating, modifying and terminating sessions with two or more participants. SIP also includes functionality for presence, event notification and instant messaging services. By exchanging session descriptions, SIP may determine the media capabilities of the target end points. Thus, sessions may be set up and modified to fit the highest level of media capabilities that can be supported by all end points. SIP can also be used to implement mobility management.

Other session layer functions like dialogue control, token management, and synchronisation may also be relevant depending on the use case chosen. In that case, such functions will be included at a later stage.

2.2.2 Communication Properties

The communication properties listed here are the ones considered important by the authors of this document. Other aspects were originally identified as important properties with regard to communication leve, called “operating context”. These, however, were dropped due to difficulties of consistency with regard to different applications and moved to the annex.

2.2.2.1 Throughput

Throughput is a measure of the amount of data transferred per unit of time. Throughput can be measured either in successfully transmitted packets (or frames) per unit of time or in bytes per unit of time. The throughput may be specified for a single connection or an aggregate. In order to be meaningful, it must be defined at which protocol level throughput is measured and thereby which protocol fields that are considered as data. Throughput is a function of the bandwidth of the underlying medium, overhead at underlying (and possibly present) layers (as viewed from the layer defined for the throughput measurements), the bit error rate, and the protocols and functions for detecting and correcting errors at underlying layers.

A related measure is *goodput*, which is the amount of data actually received correctly at the other end per unit of time (i.e. lost and errored packets are not counted). Goodput is a function of throughput and protocols and functions for detecting and correcting errors at the current layer.

2.2.2.2 Communication Delay

The *delay* is the time interval between the event that a packet is sent until it is received at the other end. For a more precise definition see ITU-T recommendation Y.1540. Keeping communication delay at a minimum is very important for some application types, in particular real-time voice and video services. Other applications may tolerate a medium level of communication delay, but are sensitive to variation in the delay (delay jitter). Some interactive applications are very sensitive to exceeding a given threshold. In such cases the goal is not to keep the delay at a minimum, but to always keep it within the threshold.

2.2.2.3 Data integrity/packet error ratio

This refers to the validity/correctness of data received. It can be compromised in a number of ways, mainly by transmission errors caused by the medium or by malfunctions of transmission hardware. The *packet error ratio* is measured as the ratio of number of errored IP packets received to the total number of (successful and errored) IP packets received.

The packet error ratio can be minimized using error detection/correction mechanisms at one or more protocol layers. There are mainly two categories of error detection/correction mechanisms: reactive mechanisms ask for retransmission after a packet error is detected, while proactive mechanisms send redundant information together with the original data. In general, all error detection/correction mechanisms increase the delay

experienced by the applications, and therefore some applications will not use such mechanisms (e.g. real-time two-way voice and video).

2.2.2.4 Packet loss

Packet loss ratio is measured as the ratio of number of lost packets to the number of transmitted packets. The packet loss pattern may also be important for some applications. For instance, certain voice and video applications are more vulnerable to loss of consecutive packets than if the packet loss is spread evenly over time.

2.2.2.5 Network congestion

Network congestion refers to the load in the network. If the network cannot carry all offered traffic, it is congested. In the application requirements later in this document, congestion will mostly be related with the amount of generated traffic.

2.3 Middleware level services and properties

The requirements of middleware are depending on the upper level requirements. Basically what the middleware does is to abstract some details of the underlying layers for the application running on top. Typical middleware technologies are Corba and Java RMI. An example of a middleware product is Resilient Telco Platform (RTP) developed by Siemens. This section lists a set of typical middleware services that are necessary to provide important dependability properties, such as availability or reliability. It also lists the properties that are commonly required by the upper (application) level, which can be provided at the middleware level through using the listed middleware services or combinations thereof. Middleware services development and investigation will be the subject of HIDENETS WP2.

2.3.1 Middleware Level Services

In this section we introduce a range of potential relevant services to Hidenets applications, to be deployed and offered as middleware level services. It is worthwhile noticing that we explicitly consider the existence of a range of very specific and basic services, which we group and refer to as *oracles*, vis-à-vis the remaining services, all of them provided by particular service *managers*. The former must be, by definition, very simple and hence reliable services, while the latter implement possibly complex functionalities, sometimes relying on the underlying oracles to improve the resilience of the services they provide.

Timeliness and trustworthiness oracles: The objective of these oracles is to exploit the intrinsic heterogeneity of computing systems, with respect to both the time and space domain. For example, a system can be more timely, or even exhibit real-time properties during certain time intervals, which can be used to provide certain services (e.g. agreement between distributed processes) that would not be feasible during the rest of the time. Or it may be possible to identify and isolate some system resources, such as when the specific network channels are created, that may be fully dedicated to perform specific functions not realizable in a homogeneous system. Further to the exploitation of intrinsic heterogeneity, systems can be constructed and designed in such a way that some components or parts are explicitly made simpler, self-contained, less prone to failures or vulnerabilities, being thus able to provide more resilient and/or timely and/or secure services. Such components can be viewed as oracles for the rest of the system, providing their services through well-defined interfaces, and thus being used as helpers for the construction of faster or more secure solutions.

Fault tolerance manager: The sources for failure are e.g. malicious or other attacks, wrong results from condensation processing, consideration of outdated data. Such failures will in many cases result in sets of contradicting data. Mechanisms must be provided to make sure that appropriate fault tolerance strategies are

used to deal with these failures. For instance, a fault tolerance manager might use fault and error handling mechanisms (see section 2.1.3) to ensure that failures are detected and removed from the network.

Consistency manager: When referring to logical consistency, the goal of a consistency handler is to perform replica management and ensure that the state of all the replicas is consistent within some defined bounds. Different degrees of (*logical*) *consistency* may be considered, which impose different requirements on the protocols for replica management, and provide different degrees of responsiveness and fault tolerance. When referring to *temporal consistency*, the objective of a consistency handler is to ensure that timing faults are correctly handled at the affected replicas. For example, a replica that becomes outdated with respect to a certain external entity whose value is being monitored, thus becoming temporally inconsistent, must be prevented to further propagate the outdated values to avoid contamination of the state of other applications. In this case a consistency handler must be able to detect the inconsistency and act upon it.

Diagnostic manager: The Diagnostic Manager monitors the components of the system in order to assess the system state. It supports other services of the middleware (QoS Adaptation, Reconfiguration and Maintenance Manager …), whose purpose is to undertake the proper actions.

QoS adaptation manager: The QoS adaptation manager has to monitor the performance of the system in order to check if the system is able to provide the required level of QoS for every specific flow of data. The QoS monitoring of lower levels (e.g. throughput, delay or connectivity) could be made using the services provided by the Diagnostic Manager. The QoS Adaptation Manager also has to adjust lower level protocol parameters and to inform the application when the required QoS cannot be sustained.

Reconfiguration and maintenance manager: The reconfiguration and maintenance manager is a configurable component that allows to: i) *reconfigure* the system in presence of faults; and ii) apply a policy of maintenance.

The Reconfiguration and Maintenance Manager uses the assessment of the state of the system obtained from the Diagnostic Manager.

The reconfiguration is part of the fault handling, in which, after a fault in the system, an attempt will be made to try and isolate the identified faults (see Section 2.1.3). The reconfiguration could activate spare components and/or assign tasks among non-failed components. The strategy for the reconfiguration can be static (in presence of a particular class of fault the manager chooses a predefined strategy) or dynamic (many strategies are defined, the one used is chosen based on the best predicted results on the dependability of the application). The application can be aware or not of this service: a transparent service is simpler to use but an application aware of this service can use it in a more precise and powerful way. This context deals with preventive maintenance; it is desired to schedule planned maintenance actions aimed at the prevention of breakdowns and failures. The primary goal of preventive maintenance is to prevent the failure of equipment before it actually occurs. Preventive maintenance is designed to preserve and enhance equipment reliability by replacing worn components before they actually fail. Software rejuvenation aimed at removing the effects of software aging before they lead to failure can also be seen as an example of preventive maintenance. The maintenance policy can be a fixed strategy or can be chosen from a set of defined policies (as for reconfiguration, the policy chosen can be based on the best predicted results on the dependability of the application). The maintenance is in general transparent to the application level. However, this must not be the case when explicit needs for application software upgrades have to be considered. In this case, reconfiguration must also ensure that new software versions can be easily installed in a safe manner. This implies the need to address additional functions, such as software licensing, software integrity, software trustworthiness and protection from software theft.

Group communication manager: We include here all the services related to group-based communication, like communication services and membership management services. Membership management usually needs the availability of failure detection services, which can be provided by a diagnostic manager. Group communication tools can be used in the construction of applications or other middleware services that involve distributed computations or cooperation among a set of participants.

2.3.2 Middleware Level Properties

Middleware level properties describe issues which can be used as performance measures for middleware performance. The properties identified in HIDENETS are:

Timeliness of data: Refers to the freshness of data, which is of most relevance when this data is received from entities in the environment, whose value is continuously varying over time. This property is necessary when some application needs to ensure that it is up-to-date (within some given bound), with respect to the context or environment in which it is being deployed. Timeliness of data is also necessary for real-time applications in general (e.g. like video conferencing, on-line gaming). Timeliness imposes constraints on communication delays to lower layers. Quite clearly, it also requires a minimum level of connectivity. Timeliness of data is compromised if pre-established bounds are not met in run-time.

Logical consistency: This property is essentially relevant when considering replication of data. It ensures that the state (the value) of every replica is consistent with each other. Consistency can be further defined as strong consistency, when every replica provides the same response to a distributed query, or as weak consistency, which allows for some replicas to be delayed with respect to others.

Temporal consistency: This property is essentially relevant in the context of real-time data representation. In brief, it ensures that at any point in time the value of some (real-time) entity stored at a replica is not too far apart from the real value of that entity at that same point in time. [49]. This applies to so-called *time-value entities*, that is, entities whose time-domain and value-domain correctness are inter-dependent. The notion of temporal consistency implies the definition of a validity constraint for the computer representations of the time-value entities (the values stored at a replica). This constraint expresses the maximum allowed difference between the representation of the time-value entity and its real value and depends on the required consistency, which is imposed by the application.

Trustworthiness of data: The concept of trustworthiness refers to the degree of confidence a service user may have that the service will perform as expected and, in particular, that it will satisfy a set of security properties. In [5], trustworthiness is considered as a similar concept for dependability. A trustworthy service is dependable with respect to security properties. In the context of communication, trustworthiness of data may be regarded as the confidence one may have that security properties for transmitted data are preserved. Some important secondary attributes of dependability that should also be considered in this context include *accountability*, *authenticity* and *nonrepudiability* (see Section 2.1.1 for precise definitions). The threats to trustworthiness include attacks from crackers or insiders, environmental disruptions or human and operator errors. The typical means by which dependability is achieved (see Section 2.1) can also be applied here.

Robustness: This specialized secondary attribute of dependability characterises systems that are dependable with respect to external faults. Therefore, the robustness of middleware solutions is especially meaningful when external faults constitute a relevant threat.

Message ordering: Different properties with respect to message ordering can be considered. FIFO ordering guarantees that messages are delivered in the same order they have been sent. Causal ordering guarantees that if a precedence relation exists between any two delivered messages, they will be delivered according to that precedence. Total ordering guarantees that any two messages are delivered to any recipients in the same order. Temporal ordering guarantees that messages are delivered in the order of their send timestamps (which requires some form of clock synchronisation to implement a useful solution). All these properties are of more relevance in the context of group communication.

Completeness, accuracy and timeliness (of failure detection): The requirement for failure detection can be specialized with particular properties that failure detectors must exhibit. The notions of completeness and accuracy to classify different classes of (crash) failure detectors have been introduced more than a decade ago by Chandra and Toueg [23], and are widely used in the distributed systems community. *Completeness* refers to the ability of a failure detector to detect every failure that occurs. *Accuracy* refers to the ability of the failure detector to not make mistakes, that is, wrongly detect failures when they do not occur. Another property that is relevant to further distinguish classes of failure detectors, in particular in systems that are not purely asynchronous, is the *timeliness* property. Timeliness refers to the ability of the failure detector to detect failures within given time bounds.

3. Applications

3.1 Overview

Applications identified in this section mostly belong to the domain of car-to-car communication. Overviews over some of such applications can be found e.g. in [1], [2], [3]. The application description in this document partly relies on the descriptions from these references. However, the focus of HIDENETS is on dependability which is not the case for the sources of the application description. This document, therefore, extends the application descriptions with all HIDENETS related subjects, in particular with dependability aspects.

Note that the applications described in this chapter are by no means exhaustive. There are other bodies and projects which have generated more complete lists of applications, see e.g. [1], [2]. The subset of applications described here has been selected for the purposes of HIDENETS only. It is noteworthy that some of the applications presented in the following share some common requirements or characteristics. The grouping of such applications into use cases is addressed in Section 5.

The application descriptions have the following structure:

- A description of the technical properties and mechanisms of the application and its representation to the user
- A section on communication and a section on middleware level requirements, both of which list requirements with regard to the communication and middleware level properties, see Sections 2.2.2 and 2.3.2.
- Where required, a section on general requirements which cannot directly be allocated to middleware and communication level properties.

3.2 Application description

An application is defined on layer 7 of the OSI communication model which relies on services of underlying layers and which has end-to-end relevance. An application involves in most cases interaction with a user but can also involve machine interaction.

The listed applications and use cases have been collected because they seem to be interesting for the purposes of HIDENETS at the beginning of the HIDENETS project. The motivation for this specific selection is that the HIDENETS members found them ideal but still challenging for further investigations. This is also reflected by the fact that they are all represented in the use cases in chapter 4. The applications that have not been considered in use cases but which were listed in the fist drafts of the document have been moved to Annex B.

Note that the fact that an application is described in this chapter does not necessarily mean that it will be covered by subsequent WPs. Rather, the WPs following WP1 will make a selection of use cases and applications that will be further investigated.

3.2.1 Floating Car Data

Floating car data (FCD) describes the process of collecting traffic flow information and calculating up-to-date information about the current traffic flow on roads. The application described here comprises both, centralized and decentralized collection and distribution of FCD.

Centralized FCD services exist already, e.g. in Germany (see e.g. Gesellschaft für Verkehrsdaten (DDG), http://www.ddg.de). The calculation of the current traffic flow model is performed by a server inside the fixed network domain. The processed traffic flow information is re-distributed to cars on the road.

Centralized FCD relies on a number of traffic information sources, such as speed measurements along motorways or information from driving cars, which are collected and processed. The redistribution of this information to vehicles on the road usually uses public networks and contains condensed and filtered information which allow to drive the best road to take. Strict control is clearly difficult to achieve, since there is no obligation to follow indications provided by this application. Major benefits could be achieved in operating contexts of high equipment ratios and high confidence (trustworthiness) on the information provided by the application. In such operating contexts it would be possible to envisage very accurate centralized monitoring of vehicle flows and consequently come up with very accurate predictions and indications. Communications considered in HIDENETS for this application are essentially one-hop and multi-hop access to fixed networks.

The decentralized version of floating car data is called decentralized floating car data (DFCD). It assumes that each vehicle which is equipped with the required devices, periodically transmits its current location, speed and driving direction. This results in a very detailed notion of the traffic flow state around receiving vehicles. In order to make sure that the amount of information remains within reasonable limits, the information is processed inside cars and condensed such that an intelligent piece of information is generated. This condensed information is then broadcasted back into the network for further distribution.

For the purposes of HIDENETS, an in-car application does not distinguish between information generated by DFCD and centralized FCD information from the fixed network. It is up to the communication layer to make sure that FCD data are distributed, regardless of the source of processing.

3.2.1.1 Communication level requirements on Floating Car Data application

The requirements with respect to the communication level properties are:

- **Throughput:** Throughput is not so much of an issue, since FCD require only little amounts of data to be transmitted (as e.g. in TMC “Traffic Message Channel” where the RDS channel with very low bandwidth is used). Lower throughput means that the frequency of data repetition may be lower but this may only have limited effect on propagation speed and data up-to-dateness
- **Communication delay:** Less critical, can be in the range of a few seconds. Nevertheless, an upper bound for the communication delay may be derived from temporal consistency constraints (see further below). Values in the range of few seconds should be adequate to still secure these constraints.
- **Data integrity / packet error ratio:** Correctness of data should remain high (packet error ratio in the range of a few percent allowed).
- **Packet loss:** Packet loss is not very critical, since data is regularly updated
- **Network congestion:** Congestion is an issue for DFCD which may happen because all kinds of information from the whole road network are repeatedly being distributed. The amount of data may, therefore, depend on the temporal consistency requirement, i.e. data which need to be available fast need to be repeated more often. It is not so much an issue for the centralized version.

3.2.1.2 Middleware level requirements on Floating Car Data application

The requirements with respect to the middleware level properties are:

- **Timeliness of data:** Data should be provided in a timely manner, where the timeliness requirement increases (i.e. lower timing bounds are necessary) with decreasing distance for DFCD. Data of the next 10 km ahead should be up-to-date within a limit of approx. 1 minute, within 100 km within 10 minutes, within 500 km within 1 hour. In the centralized version, data sent by cars are usually averaged over several minutes, aggregation and computation in the service center takes additional time. Lack of timeliness may have a significant impact on the usefulness of the application, due to temporal consistency constraints (which imposes bounds on the timeliness of data).
- **Logical consistency:** In DFCD, There will be condensation of messages by different cars. This may lead to different results which are then distributed to the neighbourhood. Different information may lead to

confusion of the driver. Therefore, logical consistency and plausibility checks need to be performed by all cars. Inconsistent data needs to be removed from the network. In the centralized version, logical consistency is less of an issue due to the centralized generation of data.

- **Temporal consistency:** It is important that only most up-to-date data is distributed in the network. This is necessary not only to ensure logical consistency, that is, that all drivers have a similar view of the actual flow situation, but also temporal consistency. Temporal consistency requirements exist because this application deals with real-time data, which is continuously changing over time, possibly becoming invalid for the purposes of the application. Even if all drivers have the same traffic flow information (logical consistency), this information must be consistent with the real situation. Each car receiving or processing FCD data needs to make sure to use only temporally consistent data, which may be achieved by using only the most recent data (possibly requiring the use of time stamps on every piece of FCD information). Specific temporal requirements (e.g. for communication delays) will depend on the dynamics of traffic flow changes. These should allow for bounds in the order of seconds, while still guaranteeing a good accuracy of the traffic flow information.
- **Trustworthiness of data:** A possible attack on DFCD is a car standing somewhere and transmitting false information into the network. It is necessary to develop methods which identify such malicious attacks and make sure that the data in the networks stems from trustworthy sources.
    - **Accountability:** It will most probably lead to acceptance problems if the originator of a traffic flow message is disclosed (possibility for police to issue speeding tickets). It is rather required to include some level of trustworthiness of the sender of a message
    - **Authenticity:** The authenticity of message content is desirable
    - **Nonrepudiability:** not required
- **Robustness:** The impact of external faults needs to be carefully investigated, however robustness does not seem to be challenging in the context of FCD
- **Message ordering:** Messages will be unordered and need not be reordered. More important is temporal consistency.
- **Completeness, accuracy and timeliness (of failure detection):** Failure detection in the sense that incorrect or outdated data is in the network needs to be performed. The ranking of the requirements are: 1. completeness – because every fault must be detected to avoid the possible contamination of condensed information, 2. accuracy – it is important to avoid mistakes when discarding supposedly faulty data, otherwise too much data may be lost, 3. timeliness –there is no specific need for timely reaction to failures, provided they are detected.

3.2.1.3 General requirements on floating car data application

Geographic extension for DFCD: Different FCD messages will be transported over different numbers of hops. Original flow information (provided by a single car about its speed, position, direction and other relevant variables) will be transported over few (2-5) hops only. After that, it is assumed that the data is condensed into something like “average speed for a piece of road”. Such condensed messages will be transported over more hops, after which they will be condensed again, and so on. The number of hops a message travels, therefore, depends heavily on the distance from the location for which the information is valid, where a general rule is that the higher the distance, the more hops a message travels.

3.2.2 Traffic sign extension

Traffic signs constitute one of the means for traffic regulation, which is independent of the kind of road and traffic characteristics. This application consists of extending traffic signs by a) allowing centralized control of the information indicated by each sign (e.g. continuous control and adjustment of speed limits) and b) allowing one-way direct communication between signs and (some) near-by cars (those to which information is concerned and should be disseminated). With respect to communication, the application is only relevant in

operating contexts with direct one-hop communication with the infrastructure (traffic signs). Remote update of traffic signs (from some central control site) is not considered here.

