

# **Device interoperability: Emergence of the smart environment ecosystems**

Version 1.0  
2010-02-12

Authors:

Juha-Pekka Soininen (VTT), Petri Liuha (Nokia),  
Antti Lappeteläinen (Notava), Jukka Honkola (Nokia),  
Kary Främling (Posintra), and Roope Raisamo (University of Tampere)

Tivit/DIEM project

*White paper*

## **Abstract**

The smart environment and device interoperability requires interoperability in three levels in physical environment: transferring bits through communication channel, using services of different devices, and understanding the meaning of information unambiguously. Smart-M3 is information interoperability solution developed in the DIEM project for smart environments. It is based on the idea to share the information of embedded devices using a common ontology models that guarantee that information is understood similarly in each device and application. The key element of Smart-M3 is a semantic information broker that is exposed as an information sharing service in service level. It enables the development of multi-device, multi-vendor and cross-domain applications based in information mash-ups. This paper presents an information interoperability approach and basic principles and research challenges related to how smart environments can be built on top of it. The core parts of Smart-M3 have been published as open source software.

## Contents

1	Introduction .....	4
2	Smart environments .....	7
2.1	Smart Environment interoperability solutions.....	7
2.2	Ecosystem needs .....	10
3	Smart-M3 interoperability platform .....	13
3.1	Objectives .....	13
3.2	Principles .....	14
3.3	Smart-M3 concept.....	15
3.4	Logical architecture of Smart-M3 IOP based smart environment.....	17
3.5	Application development .....	20
3.6	Context-Sensitive and Adaptive User Interfaces .....	21
4	Towards Smart-M3 based smart environments.....	23

# 1 Introduction

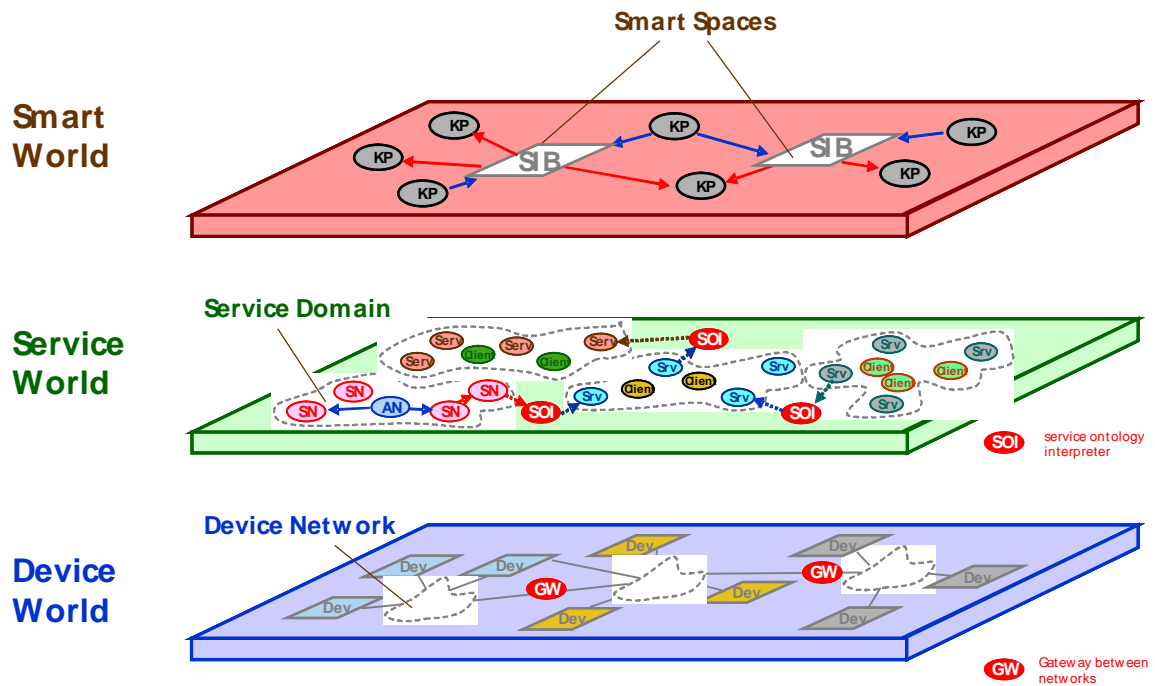
Smart environments and smart spaces are a widely studied concept in ubiquitous computing research, ambient intelligence research, and Future Internet research. They have also a very close relationship to sensor network research, ad-hoc networking research, and location-based service research. The research has been very extensive in this area. It has produced lots of scientific papers, ideas, and concepts during the over 20 year period that has been passed.

The goals and targets have been ambitious. The idea in ubiquitous computing was to shrink the size of a computer so much that you can embed a computing system into anything. The use of this computing power was not considered so much, since it is so easy to figure out possible uses. In ambient intelligence research the idea of serving peoples and processes were emphasised. Context awareness, interaction techniques, etc. were given lots of attention and the vision was that we have an environment that takes away the burden of controlling and managing all small details from us. The development of Internet into a service network created the next paradigm – the Internet of Things. The idea has been to connect all the devices into Internet and then to create networked Internet-based services and smartness.

It is widely agreed that networking devices, opening information, sharing of resources and service, etc. would be great. They would allow us to create much better, more efficient, and more capable services and systems. The expected benefits would be counted in billions of Euros. The idea has even been proved in Internet, where the cloud computing paradigm, for example has been extremely successful. But in physical spaces, in the interaction between embedded systems, in developing smart environments, no major breakthroughs have been achieved. No real large scale smart environment ecosystems exist.

Devices and Interoperability Ecosystem (DIEM) project takes a different approach. The reason why these ecosystems are missing is that current technical solutions do not fit into the business interests of the industry. We aim at developing enablers for smart environments that are simple, flexible, cost efficient, and business-wise acceptable. The key is interoperability between the devices from different domains and the opening of embedded information.

The device interoperability can be divided into three separate levels in order to clarify the different needs of them as in Figure 1. At the bottom we have device world with physical level interoperability and device networks. We need to have a capability to transfer bits between the devices. At the middle we have service world, where the applications are able to use the services also across device boundaries. At the top we have the information world, where the interoperability means that the information has the same meaning in different devices.



**Figure 1. Separation of different levels of interoperability that are needed in creating smart environments.**

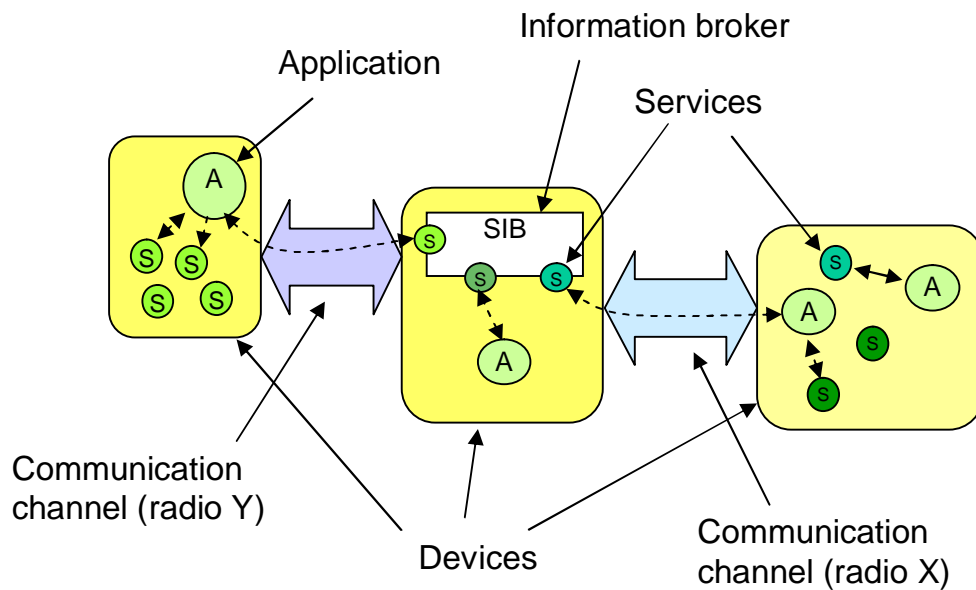
In smart environment all the three levels must exist. From the ecosystem point of view the problem is the world is extremely heterogeneous. The variety of solutions is enormous and it is impossible to support them all or even small subset of them economically feasible way. On the other hand, the variety of technologies is needed in order to be able to create feasible solutions in all possible domains.

In our approach the solution is that we focus on highest level (information level) and rely on existing solutions for solving the service and device level solutions. The core of it is the information and physical spaces such as cars, homes, meeting rooms, shops, streets, etc. The core concept is a common shared memory or information broker existing in the physical space, presentation of information using ontology models, and providing interfaces for legacy service and communication solutions. As shown in Figure 2 the information broker is exposed as a service in different service level solutions and accessed using the communication channels supported by devices.

The key benefits of our approach are that:

1. It provides completely domain independent information sharing solutions for objects and devices in the space. It is also possible to reuse and share the capabilities and information of devices and appliances that have different original purposes and uses in different domains.
2. It is use case and application independent. The solution is like a “TCP/IP” protocol for physical space. It only carries the information. It is easy to build up complete user services step by step as the Internet. It also respects the device integrity, because the final choice for what information to share and how to use the shared information depends on the user or device manufacturer.

3. It will be an open solution that is based on a service oriented architecture concept. The information sharing is offered as a service for devices and applications. Openness makes it easy to include the idea even into simple devices and appliances. Openness and independence from others' businesses is also an important criterion for multi-vendor device interoperability.
4. It is a platform for new innovations. Our approach provides a simple data and information sharing service in physical spaces that is based on ontology models. The use of ontology models provides means for having also interoperability of applications and services. This further enables the possibility to create mash-ups and other type of information, service and application collections in a completely new and yet unknown way.



**Figure 2. DIEM interoperability solution uses common information broker through existing service and device level solutions.**

In addition to great opportunities the proposed approach has also many research challenges. Providing scalability and flexibility both to very small and very large devices and to variable amounts of information will need both advanced information governance solutions and optimised computation platforms. Since the interoperability is largely based on common ontologies and data presentations derived from them, the governance and management of those ontology models will be a critical issue. It will also affect to acceptance in industry. Finally, the possibility to distribute the responsibility of user experience into several devices in the physical space gives completely new challenges and research questions to user interaction and user interface research.

The rest of this paper is organised as follows. Chapter 2 briefly introduces the smart environments and the state of the art of the research in this area. Chapter 3 gives a more detailed description of Smart-M3 interoperability solutions. Chapter 4 describes some of the research problems and approaches related to user interaction issues. Chapter 5 contains our current experience on using Smart-M3 and briefly reveals some of our future plans in selected domains. Chapter 6 summarises our results and outlines our future research ideas.

## 2 Smart environments

Smart environment is a term that not very commonly used in research community. However, some definitions do exist. In most of them the key issue is that smart environment is a physical space full of devices and services that function autonomously on behalf of an individual user [1] by providing an ambient and continuous context [2] and reacting and responding to ongoing activities [3, 4, 5]. Ubiquitous and pervasive computing, ambient intelligence or Internet of Things are used to describe basically the same idea that is also behind the Smart Environments [6,7,8,9].

The core features and attributes that are present in most Smart Environment definitions are following [10]:

1. Smart Environment is a physical space consisting of various embedded systems and electronic devices that are interconnected by wired or wireless techniques. Together these devices integrate the information of the physical world and form a digital virtual space. This physical space orientation separates Smart Environments from pure web-based environments where the actual location of the service is usually irrelevant.
2. Smart Environment should have means for perceiving, measuring and storing the context information of the entities in the environment. Context is defined as any information that can be used to characterize the state and situation of an entity [11]. The context information gathered by an individual device should be made available everywhere in the Smart Environment without human assistance.
3. The applications and devices in the environment should have means to make smart actions based on the information available in the space. The actions to be performed are only limited by one's imagination.

In our work the smart environment definition follows these principles. The smart environment is defined as *information environment and a set of interoperable devices in some physical space that enables the selected exploitation of smart services to user*. We also want to emphasise that the smartness of the environment comes from the applications that exist in the space and it is not inherent property of the environment.

### 2.1 Smart Environment interoperability solutions

*Web Services* provide a standard way for software applications running on different kind of platforms, to interact with each other over a network. Web Services implement the ideas of service-oriented architecture (SOA), which provides design guidelines and principles used during the system development and integration. The fundamental idea in SOA is to package the functionality of the system into interoperable services.

The Web Service architecture consists of three types of actors: Service Provider, Service Requester and Service Broker. The role of the Service Providers is to create and publish services for the use of others. The Service Broker entity registers and

categorizes the services published by Service Providers. Service Requesters can use the Service Broker to find the needed service and then request the service from the Service Provider.

Web Services utilize existing Web technologies such as TCP/IP and the Hyper Text Transfer Protocol (HTTP) in order to provide the communication channel between the entities of the Web Service architecture. The WSDL (Web Service Description Language) service interface consists of abstract and concrete stages. The abstract stage specifies the operations and messages of the service. The concrete stage bounds the interface to specific transport protocols and message formats. The clients can determine from the interface description operations provided by the server. The clients can then communicate with the service using SOAP-messages (Simple Object Access Protocol) that are typically serialized with XML and transferred using the HTTP over TCP/IP.