3.2.2.1 Communication level requirements on Traffic Sign Extension application

The requirements with respect to the communication level properties are:

- **Throughput:** The throughput is mostly determined by the repetition frequency. This frequency needs to be such that each passing car for sure receives the message from the traffic sign. This depends on the speed of the cars passing by, which will mostly be determined by the class of road and the speed limitation.
- **Communication delay:** One hop only, so delay is uncritical in the sense that small delay bounds are typically easy to ensure. Nevertheless, upper communication delay bounds may have to be defined in order to secure temporal consistency constraints.
- **Data integrity / packet error ratio:** Correctness of data has to remain high (packet error ratio $<10^{-4}$).
- **Packet loss:** Packet loss is not very critical, since data is regularly updated and the update frequency should allow for error recovery.
- **Network congestion:** Not an issue due to only one hop

3.2.2.2 Middleware level requirements on Traffic Sign Extension application

The requirements with respect to the middleware level properties are:

- **Timeliness of data:** Data must be provided within some bound to ensure that it is still sufficiently recent and useful. Due to one hop only communication, it should be easier to secure communication bounds.
- **Logical consistency:** Every driver should receive the same information. This should not be a major problem if information if broadcast to all drivers in the vicinity of a sign, which then receive the same information. Nevertheless, the occurrence of faults might raise some difficulties that must be addressed to guarantee the logical consistency among the different drivers' views.
- **Temporal consistency:** The information provided by traffic signs is normally stable during large periods of time. However, since it may change instantaneously, all previous information gets immediately outdated and should be immediately refreshed.
- **Trustworthiness of data:** It needs to be assured that the data stem from an official traffic sign and not from a malicious source.
- **Robustness:** The reaction to a failure should in any case be that the traffic sign stops transmitting rather than issuing wrong information. It may be conceivable that multiple traffic signs monitor each other. In this case, it may be possible to issue a message "your system may not detect all traffic signs" or the like.
- **Message ordering:** Not an issue. Messages from different signs are not related, and messages from the same sign are idempotent.
- **Completeness, accuracy and timeliness (of failure detection):** If a failure occurs, the detection needs to be: 1. timely (highest priority), 2. complete, 3. accurate (lowest priority). In fact, we need to know about failures as soon as possible, to let the driver know that some (possibly important) information may have been lost.

3.2.3 Unusual driver behaviour warning

Unusual driver behaviour warning comprises a number of applications that have different levels of relevance and safety impact. The warning is directly related to the behaviour of a car's driver. The warning is broadcast from the car whose driver shows unusual driver behaviour (where it is not excluded that the driver himself is

also warned). Forwarding of such messages is in most cases not required, i.e. the warning has relevance for the cars in the direct neighbourhood only.

The different applications are:

- **Sudden driving behaviour change warning**: A message is broadcast when the driver suddenly and unexpectedly brakes, accelerates, turns etc.
- **Drunken/sleep driving warning**: A message is broadcast when the driver behaviour is such that a limited control over the vehicle is probable.
- **Unusually slow vehicle warning**: A message is broadcast when a vehicle is unusually slow (e.g. driving at 40km/h on a motorway).
- **Change of lane warning**: A message is broadcast when a driver changes lanes without indicating it.
- **Intersection collision warning**: A message is broadcast when the vehicle is (likely to be) too fast to be able to respect the right of way regulations at a given crossroad.
- **Traffic law violation warning**: A message is broadcast when a driver disrespects the valid traffic law (e.g. overtaking in areas where it is not allowed, driving much too fast or not respecting the right of way).
- **Post crash warning**: A message is broadcast when a crash was sensed, e.g. by the activation of the airbags.

3.2.3.1 Communication level requirements on Unusual driver behaviour warning applications

The requirements with respect to the communication level properties are:

- **Throughput:** Small messages and one hop only, so throughput is not very critical
- **Communication delay:** One hop only, so delay is uncritical in the sense that small delay bounds are typically easy to ensure. Nevertheless, upper communication delay bounds may have to be defined in order to secure temporal consistency constraints.
- **Data integrity / packet error ratio:** Correctness of data has to remain high (errors $<10^{-4}$).
- **Packet loss:** Packet loss is critical, since data is not necessarily regularly updated or the time from the transmission of a message to a possibly severe consequence is extremely short.
- **Network congestion:** Not an issue due to only one hop

3.2.3.2 Middleware level requirements on Unusual driver behaviour warning applications

The requirements with respect to the middleware level properties are:

- **Timeliness of data:** There are timeliness constraints on received data, which depend on temporal consistency constraints, in general, and, in particular, on the physical distance between the source and the destination. Should data need to be forwarded over multiple hops, the distance to the location of danger is such that more time is left to react.
- **Logical consistency:** Multiple drivers may need to have a consistent view and warning of another driver behaviour, in order to react also consistently (e.g., both of them should deviate or decide to slow down in response to a warning).
- **Temporal consistency:** Temporal consistency is an issue, because the source represents real-time values (a state that changes with time, in a way which can be modeled by some function). Therefore, any information that is disseminated by this single source has temporal consistency constraints (the disseminated value might be useful or not, depending on how well it represents the “reality”). Due to one hop communication, it should not be a major problem to secure temporal consistency constraints.
- **Trustworthiness of data:** It needs to be assured that the data stem from a car which is in or may cause trouble and not from a malicious source.

- **Robustness:** The reaction to a failure should in any case be fail safe, i.e. not inject additional traffic threats into the road network.
- **Message ordering:** Not an issue due to only one hop and single source.
- **Completeness, accuracy and timeliness (of failure detection):** If a failure occurs, the detection needs to be: 1. timely (highest priority), 2. complete, 3. accurate (lowest priority).

3.2.3.3 General requirements on Unusual driver behaviour warning applications

This application has safety critical requirements in the sense that it deals with information that may significantly influence driving decisions. Non delivery of critical information (e.g., the sudden change of direction of another vehicle), or the delivery of incorrect information (e.g., the indicated direction is wrong), may lead to catastrophic consequences. Thus data integrity and availability are critical requirements.

3.2.4 Hazard warning between vehicles

Hazard warnings are warnings about road safety information which is relevant over multiple hops. The source of information may be one or more cars.

The applications in this group are:

- **Road condition warning:** A message is broadcasted when a car unexpectedly senses aquaplaning, ice, oil etc. on the road. Note that this application has to regard the general weather condition, as in winter ice or snow on the road is common and the broadcast of such information might lead to network congestion.
- **Traffic jam warning:** A warning is issued when a vehicle senses that it is stationed within a traffic jam. This information can be relayed along the road in order to inform cars approaching the traffic jam that they may need to break when they reach the end of the traffic jam. This application is particularly useful if the end of the traffic jam is in or behind a curve.
- **Cooperative forward collision warning:** Cooperative forward collision warning system is designed to aid the driver in avoiding or mitigating collisions with the rear-end of vehicles in the forward path of travel through driver notification or warning of the impending collision. The vehicle receives data regarding the position, velocity, heading, yaw rate, and acceleration of other vehicles in the vicinity. Using this information along with its own position, dynamics, and roadway information (map data), the vehicle will determine whether a rear-end collision with the lead vehicle is likely. In addition, the host vehicle will transmit position, velocity, acceleration, heading, and yaw rate to other vehicles.
- **Emergency vehicle alert warning:** An emergency vehicle broadcasts its approach, so cars ahead of the emergency vehicle can give way. The information may be forwarded in the emergency vehicle's driving direction by multihop communication.

3.2.4.1 Communication level requirements on hazard warning between vehicles

The requirements with respect to the communication level properties are:

- **Throughput:** Small messages and mostly one hop, so throughput is not very critical. The multiplication of the same message by many cars, however, may lead to network congestion but this needs to be handled by middleware.
- **Communication delay:** One or only few hops, so delay is uncritical in the sense that small delay bounds are typically easy to ensure. Nevertheless, upper communication delay bounds may have to be defined in order to secure temporal consistency constraints.
- **Data integrity / packet error ratio:** Correctness of data has to remain high (packet error ratio $<10^{-4}$).
- **Packet loss:** Packet loss is critical, since data is not necessarily regularly updated or the time from the transmission of a message to a possibly severe consequence is in some cases extremely short.

- **Network congestion:** Not an issue but repetition of messages needs to be handled properly.

3.2.4.2 Middleware level requirements on hazard warning between vehicles

The requirements with respect to the middleware level properties are:

- **Timeliness of data:** There are timeliness constraints on received data, which depend on temporal consistency constraints, in general, and, in particular, on the physical distance between the source and the destination. Should data need to be forwarded over multiple hops, the distance to the location of danger is such that more time is left to react.
- **Logical consistency:** Especially critical for those sub-applications where more than one source can generate the same information.
- **Temporal consistency:** May be critical for those sub-applications where information is generated for a specific geographic area which may persist over some period of time (e.g. road condition warning).
- **Trustworthiness of data:** It needs to be assured that the data stem from a car which is trustworthy and not from a malicious source.
- **Robustness:** The reaction to a failure should in any case be fail safe, i.e. not inject additional traffic threats into the road network.
- **Message ordering:** Not an issue
- **Completeness, accuracy and timeliness (of failure detection):** If a failure occurs, the detection needs to be: 1. timely (highest priority), 2. complete, 3. accurate (lowest priority).

3.2.5 Hazard warning of the own vehicle

This section describes applications which are based on information received by a car from sources in its environment which are processed by the in-car equipment for the purpose of detecting potential hazards. Note that the hazard information is not provided by external sources directly but only indirectly by the car's own system by evaluation of available information.

- **Pre crash sensing:** In case the received C2CC "Car-to-Car Communication" data shows that a car cannot avoid being involved in a traffic jam, the airbags can be heated and the safety belt tightened to limit the effects of the accident.
- **Corporate glare reduction:** In case the own high-beams might blind an approaching vehicle, the light is either tuned down or the driver is asked to do so.
- **Visibility assistance:** In case the received C2CC data shows that the distance to the vehicle in front is too small, a warning is issued to the driver to slow down. Note that the implementation of such a system requires a very large penetration rate and that it can also be realized with adaptive cruise or save distance control.
- **Overtaking collision warning:** A driver is warned when at the beginning of an overtaking manoeuvre it becomes evident that there is a vehicle approaching too fast from the opposite direction to successfully finish the overtaking. This application has to consider the own velocity, the velocity of the vehicle to be overtaken, the velocity of the approaching vehicle and on Basic Relations and Concept what type of road the vehicle is driving, as the number of lanes is decisive for the situation to be critical or not.

3.2.5.1 Communication level requirements on Hazard warning of the own vehicle application

The requirements with respect to the communication level properties are:

- **Throughput:** Small messages and mostly one hop, so throughput is not very critical.
- **Communication delay:** One or only few hops, so delay is uncritical for single messages. However, information need to be provided such that, given the speed of vehicles, reaction is still possible

- **Data integrity / packet error ratio:** Correctness of data has to remain high (packet error ratio $<10^{-4}$).
- **Packet loss:** Packet loss is critical, since data is not necessarily regularly updated or the time from the transmission of a message to a possibly severe consequence is in some cases extremely short.
- **Network congestion:** Not an issue due to limited geographic relevance

3.2.5.2 Middleware level requirements on Hazard warning of the own vehicle applications

The requirements with respect to the middleware level properties are:

- **Timeliness of data:** Not an issue due to one hop or few hops in most cases without storage of data.
- **Logical consistency:** Especially critical for those sub-applications where more than one source can generate the same information.
- **Temporal consistency:** Not an issue due to one hop or few hops in most cases without storage of data.
- **Trustworthiness of data:** It needs to be assured that the data stem from a car which is trustworthy and not from a malicious source.
- **Robustness:** The reaction to a failure should in any case be fail safe, i.e. not inject additional traffic threats into the road network.
- **Message ordering:** Not an issue
- **Completeness, accuracy and timeliness (of failure detection):** If a failure occurs, the detection needs to be: 1. timely (highest priority), 2. complete, 3. accurate (lowest priority).

3.2.6 Platooning

This application provides both positional and velocity control of vehicles in order to operate safely as a platoon on a highway. A platoon is formed by two or more vehicles following each other closely, controlled by the vehicle at the head of the platoon.

Platooning requires vehicle-to-vehicle communication and may include vehicle-to/from-infrastructure communication. This application functions only in the control role and improves highway traffic flow and capacity by allowing short-range headway distance following in platoon architecture. The application combines vehicle data with position and map data. Longitudinal control of the vehicle is provided in order to maintain the short-range headway following within a platoon (similar to adaptive cruise control). Lateral control via automated steering provides lane-keeping and lane change manoeuvres of platoon vehicles in a coordinated manner.

3.2.6.1 Communication level requirements on Platooning application

The requirements with respect to the communication level properties are:

- **Throughput:** Each member needs to transmit information with a data amount of 170 bits per transmission (32 bits position, 8 bits speed, 8 bits direction, 16 bits other status information, 32 bits overhead, 72 bits protocol stack overhead = approx. 170 bits). The frequency of sending out information per vehicle depends on the speed, where the frequency generally increases with speed. It shall be as often as required to make sure that, at a maximum deceleration of 9 m/s^2, the requirement to detect a change of distance in the range of 10 cm can be fulfilled.
- **Communication delay:** The time between generation of information by the platooning application and the reception on application level by a platoon member needs to remain below 100ms (taking account a maximum braking acceleration of 9 m/s^2 and the requirement to detect a change of distance in the range of 10 cm).
- **Data integrity / packet error ratio:** Correctness of data has to remain high (packet error ratio $<10^{-4}$).

- **Packet loss:** Packet loss is critical, if the data repetition frequency just fulfils the requirement (see above). Packet losses may be permitted if the frequency is high enough.
- **Network congestion:** Is an issue. If the required network resources are not available, measures have to be taken (e.g. slow down, …)

3.2.6.2 Middleware level requirements on Platooning application

The requirements with respect to the middleware level properties are:

- **Timeliness of data:** Data needs to be timely, i.e. fresh, meaning that the delay between generation and transmission onto the air needs to be kept very small. A guideline for timeliness is given by the requirement in the communication level requirements.
- **Logical consistency:** The data distributed inside the platoon must not be modified. It is absolutely necessary that all following cars receive the same control data, even in case it is transmitted by any other platoon member.
- **Temporal consistency:** As the transmitted control information may refer to rapidly changing entities, there must be mechanisms to ensure the consistency of the information made available to the application with regard to the actual value of the entities.
- **Trustworthiness of data:** The data exchanged must be strictly limited to the platoon. Only the platoon members shall be authorised to send and to receive control information, and the following cars must only process data provided by other members of the same platoon.
- **Robustness:** Required. The reaction to a failure should in any case be fail safe, i.e. not inject additional traffic threats into the road network.
- **Message ordering:** Probably not an issue
- **Completeness, accuracy and timeliness (of failure detection):** If a failure occurs, the detection needs to be 1. timely (highest priority), 2. complete, 3. accurate (lowest priority).

3.2.6.3 General requirements on Platooning application

- To keep the delay requirements, it must be investigated which communication mechanisms are suited for this purpose. It is well known that e.g. CSMA/CA as used in IEEE 802.11 cannot guarantee delays, especially in highly loaded networks.
- In order to achieve the required throughput, the communication mechanisms need to be examined for their performance, especially on medium access control level.
- A worst case situation is if two large platoons driving in opposite directions meet. In this case, the amount of radio resources needed by the two platoons increases in the affected geographic area and needs to be guaranteed for each of the platoons. This requires detection of oncoming platoons and mechanisms to re-arrange radio resources such that both platoons can continue their operation.
- At first sight, platoons seem to be suitable for one-hop communication only, i.e. the maximum extension of the platoon is restricted by the maximum radio range, minus some security margin. Larger platoons may be possible by using multihop communication. On middleware level, it may be useful to have additional methods for data condensation in order to reduce the amount of data to be forwarded by multihop communication. This must be investigated carefully
- The availability of communication resources is an absolute must.
- The integrity of data is absolutely necessary. The information distributed may e.g. contain information about braking status. If a platoon member gets wrong information about a vehicle ahead which brakes hard, this may lead to tragic accidents.

- Reliability: It is necessary that the transmission of the control data is reliable. Loss of control data packets may lead to lack of information on the receiver side and may lead to wrong behaviour. Retransmission of lost data packets is not a solution, as the control data is only valid for a very short time. Accidental and malicious attacks, e.g. by sending out false information, must be avoided. This means that some trust mechanisms need to be introduced for platoon members
- Fault tolerance: Faults in the case of platoons can lead to heavy accidents which may even lead to fatalities. Faults in the system need to be detected and fault recovery mechanisms need to be introduced.
- The distance between the cars will be proportional to the speed of the platoon so that emergency breaks will not cause cascading rear end collisions.
- Probability of cars getting too close (closer than d(v) meters) has to be less than 10^{-5}. The distance between the cars in the platoon is denoted $d_p(v)$.
- A platoon member that decides to leave the platoon has to be able to do so within 5s. The distance to the platoon has to be increased to $5 \cdot d_p(v)$. This requirement is event driven (driver turns away from the platoon) or driver requested (driver pushes a button)
- Leader leaves triggers re-election of platoon leader within $t_1(v)$ = (expression of time to reelection to be determined)...

3.2.7 Distributed black box

The “classical” black box can record informational data, such as: engine / vehicle speed (typically 5 seconds before impact), brake status (again here, 5 seconds before impact), throttle position(s), and even the state of the driver's seat belt switch (on/off). The combination of this information along with other engineering factors is indeed very valuable for motor vehicle accident investigation.

Distributing the black box functions among the neighbouring vehicles is beneficial in several aspects:

- First, cost reduction of the black box hardware: since the black box is not hosted anymore on the vehicle having the accident, it does not need to be accident proof.
- Second and most importantly, we can extend the recorded information with contextual information concerning the neighbouring vehicles, possibly the various vehicles involved in an accident. This way, investigators could draw a better picture of an accident.

A typical scenario for the distributed black box application is the following.

The Black Box function of a car A is distributed among its neighbours (B, C, D, …). Car A collects time-stamped data about its current situation (engine/vehicle speed, brake status, throttle position, direction, position, traffic and road conditions, etc.) and stores this information on B, C, D, etc. Time-stamping might use either global time (provided by GPS or Galileo) or local time. When an access to the infrastructure is available, the black-box data is backed up on a fixed server by the various cars involved in this scenario (original data on the car A and the copies on the cooperating neighbours) in order to prevent data loss or any other dependability or security related threats that might affect the availability or the integrity of the data. This scenario should work in different operating contexts, independent of the kind of roads or the road traffic density. The higher the number of equipped cars is, the lower the risk of loosing the original data is. Similar conclusions can be made with respect to the availability of access points to the fixed infrastructure. Nevertheless, it should be possible to retrieve the data from the cooperating cars, when there is no possibility for storing the data on the fixed network for some time.

3.2.7.1 Communication level requirements on Distributed black-box application

The requirements with respect to the communication level properties are:

- **Throughput:** Throughput is not a critical issue.
- **Communication delay:** Ensuring timeliness is very important for opportunistic communication both with cooperating cars when in the ad-hoc domain and with access points when in the infrastructure domain. Reducing the time needed for the switch operation between the two domains is also very important and might be difficult to achieve.
- **Data integrity / packet error ratio:** Data integrity is critical and should be ensured despite the occurrence of accidental or malicious faults.
- **Packet loss:** Packet loss may not be not very critical, since data is regularly updated.
- **Network congestion:** network congestion should be avoided to reduce latency.

3.2.7.2 Middleware level requirements on Distributed black-box application

The requirements with respect to the middleware level properties are:

- **Timeliness of data:** It is necessary to make sure that the data instances stored in the infrastructure domain are sufficient to reflect the state of the original car over time, even if some fragments of the data are lost.
- **Logical consistency:** The backed up data must be consistent, i.e. the data reaching the fixed server, or the cooperating cars, should not be modified. Only the producer must be able to write the data, the other entities have no need for write (or even read) access on the data. We can also talk about integrity of the data. Inconsistent failures might occur when the original information at the provider side differs from the data copied at the participating cars.
- **Temporal consistency:** it is important that the information disseminated in the ad-hoc domain and stored in the infrastructure domain reflects the real situation. Only most up-to-date data need to be backed-up on the infrastructure domain.
- **Trustworthiness of data:** As expressed earlier, for each car considered, there is only one writer and one reader. Only the original car should be allowed to write data, and only the data owner (or its delegates, e.g. its insurance company) should be allowed to read it.
- **Robustness:** The integrity and the availability of the black-box data should be satisfied even in the presence of accidental or malicious threats, taking into account different types of failure modes, including timing failures and value failures and different types of faults, including external faults.
- **Message ordering:** simple time-stamping of each data fragment should be sufficient for ensuring that the data can be received in the correct order.
- **Completeness, accuracy and timeliness (of failure detection):** failure detection mechanisms should be implemented at different levels to ensure that whenever faults occur, the corresponding errors are detected and signalled to ensure that they are properly handled.

3.2.7.3 General requirements on Distributed black-box application

Classical protocols such as those defined in 802.11 seem to be suited for the distributed black box application. The application relies on one hop ad-hoc geocasting while in the ad-hoc domain and on routing to a fixed backup service while in the infrastructure domain. The switch between the ad-hoc and the infrastructure domain should be notified to the application in order to reconfigure it. The amount of information produced and stored by this application is low but its freshness is critical, i.e. reducing communication latency is more important than ensuring throughput.