Web Services provide fast innovation cycles for new services and applications in situations. However, the interoperability is limited to service world level, since no support is given for understanding the information itself. Web Services are also location independent, so they do not directly support the smart environments in physical spaces.

*Universal Plug and Play* (UPnP) is a group of networking protocols that extend the idea of the original Plug and Play to networked system context. The target is to provide distributed architecture for pervasive peer-to-peer network connectivity of intelligent devices. The interoperability of various devices and services is specified in the UPnP Device Architecture (UDA), which is built on top of existing open Internet standards such as TCP/IP, UDP, HTTP, XML and SOAP. It supports features such as zero-configuration, almost transparent networking and dynamic discovery of various devices and services. It is also independent from the media, device, operating system and programming language. [12]

UDA specifies two types of devices: controlled devices (or “devices”) and control points. The controlled devices are the servers of UPnP network and their role is to respond to the request made by the control points. UPnP architecture defines the protocols used in communication between the devices and control points. The highest layers of the UDA protocol stack contain vendor-specific and UPnP forum specific information about the device. Then the UDA specific information is added and the information is structured into UDA format messages. The UPnP networking itself is based on IP addressing

The multivendor interoperability in UPnP is based on use case specific standardization of the service interfaces and because of that it does not support very fast innovation cycles.

*NoTA* is a service-oriented architecture, which implements technologies and ideas presented by Web Services and other SOAs in embedded and mobile system context [13]. It was first developed for intra-device communication to enable more efficient system modularization. The fundamental idea in the original NoTA architecture is to divide the device into independent heterogeneous subsystems that communicate through abstracted interconnect. Each NoTA subsystem provides different

functionalities that are required from the system. In NoTA Release 3 the original architecture was expanded from the intra device to cover also inter device communication.

The NoTA consists of three types of logical elements: Service Nodes (SN), Application Nodes (AN) and Device Interconnect Protocol (DIP). Service nodes are services that can be used by other nodes. Application nodes are the application functionalities composed of service calls and other logic. The DIP defines both socket based communication means supporting both message and streaming type of data flows. The NoTA defines only sockets. The actual message format can be agreed with AN and SN providers [14].

The *Semantic Web* approach presents the information in a precise format. This enables the interoperability, since computer applications can automatically interpret the meaning of the information and utilize it. The information interoperability in Semantic Web is based on defining common ontologies. RDF Schema (RDFS) and Web Ontology Language (OWL) provide vocabularies and rules for describing the concepts and relationships between those concepts, i.e. ontologies in a specific domain. Rule Interchange Format (RIF) aims to specify a format for exchanging rules in the Semantic Web. Rules defined in RIF can be used for deducing new information from ontology and then combining and using the formerly deduced information efficiently [15]. The Resource Description Framework (RDF) is used to present the ontologies in form of subject, predicate and object triples, which can be presented for example with XML [16]. SPARQL query language is used for querying data presented in RDF format [17].

The Semantic Web approach tackles essentially to the same problems as Smart-.M3 that is the information interoperability. This approach is, however, aims in extending the usability of information Internet domain and approaches taken here are not very scalable towards embedded systems.

*The Common Object Request Broker Architecture* (CORBA) standard defines a model for transparent interoperability between distributed objects on a network. CORBA supports interoperability between multiple platforms and various network programming tasks. The core component of the CORBA model is the Object Request Broker (ORB), which provides the communication methods between the objects and all interaction between CORBA objects is executed through ORB entities.

The *OSGi Service Platform* provides standardized, service-oriented computing environment that assures the interoperability of applications and devices. The only requirement is that Java Virtual Machine (JVM) must be running on the device. The core part of the OSGi Service Platform specifications is a framework that defines environment for OSGi applications called bundles. From the Smart Environment interoperability perspective, the OSGi Service Platform provides a solution to the service level interoperability.

One of major European research activities in smart environments domain was the FP6 project the *Ambient Intelligence for the Networked Home Environment* (Amigo). Its aim was to develop an interoperability platform that enables interaction between heterogeneous services and devices. The focus was on the interoperability between

devices from different manufacturers of the home network domain. The result was the Amigo open source software architecture that is based on SOA paradigm and supports service delivery and consumption on demand, and dynamic service discovery. The idea in Amigo was to create a complete middleware that also embeds the intelligence of smart environments, while utilizing existing standards and paradigms in less demanding tasks.

In building environments implementing integrated functions such as switching the power off from certain appliances, cutting off water supply and activating the burglar alarm with one single 'leaving home' command has required a lot of dedicated cabling and custom devices, installed by professionals. Different communication standards have been defined in order to provide more feasible solutions, such as LON (<http://www.lonmark.org/>), KNX (<http://www.knx.org/>) and ModBus (<http://www.modbus.org/>). However, none of these has become a global standard that all manufacturers would support. They also tend to be expensive to install, maintain and upgrade. Furthermore, they do not allow easy integration between them; in fact, they may even on purpose be designed in a way that makes interoperability more difficult due to commercial reasons.

Currently the trend is going towards 'protocol converters' that make device information available through internet protocols and SOA. The 'protocol converter' can be an ordinary computer or a cheaper and more energy-efficient solution, such as the Home Control Center (<http://smarthomepartnering.com/cms/>) proposed by Nokia. Device connectivity is implemented through adapters that convert the underlying protocols into a generic internet interfaces such as browser-compatible formats (HTML and others) for user interfaces and XML messages for machine-readable information. For successful machine-to-machine communication, the semantics of the XML messages have to be understood in the same way by both parties. The currently most used method for describing message semantics is XML Schemas. In building automation, the oBIX (Open Building Information Xchange, <http://www.obix.org/>) is an example of such a protocol. Devices Profile for Web Services (DPWS) is another initiative with similar goals. In addition to these, more generic messaging protocols exist that are intended for communication with any kind of devices (not only related to building automation). The PROMISE Messaging Interface (PMI) [PROMISE, 2009] is an example of such an interface. The IP for Smart Objects (IPSO) alliance ([www.ipso-alliance.org](http://www.ipso-alliance.org)) has similar objectives but it is unclear whether they have yet specified any messaging protocols. In practice, none of these has obtained global acceptance.