This application may pose high requirements regarding data delivery time because of the potentially high speed of the cars, i.e. data must be transferred to passing cars or infrastructure within a short period of time.

It depends on an acceptable throughput, but throughput is not a critical issue. Relevant QoS attributes are delivery time and data correctness.

Dependability-related Properties: The main dependability and security properties to be ensured are:

- Availability of the black box information: this is the main goal of the application, the information availability must be maximised despite faults.
- Confidentiality and privacy, i.e. the black box information should be accessible only from authorised parties.
- Integrity of the black box information: the original information produced by car A should not be modifiable, either by A's driver or by the other cars hosting copies of the original data (B, C, D cars), or by any third party.

Threats: The dependability properties listed above should be satisfied even in the presence of accidental or malicious threats, taking into account different types of failure modes, including timing failures and value failures and different types of faults. Both permanent and transient faults should be taken into account.

The types of faults to be tolerated are:

- Accidental or malicious faults affecting the availability of black box original information at the provider side (typically the crash of the car but this can also be the theft of the car),
- Accidental or malicious faults affecting the availability of the black box information copies on the cooperating cars (e.g. the crash of the neighbours B, C, D),
- Accidental or malicious modification of the black box information (either by the information provider, i.e., car A, or its driver, or by a third party, e.g. the insurance company),
- Malicious read access to the black box information (only authorised parties can access the data),
- Denial-of-service attacks on the distributed black box service (malicious or selfish).

Concerning failure mode assumptions, both content failures (e.g. incorrect values) and timing failures (e.g. service delivered too early or too late) have to be taken into account. Such failures might occur as a result of faults at the information provider side, at the receivers side (neighbouring cars), or during the information transmission through the wireless communication links or through the fixed infrastructure when data is backed up on the dedicated servers.

This list of failure assumptions has to be completed by the specification of the acceptable degraded operation modes. In the case of the distributed black-box application, examples of degraded operation are when the data is available at the provider side (i.e., car A) without having the required number of copies at the neighbours cars, or vice-versa (copies are available whereas the original data is lost). Such degraded operation modes might occur when connections to the infrastructure or to neighbours are limited or impossible:

- Car A is alone in the countryside, has no neighbours, no access points, no access to infrastructure. In such a case, if A crashes, there is no backup of the black box information.
- Car A has a few neighbours, in the countryside; there are no access points and no access to infrastructure. In such a case, if A and its neighbours crash together, there is no backup of the black box information.

Some additional general requirements: Quantitative objectives could be defined to assess data availability. Such quantitative requirements can be formulated as follows:

- In case of a sudden accident (completely destroying the CPU) the probability of being able to restore 90% or more of the data must be greater than $1-10^{-4}$: P(90% of the data can be recovered) > $1-10^{-4}$.
- In case of hazard warning, within 1 sec the data has to be distributed so that 99% can be recovered sub sequentially with probability $1-10^{-6}$. This should also work for 10 cars in a group accident within radio range

3.2.8 Maintenance and software updates

The status of cars with respect to maintenance is transmitted to a central agency. The transmission of information can either be initiated by a car driver or passenger or automatically and regularly. Maintenance can then be performed by a workshop, provided that the failure can be removed by pure means of software.

Software updates and new software can be transmitted into cars. This involves software for new car-to-car applications. Such updates may be subject to payment which requires proper authentication and possibly billing mechanisms in the fixed network domain.

3.2.8.1 Communication level requirements on Maintenance and software updates application

The requirements with respect to the communication level properties are:

- **Throughput:** A software update will first check whether the throughput available is sufficient to download the data in a sufficiently short time which should remain clearly below 5 minutes in all cases. Ideal would be a download within less than one minute. So the required throughput depends heavily on the amount of data to be transmitted. For 100 kbyte of data, ideally 13 kbit/s and minimally 2.5 kbit/s is needed; For 1 Mbyte of data, ideally 133 kbit/s and minimally 27 kbit/s is needed; For 100 Mbyte of data, ideally 13.3 Mbit/s and minimally 2.7 Mbit/s is needed. The general rule is: the more, the better.
- **Communication delay:** Delay is less critical but is usually closely related to throughput and may have some influence on the traffic flow control protocol.
- **Data integrity / packet error ratio:** Data has to be absolutely free of errors above the transport layer.
- **Packet loss:** Is of less importance but may have influence on traffic flow control.
- **Network congestion:** Closely related to throughput.

3.2.8.2 Middleware level requirements on Maintenance and software updates application

The requirements with respect to the middleware level properties are:

- **Timeliness of data:** Not really an issue because the SW maintenance will rely on a one-to-one connection.
- **Logical consistency:** Not really an issue because the SW maintenance will rely on a one-to-one connection.
- **Temporal consistency:** Not really an issue because the SW maintenance will rely on a one-to-one connection.
- **Trustworthiness of data:** This needs to be assured in every sense, since false SW in the car is safety critical
- **Robustness:** The situation may occur that the connection is interrupted, be it by system internal or external reasons. There must be methods to cope with this situation.
- **Message ordering:** Not really an issue because the SW maintenance will rely on a one-to-one connection, so correct order after the transport protocol is assumed.
- **Completeness, accuracy and timeliness (of failure detection):** If a failure occurs, the detection needs to be: 1. complete (highest priority), 2. accurate, 3. timely (lowest priority).

3.2.8.3 General requirements on Maintenance and software updates application

- Probably the most critical point is if the communication is interrupted. There must be means to
    - Resume a maintenance or SW update process in order to keep update times low (especially if the process is interrupted multiple times) or, if this is not possible, restart it.
    - Make sure that an interruption does not lead to inconsistencies or even to safety critical states of the vehicle.

3.2.9 Blackboard application

The blackboard application shall serve to distribute information which is relevant for a certain geographic area. The notion behind is that a lot of information can be broadcast into the network but the user only wants to see the information which is relevant for him. A major share of information is relevant for a certain geographic area only, such as speed traps, fuel prices, restaurant offers, warning about a slippery road, etc. This special type of application assumes that the information is not necessarily permanently repeated by its source but sent only once (or with larger intervals). This means that the distribution has to happen by cars on the road which means also that the cars are the ones to store the data and to make sure the data does not disappear. A car may have received the message outside the relevant geographic area already but it is displayed only upon entering the area. This may be relevant for cases where no car is in the considered region but the message shall still be preserved in the network.

3.2.9.1 Communication level requirements on Blackboard application

The requirements with respect to the communication level properties are:

- **Throughput:** The messages that can be transmitted can range from few bytes (pure text message) to several Mbytes (e.g. commercial video). The relevance, however, is mostly geographically limited, which should enable the use of broadcasting (one transmitter, multiple receivers). The throughput should be such that the time required to transmit the entire message remains well below a typical contact duration of two cars or a group of cars (typically well above 20 seconds).
- **Communication delay:** Delay is less critical but is usually closely related to throughput and may have some influence on the traffic flow control protocol.
- **Data integrity / packet error ratio:** Data should be free of errors above the transport layer (packet error ratio < 10^{-6}).
- **Packet loss:** Is of less importance but may have influence on traffic flow control.
- **Network congestion:** It may be required to define priorities in order to make sure that important messages come through (such as announcements about closed routes).

3.2.9.2 Middleware level requirements on Blackboard application

The requirements with respect to the middleware level properties are:

- **Timeliness of data:** The data may outdate, so timeliness is a critical point (see also temporal consistency)
- **Logical consistency:** Blackboard messages may be permanent (e.g. commercials) or have a transient nature (such as police speed control announcement). In the latter case, logical consistency does play a role because such messages may outdate and should then disappear from the network.
- **Temporal consistency:** Outdated data needs to be removed, new data needs to be distributed quickly.
- **Trustworthiness of data:** Trustworthiness is critical in the sense that, if everyone can place his message on the air, the user acceptance will drop dramatically.
- **Robustness:** External faults may lead to logically inconsistent messages in the network. Means have to provided to avoid this. An acceptable behaviour for the user is that (potentially) faulty messages are not displayed.
- **Message ordering:** Not really an issue because the blackboard application will rely on a data connection, so correct order after the transport protocol is assumed.
- **Completeness, accuracy and timeliness (of failure detection):** If a failure occurs, the detection needs to 1. accurate (highest priority), 2. correct, 3. timely (lowest priority).

3.2.9.3 General requirements Blackboard application

- The messages need to be kept in the network, i.e. even if no car is inside the relevant area, the message may be distributed outside the relevance area and activated only when entering the area.

3.2.10 Video conference

The video conference application includes two-way real-time voice and video communication between two or more terminals. It may be used for personal communication, office/business purposes, for group communication at a crash site, or for access from an ambulance to medical expertise at a hospital.

For the audio and video components, relevant QoS attributes are throughput, delay, and packet loss. These attributes will, together with other parameters like video coding and audio coding, determine the video and audio quality.

A special case of video conference is "**access to medical expertise**". Ambulance personnel at an accident scene may need to communicate with medical expertise at the local or a central hospital by use of video communication as well as voice, images, and/or data transmission. The application "Access to medical expertise" runs between the ambulance and the hospital over a multimedia session. The application may be used while the ambulance is at the accident scene and while heading back to the hospital with the injured. It may be of vital importance to keep the multimedia session running while driving, possibly switching between the different access points and technologies that are available. This involves seamless handovers between access points and even handovers between access points of different technologies.

For group communication at a crash site, broadcast/multicast communication and secure communication groups are needed.

3.2.10.1 Communication level requirements on video conference application

The requirements with respect to the communication level properties are:

- **Throughput:** Throughput requirements mainly depend on the characteristics of the video component since this normally requires higher throughput than the audio component. The actual requirements will depend on the video coding, video frame sise, and the frame rate. Typically, video conference data rates are in the range 16 - 384 kb/s (ITU-T rec G1010.
- **Communication delay:** The delay requirements for the video conference application are strict. End-to-end delay values below 150 ms are normally presumed (ITU-T Y.1541). The delay variation part of this should be below 50 ms (Y.1541). This is network performance and not mouth to ear. Packet insertion time is included. The requirement of the maximum delay will be the sum of these two values.
- **Data integrity / packet error ratio:** To what extent the perceived video quality is affected by errored packets depends highly on the video and audio coding used. Some coding techniques have a high level of error protection in-built while others have none. Further, the human eye and ear is tolerant to some level of erroneous and lost information in the audio and video. Therefore, some degree of packet errors will be acceptable depending on the specific video coder and amount of error protection in use.
- **Packet loss:** Loss requirements for the voice and video components will depend on the ability of the voice and video coding to handle packet loss. ITU-T recommendation G.1010 introduces a requirement of less than 1% packet loss ratio end-to-end for video conference applications. Furthermore, the voice and video components must be synchronised within 80 ms to provide "lip-synch". In an emergency scenario the video conference quality should be as high as possible and therefore loss requirements are strict. Nevertheless, reduced quality can be tolerated if the alternative is to close down the conference call.

3.2.10.2 Middleware level requirements on video conference application

The requirements with respect to the middleware level properties are:

- **Timeliness of data:** This application has timeliness requirements, in the sense that lack of timeliness will result in degradation of the audio and video signals, possibly to unacceptable levels. However, occasional timing failures may be acceptable. The middleware should ensure that the probability of timing failures stays close to a small value.
- **Logical consistency:** Every participant to the video conference should receive the same information with approximately the same delays. However, no strict logical consistency requirements exist.
- **Temporal consistency:** Not an issue.
- **Trustworthiness of data:** Depending on the nature of transmitted data, there might be privacy and/or confidentiality requirements.
- **Robustness:** The application should still work despite external faults. Should, however, external faults occur, acceptable behaviour might e.g. be a reduction in frame rate or video resolution, whereas the availability of the voice channel has to be maintained.
- **Message ordering:** Communication between the sender and each receiver should take place through FIFO channels.
- **Completeness, accuracy and timeliness (of failure detection):** If a failure occurs, the detection needs to be 1. accurate (highest priority), 2. timely, 3. complete (lowest priority). A participant in the video conference should not be removed unless it has effectively failed. When this happens, the failure should be reported and managed as soon as possible. No failure should remain undetected.

3.2.11 Online Gaming

We consider here only interactive online games with real time constraints. Other games with less interaction, such as chess, can rather be considered as interactive data, see section 3.2.15. Interactive games can be played between cars on the road which are not too far away from each other or, in suitable scenarios, with partners in the fixed network. Online gaming applications will have real-time requirements different from other applications identified here which is the reason for listing them separately. Relevant QoS attributes are delivery time and data correctness. Delivery time depends partly on throughput, delay, and packet loss.

3.2.11.1 Communication level requirements on online gaming application

The requirements with respect to the communication level properties are:

- **Throughput:** Throughput requirements may be strict depending on the amount of contents that will be transferred. Delay (Round-Trip Time) and packet loss ratio may influence achievable throughput experienced.
- **Communication delay:** The delay requirement is strict. End-to-end delays below 150 ms are normally presumed (ITU-T Y.1541). The access network component of this delay is for further study. The delay variation part of this should be below 50 ms (Y.1541).
- **Data integrity / packet error ratio:**
- **Packet loss:** A general packet loss ratio objective of 10^{-3} and packet error ratio objective of 10^{-4} is specified in Y.1541.

3.2.11.2 Middleware level requirements on online gaming application

The requirements with respect to the middleware level properties are:

- **Timeliness of data:** Different kinds of data may be transmitted, typically with different timeliness requirements. For instance, control information related to interactive events (game play/control) may

have to be transmitted within stricter and smaller time bounds than other game related information, such as player names or setup related information.

- **Logical consistency:** Every participant must receive all pieces of control data in order to have a consistent view of the game. There are strong logical consistency requirements.
- **Temporal consistency:** Temporal consistency may be an issue for certain kinds of game related data, and also depending on the application design. If the game "world" is modelled, kept and continuously updated by a central server, then every participant must have a temporally consistent view of this world.
- **Trustworthiness of data:** This should not be an issue.
- **Robustness:** It may be interesting to keep some state information despite external faults affecting the application. This information may be useful upon game restart.
- **Message ordering:** Communication may have to secure total order properties on the delivery of events to participants.
- **Completeness, accuracy and timeliness (of failure detection):** If a failure occurs, the detection needs to be 1. accurate (highest priority), 2. timely, 3. complete (lowest priority). A participant in the game should not be removed unless it has effectively failed. When this happens, the failure should be reported and managed as soon as possible. No failure should remain undetected.

3.2.12 Audio and video streaming

Streaming audio is for instance used in radio programs, and music, while streaming video may be used in video on demand and TV applications. Relevant QoS-attributes are mainly throughput and packet loss rate, but delay variation must also be considered. These attributes will, together with other parameters like audio coding, determine the audio quality.

3.2.12.1 Communication level requirements on audio and video streaming applications

The requirements with respect to the communication level properties are:

- **Throughput:** In audio streaming throughput requirements are generally loose since the amount of data that is transferred is quite low. In video streaming, throughput requirements will depend on the video coding, video frame sise, and the frame rate. Typical data rates for video streaming are presented in G.1010 as 16 - 384 kb/s, but also higher data rates might be relevant depending on intended application use.
- **Communication delay:** The delay requirement is much weaker than for real-time applications. It is first of all a delay variation requirement, and this depends again on buffering capabilities in the terminal equipment. If delay variation is large, the jitter buffer will not be able to compensate and packets will be considered as lost, even if they arrive some time later. 5 – 10 seconds delay variation is often presumed over mobile networks. ITU-T recommendation G.1010 presents a requirement of less than 10 seconds one-way delay for video conference applications.
- **Data integrity / packet error ratio:** To what extent the perceived audio and video quality is affected by errored packets depends on the video and audio coding used.
- **Packet loss:** Generally, voice applications can tolerate a packet loss ratio of a few percentages depending on codec and quality requirements (MOS / E-model). For high quality audio streaming G.1010 has a packet loss requirement of 1% end-to-end. Video streaming applications should not be exposed to packet loss rates above 1% (ITU G.1010).

  In general, the requirements of audio and video streaming applications are stricter when such applications are used for emergency purposes.

3.2.12.2 Middleware level requirements on audio and video streaming application

The requirements with respect to the middleware level properties are:

- **Timeliness of data:** Theses applications have more relaxed timeliness requirements than the video conference application. This is because the only visible effect of loose timeliness is extra delay. Nevertheless, the application has to assume an upper bound on the timeliness of data in order to configure the sise of buffers and other time-related parameters.
- **Logical consistency:** Since video streams are independent, there are no logical consistency requirements.
- **Temporal consistency:** Not an issue.
- **Trustworthiness of data:** Depending on the nature of transmitted data, there might be privacy and/or confidentiality requirements.
- **Robustness:** The application should still work despite external faults. If this is not possible, then service will simply be interrupted.
- **Message ordering:** Communication between the sender and each receiver should take place through FIFO channels.
- **Completeness, accuracy and timeliness (of failure detection):** Since there are not multiple participants, the properties of (crash) failure detection are not relevant.

3.2.13 Streaming data

Streaming data is transmission of information that is continuously updated like for instance positioning information or a patient's pulse or blood pressure. Even though such an application has high quality and performance requirements, it does not depend on the correct reception of every single message. Relevant QoS attributes are throughput and data correctness. Data transmission can be both from the fixed network or originating from a vehicle.

Note: positioning information from the ambulance to the emergency central and hospital could possibly be integrated with the functions in the Traffic Flow Control application, or it might be a stand-alone application.

3.2.13.1 Communication level requirements on streaming data application

The requirements with respect to the communication level properties are:

- **Throughput:** Throughput requirements will depend on the amount of data that is transferred, e.g. they may be loose for positioning information from the ambulance, and higher for transmitting patient information.
- **Communication delay:** The delay requirements are much weaker than for real-time applications. It is first of all a delay variation requirement, and this depends again on buffering capabilities in the terminal equipment. If delay variation is large, the buffer will not be able to compensate for this and packets will be considered as lost, even if they arrive some time later. 5 – 10 seconds delay variation is often presumed over mobile networks.
- **Data integrity / packet error ratio:** Depending on how this application component is used, it may have a requirement for correct information, i.e. if a packet is received with incorrect content the packet is discarded rather than presenting incorrect information.
- **Packet loss:** If a packet is lost, it will not be resent since it is updated in a short while anyway. But loss should nevertheless be kept at a minimum when this application component is used in emergency applications.

3.2.13.2 Middleware level requirements on streaming data application

The requirements with respect to the middleware level properties are:

- **Timeliness of data:** The requirements on timeliness depend on the kind of data being streamed and on its purposes. For instance, if this is only log data, then no timeliness requirements exist. However, if this is a stream of control data, then it might have to be received within certain timing constraints.
- **Logical consistency:** Not an issue.
- **Temporal consistency:** If the transmitted data corresponds to continuously changing entity values, then there are temporal consistency requirements. In this case the transmission rate (throughput) should be high enough to secure such requirements.
- **Trustworthiness of data:** Depending on the nature of transmitted data, there might be privacy and/or confidentiality requirements. It might also be necessary to ensure that data is received from a trusted sender.
- **Robustness:** The application should still work despite external faults. A fail-safe state might have to be identified, to which the application should switch when it is not able to handle some fault.
- **Message ordering:** Communication should take place through FIFO channels.
- **Completeness, accuracy and timeliness (of failure detection):** Only timeliness is relevant in this context. Failure detection should be timely in order to allow for fast switching to the fail-safe state (when applicable).

3.2.13.3 General requirements on streaming data application

The dependability requirements of streaming data will depend on its contents and use case.

3.2.14 Non-interactive data communication and messaging

Non-interactive data and messaging can happen on the one hand between cars only, possibly involving infrastructure (to forward the messages if the vehicles are too far apart from each other), or between cars and some device located anywhere in the networked world on the other hand. The files/messages/data can contain text, images or recorded voice.

Examples of applications in this category are documents upload and download, and calendar and email synchronisation. Also TV and radio programs, Video-on-Demand (VoD), and music may be downloaded as files and played out later.

Relevant QoS attributes for the Non-interactive data communication and messaging applications are throughput, reliable transfer (that the data actually arrive) and data correctness.

Typically, this is best-effort data using TCP. With TCP delay (Round-Trip Time) and packet loss will influence the achievable throughput.

The Non-interactive data communication and messaging applications may also be used in cases where the content is of a critical nature, and the applications will in this case require high dependability and some will also have stricter delay requirements. A special application is **online notification to hospital**, where users may transmit information to an emergency centre or a hospital. Messages may be of different formats like for instance text, images, or video. This kind of information may be used for enhanced planning of the rescue operation, and retrieving the injured people's earlier case history. Online notification may also be transferred from the ambulance to the hospital, for instance giving important information on a patient.

Some applications like for instance calendar and email synchronisation may have high requirements with respect to data consistency.