## **2.2 Ecosystem needs**

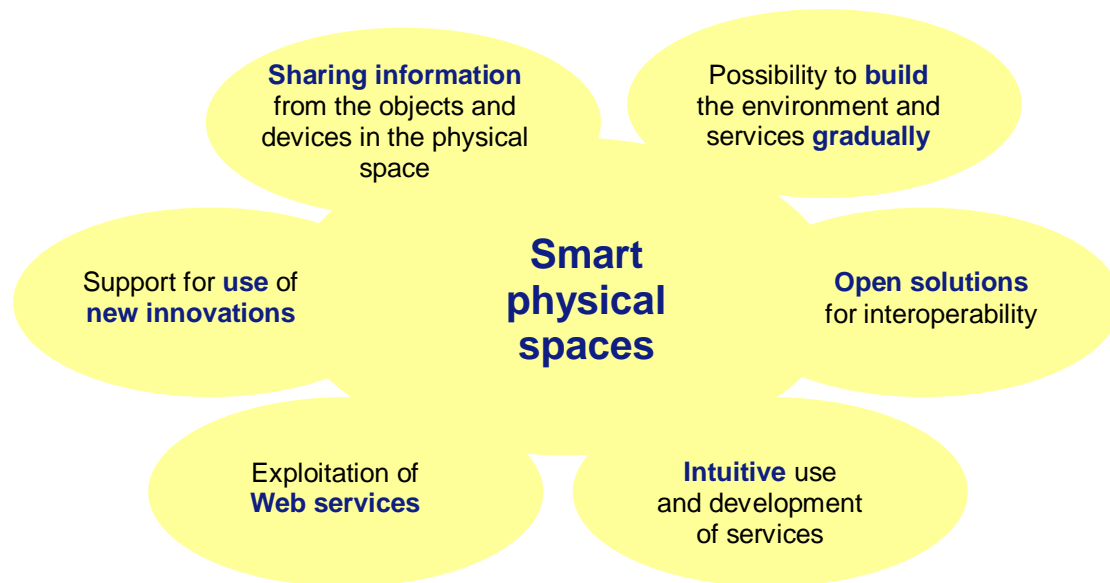
Most of the smart environment research results have not been used in real life systems. The best examples are RFID based system for supporting logistics in manufacturing and transportation, the wireless service networks for mobile devices (uPnP) and home entertainment appliances (DLNA), and smart card based solutions for payment or access/configuration management.

The typical reasons why smart spaces do not exist are the following:

1. Solutions that have been developed are domain-specific, enabling only very limited sets of applications and no possibilities to exploit the investments for other purposes.
2. Solutions are typically completely closed systems that have to be acquired and installed when physical spaces are built or constructed. The completeness also means that the whole system must be bought at the same time leading to significant initial costs.
3. Smart space applications need co-operation of several companies and it is difficult to combine the business interests or business models so that win-win conditions are met.
4. Solutions are too complex. They are developed for personal computers, Internet and servers and they do not shrink to embedded systems and mobile devices that have limited resources. It may also mean that the cost to implement the needed functionality to embedded system easily becomes too high.

If we want to overcome these showstoppers we need to understand the principles of business ecosystems and how the win-win situation can be created. The basis of smart environment is clearly an infrastructure that enables new innovations. A good analogy for this is the Internet. The whole value of Internet was only seen after 20 years of its first appearance,

The main issues are described in Figure 3. Opening the embedded information from the embedded systems is naturally a key issue. A big issue here is that it must happen voluntarily and so that the device and object manufacturers have the total control over the information to be opened. The value of smart environments comes from the use of this embedded information and not necessarily to the information provider. Therefore the motivation to information providers must be explicitly stated and the burden of opening of information must be minimised. Open solutions, simple and scalable solutions are one way achieving these objectives through low cost implementations. Also to possibility to link the open information to Internet and Web services may offer possible benefits also for information providers.



**Figure 3. Requirements for feasible smart environment solutions.**

Another key issue is how to move from current situation into smart environments. The most obvious observation is that we have an environment around us already and it contains lots of investments and features that can be exploited. No one will start from scratch and rebuild the world. Therefore we need solutions that can be integrated and extended gradually. This also includes the need for flexibility. It is not possible to figure out the applications that exploit the information or the technologies that will be used.

Finally, the smart environments must be acceptable for end users and for the owners of the physical spaces hosting these capabilities. This means that the applications must bring added value to the users and that usability of them is taken care of. In case of smart environments this may introduce completely new kind of challenges, since the application can be a multi-device application or even something that emerges from the context in which the users and the environment exist. This may require human interaction with the space instead of traditional human-machine type of interaction.

As a conclusion, we need to make information available in digital format in physical space from all objects and devices. It means the connecting of real physical world with information world. This is both a technical and a business challenge. Technically it means that we need a "TCP/IP" type of enabler for physical space information exchange. "TCP/IP" type refers to easy and commonly accepted accessibility. Business-wise we need low cost, open solutions that have platform capabilities, e.g. it has to be agnostic of use cases, information types and applications. It has to support existing and legacy solutions, and developing new, local mash-up applications of various types.

### **3 Smart-M3 interoperability platform**

Smart-M3 interoperability platform (IOP) is based on the idea of exploiting existing communication networks and service oriented architecture approaches in exposing the Semantic Web type of information sharing solution. The core of the IOP is the Smart-M3 information sharing solution being developed in Tivit's DIEM and Artemis-JU Sofia projects. The reference design of Smart-M3 has been published in open source at [www.soureforge.org](http://www.soureforge.org) in Smart-M3 project.

#### **3.1 Objectives**

The main objective of Smart-M3 IOP is to open the embedded data in the devices and objects in our current physically accessible environment to applications so that they can create better and more cost, energy, and resource optimised services to users. The Smart-M3 IOP is targeted to different physical environments and application domains and the aim is to be able to use web tools and business models in this emerging ecosystem. The vision is that Smart-M3 IOP opens the local information and together with Web services and Semantic web concepts revolutionizes the application development and application possibilities.

Smart-M3 IOP will be an open solution meaning that is freely available and modifiable. The purpose is to create a solution that is easy to integrate and adapt into various products and systems.

Smart-M3 IOP will be a scalable solution. Current reference implementation is targeted to devices that have a computational capacity of modern multimedia terminal. There is need to be able to scale the approach both upwards and downwards. Upward scaling is needed with respect to amount of information, numbers of users, and intelligence of reasoning. Downward scaling is needed for being able to open the information of low capacity (or no capacity) devices and objects that we have everywhere.

Smart-M3 IOP will be vendor independent. The aim is remove the dependencies to operating systems, computing platforms, service level solutions, and communication technologies. The intention is to provide portability and enforcement support for all. The target is an ecosystem, where Smart-M3 IOP is part of commodity and users can select their devices based on some other differentiating factors.

Smart-M3 IOP will be a multi device solution. We will aim towards seamless operation of smart environments, where smart applications exceed the boundaries of devices.

Smart-M3 IOP will be a multi domain solution. Possibility to use information across domain boundaries is one of the key objectives. Cross domain information creates completely new possibilities for application developers.