Data backup from the distributed black box application on neighbouring cars and on the fixed network. This application may pose high requirements regarding data delivery time because of the potentially high speed of

the cars, i.e. data must be transferred to passing cars or infrastructure within a short period of time. It depends on an acceptable throughput, but throughput is not a critical issue. Relevant QoS attributes are delivery time and data correctness (see 2.2.2).

3.2.14.1 Communication level requirements on non-interactive data communication and messaging application

The requirements with respect to the communication level properties are:

- **Throughput:** Throughput requirements are closely linked with user satisfaction. In general, the higher the throughput, the higher the satisfaction. However, there is a lower limit below which user satisfaction drops dramatically which depends on the type and the amount of contents that is to be transferred.
- **Communication delay:** The application has loose requirements regarding transmission delay, i.e. delay requirement depends on requirements from higher layers, i.e. timer values etc, since such high delay will be recorded as packet losses and impact throughput.
- **Data integrity / packet error ratio:** Messages should have a very high probability of reaching the receiver.
- **Packet loss:** 10^{-3} packet loss ratio is normally satisfactory for TCP applications.

3.2.14.2 Middleware level requirements on non-interactive data communication and messaging application

The requirements with respect to the middleware level properties are:

- **Timeliness of data:** Timeliness requirements depend on the application concerned and the criticality of the information transferred. For instance in the case of emergency vehicle warning, timeliness of data is clearly important.
- **Logical consistency:** Not an issue.
- **Temporal consistency:** Not an issue.
- **Trustworthiness of data:** Data must be sent reliably and it might also be necessary to ensure that data is received from a trusted sender.
- **Robustness:** The application should be robust. Since there are no timeliness requirements, data transmission can always restart after recovering from a fault.
- **Message ordering:** Not necessarily an issue. Ordering requirements may be imposed by the application.
- **Completeness, accuracy and timeliness (of failure detection):** Completeness is the most important property of failure detection, in order to ensure reliable delivery.

3.2.15 Interactive data

Interactive data communication means that the user receives responses from other users or servers within a limited time, where the tolerable delay depends on the application. This application involves interactive data exchange between cars and data access to the Internet. Examples of such applications are web browsing, document sharing, and some collaborative gaming applications.

Relevant QoS attributes are data correctness and throughput.

Web shopping is a particular instance of interactive data that poses special requirements to both middleware and communication infrastructure. While browsing for products can be viewed as ordinary web browsing, the payment part is transaction-based with special requirements regarding security, dependable delivery of the messages within a reasonable time limit, and exactly-one delivery of packets. Web shopping requirements are not considered below.

3.2.15.1 Communication level requirements on interactive data application

The requirements with respect to the communication level properties are:

- **Throughput:** Throughput requirements will vary according to the amount of data that is to be transferred. Achievable throughput may be influenced by round-trip delay and packet loss rate.
- **Communication delay:** Applications in this category are typically based on TCP, and the delay (Round-Trip Time) requirements are lower than for Online Gaming, Voice Call, and Video conference, but higher than for streaming and file transfer applications.
- **Data integrity / packet error ratio:** Data integrity and packet errors are handled by TCP. However, incorrect packets are interpreted by TCP as lost packets, leading to a retransmission of these packets together with a decrease of the throughput by flow control. It is well known that this leads to massive problems. Therefore, packet error ratio must be kept low (in the range below a few percent).
- **Packet loss:** Packet loss is critical with respect to TCP, see previous point.

3.2.15.2 Middleware level requirements on interactive data application

The requirements with respect to the middleware level properties are:

- **Timeliness of data:** Different kinds of data may be transmitted, typically with different timeliness requirements.
- **Logical consistency:** If the application involves several participants, then they may need to receive the same data in order to keep consistent with each other.
- **Temporal consistency:** This should not be an issue.
- **Trustworthiness of data:** It may be necessary to ensure that data is received from a trusted sender.
- **Robustness:** Depending on the specific application, robustness requirements will vary. While for web browsing robustness is not fundamental, collaborative applications require that at least a safe-state is always achieved in spite of faults.
- **Message ordering:** When interaction is among several participants, it is usually important to secure that messages are delivered in total order.
- **Completeness, accuracy and timeliness (of failure detection):** The importance of each property depends on the specific application using the interactive data.

3.2.16 Mobile mission control centre

Collaborating mobile entities may require the establishment of restricted ad-hoc networks – excluding any entities that must not participate to the collaboration (however they may participate in transmission) – without requiring the exact predefinition of the actual members. The hierarchic nature of such brigades raise the idea to previously choose an entity in the brigade that has a strong connection to the infrastructure domain (through broadband wireless, satellite or traditional cellular phone connection) and serves as a proxy (or provisional access point, as well if necessary) for the local ad-hoc network. An example for such a chosen entity of a brigade would be a mission control centre for a police mission (obviously the same can hold for a fire brigade, road construction, TV report brigade etc.). The functionalities of the mission control centre cover buffering messages between undependable networks (between the ad-hoc one and its own “uplink“), transfer of priority information across low or medium bandwidth networks, message prioritization, routing, authentication, caching and proxy services.

3.2.16.1 Communication level requirements

- **Throughput:** The performance and feasibility highly depends on the communication throughput among brigade members (e.g. utilization of the available bandwidth) as well as through the proxy (e.g. caching strategies).
- **Communication delay:** The feasibility of the application highly depends on the communication delay if scheduling and coordination of the members' move is required, otherwise this dependency is very low.
- **Data integrity / packet error ratio:** The application has only standard requirements on correctness.
- **Packet loss:** High packet loss ratio can reduce the performance of the application, but generally, packet loss can be handled by traditional methods.
- **Network congestion:** The central role of the mission control centre in the communication can be a source of network congestions, but even so, the feasibility of the application does not really depend on this issue.

3.2.16.2 Middleware level requirements

- **Timeliness of data:** The application has only standard requirements in the timeliness of data. Each member of the brigade has to receive actual data. Most of the data to be transferred are static or changing with a very low frequency. Strict requirements may arise when data on movement scheduling or coordination have to be communicated.
- **Logical consistency:** The application has only standard requirements in logical consistency. The mission control centre has to maintain a consistent view of actual data over the brigade.
- **Temporal consistency:** The application has only standard requirements in temporal consistency. Caching function requires standard functions to remove outdated data. The communication delays within the brigade should not endanger the consistency of the typically infrequently changing mission data.
- **Trustworthiness of data:** The application requires authorisation and authentication to maintain the restricted ad-hoc network.
- **Robustness:** As the mission control centre is a single point of failure in the communication between the restricted network and the infrastructure, the feasibility of the application depends on its robustness.
- **Message ordering:** The application has no special requirements in message ordering.
- **Completeness, accuracy and timeliness of failure detection:** The application has no special requirements against the applied failure detection techniques.

3.2.16.3 General requirements

When considering content failures, the central role of the mission control centre (caching, proxy, authentication ...) has to be taken into account. Timing failures are an issue only if real time scheduling and movement coordination of the brigade members is required. The mission control centre has to separate the restricted ad-hoc network and its uplink to the fixed network to confine failures in the two separate regions. During attacks against the restricted ad-hoc network, malicious behaviour may result in consistent failures. Application in a police or military domain may have special security requirements. Failures can lead to severe injuries or to the loss of lives, in particular in cases where the mission control centre has direct control functions over the other brigade members who may rely on faulty mission data.

3.2.17 Ad-hoc service providers

Public ad-hoc networks can be limited to the entities "near" a given geographic location (e.g. a hotel, a restaurant, a taxi station ...), if the set of the potential service providers is strongly limited by the geographic distance of the requester and the provider candidates. (The provided service and its way of provision are out of the scope of this application.) A service requester may send a broadcast query (e.g. a taxi call) into this ad-

hoc domain without uniquely addressing a chosen provider (remaining at the same example: without addressing a single taxi car or even the cars of a single taxi company) to discover potential providers in its neighbourhood. The communication is started by this general poll, continued by several replies possibly including the offered quality of service or service level agreement (e.g. when they can start providing the service, their performance and dependability parameters, certificates, price information ...) from volunteering service providers in the ad-hoc domain, and completed by a peer-to-peer communication between the original requester – who decides based on his own business logic (e.g. fastest reply, preferred certificates, best price, or any complex strategies) – and a chosen provider. Although the messages of the second and third phase are transmitted by the ad-hoc network, too, these message exchanges are of a "traditional nature" (one-to-one communication that is routed based on the known addresses of the partners).

One of the core problems in this context is a fair distribution of discovery messages simultaneously while avoiding a complete flooding of the communication channel, and the geographic limitation of the broadcast. The addressing of the broadcast messages is based on the service type and it is neither communication unit nor location dependent. The subsequent peer-to-peer communication, which may include a voice communication, uses the traditional communication after a switch over.

3.2.17.1 Communication level requirements

- **Throughput:** The application's performance and feasibility depends on the throughput only to a small extent.
- **Communication delay:** The application's performance and feasibility depends on the communication delay to some extent, but it highly depends on the equality of the delay, since the competition of the providers (the broadcast receivers) can be highly influenced by unequal delays.
- **Data integrity / packet error ratio:** The application has only standard requirements on correctness. High ratio of damaged or erroneous packets/data can reduce the performance and reliability of the application. Incoherent data can be detected on a lower level and retransmitted immediately, and coherent but false data can be detected on application level. Partially damaged or erroneous data may endanger a fair broadcast.
- **Packet loss:** High packet loss ratio can reduce the performance and reliability of the application, but generally, packet loss can be handled by traditional methods, this aspect offers no research challenges.
- **Network congestion:** The broadcasted polls and their competing replies may cause network congestions that may highly influence message ordering.

3.2.17.2 Middleware level requirements

- **Timeliness of data:** The application has only standard requirements in the timeliness of data.
- **Logical consistency:** The application requires no special logical consistency properties.
- **Temporal consistency:** The application's feasibility highly depends on the temporal consistency of the polling messages and their replies.
- **Trustworthiness of data:** The application's feasibility highly depends on an authentication method that guarantees the unreputiability of the messages of the requesters and offering providers.
- **Robustness:** The application has no special requirements in robustness.
- **Message ordering:** The application's feasibility may highly depend on the correct ordering of the reply messages. (It depends on the business logic of the requester that may rely on this ordering.)
- **Completeness, accuracy and timeliness of failure detection:** The application has no special requirements against the applied failure detection techniques. Undetected communication failures endanger a fair competition if they affects a part of the ad-hoc network only.

3.2.17.3 General requirements

Geographic extension: The range of single polls can be supposed to be less extended. Not an issue.

Failure mode assumptions: Content failures of both service requests and service offers always have to be taken into account. Timing failure is not to be considered, but for guaranteeing the fair competition, message ordering is to be kept. A message transmission delay within some seconds (at most) is to be guaranteed, transmission delays of broadcasted messages before delivering at the first potential provider is not critical. Both content and timing failures might occur as a result of faults at the message originator side, at the receivers side, during the information transmission through the wireless communication links, or as a result of faults and malicious behaviour at the transmitting partners. Malicious behaviour may result consistent failures as well. None of the failures may lead to severe injuries or to the loss of lives.

4. Use cases

According to the definition, a use case is a set consisting of (one or more) applications, the actors and roles involved and the identification of the affected dependability domains. The identified applications for a use case are assumed to occur in a certain context where these applications typically appear together and interact with each other. The actors and their roles represent the glue of the use case and are important for the kind of interaction between the applications. In that sense, they have an impact on architecture dicussions expected in subsequent WPs.

Each use case is complemented with failure modes and challenges. These challenges go beyond the challenges in the descriptions of single applications in the sense that they already take into account the interaction between different applications in the use case, taking into account the actors and their roles. The purpose of the failure modes and challenges is to initially stimulate architecture discussions and resulting solutions for dependability in subsequent WPs.

In general, the use cases described below shall serve as the basis for further investigations in the HIDENETS project.

4.1 Platooning use case

4.1.1 Selected Application(s)

The only application involved in this use case is “Platooning”, see section 3.2.6, where the leading car sends control information to the following cars, which immediately have to process the control data and act accordingly. For this, a very fast and reliable one-hop/multi-hop data transmission is indispensable.

4.1.2 Actors and their roles

Platoon member: Each platoon member must transmit its position and speed information periodically. Accelerations and in particular breakings needs to be transmitted additionally. The cars following have to process the received information and act accordingly, e.g. adapt their speed, brake and steer according to the received information.

Head of Platoon: Additionally to all other platoon members, the car at the head of the platoon needs to add this information (*i.e.* being the head of the platoon) to each transmission.

End of Platoon: Additionally to all other platoon members, the car at the end of the platoon needs to add this information to each transmission.

Data forwarder: If multiple cars follow the leading car, the control information might have to be transported via multiple hops in case the cars at the end of the platoon are out of reach of the head of the platoon. Some cars then also have to act as data forwarders.

4.1.3 Challenges and failure modes

There are a number of challenges and failure modes related to platooning. They are ranked by their importance.

Very important challenges and failure modes:

- If messages arrive at a platoon member with too much delay, this may result in too late reactions to a maneuver in the platoon, leading to safety problems. The sources of delay in the networking part needs to be investigated and appropriate means for timeliness are required.
- It is essential that all messages sent out can reach all addressed platoon members. This touches throughput but also transmission errors. Means to ensure sufficient throughput and a reliable communication link are needed.
- A worst case situation is if two large platoons driving in opposite directions meet. In this case, the amount of radio resources needed by the two platoons increases in the affected geographic area and needs to be guaranteed for each of the platoons. This requires appropriate means to detect such events and to re-distribute the available radio resources quickly and efficiently.
- Any information inside the platoon needs to be trustworthy. False messages may lead to traffic accidents.

Important ones:

- Larger platoons may need not only single but multihop communication. This may introduce additional delays and use more radio resources. It is unknown whether multihop communication for platooning is possible and whether mechanisms for the efficient use of radio resources in this case exist.
- Multihop communication in large platoons may lead to increased data volumes which increases required resources. A solution may be to condense platoon data for multihop communication.
- Data integrity is important. The information distributed may e.g. contain information about breaking status. If a platoon member gets wrong information about a vehicle ahead which breaks hard, this may lead to traffical accidents.
- A fault in the system may cause false messages which are a hazard to traffic safety. Possible solutions will involve fault detection and recovery mechanisms

4.2 Infotainment and work with highly mobile terminals

The scene is a car, taxi, bus, train etc, and also pedestrians may be included. A worker wants access to his firm’s Intranet to get access to or upload important documents, update his calendar, check email, meet some deadline etc. While travelling, he may take part in a teleconference discussing important strategic decisions before the upcoming meeting later that day.

Other travellers want to play online games during the travel, watch TV or a video, collect tourist information about the scenery passing by or shop products on the web etc.

In this use-case we envision a multitude of different devices within and outside of cars, and people with different preferences. We also expect a wide diversity in the applications requested. How the ability to use more than one access point, if possible, can improve the dependability and how the high mobility of the clients/terminals (possibly frequently changing communication channels and network locations) influence the communication delay will be a main motivation. Multiple alternative access points and technologies to a fixed infrastructure will therefore be assumed. Possibly multihop communication over an ad-hoc network is used to reach the fixed-network gateway.

Floating Car Data (see 3.2.1) is not included in this use-case, since this application is covered in another use-case (Assisted transportation, see section 4.4). However, it might be possible to merge them if necessary.

4.2.1 Selected application(s):

All applications involve Internet access on the move. The applications that are relevant for this use case can be listed as follows:

1. **Information**
   - **Tourist information** - Users in cars and pedestrians may request information that is suited for their current geographical position or context. Also information about future locations and contexts may be relevant. Such information may be requested by the users on-demand or alternatively one could imagine some kind of subscription the information is pushed down to the users as messages or streaming audio/video/data. Thus, many types of applications may be used for delivering tourist information, and location/context awareness may be a particular (add-on) aspect of all these applications. Potential applications and their requirements can be found in sections 3.2.12 - 3.2.15.
   - **Floating car data** - Floating Car Data (see 3.2.1) is not included in this use-case, since this application is covered in another use-case 4.4. However, it might be possible to merge them if necessary.
2. **Entertainment**
   - Music, radio - these applications and their requirements are described in 3.2.12.
   - Radio, Video, TV - these applications and their requirements are described in 3.2.12.
   - On-line Gaming - these applications and their requirements are described in 3.2.11.
3. **Office**
   - **Calendar and email synchronisation -** these applications and their requirements are described in 3.2.14. In addition to the communication level requirements, calendar and email synchronisation may have high requirements with respect to data consistency.
   - **Documents upload and download** - this application and its requirements are described in 3.2.14.
   - **Document sharing -** this application and its requirements are described in 3.2.15.
   - **Teleconference –** this application and its requirements are described in .
   - **Voice call -** this application and its requirements are part of the description in .
4. **Web-shopping -** this application and its requirements are loosely described in 3.2.15.

4.2.2 Actors and their roles:

End user(s): Human interfacing with the applications, and/or interacting with another end user

Server: Providing tourist information, music, video, broadcast, gaming, shopping

Terminal: Either hand-held or integrated in a car. The users interface to the services/applications.

Routers: Terminals or dedicated routers that route traffic between end points.

Gateway: Connects the ad hoc domain or a terminal to the fixed infrastructure and the Internet.

4.2.3 Failure modes, Challenges and subjects for further investigation:

This use case consists of different application which will experience and be affected by different modes of failures. For example tourist information is not a very time critical application, but it will not work properly if messages are too much delayed and therefore out of interest for the user. Another risk is that the service may deliver messages that are not relevant for the current position. For the music, radio, video and TV applications a large receiver buffer may solve some of the failure modes. However, to many messages out of order and too much content error will be perceived by the user as noise and poor audio and video quality. For

the online gaming application a buffer may not be appropriate due to possible real-time requirements. For the document uploads and email/calendar synchronisation it may be critical to make the application terminate with a complete result. Inconsistency between the different versions may cause difficulties. Teleconferences share some of the same failure modes as online gaming due to real time requirements. Web-shopping share some of the failure modes with office synchronisation, like importance to complete. It is also important to deliver messages within a threshold, so that the application will not terminate without completeness.

From the communication level requirements described for each application in Chapter 3, it is already concluded that different applications require different treatment with respect to e.g. delay, throughput, jitter and packet loss. To manage the resources in the network, mechanisms for differentiated service delivery and routing is necessary. This might also be necessary due to different capabilities in routers and gateways. Radio resource management must also consider this heterogeneity in application and equipment requirements and capabilities.

Since ad hoc networks are based on wireless communication and mobile nodes, the service delivery may be interrupted due to several reasons. Resilient routing techniques are necessary to ensure that traffic can be quickly rerouted due to loss of neighbours or congestion. It should also be possible to roam between different gateways. Since applications require different treatment, differentiation should also be considered in the case of rerouting and roaming.

Wireless and highly mobile networks might also operate with tailored transport layer and media layer mechanisms (e.g. packet resending) to improve the throughput and the delay in the wireless environment. Some applications like web shopping do not require very strict delay bounds; however they require hard guarantees with respect to session completeness and correctness. Communication delays in dynamic multi-channel/multi-protocol/mobile environment and effective store and forward mechanisms in the ad hoc domain will be important in that respect.

A short list on challenges and candidate research topics will be as follows:

- Highly resilient routing techniques
- Differentiated routing
- Differentiated resilience
- Effective and stable mechanisms for roaming
- Effective RRM, possible across different radio technologies
- Tailored transport layer and media layer mechanisms
- Effective store and forward in the ad hoc domain
- Server response time.
- Privacy and confidentiality in the ad hoc domain
- Effective QoS specifications and adaptation strategies

4.3 Car accident (incl distributed black-box)

This use-case evolves about a scene with an accident on a road, involving cars and other road users. The use case covers mainly what happens after the accident but also involves some issues directly before and during the accident. General driver assistance and collision avoidance are not part of this use case, they are treated in the use case “assisted transportation” in section 4.4.

Directly before the accident, the distributed black-box functions of the cars in the area collect time-stamped information. This information is backed up to other cars as they pass, as well as to fixed-network servers whenever access to the fixed infrastructure is available. The higher the percentage of cars equipped, the lower is the risk for loosing data.

Right after the accident, many people may try to call the emergency services, call home, and send text and multimedia messages, at least in motorways with high traffic density. This may cause congestion in the radio access network, both in WLAN-type technologies and in mobile networks. It is a challenge for the networks to be able to prioritise between the requested services according to some pre-determined parameters (service type, user class …).

Some time after the accident, an ambulance is approaching and the cars along the road are notified either directly through the ad-hoc network or via a central unit that broadcasts the message to the cars along the particular road as it is approaching. The alarm center personnel may already at this stage know the names of the persons involved in the accident, and even pictures may have been transferred from the accident scene. Essential data on the injured, along with possible pictures of the crash scene, may be transferred to the ambulance while on its way.

Arriving at the place in question, there may be a need to communicate with medical expertise at the local or a central hospital by use of voice, video and data transmission (multi-media application). There may also be a need for group communication with other emergency teams at the site. Heading back to the hospital with the injured there will be a need to transmit information on the positioning of the ambulance to communicate that it is approaching the hospital and at the same time maintain the multimedia connection with the medical expertise.