## **3.2 Principles**

The main principle of Smart-M3 IOP is that it focuses on opening and sharing of information only. It does not include any control or service requests as such, instead it is an example of publish/subscribe system. Another main issue is that the interoperability agreement is done at the information level only. Common use case specific or domain specific ontology model as a basis of information and common Smart-M3 IOP specific data format in the sharing are issues that need to agree with companies that develop products that participate in smart environment. More detailed analysis of the key principles of Smart-M3 IOP has been done in Sofia project [18]. The most important of those are briefly summarised below.

### 1. The open information principle

The purpose of the Smart-M3 IOP is to open and enable the sharing of information for devices and application. The information is represented in a uniform and use-case independent way and the information interoperability and semantics are based on common ontologies that model the information.

### 2. The simplicity principle

The Smart-M3 IOP deals with information only. The purpose of information and how it used is out of scope of Smart-M3 IOP. The information level is use-case agnostic. The architecture is kept as simple as possible.

### 3. The service principle

The information sharing is offered as a service in service-level solution (SOA). The interface to service level is defined as part of Smart-M3 IOP, but the service level solution can have multiple alternatives. The Smart-M3 IOP can offer the same information service for several different SOA at the same time.

### 4. The agnostics principle

The IOP is agnostic with respect to ontology, application programming language, service platform, communication layer, and hosting device/system.

### 5. The extensibility principle

The Smart-M3 IOP does not provide a-priori defined functionalities to manipulate the information, in addition to inserting and removing the information for sharing. Smart-M3 IOP functionality may be extended with domain ontologies and with information manipulation applications if these “applications” become usable through tools of the development environment. These extensions must not disable any normal features.

### 6. The evolvability principle

The Smart-M3 IOP should support applications that are not affected by possible changes/extensions to information sharing service.

#### 7. The notification principle

Applications may subscribe to be alerted upon an information-change event.

#### 8. The security and trust principle

When access control to information is required, it may be handled both at Service Level and at Information Level. Current Smart-M3 IOP definitions do not address these issues yet. Security, privacy and trust management may become a Smart-M3 IOP extension, according to the “extensibility principle”

#### 9. The legacy principle

Smart-M3 IOP exploits legacy (i.e. existing) service and communication level solutions. Legacy (i.e. existing) devices can access and exchange information using Smart-M3 IOP through a simple, use-case independent protocol that is part of Smart-M3 IOP.

#### 10. The scalability principle

The Smart-M3 IOP should scale with respect to the number of users, the number of devices, the resources available on each device, the amount of information, the number of information sharing service, etc.

#### 11. The quality principle

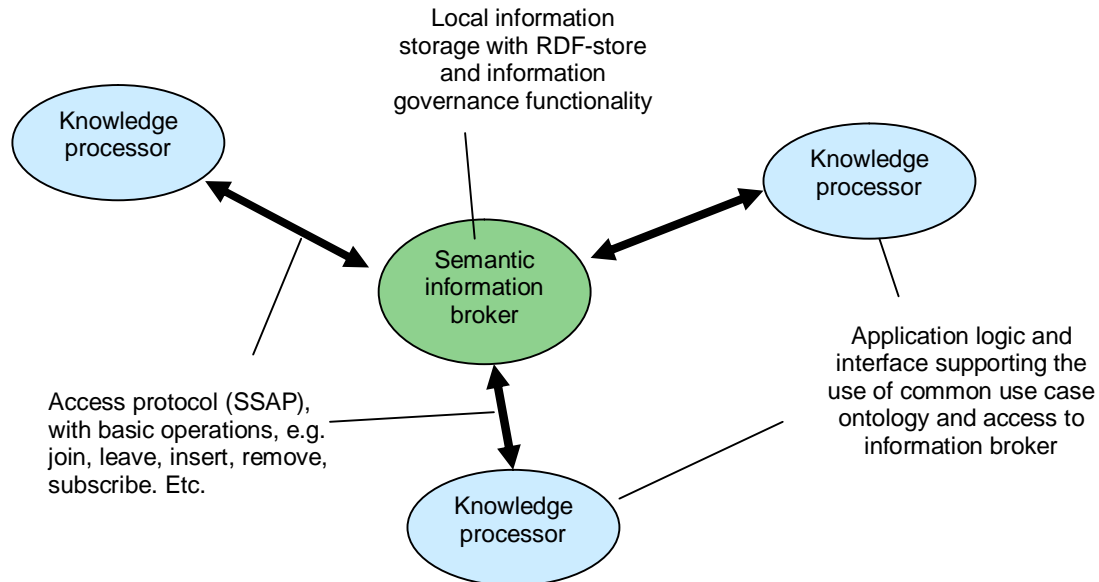
It must be possible to evaluate the quality characteristics of Smart-M3 IOP localizations and applications at development time and to measure at execution time.

### **3.3 Smart-M3 concept**

The information level view of Smart-M3 concept is presented in Figure 4. The basic information level elements are

- Semantic information broker (SIB) that is information world entity for storing, sharing and governing the information of one smart space as RDF (resource description framework) triples. RDF triplets present the information in subject-predicate-object expressions.
- Knowledge processor (KP) An information world entity that processes information and contributes to and/or consumes information content from SIB according to ontology relevant to its defined functionality. A KP can be enough for an application to make sense; however it needs one or more partner KPs for useful sharing of content, implying an agreed semantics.

- Smart space access protocol (SSAP) A protocol used by knowledge processors when accessing SIB. The SSAP defines the basic messages for joining, leaving, inserting, removing, updating, querying, and subscribing.



**Figure 4. Information level view on Smart-M3.**

One or more SIBs create a smart space (SS) that is a named search extent of information. A smart space is logical entity composed by a set of SIBs.

The Smart-M3 SIB provides cross domain search extent for information for independent knowledge processors. The information level is implemented on top of one or multiple SOA style service networks. There is no Smart-M3 specific service level architecture, thus M3 search extent has no limitation in physical distance or transport. The choice of underlying SOA implementation may pose limitations especially when we have an SOA targeted for physical spaces.

The simple operation principle of Smart-M3 is following:

1. Knowledge processors discover the SIB service (or Smart Space) using discovery and communication mechanisms offered by the device hosting the SIB.
2. KPs join the smart space using SSAP join message.
3. KPs access the SIB using SSAP insert, remove, update messages or react to changes of information they subscribed with respective message. The information is stored into the SIB.
4. Environment interacts with KPs in the way how it is design in applications that are in KPs (so, the smartness of the environment is actually programmed in the KPs)
5. The KPs leave the smart space using SSAP leave messages.

The Smart-M3 is meant for opening the information. It does not guarantee the performance and it is not meant for sending commands between devices (or KPs). That kind of interoperability should be implemented using service level capabilities.