Afterwards, what really happened in the accident can be found by investigating data collected by the distributed black box application.

Traffic information on the basis of floating car data that could guide cars around the accident by presenting alternate roads is not included in this use-case, since this application is covered in another use-case (assisted transportation). However, it might be possible to merge these use cases if necessary.

4.3.1 Selected application(s):

The applications will in this case require high dependability due to the emergency nature and some will also have real-time requirements:

1. **Emergency communication**
    - **Emergency vehicle warning** - this application and its requirements are described in 3.2.14.
    - **Online notifications to hospital** –- this application and its requirements are described in 3.2.14. Due to the emergency nature of this use case, the dependability and security requirement of this application will be higher in this use case than in the normal case. Requirements regarding transmission delay are loose, but messages should have a very high probability of reaching the receiver.
    - **Access to medical expertise (multimedia)** – description and requirements for the components of this application can be found in the sections , 3.2.15, and 3.2.13. Due to the emergency nature of this use case, the dependability and security requirement of these application components will be higher in this use case than in the normal case.
    - **Group communication at crash site** - description and requirements for the components of this application can be found in the sections , 3.2.15 and 3.2.14. Due to the emergency nature of this use case, the dependability and security requirement of this application will be higher in this use case than in the normal case. Further it is essential that these applications work when terminals are connected via a single hop to the infrastructure, but also when no connection to infrastructure exists (ad-hoc only).
2. **Distributed black box -** this application and its requirements are described in 3.2.7.
    - Automatic backup on neighbouring cars in the ad-hoc domain,
    - Automatic backup on the fixed network whenever possible

Generally, application components used in emergency applications will have the same or higher requirements to quality/performance as compared to the “internet access on the move” applications. Further,

they will have higher requirements regarding dependability, integrity and stability, for instance having priority over other applications when communication resources are scarce, as often is the case at crash scenes, or keeping a session while on the move.

4.3.2 Actors and their roles:

End user: Human interfacing with the applications and/or interacting with another end user. This is relevant for communication with medical expertise. The black-box application should most of the time run automatically, in a transparent way, without user intervention. Built-in mechanisms should be developed to manage automatic backups on neighbouring cars located in the vicinity as well as on the fixed network. Nevertheless, the user should have the possibility to activate the backup whenever needed.

Server: Providing emergency vehicle warning, patient information, and collecting "black-box information".

Terminal: Either hand-held or integrated in a car. Serves as the users' interface to the services/applications. Emergency vehicle warning, online notification and part of access to medical expertise will involve terminals. For the distributed black-box application, a terminal embedded in the car could display the information collected as well as the data summarizing the backup process. When an accident has occurred and an information access is requested, a secured terminal should be used for ensuring access rights if we want to give the possibility to the car owner or other authorised third-parties to display the data on the car.

Routers: Terminals or dedicated routers that route traffic between end points.

Gateway: connects the ad-hoc domain or a terminal to the fixed infrastructure and the Internet.

Atomic data provider: Cars which broadcast "atomic" information about their current position, driving direction and speed.

Data forwarder: A data forwarder takes atomic or condensed data and forwards it into the network without modification. Data forwarders can be fixed stations and cars.

Data processor: Cars or servers in the fixed network collecting data from other cars and condensing it in order to build an image of the traffic situation (e.g. for road segments). Input data can be (1) atomic data from other cars or (2) already condensed data which is updated or further processed, taking into account several information sources. Output can be (1) newly condensed data or (2) updated data or (3) data with a higher level of condensation.

4.3.3 Failure modes, challenges and subjects for further investigation

The applications in this use case are associated with slightly different failure modes. Emergency vehicle warning and online notification to hospital will not function properly if the messages are too much delayed or contain too many errors to be interpreted. The delay bound is exceeded and the messages are useless if the ambulance is ahead or very short behind the message. The multimedia applications associated with access to medical expertise will consider message delay, errors and out-of-order messages as a failure mode due to bad video and audio quality. Group communication at crash site can contain both notifications and multimedia communication, and will share failure modes with the application mentioned above. In addition, there may be failure modes with respect to group connection and management. There may exist messages that only make sense if all of some specific group members can receive them.

Some details about failure modes for the distributed black-box can be found in Section 3.2.7.

The overall challenge for emergency communication applications is to give the ambulance a dependable and secure network connection to the central site (hospital or alarm central) while driving. This may be achieved for instance by being able to utilize and seamlessly switch between several different network technologies and access points. Further the emergency applications should be given priority over other applications. Even emergency applications with relatively loose requirements for performance/quality (throughput, delay, loss), may have high requirements regarding reliability, and should be given priority. Techniques for differentiated service quality, differentiated resilience, highly resilient routing and differentiated rerouting may help to

achieve this. The radio network is a particular important area for studying these issues, and effective radio resource management techniques should be studied.

For some applications, it must be possible to roam between different gateways/access points. For the ambulance, particular requirements exist as to keeping the session while on the move. For the distributed black-box application, ensuring timeliness is very important for opportunistic communication both with cooperating cars when in the ad-hoc domain and with access points when in the infrastructure domain. Fast and seamless handover mechanisms, possibly between different access network technologies, should be studied. Differentiation should also be considered in the case of roaming and handover.

Wireless and highly mobile networks might also operate with tailored transport layer and media layer mechanisms (e.g. packet resending) to improve the throughput and the delay in the wireless environment. Server response time and effective store and forward mechanisms in the ad hoc domain will be important in that respect.

A short list on challenges and candidate research topics could be as follows:

- Highly resilient routing techniques
- Differentiated routing
- Differentiated resilience
- Effective and stable mechanisms for roaming
- Effective RRM, possibly across different radio technologies
- Tailored transport layer and media layer mechanisms
- Effective store and forward in the ad hoc domain
- Server response time.

Considering the case of the distributed blackbox application, classical protocols such as those defined in 802.11 seem to be suited for this application and it is not necessary to develop new network protocols. The application relies on one hop ad-hoc geocasting while in the ad-hoc domain and on routing to a fixed backup service while in the infrastructure domain. The switch between the ad-hoc and the infrastructure domain should be notified to the application in order to reconfigure it. The amount of information produced and stored by this application is low but its freshness is critical, i.e. reducing communication latency is more important than ensuring throughput.

It is mainly concerned with the tolerance of faults affecting data, and not computations. Thus, the main challenges concern the development of efficient solutions to the tolerate faults affecting data, and not computations. Thus solutions should be designed to ensure the consistency, integrity, availability and confidentiality of the data itself.

- Availability of the black box information: this is the main goal of the application, the information availability must be maximised despite faults.
- Confidentiality and privacy: the black box information should be accessible only from authorised parties. Only the original car should be allowed to write data, and only the data owner (or its delegates, e.g. its insurance company) should be allowed to read it.
- Integrity of the black box information: the original information produced by the car at the provider site should not be modifiable, either by the driver or by the other cars hosting copies of the original data, or by any third party.
- Logical consistency: the backed up data must be consistent, i.e. the data reaching the fixed server, or the cooperating cars, should not be modified. Only the producer must be able to write the data, the other entities have no need for write (or even read) access on the data.

- Temporal consistency: it is important that the information disseminated in the ad-hoc domain and stored in the infrastructure domain reflects the real situation. Only most up-to-date data need to be backed-up and restored in the infrastructure domain.

4.4 Assisted Transportation

The Assisted Transportation use case covers the situations in which driving by car is subject to general constraints, like time constraints, route constraints (the need to pass by specific locations), or both. In this use case, several applications might be used, independently or in combination, in order to better assist the user in achieving its goals. These applications will provide information concerning the several issues that may determine, in particular, the route taken by the car and its speed: road conditions, traffic conditions, weather conditions, vehicle components conditions. To help the user in making good decisions, assistance should be provided by exposing the several alternatives and some estimation of their goodness. Doing this in a dependable way clearly requires the use of appropriate communication solutions to support all these applications.

As for the challenges that must be addressed in this use case, the distributed, heterogeneous and unpredictable nature of the environment in which the several applications will be deployed will make it harder to provide the necessary dependability guarantees. Therefore, it will be necessary to devise solutions to deal with the general challenge imposed by unpredictability. More specific challenges can nevertheless be identified and are listed in Section 4.4.3. It is necessary to note that, in any case, the specific user requirements, which depend on the nature and goals of the transportation, will ultimately provide a measure for the required dependability, for instance in terms of availability, reliability and integrity needs.

4.4.1 Selected application(s):

1. **Floating car data (FCD)**
   - One of the most important pieces of information that should be available in order to plan an adequate route (one which provides high probability of fulfilling the space/time requirements) is information about traffic jams, traffic congestion, and accidents on the road, which has a large impact in terms of the travel speed and probability of meeting deadlines. An FCD application (see description and specific requirements in Section 3.2.1) can provide this information, which is collected and distributed through messages exchanged between cars over ad-hoc communication channels. This requires multihop communication, in particular in the case of centralized FCD. Therefore, an FCD application can play a fundamental role in the assisted transportation use case.
2. **Traffic sign extension (TSE)**
   a. With currently existing technology, traffic signs can become “intelligent” and may easily be extended to proactively disseminate information to relevant vehicles passing by. This can be done, for instance, by means of event dissemination through wireless connections, which should then be processed by receiver modules or other applications in the vehicles. Such an application is clearly relevant for the assisted transportation use case and its description and requirements are provided in Section 3.2.2.
3. **Hazard Warnings and Information from other vehicles (HWI)**
   - Overall safety can be improved by using applications that provide hazard warnings or sporadic information, both from other cars on the road (see Section 3.2.4 for details and requirements) or even information generated from the own vehicle (see Section 3.2.5). This information might help preventing accidents – which would definitely compromise the objective of meeting certain deadline requirements. Hazard warning applications definitely fall under the scope of an assisted transportation use case.

4. **Unusual driver behaviour warning (UDBW)**
   - This application, which is described in Section 3.2.3, has similar relevance in the assisted transportation use case as the HWI applications mentioned above. The objective is to use all the possibly available information that might help the user in meeting all the restrictions within the expected or anticipated bounds (dependably), which can be better achieved by increasing the safety margins.
5. **Interactive data between cars/terminals (ID)**
   - Even if there exist applications to support collection of information in an automatic way, it is certainly good to have additional sources of information, namely by allowing the establishment of direct communication channels to support interactive communication with nearby cars. With such an application (which involves real-time requirements, as described in Section 3.2.15), even if this is the only available, it may be possible to conceive an assisted transportation use case in which assistance is provided by some "guiding" vehicle using interactive communication with the guided car drivers.
6. **Interactive data from internet**
   - This is probably a marginal application for the assisted transportation use case. However, one can easily imagine several situations in which access to the internet while on the move might be useful to assist transportation. While the information that is made available through the other applications (DFCD, Traffic sign extension, etc) is severely formatted and semantically limited, the Internet is a source of (almost) unlimited, unstructured and semantically varied information, which might be of help if browsing and searches are done in the appropriate way. In general, richer context information can be retrieved by users, increasing the probability of finding good solutions in very specific cases, for which automatic support is of no help. In this sense, interactive data can be considered as a last resort application for assisted transportation. This application is described in Section 3.2.15.
7. **Maintenance/Software updates (MSU)**
   - A maintenance and software updates application falls in the scope of the assisted transportation use case as it can be used to improve or correct the behaviour of software components in a vehicle, keeping them up to date with respect to new versions released by a manufacturer (e.g. communication software, information repositories – maps, servers, etc –, in-car monitoring software). Maintenance information can be exchanged with central servers when a vehicle is put into operation and does initial verifications. Maintenance activities can be handled in different ways depending on their criticality (for instance, the user might be warned of an imminent failure, and possibly prevented to drive before some maintenance is done). The requirements for this application are provided in Section 3.2.8.

4.4.2 Actors and their roles:

End user: The vehicle driver, mainly, or someone who might be helping the driver by keeping track of information provided by some application.

Terminal: The information collected through the different applications (DFCD, Traffic flow control, Traffic signal extension, Hazards warnings) may be displayed on a terminal embedded in the car, in a more or less integrated way (which depends on how and which of the several applications are integrated). Hand-held terminals may also be used for some applications (internet access on the move or interactive real-time communication with users). Other applications (maintenance and software updates) require minimal user interaction.

Atomic data provider (in DFCD, HWI): Cars which broadcast "atomic" information about their current position, driving direction and speed, which can be used as input to hazards warning applications (e.g. detecting stopped cars in the way).

Data processor (in DFCD): Cars or servers in the fixed network collecting data from other cars and condensing it in order to build an image of the traffic situation (e.g. for road segments). Input data can be (1) atomic data from other cars or (2) already condensed data which is updated or further processed, taking into account several information sources. Output can be (1) newly condensed data or (2) updated data or (3) data with a higher level of condensation.

In car data providers (HWI, UDBH, MSU): Sensors and intelligent nodes in the car act as data providers for Hazards warnings, unusual driver behaviour warnings and for maintenance purposes. In-car communication may be relevant for the interconnection of these actors. Real-time concerns are usually important at this level of abstraction (intra-car communications). However, in general, this level of interaction is abstracted by the car itself, which is the main actor.

Data forwarder (in DFCD): A data forwarder takes atomic or condensed data and forwards it into the network as is. Data forwarders can be fixed stations and cars.

Server (in TFC, MSU, IAM): Front-end servers that are able to process positional, traffic related or maintenance related information sent by cars, replying with adequate responses. A server may just process information requests and reply with the requested information.

Routers, gateways, etc (in TFC, MSU, IAM): These are necessary actors in applications that communicate through fixed network infrastructures, involving multi-hop connections.

4.4.3 Challenges

In order to provide convenient assistance to car drivers, it is fundamental to ensure that all information received and processed within a car is up to date, that is, consistent with the real conditions of the environment. Therefore, timing failures represent a serious risk for the correct behavior of some applications considered in this use case, namely floating car data, traffic flow control and warning related applications. In fact, if timing failures are not treated appropriately, they are transformed into value failures, meaning that users will not just have a late or delayed view of actual conditions, but a wrong view at a given moment. Various techniques, including filtering at central processing servers (in the case of TFC), can be used to remove inconsistent data, but it is still necessary to handle timing failures at the car side. The risk of timing failures is more serious in the case of applications based on multi-hop communications.

On the other hand, in general, omission failures do not represent such a serious risk, provided that it is possible to guarantee minimum levels of connectivity and avoid complete disconnection. This is because in assisted transportation, it is normally necessary to continuously retransmit updated information, thus allowing an application to recover from previously missed information. Such a continuous transmission of information has however a cost, in all the cases where the transmitted information is redundant. Important resources could be saved in these cases, if event based communication was used instead, just to communicate relevant changes to previously transmitted information. An interesting research issue is therefore to better understand this compromise between required resources and the possible impact of omission failures.

Trustworthiness issues are relevant in the remote software update application, particularly if these updates concern relevant software for the car safety. In most of the other applications, authenticity of content is desirable, but should not be an issue when clients must register and servers are well identified. In the case of traffic sign extension, it may be an issue to identify and authenticate the information provider.

Complete disconnection or crash failures are not a fundamental problem, provided that these failures do not lead to incorrect information being provided. At worse, applications should behave in a fail-safe manner, not producing any more output.

4.5 Brigade communication

4.5.1 Selected application(s)

The only application involved is “Mobile mission control centre”, where collaborating mobile entities establish restricted ad-hoc networks without the exact predefinition of the actual members. To keep us as far as possible from military applications, let us consider a road reconstruction brigade that works on an extended section of a road, which is not closed for the passing traffic, but the passing-by traffic is controlled by provisional traffic lights. Besides collecting and distributing reconstruction work related data, the mission control centre plans and schedules the moves of the reconstruction machines and controls the passing traffic.

The proxy can download a large amount of mission related background information (e.g. maps, construction plans, actual mission data, weather forecast, satellite photos etc.) from the fixed main infrastructure and then it can relay them to the local mobile units (as a proxy does it), and it can multiplex the traffic between the infrastructure and the local mobile units (thus serving as a provisional access point, if necessary). The centre itself can be mobile as well, and the members of the restricted ad-hoc network are the entities of the given company that are willing to collaborate and are "in the nearby".

4.5.2 Actors and their roles

- **Brigade members:** Brigade members (e.g. reconstruction machines, material delivering trucks and staff transporting cars that join and leave the actual working area) may connect to the restricted ad-hoc network after authentication and then may use the services of the proxy (access point). They may get plans and schedules for their moves in the working area, but following these plans is the responsibility of the driver of the given brigade member.
- **Mission control centre:** This specially equipped car – that serves as the office of the brigade manager – has special equipments for the communication with the background servers in the infrastructure domain, serves as a gateway between the ad-hoc and infrastructure domain transmitting messages and eventually caching information from the infrastructure domain, and controls the ad-hoc network and its traffic. It may plan and schedule the moves of the participating cars.
- **Background providers:** Service providers in the infrastructure domain that reply queries originating from or transmitted by the mission control centre.
- **Outsiders:** Cars that are not member of the brigade. They may participate in the communication as non-dependable transmitter nodes, but they must not play an active role in the communication. (Outsiders must not directly influence safety-critical control, must not be able to read/write/modify business-internal or mission-internal data, but they may provide data about the actual car flow, which is information that can serve as input for some decisions in the mission management.) Outsiders may get orders from the mission control centre through the traffic lights.

4.5.3 Challenges

The devices of a road reconstruction brigade are exposed to an increased risk of accidental faults. (This is equally true for police or fire brigades, as well.) Only the treatment of faults of the centre and that of faulty communication lines offer research challenges. The restricted nature of the ad-hoc network may provoke intrusion attempts of any kind. Through tampering with mission data one may attempt to impede a mission.

Content failures are to be taken into account in the case of faults of the communication channels, of the caching, of the planning/scheduling of moves or of the traffic control. Timing failures can arise from faults in the caching strategies. Measures are to be taken to detect communication channel and member outages, and data damages. Inconsistent failures might occur when accidental faults happen, but malicious faults – caused by an intruder – can turn into well designed consistent failures.

Caching and proxy services should be further investigated. A generally flexible but secure way of identification of potential brigade members has further security related aspects. The dependable use of the independable outsiders in communicating traffic control data and confidential data raises further dependability questions. The use of the outsiders implies routing aspects as well (they should not route messages to the far from the working area). The mission control centre has to efficiently transmit high-priority-small-size data and low-priority-large-size data (e.g. engineering information) as well. From the bandwidth differences between the local ad-hoc network and the dedicated link to the infrastructure domain other networking issues may arise.

4.6 Service discovery in ad-hoc networks

4.6.1 Selected application(s)

The only application involved is “Ad-hoc service providers”, where public ad-hoc networks are limited to the entities "near" to a given geographic location to guarantee that the set of the potential service providers is strongly limited by their geographic distance from the requester. A restaurant (or a similar smaller business) may offer a taxi call service to its customers on leave. The restaurant is not bound to a single taxi company, but it rather calls the actually "nearest & best" car every time. The actually available taxis are ranked based on the preferences of the restaurant (or even of the customer). The ranking function is not public for the taxi drivers, but it can depend on the reply order of the taxis, on the estimated arrival of the taxis at the restaurant, on the price system and reputation of the taxi companies etc. The restaurant dispatches the actual carriage to the best-ranked taxi car.

4.6.2 Actors and their roles

- **Service requester:** The restaurant that initiates the poll, collects the replies, ranks them and dispatches the carriage.
- **Interested providers:** Taxis (taxi drivers) that are actually near to the geographic location of the service requester and are volunteering to provide the requested service.
- **Uninterested providers:** Taxis near to geographic location of the service requester that could provide the requested service, but are not actually volunteering (busy or lazy taxis).
- **Outsiders:** Cars near to the geographic location of the service requester that cannot provide the requested service. However, they can participate in transferring the broadcast message.

4.6.3 Challenges

The accidental flooding of the complete network must be avoided. Competing providers may try to corrupt the messages and the system components.

Content and timing failures can appear on the receiver side of the messages when wrong or outdated information is received, or not all competing messages are received on time. The most important failure modes in this use case are:

- in the service request phase: blocking of messages, flooding the network with the message, omission of a subset of the addressees, late delivery (with respect to send-time, obsolete messages), late delivery (with respect to the first delivery of the given message, unfair delivery), consistent corruption of all instances of the message, corruption of some instances of the message

- in the bidding phase: blocking messages to an addressee, blocking messages from a sender, late delivery (with respect to send-time, obsolete messages), change of order of the messages, corruption of messages

Special attention must be paid to the detection of errors that may lead to an unlimited broadcast or other kind of flooding. The network may require recovery mechanisms after a flooding. Security threats may produce consistent failures as well. No failures of the system should be treated as safety critical, however, inconsistent or non-timely delivery of messages can get business critical by ruining the trustworthiness of the system.

The main challenges are how to limit the broadcast to the cars near to geographic location of the service requester, how to fairly distribute the messages and simultaneously avoiding a complete flooding of the communication channel. The fair distribution of the broadcast poll and fair transfer of the replies should be guaranteed even if competing service providers are included in the message transfers. The application can be ruined if the reply (and the offer of the quality of service within) cannot be enforced afterwards (it implies security issues)

5. Business impact analysis

This chapter describes the economic impact related with the introduction of dependability services for the sake of more dependable systems to be used by applications which will be further investigated in the HIDENETS project.

It discusses general effects which occur if more dependable services are introduced and describe the effects which are perceived by the system user on the one hand but also take into account general economic effects on the other hand.

The contents in this section represent the state of the information available in HIDENETS at the point in time when this deliverable is released. Since the available material is scarce and it is difficult to derive it from other available material, HIDENETS intends to further work on it and to refine the business impact analysis in the course of the HIDENETS project.

5.1 Related work on business analysis and dependability for car-to-car communication systems

Most of the applications will be used in a car-to-car communication environment. There is only little material available on the business aspects of car-to-car communications as well as on the business effect of increased dependability in communication systems in general. Two main publications in this field are [20], [21]. There, the general problems and difficulties with respect to the market introduction of car-to-car communications are analysed with the result that the mechanism here is different from the normal process of equipping cars with new technology. This is due to the fact that the functioning of an application depends on whether other cars are equipped with car-to-car devices as well, whereas other new technologies have a direct benefit for the user.

This business impact analysis does not consider these general market introduction mechanisms. It rather assumes that the cars are equipped with the required technology. It is the lack of dependability and the introduction of dependability means at which we are aiming our interests.

5.2 Business effect of the introduction of more dependable services

The first thing to be considered in the business analysis is why, from a business perspective, a certain degree of dependability is required. This immediately leads to the question what needs to be payed for having increased dependability.

As to the first point, dependability is often related with the **acceptance** of an application or a set of applications. The issues to be considered in this respect are:

- Safety criticality. Whenever an application is safety critical, i.e. its failure may lead to severe injuries or even the loss of lives, the application is acceptable only if the failure rate is close to zero. This point is important for both the service users, since it is their health which is at stake and insurance companies which will not be willing to pay for a failure of a system which is not proven to be safe.
- User satisfaction. Many applications considered in HIDENETS rely on a certain throughput, a bounded delay or a response time below certain limits. The user will not be willing to use an application which annoys him to the point of dissatisfaction.
- Costs. Many applications remain among cars, so it is probable that these applications can be used for free. Others, however, rely on communication with the fixed network, where the assumption is that at least the fixed network access is subject to charges. The communication service used for such

applications must be competitive with regard to costs which clearly limits the effort that can be spent for dependability.

The second point, the **costs** to be paid for increased dependability, has a number of aspects which are related with the following points:

- Usually, if a service is improved in a cooperative communication system, this causes increased effort for management related communication between the involved devices. This effort cannot be used for the communication of application anymore which decreases the share of bandwidth available to the system users. This conflicts with the requirement to deliver the highest possible QoS at lowest possible costs.
- Dependability is always obtained through the deployment of redundancy - for detection and recovery or for masking - so it implies higher cost than non dependable solutions due to
  - Development of the dependability solutions. This relates to both, software and hardware effort
  - Costs for additional Hardware, be it additional memory in the good case to additional chips and more room on the boards for redundancy solutions.
- There are applications which are more important than others in the sense of availability, see e.g. the explanations above on safety criticality. Combinations of differently prioritised applications at the same location may require that some applications supersede others. The effect is a service interruption or, in the worst case, a lasting degradation of applications with lower priority. This again has effects on the acceptance of applications, see above.

In order to assess the impact applications have on each other, it may be useful that subsequent WPs define a clear order of priorities for all of them, possibly in relation to the use case where they are applied.

5.3 Business impact analysis of use cases

The business impact analysis is performed on a per-use-case basis, i.e. this clause contains a section for each use case and the applications related with the use case.

For each use case, we shortly list the applications involved and proceed with the business analysis according to the following outline:

1. Assessment of alternative technologies. Some applications need not necessarily rely on ad hoc network technology, as mainly considered in HIDENETS, but may just as well use other networking technologies instead. This will be analysed with respect to the competition for the technology used in HIDENETS.
2. Analysis of the costs to be paid for the introduction of versus the benefit from the introduction of a higher level of dependability. This can only be a very rough estimate at this point in time, since HIDENETS is only just starting to work on these issues, so the required results for this analysis are not yet available.
3. Relation of business impact with challenges and failure modes, where applicable.

5.3.1 Platooning use case – business impact

Platooning has only a single application “Platooning”, as described in Section 3.2.6.

5.3.1.1 Platooning use case – Alternative technologies

Platooning is an application which uses communication solely between the cars involved in the Platoon. Additionally to the information exchanged via communication means, some sensor data is usually used in order to increase the safety level of the application. The additional data is information about distance to the car ahead which can be provided by e.g. stereo video, radar or ultrasound sensors.

There is **no competing technology** for this application available. In particular, communication networks with a centralized structure are not suited for this application because lack of coverage and additional network delays make it impossible to use them.

The car-to-car communication technology may, however, have rather low bandwidth due to the relatively small amount of data to be exchanged.

5.3.1.2 Platooning use case – costs and benefits

Due to its very local nature, the direct benefit from platooning goes mainly to the drivers of cars who are members of a platoon. It is a service which remains solely within this platoon, i.e. there is in the first instance no third party having a benefit from it. This also means that car owners who will have to pay for it, where the costs are mainly related to the equipment that is built into the car.

If it is the car driver who has the main benefit, there must be some pay back of the investment. The potential benefits for a car driver are:

- Increased comfort. When driving in platoon, the responsibility of the driver is mainly to steer the car. Acceleration and deceleration are done automatically by following the car ahead.
- Reduced accident probability. Rear end collisions may become less probable. This, however, still requires proof which is out of the scope of this project.
- Lower fuel consumption. Due to the automatic control. Lower distances between cars should become possible. Driving in the slipstream of the cars ahead together with a constant speed reduces fuel consumption.
- Constant and selectable travel speed. A driver can select a platoon with the speed he prefers and join it. He may also change platoons if the head car's driving style does not meet his expectation.

Beyond the direct benefits for a single car driver, there may be **general economic effects** from platooning. Among them are:

- Reduced fuel consumption (see above)
- Due to smaller distances between cars, roads can be used more efficiently, thus avoiding traffic jams.
- Reduced amount of traffical accidents (to be checked)

So it may happen that there will either be additional incentives from governments to equip cars with platooning equipment or that some countries introduce regulations which require to equip new cars with it.

In any case, platooning can be classified as highly safety critical. This means that the use case will only be accepted if the availability and reliability of platooning is so high that catastrophic failures happen only with probability very close to zero. Given this, there is a high probability that the means required to achieve this high level of dependability will be rather costly. Since HIDENETS is only starting to work on these issues, it is not possible at this point in time to assess the additional costs involved for dependability measures and to relate this with the benefits for car owners. However, should there be government regulations which require platooning, the additional costs are of minor importance.

It is also obvious that the high level of dependability and availability required for safety reasons will have an effect on other applications. This means that a platoon MUST have sufficient resources available at any time, regardless of the applications in its surrounding. Therefore, it can be expected that platooning will have a high priority and will supersede other applications in the vicinity. The expected effect is that other applications may suffer QoS degradations.

5.3.2 Infotainment and work with highly mobile terminals use case – business impact

The applications included in this use case are:

- Tourist Information (described in Section 4.2.14 Location-/context-based information services)

- Music and radio (described in Section 4.2.11 Audio streaming)
- Video and TV distribution (described in Section 4.2.12 Video Streaming)
- On-line gaming (described in Section 4.2.10 Online Gaming)
- Calendar and email synchronisation (described in Section 4.2.15 Non-interactive data communication and messaging).
- Documents upload and download (described in Section 4.2.15 Non-interactive data communication and messaging).
- Document sharing (described in Section 4.2.16 Interactive data)
- Teleconference (described in Section 4.2.9 video conference).
- Voice call (described in Section 4.2.8 voice call)
- Web-shopping (described in Section 4.2.17 Web shopping)

The application "Floating Car Cata" (see 3.2.1) could have been included in this used case, but is omitted since it is covered in the use-case "Assisted Transportation" (see 4.4) already.

5.3.2.1 Infotainment and work with highly mobile terminals use case – Alternative technologies

The focus of this use case is information and communication between the wireless and the wired domain. Some of the applications in this use case could use other technologies. For distributing TV, for instance broadcast technologies could be used for distributing television, and the cellular voice network could be used for producing voice calls. However, there is no single alternative technology that may be used to support all these applications. In fact, our idea is to use a multitude of technologies to produce communication services for the applications, and select the technologies that at every point in time is best suited to the applications.

5.3.2.2 Infotainment and work with highly mobile terminals use case – costs and benefits

Benefits for users:

- cheaper services since "cheaper technologies" may be used
- increased coverage – coverage via ad-hoc networks in neighboring cars to a WLAN base station
- increased bandwidth – use neighboring car's access to a WLAN base station even though my own car is not within the coverage area of that base station

Benefits for providers:

- increased coverage for cheap, high bandwidth technologies (WLAN)
- ability to produce the services at a lower cost when customers are within range of cheaper technologies, and even extend this range with ad-hoc networking
- differentiated reliability for the applications during failures enables differentiation in customer segments (business customers vs regular customers)

5.3.3 Car accident use case – business impact

The applications included in this use case are:

- Emergency vehicle warning (described in Section 4.2.15 Non-interactive data communication and messaging)
- Online notifications to hospital (described in Section 4.2.15 Non-interactive data communication and messaging. Note however that due to the emergency nature this application, the dependability requirements will be higher than for a normal application in this category. Although requirements on transmission delay are loose, but messages should have a very high probability of reaching the receiver)

- Access to medical expertise (multimedia) (described in Section 4.2.9, 4.2.16 and 4.2.13. Note however that due to the emergency nature this application, the dependability requirements will be higher than for a normal application in this category.)
- Group communication at crash site (described in Section 4.2.9, 4.2.16 and 4.2.13. Note however that due to the emergency nature this application, the dependability requirements will be higher than for a normal application in this category. In particular, it is essential that these applications work when terminals are connected via a single hop to the infrastructure, but also when no connection to infrastructure exists (ad-hoc only).
- Distributed black box (described in Section 4.2.19 Distributed black box)

5.3.3.1 Car accident use case – Alternative technologies

For most of the applications mentioned for this use case, communication will take part in both the ad hoc domain between cars and also into the infrastructure domain. For communication between cars we are not aware of any competing technologies. For communication to and in the infrastructure domain there might be other technologies. Many countries are planning to build special purpose emergency networks solely for emergency communication. Such networks will probably cover group communication at site, online notification to hospital and access to medical expertise. This differs from the Hidenets approach where the goal is to exploit and improve public communication infrastructure to meet the dependability requirements of emergency communication.

Concerning the distributed black box application included in this use-case, this concept is original and has several advantages that will make it attractive to the market. To the best of our knowledge, there is no similar application in the market yet. We can mention the Celestica's eDevices technology acquired by IBM. It consists in an embedded computer equipped with GPS and GSM communications. It has been deployed by Norwich Union in UK for a "pay as you drive insurance". The device in the vehicle uses GPS to record the location where a vehicle is driven as well as information which can be used to calculate the vehicle average speed. The black box measures vehicle usage and transmit real-time data for analysis via GSM. The program also allows examination or replay of any trips taken by the vehicle, enabling the company to conduct usage and risk analysis to refine its billing process.

Compared with the solution investigated within Hidenets, this solution is rather limited: GPS-only positioning, GSM communication, type of applications of limited scope, etc. The collaborative approach taken by the project is aimed at developing a real virtual black-box, similar to traditional black boxes, taking advantage of the mutual resources available in the ad-hoc domain to enforce fault tolerance and improve data availability and integrity. These advantages, along with the reduced-cost of this approach make this solution appealing.

5.3.3.2 Car accident use case – costs and benefits

The benefits of offering the applications relevant for a car accident are indisputable very high. And the cost of building the mechanism and infrastructure needed to support these applications can be worthwhile. With respect to cost, the Hidenets approach will be less costly than a national special purpose emergency network due to several factors.

- Can take advantage of existing public infrastructure, both WLAN and cellular based
- With strict dependability and service differentiation, the resources can be used to public communication in the absent of emergencies,

Concerning the distributed black box application, it is clear that the technology needed to deploy this application is less costly, when compared to traditional black boxes that are based on specific hardware. The benefits are also high. For example, the capabilities offered for analysis of the causes of an accident are improved since besides recording information concerning the corresponding car , the distributed black-box application can also be used to record contextual information about the environment and the other cars in the in the vicinity.

5.3.4 Assisted transportation use case – business impact

The applications included in this use case are:

- Floating car data (FCD) (described in section 3.2.1)
- Traffic sign extension (TSE) (described in section 3.2.2)
- Hazard Warnings and Information from other vehicles (HWI) (described in section 3.2.4)
- Unusual driver behaviour warning (UDBW) (described in section 3.2.3)
- Interactive data between cars/terminals (ID) and internet access (described in section 3.2.15)
- Maintenance/Software updates (MSU) (described in section 3.2.8)

5.3.4.1 Assisted transportation use case – Alternative technologies

For the applications mentioned in this use case, communication will take part in the ad hoc domain between cars as well into the infrastructure domain. Some features of FCD application are also available in commercial technologies, for instance there are many GPS software for route planning. However, these technologies differ from the Hidenets approach because they are based only on geographic and signalling information. In general they do not take into account traffic flow information, such as traffic congestion, traffic jams, accidents on the road, or any other information acquired by other cars to trace a route. An alternative technology closer to our approach, to collect traffic flow information and to calculate up-to-date information about the current traffic flow on roads, already exist in Germany (see e.g. Gesellschaft für Verkehrsdaten (DDG), http://www.ddg.de). The calculation of the current traffic flow is performed by a server inside the fixed network domain. Instead, our approach can be performed in a decentralized version of floating car data. This results in a very detailed notion of the traffic flow state around receiving vehicles.

For the traffic sign extension we have no knowledge of competing technologies. For this application the communication is essentially between cars and infrastructure. The closest application that we are aware of is automatic toll payment, which is used in several countries in Europe. Some of these systems are based on the use of RFID technology, and there are projects to use location based technology to support this application.

Nowadays, hazard warnings about the car itself are usually offered to a driver. However, our approach rely on hazard warnings about the surroundings' behaviour, for instance a post crash warning, a sudden driver behaviour change, a drunken/sleep driver warning, etc (see Sections 3.2.3 and 3.2.4 for more details). To our knowledge, there is no competing technology based on hazard warnings considering the behaviour of other vehicles.

5.3.4.2 Assisted transportation use case – costs and benefits

There are many benefits offered by these applications concerning assisted transportation use case. A number of capacities more or less innovative are provided by cars. For instance, the driver will take advantage of navigation facilities, provided by Floating Car Data, such as a more accurate tracing route, based on updated information provided by other cars. Route guidance, localisation of other cars (for example a friend) and interactive data between cars are examples of other facilities offered to the driver. However, the most obvious benefit of these applications is the lower fuel consumption, due to optimised trace routes, or interactive guidance, and saving time by avoid traffic jams, higher traffic flows, etc.

Safety support is provided through dissemination of information in regard of hazard warnings and unusual behaviour of other cars in the neighbourhood. These applications could avoid accidents occurrence. The cooperation support offered will also allow accidents occurrence to be detected by others quickly. This service will benefit drivers affected by an accident, once the assistance will arrive faster.

The sign extension application could be used to enforce laws that are neglected. This would benefit all drivers on a road. However their implantation depends on governmental acceptance. Certain levels of

accuracy and availability must be met, once the speed limits need to be guaranteed. Additionally, security must be guaranteed, once malicious information could harm the traffic flow or even establish a dangerous zone for the drivers.

Although performance degradation of FCD and TFC applications does not disturb car functionalities, inaccurate response time could generate trace routes out of date, useless for the driver, for example. A worst case concern hazard warnings or unusual behaviour alerts. Unbounded delays or inaccurate response time can confuse drivers. Monitoring and adaptive approaches tend to reduce the degradation levels and, more than this, they must warn the driver about the actual services coverage. A reliable knowledge about the coverage reflects directly on the driver expectation. Some costs on development of useful assisted transportation applications concerning to network facilities, monitoring and replica management.

Trustworthiness is a desired requirement to applications relying on internet access. Maintenance and software updates costs are increased by needed security approaches, such as certification and cryptography protocols.

5.3.5 Brigade communication use case – business impact

Brigade communication has only a single application “Mobile mission control centre”.

5.3.5.1 Brigade communication use case – Alternative technologies

Brigade communication requires the equipment of the brigade vehicles only, because the communication between the mission control centre and the brigade members or among the brigade members may not get stuck in the absence of external resources. The mobile mission control centre application does not consider the remote control of the brigade members by the mission control centre (as opposed to the platooning application), therefore it is less safety critical and it has weaker dependability requirements.

The alternative technologies are based on either static brigades (as opposed to ad-hoc organisation) or on common communication channels of all brigades of a single company. While the first alternative solution obviously reduces the flexibility of building brigades, the second one shifts the responsibility of separating the brigades to the application level.

5.3.5.2 Brigade communication use case – costs and benefits

Due to its very local nature, the direct benefit from brigade communication goes mainly to the owners of the brigade. It is a service which remains solely within this brigade, i.e. there is in the first instance no third party having a benefit from it. The costs are mainly related to the equipment that is built into the brigade member vehicles and into the mission control centre.

If it is the brigade owner who has the main cost, there must be some pay back of the investment. The potential benefits for a brigade owner are:

1. Reliable communication channel between the brigade members and the back-end information infrastructure of the owner through the proxy function of the mission control centre. Any member of the brigade can get up-to-date additional information from the back-office in any unexpected cases.
2. Reliable communication channel between the brigade members and the on-site mission control centre. The mission control centre can coordinate and (re)schedule the actions of the members continuously that makes any mission more efficient.
3. Increased flexibility in building brigades. There is a brigade specific communication medium without requiring a static predefinition of brigades. Members may leave or join to the mission as actually required without additional management activities.

As the target audience of the use case is construction brigades, firefighter, police and military brigades, their more flexible and more efficient cooperation has obvious general economic effects as well. This general economic benefit may lead the governments or road traffic regulation authorities to require the device

manufacturers to build in an obligatory support for these brigades of general interest. Otherwise brigades can not count on the support of passing by cars for transferring messages as those have no benefit from offering their resources.

The classification of the brigade communication regarding safety criticality highly depends on the (non-) autonomicity of the communicating devices.

- When only devices with local intelligence (e.g. vehicles with autonomous human drivers) are connected, the broken/corrupt communication channels should not lead to severe injuries or to the loss of lives.
- The safety criticality increases if the communicating devices rely on the communicated information without being able to check the immediate consequences of their actual actions. (E.g. when a machine moves materials to places that cannot be directly seen by the driver, or when soldiers fire to places that cannot be directly seen by them.)
- It is highly safety critical if devices without local intelligence (e.g. traffic lights in a construction area) are connected.

The security requirements of the brigade communication depend on the field of application, too. (The security requirements of a road construction obviously differs from that of a police/military mission.) The applicability of the HIDENETS architecture will highly depend on the safety and security parameters that can be provided by him. But as the expected costs of building a communication system that is based on this architecture and that offers the necessary dependability parameters for most of the targets of this use case is not much more than building a traditional one without these extra features, it seems to be highly probable that the benefits will justify the extra costs.

5.3.6 Broadcasting use case – business impact

Broadcasting has only a single application “Ad-hoc service providers”.

5.3.6.1 Broadcasting use case – Alternative technologies

Ad-hoc service providers application targets the fair and geographically limited distribution of service requests among the locally available service providers and the reliable and confidential collection of actual service offers to be evaluated by the requestor according to his personal preferences. The bidding providers must be reliably notified about the acceptance/refusal of their offer.

There is no competing technology that can guarantee a fair competition among the geographically nearly located service providers. In particular, centralized service dispatchers do not have enough real-time information about the providers and they are not able to implement the different special evaluation aspects of the customers.

5.3.6.2 Broadcasting use case – costs and benefits

The critical point in the spread of the ad-hoc service providers application is how to gain a critical mass of services providers over to invest into it. The main job is to break the vicious circle in which providers do not invest as long as there are not enough customers and customers do not use the application as long as there are not enough providers to guarantee a reliable service offer. The circle can be broken the easiest in case of services where the expected profit on the increased competition among the providers is the highest.

After reaching the critical mass in the equipment ratio, service customers may benefit from

- the competition of more providers,
- the competition of more actual offers,
- the flexible tailorable evaluation process.

Service providers may benefit from

- the quicker return of better offers,

- better adjusting their offers to the actual parameters.