Implementations of Smart-M3 provide a uniform, use case independent service API for sharing information in a Smart Space. The possibility to expose Smart Space service APIs concurrently through multiple domains and transport technologies makes the information currently isolated in multiple heterogeneous embedded domains available (i.e. to be monetized) by using web programming tools and methods without compromising power, safety and performance requirements of the embedded domains. As an example this would allow an application programmer who programs for a mobile platform to access contextual information in a car, home, office, football stadium etc in a uniform way and improve the user experience, without compromising real-time requirements of the embedded system.

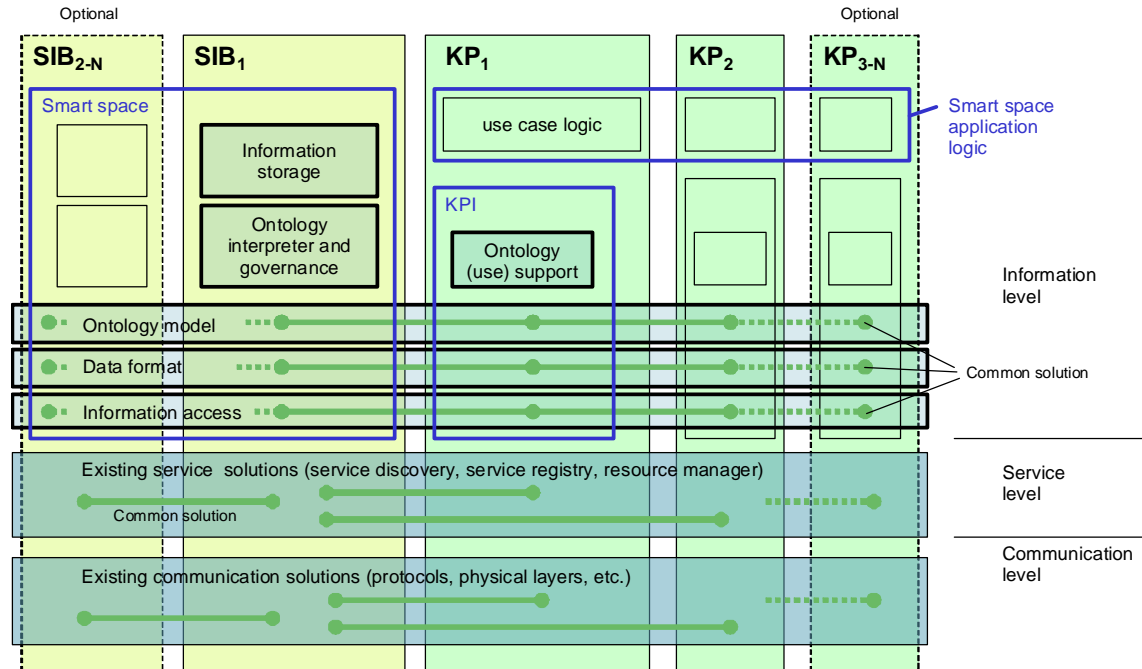
SIB stores information in RDF (Resource Description Language) format, which is a W3C standard. RDF and the use of SSAP are the only limitations for the implementation of Smart-M3 IOP. There are two reasons for using the RDF. First, it gives ability to join data from vocabularies from different business domains, without having to negotiate structural differences between them. Second, when combining the Smart Space concept with RDF, Smart-M3 not just merges the information of the embedded domains with the information in web but also makes the vast reasoning and ontology theory, practice and tools developed by the *semantic web community* available for Smart Space application developers. Smart-M3 is not about creating a new local semantic web or any other semantic web; it's about making the heterogeneous information found in embedded domains available for the semantic web tools.

The focus in the information level in Smart-M3 addresses value in application development by abolishing the need for a priori use case standardization familiar in service level solutions such as DLNA and Bluetooth. Furthermore, Smart-M3 shall abolish design time freezing of the address of the used service API, as is the case of WebServices. We propose an ontology governance process as the alternative to use case specific service API standardization. In simple cases this would mean agreeing on common ontology models. In more advanced cases the ontology governance process would agree and adopt new vocabularies using RDF and RDFS (RDF schema) defined constraints. Additionally the vocabularies could be further constrained by domain specific ontologies. Finally, Knowledge Processors (KP) can actively monitor and update the information in the RDF store based on other kinds of reasoning rules.

### **3.4 Logical architecture of Smart-M3 IOP based smart environment**

The logical architecture of smart environment (SE) based on Smart-M3 IOP is presented in Figure 5. It presents the main building blocks and concepts. The practical smart environment consists of one or more SIBs and two or more KPs. The SIBs may or may not create a single smart space (even though in the Figure 5 they do). The

participating SIBs have to share service and communication level solutions with each other and at least one solution (at service and communication level) with each KP in the smart environment. The KPs do not necessarily have to share any communication of service technologies.



**Figure 5. Logical architecture of smart environment.**

Smart space application (which is something that interacts with physical world or user) is a logical collection of a set of KPs (all KPs in smart environment do not have to participate) functionality. The requirement is that KPs either produce or exploit the information stored by one of the SIBs in the SE.

The requirement for having smart space applications is that the KPs and SIBs involved have common ontology model, data format, and information access solutions.

- Ontology model is a specified model of the information that defines the meaning of it. It can be either use case or domain specific also.
- Data format is RDF triplets (predicate, subject, object) as explained in previous chapter.
- Information access solution is SSAP protocol. SSAP is basically a simple protocol that defines the basic messages (or service operations of SIB) between KP and SIB communication. SSAP is defined in detail in [19] and in Table 1.

**Table 1. SSAP operations (from [19])**

<b>Name</b>	<b>Description</b>
Join	Begins a session between KP and SIB
Leave	Terminates the session
Insert	Inserts information into the smart space
Remove	Removes information from the smart space
Update	Combination of remove and insert operations
Query	Queries information within the smart space
Subscribe	Sets up a persistent query
Unsubscribe	Terminates a persistent query
Results indication	Updates the result set of a persistent query
Unsubscribe indication	Notifies a knowledge processor of a smart space initiated termination of its subscription
Leave indication	Notifies a knowledge processor of a smart space initiated termination of the session

The logical structure of the SIB consists of RDF storage and ontology interpreter and governance functions in addition to common parts. RDF storage is a memory (or database) for keeping the RDF triplets. The ontology interpreter and governance part can be divided into main three parts: the support for multiple SIBs, support for SSAP operations, and interfaces to service level.

The idea in supporting multiples SIBs is that each Smart-M3 smart space can be constructed by physically distributed RDF stores. This allows implementations where the personal information of a family is stored at home but it is augmented by non-personal information in, for example, Foreca weather service and there are copyright and privacy reasons not to merge the information. In order to enable these types of business models we are developing the concept of deductive closure towards distributed deductive closure. This work is currently in early phase.