Other service providers currently dominating the market of the given service may be counterinterested in the spread of this application. There will be a general economic effect from increased competition of the service providers.

The ad-hoc service providers application is not safety critical, but its acceptance highly depends on its fairness, reliability and confidentiality. As the communicating partners are (may be) competing service providers, they may be interested in attacking the message flow to and from their competitors. Therefore the application has to be highly robust against malicious threats as well. In addition, the application has to limit the broadcast of service requests in a very reliable way, otherwise the application may unnecessarily consume large amounts of resources while forwarding messages out of the targeted geographic area. This not safety critical application may not require large bandwidth or very reliable low-level communication channels, as this would increase its cost in a non-acceptable extent.

6. Preliminary Reference Model

Reference models (RM) generally are collections of Best Practices which have been collected in the course of a specific project or in a specific field. Examples for such RM are the ISO OSI (Open System Interconnection), [28], RM and RM ODP (Open Distributed Processing), [29]. Both example reference models have the following general contents:

- Generic collections of issues incurred during development of respective systems
- A structured collection of
  - Terminology
  - Best practises for system design, structuring and development
  - "Checklists" with all possible features and options a system can require
  - With generic contents
- They shall serve as a framework for concrete developments
  - Selection of what is needed
  - Guidelines and principles for development processes and methods used
  - Starting point for more concrete methods (e.g. CORBA, UML)
  - Tailoring to specific system development project always required

The HIDENETS RM shall be a result from the overall HIDENETS project which follows the guideline above. HIDENETS covers different aspects of dependability in distributed communication systems in different WPs and, therefore, each WP is a potential source of best practices for the reference model. The results will be generalized and structured such that they are widely generic and independent from specific applications or systems.

Beyond this, the HIDENETS RM will contain a general framework with a taxonomy and definition of terms which are important for the considered field.

This chapter contains the preliminary HIDENETS RM. It can only be preliminary in month 9 of the HIDENETS project, when this deliverable is released, since most of the WPs are only at their beginning and the results not yet fixed. The following chapters will outline the expected contents of the reference model which is due in month 18 of the HIDENETS project and the final lessons learned documentation which is due at the end of the HIDENETS project.

6.1 Reference Model General Part

The general part of the HIDENETS RM will contain issues which are mostly part of this documentation already.

The dependability framework, as described in Section 2.1, will be integrated into the HIDENETS RM. The dependability framework mostly relies on dependability related publications which set the guideline for terminology and properties in this field. Since dependability is an integral part of HIDENETS, it will appear there again.

Communication level services and properties, as described in Section 2.2, as well as middleware level services and properties, as described in Section 2.3, will appear again in the HIDENETS RM. The descriptions there will be shortened and revised in order to reflect the points which are relevant and suitable for an RM. The terminology used there will be defined carefully and structured in the sense of a taxonomy.

Beyond the communication and middleware level services and properties, the RM will include a reference network architecture and an architecture model of a node. The reference network architecture will present ad-hoc and infrastructure network domains with accompanying high-level descriptions of the network elements and technologies. The node architecture model will show the main functional structure of a node

(car/terminal). A detailed picture of the node architecture introducing functional building blocks at different layers is being developed by WP2 and WP3 together. A detailed description of these blocks and the interfaces between them will also be elaborated in these WPs. In the RM however, only a simplified picture of the node architecture will be included. This will be a major part of the input from WP2 and WP3.

The purposes of the node architecture are:

- Listing and structuring of the functional building blocks of a HIDENETS node
- Displaying the relations between different building blocks
- Define which blocks should contain various dependability related functions

The functions in the middleware and communication domain will be listed and allocated to blocks in the simplified architecture model.

A first version of the simplified node architecture is shown in Figure .6.1. The node consists of some hardware (HW) that may be installed in a car or be part of a separate terminal, and some software running on it. One particular piece of hardware is the network interface card that allows the transmission of information out on the network. Other relevant hardware parts may for instance be GPS devices.

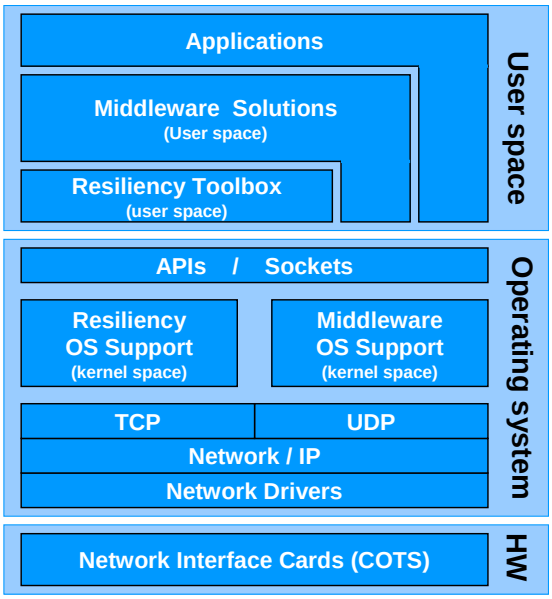

Figure .6.1: Simplified node architecture

The node software may be part of the operating system or it may be implemented in user space. Regular applications that may be installed and run by users are always implemented in user space. Since user space applications are thought of as potentially untrusted they are only allowed to access the operating system functions through well-defined APIs or sockets. On the other hand, software that is included in the operating system is thought of as trustworthy and is allowed to use operating system functions, read variables and even interact with low-level hardware.

Resilience functions in user space may be implemented as separate functions, included within middleware, or built into the applications themselves. In the operating system, resilience functions may be categorized in three main blocks: Middleware OS support, Resiliency OS support, and resilience support in the network protocols (TCP/UDP/IP).

6.2 Reference model: expected results from WP2 – Resilient architecture and middleware

Middleware solutions can be used as building blocks for the development of complex systems and applications. They provide enriched functionalities and an abstraction of the underlying processing and communication infrastructures properties. For the design of middleware services it is important to understand several conceptual models with impact on the functional and non-functional properties that can be offered to applications. Relevant conceptual models that we will consider in HIDENETS include failure, synchrony, interaction, topology and complexity models. In general, WP2 will contribute to HIDENETS with the design and development of middleware services that provide improved functionality over state-of-the-art solutions, based on new protocols and advances on the those conceptual models or combinations thereof.

6.2.1 Conceptual models for the design of middleware services

The definition of a system model implies making assumptions about a certain number of aspects. Which aspects to consider and how they can be used to classify different system models is what we address in what follows. In the context of HIDENETS, we are interested in modelling distributed systems. In this kind of systems, where computations involve several processes or nodes that need to exchange information among them, a major concern has to do with modelling the communication between processes. Besides that, assumptions about how local computations will take place are also relevant when it comes, for instance, to deal with real-time issues.

With respect to communication, we are essentially interested in message-passing models, which can be characterized by the assumptions they make about the following four concerns: *failure*, s*ynchrony*, *network topology* and *message buffering*. On a higher level of abstraction, different forms of *interaction* among processes can be considered, which may have impact in terms of middleware service properties. On a different but not less important dimension, other aspects relative to the distributed system *complexity* should also be considered.

Failure models

In the development of dependable systems, failure models are of extreme importance, since a system is made dependable by applying fault tolerance measures to tolerate the assumed faults. In HIDENETS we are essentially interested in communication failures, which are perceived at some service interface during the execution of an interaction. While a general framework for dependability, as described in section 3.1, should be used, specific failure models, defined by the number and classes of failures that must be tolerated, can be considered. There are three fundamental classes of failures that may be considered: the *omissive* class (including crash, omission and timing failures), the assertive class (including syntactic and semantic failures) and the arbitrary class, which includes all the other failures not covered by the former classes. A special case of arbitrary failures are the Byzantine failures, which refer to inconsistent semantic failures, such as when a process sends different copies of a message to different recipients.

A failure model can be made weaker or stronger. A weaker model is less restrictive with respect to the kind of failures that may happen, but is more demanding in terms of fault tolerance measures to achieve dependability. When the probability of the occurrence of a failure is negligible, the failure model can be more restrictive without compromising dependability, thus reducing the fault tolerance efforts. Finding the adequate failure model has implications on the final system complexity and depends on the required resilience vis-à-vis the actual infrastructure and operating conditions (which determine the probability of failure occurrence). In HIDENETS we intend to explore weaker failure models, more appropriate for the kind of environments that are considered in the use cases defined in WP1, even at the cost of increasing the complexity of the solutions that are needed to address the assumed classes of failures.

Synchrony models

Synchrony models allow characterizing a system with respect to the existence of notions of *time* and *timeliness*. The weakest model is the *asynchronous* model, also called *time-free*. A time-free model makes absolutely no assumptions related to time, which means that message delivery and process response time can be arbitrarily large. Furthermore, in time-free models there are neither clocks nor any other form of measuring time intervals. On the other extreme, the strongest model is the *synchronous* model, in which communication and processing delays are always bounded, and the bounds are known. Intermediate models are known as *partially synchronous* models. They do only the necessary time and/or timeliness assumptions that will allow satisfying some required properties with the appropriate dependability. For example, the Timed Asynchronous model, [47] , only assumes the availability of local clocks with a bounded rate of drift, and the Timely Computing Base model, [48] , assumes that some parts of the system are synchronous while other parts may be asynchronous. The former is powerful enough to allow the detection of timing failures, while the later also provides the means for a controlled and timely reaction to those failures. The use of partially synchronous models is attractive for the work in HIDENETS, since these models allow capturing the dynamics and heterogeneity of the operational environments considered in the use cases.

Topology models

The network topology determines how messages can be routed from one process to another. System models that assume a network topology where all the nodes are connected to each other are the most simple and general ones. However, the cost for having such nice models is that it may be harder to enforce the assumption of a completely connected graph. Usually this requires the execution of low-level routing protocols, which must be constructed over weaker assumptions about the network topology. At some point, assumptions about the actual network structure, that is, how the nodes are physically connected with each other, may have to be done. At this level, the network topology may assume well-known forms such as a ring, a mesh or a tree. The problem gets even more complex when the actual network topology may be changing over time, which is usually the case of networks based on ad-hoc connections. This is particularly relevant in HIDENETS, since we consider operation in both infrastructure and ad-hoc network domains. Dealing with dynamic topologies is therefore another source of complexity for the solutions to be developed in HIDENETS.

Message buffering models

In distributed systems, using message-passing models, the time it takes for a message to be transmitted from the sender to the receiver is non-negligible. This can be abstracted by considering that a link connecting two processes has some buffering capacity to store one or several messages being sent. In fact, models can assume either finite or infinite buffers. In practical systems the capacity is known to be finite, but it is seldom large enough to make infinite buffering a reasonable and commonly used abstraction. How messages are ordered within buffers is another issue. The weakest assumption is that any ordering is possible, which means that any two undelivered messages can be received in any order, independently of the order in which they were sent. More restrictive are models that assume FIFO (first-in-first-out) buffering, that is, they assume that messages (if not lost) are received in the same order in which they were sent. In any case, higher level interaction models usually provide stronger abstractions with respect to ordering properties, as described next.

Interaction models

Communication among distributed system processes can be modelled with more powerful and more interesting abstractions, delivering properties not available at lowest communication levels. A distributed application can be constructed more easily if the appropriate interaction model is used. Of course this requires the availability of the communication primitives implementing that interaction model. Typical interaction models include the Publisher/Subscriber, the Client/Server and the Group-based model. In HIDENETS, different interaction models might be better suited for the various applications considered in the use cases. Publisher/subscriber models are essentially event-based and provide anonymous communication, which may be adequate for applications in the ad-hoc domain in which transmitted (published) information is relevant for possibly many (unknown to the sender) receivers (subscribers). The client/server model is well suited, for example, in applications where interactions with a server in the infrastructure domain take place.

Finally, group-based communication usually provides a set of rich communication primitives, with strong message ordering properties (e.g., total order or causal order), which may be useful in applications involving groups of processes performing a common task or coordinating with each other, such as in platooning. Whichever is the case, middleware services should be developed having in consideration HIDENETS high dependability requirements.

Complexity models

When developing middleware algorithms and protocols, heterogeneity and scalability issues become decisive. The complexity of the environment measured in terms of the heterogeneity of target (hardware and software) systems, as well as in terms of the number and distance of nodes is directly proportional to the complexity of the solutions. An approach that is commonly taken to address heterogeneity problems is to use common languages, allowing for the interoperability of components. HIDENETS will explore the concept of *architectural hybridization*, not only making use of the intrinsic heterogeneity of components in terms of their properties (reliability, synchrony, security), but assuming at the architectural level the existence of such heterogeneity in order to make use of it in the development of the middleware solutions. In terms of scalability, HIDENETS assumes a possibly large number of components, which will imply increased complexity of the solutions. One possible approach to the problem is to use hierarchical composition of the solutions, thus reducing the complexity at each hierarchical level. For instance, a model of hierarchical composition would allow for information in traffic flow control application to be condensed at each level before being sent to another level, thus reducing the sise of information and the complexity of the solutions.

6.2.2 Expected results

Node architecture

WP2 will work together with WP3 in the definition of the HIDENETS software architecture of an (ad-hoc) node. This will encompass the definition of several building blocks and interfaces, whose description will be done in these WPs. WP2 will focus more specifically on middleware services to be located on upper layers of the architecture, in user space. Nevertheless, the relations, dependencies and interactions among the several service blocks, not restricted to middleware services, will be described in WP2.

Architectural hybridization

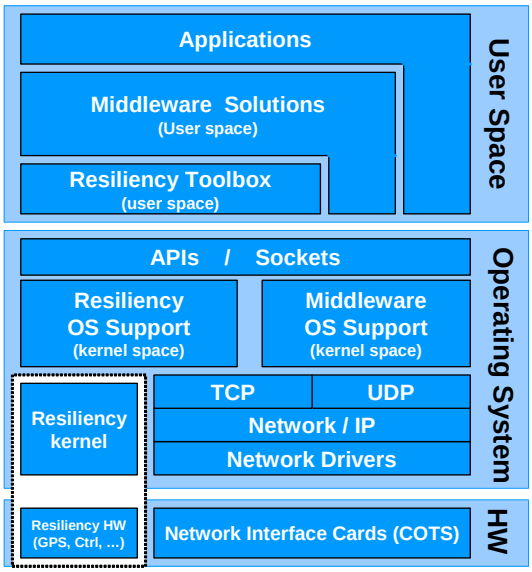

Figure 6.2: Resiliency services and protocols

WP2 will define architectural solutions to deal with the required dependability needs, in particular exploiting architectural hybridization, allowing the provision of specialized oracle services to deal with reliability, synchrony and security requirements. Figure 6.2, derived from Figure .6.1, illustrates in a very simplified

manner the concept of architectural hybridization to be exploited in WP2. The architecture encompasses a well defined part, a resiliency kernel, with its own synchrony and reliability properties, which is specifically concerned with the provision of specialized support services, securing more stringent properties due to this hybrid design and the provided services with reduced complexity.

WP2 will design a set of middleware services/mechanisms (as identified in the general architecture) that are able to provide resilience for the use cases identified in this document. This encompasses the definition of the necessary protocols and algorithms necessary to implement the services. In HIDENETS, we distinguish the ad-hoc network and the infrastructures domains. Therefore, some of the services will be more oriented towards operation in the infrastructure domain (or in ad-hoc to infrastructure interactions), while others will be fundamentally designed for operation within the ad-hoc domain. WP2 will also provide implementations of the developed services, as required for the construction of proof-of concept demonstrations of the purposes and benefits of those services.

Oracle services and protocols

WP2 will provide the definition of the oracle services to reside on the resiliency kernel. This will serve as a basis for the work to be then continued in the scope of WP3, concerning the definition of strict timeliness and trustworthiness requirements. Therefore, there will be a strong connection between WP2 and WP3 with respect to the definition of these services. The implementation and the construction of proof-of-concept prototypes of these oracle services will be done essentially in the scope of WP3.

6.3 Reference model – expected results from WP3 – Resilient Communication

As Internet enters into the wireless scene new business opportunities arises and new services are offered to both the business and the residential markets. We see a trend towards the demand of always being connected to a network, even on the move. Dependability and QoS has not been focused in such scenarios but we expect this will change as increased interests for such services will emerge. This means that new challenges for network infrastructure and resilient communication will appear. One such aspect is the ability to use the different access networks in combinations to enhance the user experience and the network resiliency. This may be a promising business case.

To support dependable services it is crucial that we design network topologies, routing and resilience strategies to ensure an appropriate quality of service in a cost efficient manner, even in case of failure. The best solution for a given network depends on a number of factors, including reliability requirements. Methods need to be developed for formulating these requirements in a suitable manner. For such purpose ‘ideal’ structures will be studied by mathematical and simulation methods. It is also necessary to develop methods for mapping between real networks and these more ‘ideal’ structures in a way enabling them to be implemented in an appropriate way.

Different user groups and different applications will have different requirements. To design a cost efficient network with appropriate routing and resilience strategies implemented it is therefore essential to identify different dependability and security requirements and to offer services adapted to the different needs. In a failure situation, when resources become scarce, we can then prioritise certain applications/users/services over others (differentiated recovery and reconfiguration) to be able to retain the most critical applications. Besides the car-to car scenario WP3 will therefore investigate use-cases were differentiation between users and applications within the wireless access and ad-hoc domains - and in terms of traffic management and priorities in failure situations - with favour can be implemented.

In contrast to the fixed network, the ad-hoc component may be highly dynamic due to the addition, removal, mobility of the nodes, changing radio propagation conditions, and even routing disruption attacks. Multi-path (re) routing algorithms will be developed that are able to cope with fast changing topologies as well with the mobility of the (ad-hoc) network. The use of simultaneous active interfaces between the ad-hoc and the fixed network components (multi-homing) will be investigated regarding dependability.

Modern wireless networks employ robustness against transmission errors on various protocol layers. The amount of added temporal and information redundancy is thereby in trade-off between increased transmission volume and delays on one side and increased bit error rates and packet drop rates on the other side. A joint optimisation of the redundancy across all protocol layers will be performed and with the aim of increasing efficiency of the end-to-end transmission in a large set of relevant scenarios (channel conditions, traffic models, etc.).

Different dependability, security and real-time requirements will require different solutions. For the most demanding services the supporting mechanisms could be the most expensive ones, at least in terms of resource usage. Of course, solutions have to account mainly for those requirements which are “more strict", that is which are utmost relevant with respect to the criticality of the offered service, as perceived by the user/application. However, such requirements are frequently in trade-off with each other, e.g. increased dependability may lead to decreased performance. In such cases, the proposed solutions have to accommodate a balanced harmonization among dependability, security, and real-time aspects.

Solutions for communication protocols that can support services with strict dependability, security and real-time requirements, accounting for both accidental and malicious faults in networks of mobile devices, will be sought for. Examples are emergency services and some business critical services. A major goal will be to pursue efficient resource utilisation given the stated requirements.

Beyond the contribution to the network and node architecture in the general part, WP3 will make specific contributions to the HIDENETS RM, as outlined below.

6.3.1 Expected results from WP3

Network architecture

WP3 will deliver a high level description of the HIDENETS network architecture, i.e. a description of ad-hoc and infrastructure network domains with accompanying high-level description of the network elements ad-hoc node including access technologies for an ad-hoc node. Further description of gateway, access point, base station, WLAN functionality (ad-hoc and infrastructure mode), GPRS and UMTS radio functionality, and a summary of relevant network management and control functionalities. Other network elements may be included as applicable.

Software node architecture

WP2 and WP3 have together started to draw a detailed picture of the software architecture of an (ad-hoc) node, introducing blocks at different layers. A detailed description of these blocks and the interfaces between them will be elaborated in these WPs. For inclusion in the reference model we will produce a simplified picture with fewer building blocks and describing the interrelations between these blocks. Accompanied with it we shall deliver a clear picture of how the different tasks interact.

Also, the integration of resilient solutions at different layers should not make the software less stable or robust because of interaction effects (e.g. common access/write of a variable by blocks at different layers). Therefore, a thorough analysis of the cross-layer design will be added to the block interaction discussion. This analysis shall focus on cross-layer design techniques and on potentially conflicting interactions between blocks.

Fault analysis

WP3 will do a fault analysis, identifying possible faults at the communication level and analyse the consequences of such faults within the HIDENETS architecture. The result of this fault analysis will be delivered to WP1 as input to the discussion of possible solutions to deal with the identified failure and service degradation modes that need to be addressed.

Recovery schemes

WP3 will in the first run give a state-of –the art overview of resilience schemes, including aspects such as protection and restoration mechanisms, differentiated resilience, multi-path routing and broadcasting and

multi-cast schemes to be included in D3.1.1. This will be completed with new developments as part of general WP3 work. This will be input to the reference model in WP1 for the discussion of basic fault tolerance strategies to address dependability and resilience requirements identified from the use cases and the layers at which these mechanisms should be applied.

6.4 Reference model – expected results from WP4 – Quantitative evaluation

Complex software and hardware systems are widely used in different applications and they have become pervasive in many fields of human activity. Each system demands some specific properties, as a certain level of availability, reliability, performance or quality of service (QoS), whose quantitative evaluation has become a key issue in information technology and computer science.

The quantitative evaluation of these system's properties is performed following two basic approaches: **measurement-based** and **model-based** (both analytic and simulation). In the first approach, the required measures are estimated from measured data using statistical inference techniques, and the data are measured from a real system or from its prototype. It is usually an expensive approach, since it requires to build a real system, take the measurements and analyze the data statistically. For this reason it is usually applied to analyze a well-specified subset of the whole system, for example considering some critical system components or protocols. On the contrary, the model-based approach is inexpensive and easier to perform, since the system evaluation does not require to build and measure the system, and it can be used to analyze the whole system behaviour at various levels of abstraction. The model-based approach can be used for system assessment in all phases of the system life cycle. During design phase, models give an early validation of the concepts and architectural choices; allow comparing different solutions to highlight problems within the design and to select the most suitable one. During the operational life of the systems, models allow to detect bottlenecks and to suggest solutions to be adopted for future releases. Moreover, the sensitivity analysis can be carried out after modelling, and it allows identifying system bottlenecks, highlighting problems in the design, and identifying the critical parameters, that are those to which the system is highly sensitive.

The assessment of the dependability-related attributes of the HIDENETS system is a very challenging topic due to its characteristics:

- different networking scenarios, dealing with both ad-hoc/wireless multi-hop domains and infrastructure network domains;
- large number of components;
- highly dynamic network topologies;
- heterogeneity of design of and constant evolution;
- variety of threats (malicious or accidental).

Actually such characteristics are not strictly related to HIDENETS system only, but they are common to many other contemporary application fields having a high interconnectivity between different infrastructures with seamless interactions.

The system complexity previously described leads to several problems that need to be addressed in performing a quantitative evaluation.

- The *state-space explosion problem* affects state-space analytic models and occurs when the size of the state-space (and then the memory requirement) is too high. The point is to find a right balance between the complexity of the models and the level of detail of the analysis. Being able to describe critical complex systems by accounting at the same time for all the relevant aspects is not trivial at all. The models built for evaluating the measures of interest are always a trade-off between correctness of representation of the real systems behaviour and capability to solve the model equations to obtain the

measures. On one side, we would like to perfectly represent the system's behaviour accounting for all the details that can have an influence on the measures of interest. On the other side, we aim to efficiently solve the obtained models despite their high level of complexity.

- The *rare events problem* affects simulation and occurs when there are events that occur at very different time scales, so the computational time needed to obtain a statistically significant solution becomes no acceptable.
- Concerning the *experimental analysis*, there are two main problems: i) the identification of the interesting situations in which to "test" the proof-of-concept laboratory set up, since it is a quite expensive approach that can not be easily tuned during the course of the project, and ii) and the deployment of non-interfering monitoring systems that could affect the meaningful of the observations.

Therefore it appears quite evident that each technique has its own advantages and disadvantages, and its application may be more profitable to analyze some parts of the system rather than others. If we see the HIDENETS system, at a given level of abstraction, as a set of interacting components, we could map the available evaluation techniques to the components they are more fit to be applicable. This is not a partition, since very likely components and subsystems can be analyzed using several evaluation techniques. This approach leads us to consider also the opportunity to exploit the possible interactions among the different evaluation techniques. This is the rationale of the holistic approach: several (maybe complementary) evaluation techniques cooperate to reach a common objective, that is the end–to-end quantitative assessment of dependability and QoS indicators, both as system's characteristics and as perceived QoS of the services offered to the clients or users.

6.4.1 The HIDENETS holistic approach to quantitative evaluation

The application of the holistic approach allows to define a "common strategy" using different evaluation techniques, applied to the different components and subsystems, thus exploiting their potential interactions. In the evaluation of systems like HIDENETS a single technique is not capable to tackle the whole problem. The idea underlying the holistic approach follows a "divide and conquer" philosophy: the original problem is decomposed in more simpler sub-problems that can be solved using appropriate solution techniques; then the solution of the original problem is obtained from the partial solutions of the sub-problems, exploiting their interactions.

The vision of the holistic procedure to pursue a common objective is the following:

1. *Identification* of a common objective (i.e. the end-to-end evaluation of some system dependability attribute), that can not be adequately solved using a single evaluation technique.
2. *Decomposition* of the entire problem in a set of simpler sub-problems.
3. *Solution* of the single sub-problems using an appropriate evaluation technique (partial solutions).
4. *Reconstruction* of the solution of the original common problem, exploiting the interactions among different evaluation techniques (complete solution).

Some of the possible interactions among different evaluation techniques are the following:

- *Cross validation.* A partial solution validates some assumptions introduced to solve another sub-problem, or validates another partial solution (e.g. a simulation model can be used to verify that the duration of an event in an analytic model is exponentially distributed).
- *Solution feedback.* A partial solution (or a part of it) obtained by applying a solution technique to a sub-problem is used as input to solve another sub-problem possibly using a different technique (e.g. a critical parameter in an analytic model is obtained using experimental evaluation).

- *Problem refinement.* A partial solution gives some additional knowledge that leads to a problem refinement (e.g. the architecture of a component changes since it is recognised to be a system bottleneck).

6.4.2 Expected results from WP4

In summary, the expected contribution is twofold.

At one side we get advances in the development of appropriate evaluation techniques capable to cope with the complexity problems previously outlined.

At the other side, we show how the HIDENETS holistic approach can be used to provide a proper response to the specificities and difficulties to quantitative evaluation of the HIDENETS environment. The rationale is to exploit the collaboration among different evaluation techniques in order to obtain a more accurate evaluation of the final QoS measures. In so far we did not mention and do not intend to provide a physical integration of the techniques, but we just highlight the importance of exploiting the logical interdependencies among different evaluation methods.

6.5 Reference model – expected results from WP5 - Design methodologies and testing

In this section we present the two directly related focal points of WP5's contribution (1. modelling and model based development and 2. testing methodology) to the Hidenets RM separately.

6.5.1 WP5 – Modelling and model based development

All modern development techniques for complex systems rely on some kind of modelling. Models are used (i) *to record the specification and design of a system as unambiguous as possible,* (ii) *to allow different kinds of analysis of the design* and (iii) *to support the documentation of the development*.

In latest techniques (e.g. OMG's Model Driven Architecture) modelling plays even a more central role, the whole development process is seen as an evolution process of system models. Nowadays modelling in application development does not only mean a general description of business entities to document business logics, but even implementation related information like platform specific details, deployment, configuration etc. are modelled. Good documentation of this kind of information is even more useful when the applications are to be developed for complex, very special platforms like the one in Hidenets with its special communication and middleware services. Different kinds of qualitative and quantitative analysis require the underlying model to contain many implementation related information that cannot be described in a platform independent model. The (manual, automatic or partially automatic) implementation of the application can be based on the system model only if a platform specific model of the system exists. (Implementing automatic code generation lies beyond the scope of the Hidenets project.)

The development of applications for a special platform requires the representation of domain specific and platform specific entities and constructions. The platform specific entities mainly represent the different domain specific entities of a model **of the application programming interface (API) of the platform.**

Each special platform requires a special programming practice to utilize its strengths to the best. This best practice can be documented in the form of **design patterns** to support the reuse of existing design in application development activities. The patterns provide **development guidelines** by describing some special parts of a system design (including some system entities, their relation to each other and to the rest of the system, the best settings of their parameters and configuration, ...) representing the best practices to solve a

special design issue. In the best case the patterns are documented in such a way that they can be directly inserted into the system model during development, where the given special design issue arises.

A purely practical but very important condition of creating platform specific models is the existence of a model editor that supports the representation of the domain specific entities. Such a **domain specific editor (DSE)** provides special tooling for the application development for a given platform, supporting the easy modelling of the special entities (and of their relations and special parameters ...) of the platform. The same editor can be utilized when documenting the best practices of the application development for the given platform in the form of design patterns. A domain specific editor in the practice helps the application developers by providing a predefined set of platform specific extensions of general model elements. Contrary to a general model editor there is no need to specialize general model elements each time when a special one is required in the application model. By giving up generality a DSE can provide elements of the given platform API to model out of the box.

6.5.1.1 The Hidenets modelling approach

Application developers require an unambiguous description of the platform interfaces of the communication and middleware services, which will be provided by the Hidenets platform to support special applications and services in ubiquitous communication scenarios. Therefore a formal model of the service interfaces will be elaborated in form of standard UML models to support application development.

Based on this formal description a domain specific editor will be created for supporting the development of applications that will utilize the services of the platform. The adequate results of the evaluation and testing work in the project will be summarised into design patterns that help application developers to reuse these results during designing Hidenets based systems. The application of these standardized solution fragments can help the evaluation and testing of the complete application design, if the Hidenets evaluation framework and the Hidenets testing framework will prepare the corresponding standardized model fragments in modelling languages that are appropriate for the chosen evaluation and testing tools. The formal model, the DSE and the design patterns allow the easy and exact specification of each Hidenets specific parameters of every entity in the application model, thus supporting the application implementation step.

6.5.1.2 WP5 Design methodology - Expected results

At the end of the project we expect to posses a specialized modelling approach for developing applications for the Hidenets platform. This modelling approach has to support the application development by providing (i) *a formal model of the application interface of the Hidenets architecture*, (ii) *application development guidelines*, (iii) *design patterns*, and (iv) *a domain specific editor*. Items (i)-(iii) will be part of the reference model, whereas the editor will be a tool to utilize the reference model in development projects.

6.5.2 WP5 -Testing methodology

As will be explained in Section 6.5.2.1, the characteristics of mobile applications and services provide new challenges for the testing technologies. Still, the development of appropriate methods has been seldom explored so far, and remains an open issue. Our contribution will be focused on the highest layers of mobile systems. The objective is to develop solutions for the testing of applications and middleware services in mobile settings, with consideration for both functional and robustness requirements. Section 6.5.2.2 discusses first direction for our investigation and expected results.

6.5.2.1 Challenging issues in testing mobile computing systems

The problems raised by mobile computing systems are discussed hereafter, by considering both fundamental issues of testing and technological issues. There are three fundamental issues: (i) *the determination of testing*

levels, (ii) *the test selection problem* and (iii) *the so-called oracle problem* — or how to determine the correctness of the test outputs. Technological issues concern the platform required to control and observe test experiments.

The determination of a testing level requires the determination of the (sub-)system that will be the target of testing, and of its interface to its surrounding environment. Typically, the testing levels are determined in the framework of an integration strategy that progressively aggregates the system components until a system level test is performed. For mobile applications and services in ubiquitous communication scenarios, a difficulty is that the notions of system boundaries, and of system components, are not as clear as for traditional applications. Moreover the composition of the system may continuously vary during testing. Generally speaking, the determination of the testing levels should depend on the target application and on the target properties to be verified. To the best of our knowledge, there currently exist no practical guidelines to address this problem.

There exist two traditional approaches in software testing for the test selection techniques: white-box selection approaches, and black-box ones.

White-box approaches are based on structural coverage criteria. They are applied for small pieces of software, typically at a unit testing level. An underlying assumption is that the behaviour of the target piece of software is governed by its structure. But in mobile applications, this assumption may not hold. Software running in mobile devices depends on not only the application logic, but also on conditions on contextual parameters that are not explicit in the applicative code. Such a situation is exemplified by [31].

Black-box approaches need a model of the application functions. Currently, there is no standard specification language for mobile computing systems. Some authors have proposed using UML or Mobicharts [32][33][34][35], but it remains to be studied to what extent such formalisms may cover key aspects of mobile computing. The lack of appropriate specification approaches is obvious from the concrete examples of test experiments reported in the literature: it seems that the typical practice is to select test cases manually from the informal application requirements.

Hence, finding effective test selection techniques for mobile applications is still an open problem.

Another problem that needs to be investigated concerns the development of an oracle procedure to compare actual test outputs with the expected ones. The best solution is to have an automated oracle based on a formal specification. However, as mentioned above, the specification of mobile applications is itself a problem. Moreover, in mobile settings, it may be the case that part of the inputs is uncontrollable and unobservable. For example, if a device is tested in a real environment, neighbouring devices can join and leave a network in an unpredictable manner. The resulting inputs for the target device cannot be determined, and the expected outputs cannot be decided. The solution in this situation would be to develop *partial* oracles that perform (weak) plausibility checks. Examples of such oracles for mobile applications are provided in [36][37].

The choice of the test platform must be done with regards to the facilities it offers to observe and control the input and output events, received or transmitted from each component. This is typically done by means of Points of Control and Observation (PCO). A PCO can be global or local. Compared to traditional systems, mobile computing systems need a higher complexity of PCOs in both cases, in order to adapt to the high dynamicity due to the mobility of entities and rich contextual parameters.

The test platform may be more or less realistic with respect to the operational environment. In practice, it is difficult and expensive to test some mobile applications that require checking the combined logic of the involved hardware and software components. For example, in automotive domain, realistic platforms involve prototypes that are implemented into real cars (see e.g., [38]). The cost of such platforms, as well as controllability and observability constraints, implies that part of the testing activities may preferably be performed using emulation/simulation facilities.

Generally speaking, there is a trade-off to be found between the need for realism, and the need to support fine-grain control and observation to implement a given test strategy.

6.5.2.2 Perspectives and expected results for testing

WP4 will contribute to the development of a test strategy to support the testing of mobile applications and middleware services in a *simulated* environment. A discussion of the test platform and test strategy to be considered is provided below. Our investigation will be supported by case studies.

A test platform for mobile-based applications typically consists of three categories of components: Application execution support, Network simulator, and Context controller. Concrete examples of platforms built according to this generic architecture are provided in [39][40].

The Application execution support is needed to emulate the executive support for the applicative code. A requirement is to provide an interface to the context controller and network simulator, so that the application can interact with them as if it would interact with a real environment. Since applications and services connect over wireless link, the simulated environment must support a model of the underlying network. The network simulator is responsible for simulating the full functionality of a real wireless network. Network simulators like ns-2 or GlomoSim can be used. The context controller is needed to simulate context information. Applications exploit context information to take different actions adaptively. Context is also needed by the network simulator to set up input parameters for imitating the real network characteristics. There have been several toolkits for simulating contexts in recent years (see [39][40][39]).

This simulated environment needs PCOs to control and observe the test experiments. Depending on the target application and test strategy developed, PCOs can be integrated in each component or can be realized as a global component added to the architecture.

6.5.2.3 Test strategy

Model-based approaches will be investigated for testing both functional and robustness requirements. As the specification of mobile computing systems is not a mature issue, a first step will be to investigate adequate models for both the application functions, and their environment. There will be close interactions in this direction with Task 5.1 on design methodologies. The specification of the environment, including the threats that may affect the application, is expected to be the most challenging issue.

As regards test selection, compared to traditional software, there is a shift in the number and nature of input parameters to be considered: contextual parameters and their variation over time have to become first-class citizens. There is a need for methods to combine these parameters with inputs linked to the logic of applications, so as to obtain effective test data at a realistic expense of effort. Model-based probabilistic methods for test generation [43] should be relevant to account for the uncertain characteristics of the environment, and to guide the sampling of meaningful input combinations.

7. Outlook

This document has been put together by HIDENETS WP1 and contains a collection of material which is important for the HIDENETS project in order to add more detail to the precise subjects that shall be further investigated in the course of the HIDENETS project-.

The application and use case descriptions are stable as such. Minor complements and modifications to the requirements may result as a consequence of the detailed investigations of WPs 2-6, but these modifications will be documented in these WPs.

WP1 will work in close cooperation on the definition of the reference model. This will require a through analysis of the results form other WPs and the derivation of abstract descriptions and best practices which will then be compiled into a suitable form for the reference model.

WP1 intends to further work on a more detailed business impact analysis. The approach is to develop a questionnaire on the effect and the costs of dependability and to ask the industry companies inside HIDENETS, the Advisory Board members and other „friendly“ companies, e.g. SA Forum members, to fill it in. This shall then be analysed and put together into an analysis which shall have a stronger basis than the material contained in this document. However, this approach bears a significant risk that the companies will not be willing to disclose their internal knowledge on business effects which will jeopardise the success of this undertaking.

Annexes

A Annex: Operating contexts

The operating contexts described in this section have been originally identified and are recorded here for the sake of completeness, although they are not represented in the remaining document. They were supposed to serve as example environments which have a major impact on the performance or in some cases the feasibility of the applications listed in the earlier chapters.

A.1 Kind of road

The kind of road is quite relevant with regard to road density, average speed on roads, number of lanes (and consequently car densities). The kinds of roads identified are:

- **Motorway**: A motorway has usually two or more lanes per direction, the directions are separated from each other. Average speeds are high when the traffic is flowing freely. Microwave propagation is mostly of the rural area type.
- **Country road**: Country roads have usually one lane per direction, speeds are medium to high when flowing freely. Microwave propagation is mostly of the rural area type.
- **City road**: Dense road network with low speeds and frequent stop-and-go traffic (traffic lights, crossings). Microwave propagation is mostly of the urban area type.

A.2 Road traffic density

The road traffic density is usually directly related to the average speed and the spacing between cars. High density usually means low speeds and vice versa. The following operating contexts are considered in HIDENETS:

- **Night**. Low traffic densities, high speeds
- **Day**: Free flow traffic, medium traffic densities, medium speeds
- **Traffic jam**: High traffic density, very low speed

A.3 Equipment ratio

The share of cars which are equipped with car-to-car communication equipment has a huge impact on the performance and in some cases the feasibility of an application. Some applications function satisfactorily with very low equipment ratios already, whereas other applications require 100% equipment ratio (e.g. intersection collision avoidance).

The following operating contexts are defined:

- 5% equipment ratio
- 10% equipment ratio
- 20% equipment ratio
- 50% equipment ratio
- 100% equipment ratio

A.4 Communication Partner / involved network domains

The communication partner and the involved network domains are partly restricted to a specific combination for some applications and have a major impact on the performance of other applications. The following operating contexts are defined:

- One hop cars: Only a single hop between transmitter and receivers. This is mostly relevant for road safety applications.
- Multihop cars: connections between cars over multiple hops, without involving infrastructure
- One hop fixed point: One hop connection with a fixed point. The application data is exchanged between the car and some server in the fixed network domain.
- Multihop fixed network connection. The application data is exchanged between the car and some server in the fixed network domain, possibly involving multiple hops in the ad hoc domain.
- Vehicle cluster connection: Connection between cars, possibly using the fixed network for interconnection. The fixed network may provide additional services, such as a location registration and authentication service.

B Annex: Applications which have been identified but which do not occur in use cases

B.1 Tolling

This infrastructure application works on toll roads and uses communications for toll collection without the need for toll plazas along the roadway. Communication is between a fixed station and a car driving through (possibly at low speeds).

The application can be designed to eliminate the need for vehicles to stop at toll plazas, thereby reducing stop-and-go traffic near toll collection areas. This application would reduce congestion and improve traffic flow on toll roads.

B.2 Parking guide

A parking guide can be implemented as a purely distributed application or can make use of infrastructure. Both versions are described here. Note that the in-car application need not necessarily know whether or not infrastructure is involved or not.

Parking guide without infrastructure:

Cars leaving a car park broadcast the location and the exact position of the potentially free parking lot. The information is gathered by other cars and broadcast to a broader geographic area. If a car enters a parking lot that has previously been reported to be free, it broadcasts the new status again.

Parking guide with infrastructure:

The version with infrastructure receives the same information as the version without infrastructure. In addition, the servers in the fixed network domain can obtain additional information, e.g. from parking houses on the number of available parking lots, prices, opening hours etc. The most up-to-date information is transmitted into the car-to-car network. Multihop relaying is possible.

B.3 Friend finder

A friend finder shall enable users to find out where a person he knows is currently located. This application requires a location function which may be located in the fixed network or can be distributed among vehicles.

B.4 Intersection collision avoidance

This application warns drivers when a collision at an intersection is probable. Infrastructure sensors and/or DSRC communications can be used to detect all vehicles, their position, velocity, acceleration, and turning status while approaching an intersection. Weather status and the road shape/surface type can be variables for calculating the likelihood of a collision. The infrastructure unit or the in-vehicle unit determines when a collision is imminent and issues a warning to either a specific vehicle or all drivers in the vicinity, depending on the warning strategy.

The options for implementing this application can be differentiated based on the following criteria:

- whether or not DSRC technology is used to sense vehicles approaching the intersection,
- whether the application intelligence (judgment of collision potential) is in the infrastructure or in the vehicle, and
- whether the warning is given via DSRC or via some other means (e.g. warning lights, variable message signs, etc.)

One potential combination of these elements is an application in which the infrastructure determines the location of vehicles through infrastructure sensors (radar, cameras, etc.) and transmits this information to vehicles in the vicinity. If the in-vehicle application determines that a collision is imminent, it provides a warning to the driver.

Other combinations are equally conceivable.