The support for SSAP operations means the implementations of defined in SIB interface specification. The basic functionality is defined, but there are open issues related security, authentication, and query languages, for example. In query languages, the template query and Wilbur query are currently supported in reference implementation.

Interface to service level is needed for implementing the SOA specific functionality that is needed for communication.

More details about reference implementation of Smart-M3 are given in [20].

### **3.5 Application development**

The concept of application is different in smart environments and traditional devices. Smart environment offers a possibility to share information and services in a very flexible way as long as some key issues are supported. Some of the key issues are the following:

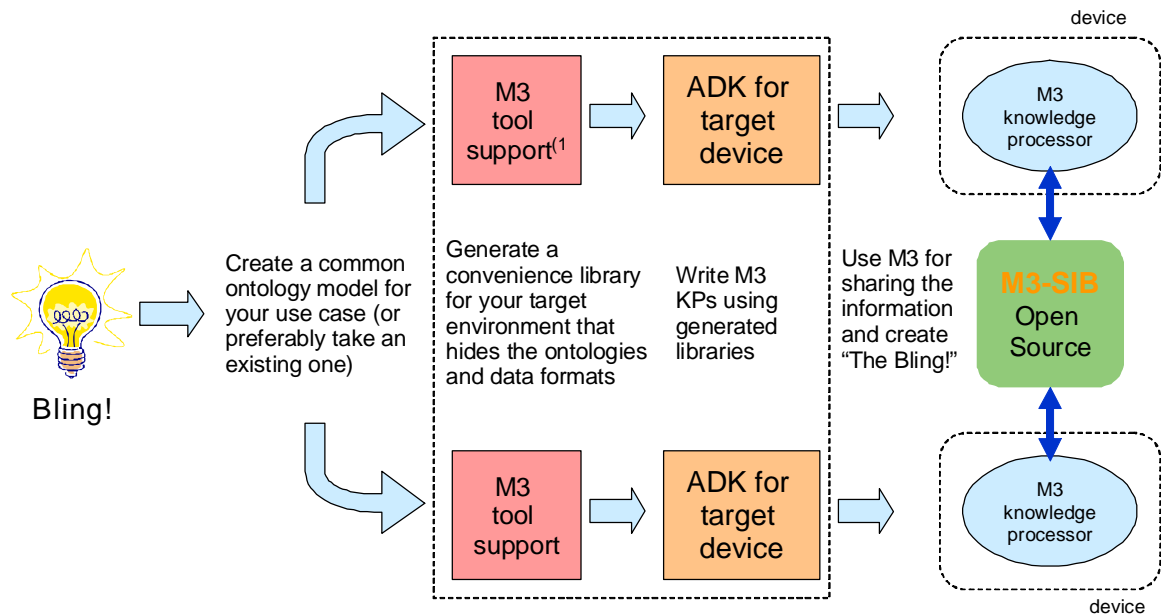
1. How to support the publishing and accessing the information in Smart-M3 SIB to/from device specific applications?
2. How to create applications in knowledge processors based ontology models?
3. How to manage the common ontology base for meaningful sharing of the information through SIB?
4. How to create information mash-ups and new multi-KP applications based on them?
5. How to encapsulate and reuse the aggregated information or supporting functionalities in Smart-M3 context?
6. How to let new user experiences emerge from dynamic environment through ad-hoc collaboration of several devices and KPs?
7. How to interface the application KPs, the devices and appliances in environment, and the users?

The smart space (or environment) applications can be divided into three basic categories. The most trivial is a single device application (KP) that exploits the shared information from Smart-M3 SIB. The adequate prerequisite for this kind of application is that its logic is based on same ontology model that the information provider has had. The second application type is a distributed application that means that the application logic (or the responsibility of user experience) is distributed to several KPs in different devices. Typical examples of this are for example distributed control or entertainment system with pre-designed logic and features. The third type is dynamically emerging user experience resulting from information or service mash-up. Key feature is that the KPs in participating devices are autonomous and designed the react to published information.

Another view to smart space application is to look what kind of KPs and support from them is available for application. In Smart-M3 the key idea has been to focus into interoperability of information and to try to keep the needed middleware as thin as possible. In principle all the KPs are therefore application specific and the middleware functionality is mostly in the operations performed by the SIB. In practice it is clear that in smart environments there are functionalities that are more or less common and that could be encapsulated as reusable KPs and integrated as a part of interoperability platform. These KPs can be divided into core KPs needed to support basic smart environment and SIB operations (authentication, security ...), common KPs used for improving the general usability of Smart-M3 (performance adaption, quality ...), and domain specific KPs used for improving the usability of smart environments for some specific purposes (information aggregation, etc.), for example. This is a very initial division and other ways of categorisation also exists.

In current Smart-M3 interoperability platform only partial solutions and ideas exist for the application development problems in general. The Smart-M3 interoperability platform now offers the basic SSAP protocol, insert/storage/query functionalities for

information, and reference implementations of KPs in various target environments (ISO-C, Java, Linux, TRON). More details about KP implementations can be found in [21] and [10].



**Figure 6. Ontology based application development flow.** <sup>1)</sup> M3 ontology compiler developed by Åbo Akademi (available at [www.sourceforge.org](http://www.sourceforge.org) Smart-M3 project).

The current application development flow is shown in Figure 6. The publishing and accessing the information in target specific application development environments is supported by target specific function libraries from defined common ontology models. The development of KPs is done using target specific tools. This approach is answering to the challenges 1 and 2 in the previous list.

In addition to the approach in Figure 6 there are needs for an application development system for smart space applications that span over several devices and KPs, and for application configuration support with dynamic creation of information mash-ups and new user experiences. This kind of application development environments should be based on governed and accepted ontology models and on ontology driven design principles. Some of these ideas are being developed in Sofia project.

### 3.6 Context-Sensitive and Adaptive User Interfaces

The DIEM project and Smart-M3 platform offer many interesting challenges for research on human-technology interaction. The work in this field focuses on user-centered design and evaluation methods for ubiquitous computing, new interaction methods to manage environments and to use different types of devices, and flexible user interface platform that allows the interfaces to be easily migrated between different devices.

The first and central part of user-centered design and evaluation process is to gather user requirements and to create scenarios. This work is done in collaboration with end-users and project partners. The results from the initial phase were scenarios that

made use of Smart-M3 platform in new kind of use cases. The scenarios also link together different application areas that are a part of the project, such as building automation and public spaces.

When designing in a multi-device environment where the user may choose between different interaction modalities and methods, the traditional interaction design and evaluation methods are often not enough to express all the necessary variables. This is why a central part of the project is to create and further develop such new design and evaluation methods that are usable in this new and ever changing context of use. The user interfaces are often based on multimodal interaction and may be changed dynamically based on the present context of use. Multimodal interaction provides the users with excellent opportunities to have more natural, efficient and expressive user interfaces in the future.

Adaptation is also an important issue in DIEM systems, as in some contexts of use some interaction methods may be recommended or function better than the others ones. These contexts may include different lighting conditions and different ways of moving, such as walking, running or driving a car. Since the solutions DIEM is developing are general and they are meant to be applied with all kinds of interoperable device environments, it is not possible to know in advance which interaction methods are available in the user's device. This is why the runtime environment must be aware of the capabilities of all devices and smart spaces that are connected to it. It should adapt to the present technical environment by providing those interaction methods that are possible to be used with the devices and smart spaces around the user.

One of the generally usable results of research on human-technology interaction is creation of new kind of user interface runtime environment. The starting point is open source software. The platform can be used to run multimodal and adaptive user interfaces in a kind of web browser environment: the user interface is loaded from the web and the software is executed in the user's device. However, in contrast to normal web browsers, the runtime offers direct linking to all resources available in the device, and it is not restricted to what is available in a normal web browser. It is possible that browser in a traditional sense is not even visible, but the runtime is using web technologies to deliver the dynamically loaded and run software and execute it. The runtime implementations are based on Javascript and Qt libraries. The programs are dynamically customizable. Some prototype platforms developed in the project are available as open source distributions [22, 23, 24].

Nokia Starlight [24] supports the use of new interaction technologies. Directly manipulated multitouch user interfaces, combined with rich visual content, are today's trend in mobile devices. The same technologies have also been recently introduced to PCs by e.g. Dell and HP with Windows 7. These new devices usually provide the full extent of new capabilities only to native applications. There has not been a uniform way a for web developer to embrace the new interactions and rich visual content.

We have been looking into bringing multitouch, haptic feedback and gestures to the web framework in order to accelerate web usage with mobile devices and laptops. We have also adapted the Apple proposed CSS transforms, transitions and animations on top of Qt 4.6. These can effectively be used for creating visually appealing effects.

## 4 Towards Smart-M3 based smart environments

The DIEM project has created a baseline for smart environment ecosystem. Smart-M3 interoperability platform is a concept and preliminary solution that already now implements the main requirements of smart environment infrastructure. It is open, simple, vendor independent, agnostic to use cases, domains, and information, scalable to various kinds of devices, and a link between real-world and Internet. It can be implemented on top of almost any service-level and communication solutions offered to physical spaces and together with them it can create an infrastructure solution that is so easy to take into use that it revolutionises our every day lives.

There are many challenges left. The current technical solution is a prototype that implements the core features needed to demonstrate the idea. There are big research questions related to security, performance, and scalability that needs to be improved. Current version is not protected by hostile behaviour of environment. It does not protect the privacy of users. When the environment, use cases, and applications get more complex, the performance of SIB, subscriptions and queries needs to be improved and the functionality needs to be enhanced. The Smart-M3 IOP is about information. The likely scenario in real-life environment is that information of opened in each appliance separately. The scalability towards very simple devices is also critical. It also creates big demands to the performance of service-level solutions, so that energy efficiency and performance criteria are also met there. Seamless integration between Smart-M3 and NoTA, for example, is essential.

Common ontology is the unifying factor in Smart-M3 smart environment and therefore also the biggest challenge for the acceptance. Fortunately it does not mean that there must be one, single common ontology. Instead the ontology can be fragmented into many domain or use case specific ontologies, which makes the problem much smaller. Anyway it means that the ontologies must be managed and their evolution and development must be controlled. This is especially essential for embedded system that can not be so easily changed once they are installed. With more complex devices it is possible to dynamically to adapt to different versions of ontologies using the same techniques as in Internet.

Smart environment applications can span over many devices and appliances. The user interaction is therefore a potential problem, since the user is interacting with the physical space and its services instead of some specific device. The problems related to how to visualise the services for the user, how to create the feeling of control, and how make the use of services intuitive, are likely the most important.

Smart-M3 information platform is also an open source project. Open source has become a viable alternative for industrial use, but it also means that new kind of thinking is needed in developing organisations. In DIEM project there is a strong intention to combine both commercial utilisation of results with open innovation of needed infrastructure, i.e. Smart-M3 information platform. It is also understood that the concept is evolving and new research is needed to make it better. The open solution, independence of companies and business, possibility to create own business solutions based on Smart-M3 are crucial for industry. At the same time new attitude towards opening the information in embedded systems is needed. Otherwise the cross-

domain mash-ups are not possible. The willingness to invest money for research and development of open source platform that can be used by the whole community is needed as well. When these needs are understood, the path to exploitation of information in completely new way is open and smart environments can become reality.

## References

- [1] Cummins S., Davy A., Finnegan J. & Carroll R. (2003) State of the Art: Middleware in Smart Space Management. M-Zones Deliverable 1: State of Art Surveys: Release 2, pp. 99–135, May.
- [2] Daoud F. & Nomura T. (2002) Preface to 'smart spaces'. In: Journal of Network and Computer Applications, volume 25, issue 4, pp. 239–242.
- [3] Singh R., Bhargava P. & Kain S. (2006) State of the art Smart Spaces: Application Models and Software Infrastructure. In: ACM Ubiquity, volume 7, issue 37, pp. 2–9, September 26 – October 2.
- [4] Essa I. A. (2000) Ubiquitous Sensing for Smart and Aware Environments. In: IEEE Personal Communications, volume 7, issue 5, pp. 47–49, October
- [5] Dalton G., McDonna A., Bowskill J., Gower A. & Smith M (1998). The Design of smartspace: A personal working environment. In: Personal and Ubiquitous Computing, volume 2, number 1, pp. 37–42, March
- [6] Weiser M. (1991) The Computer for the Twenty-First Century. Scientific American, pp. 94-100, September
- [7] Saha D. & Mukherjee A. (2003) Pervasive Computing: A Paradigm for the 21st Century. IEEE Computer, pages 25-31, March
- [8] ITU Internet Reports (2005). The Internet of Things – Executive Summary, 29 p., November
- [9] Aarts, E. (2004). Ambient Intelligence: a multimedia perspective. IEEE Multimedia, volume 11, issue 1, pp. 12-19, January – March
- [10] Kiljander, J. (2010) Reference implementation of interoperable entity for smart environments. Master's Thesis, University of Oulu, Oulu, 87 p.
- [11] Dey A. (2001). Understanding and Using context. Personal and Ubiquitous Computing, volume 5, issue 1, pp. 4-7, February
- [12] UPnP Device Architecture 1.1 (2008) Contributing members of UPnP forum, 136 p., October 15
- [13] NoTA project Home Page (20.10.2009) URL: <http://www.notaworld.org/>
- [14] Hurskainen M. (2007) Interconnect for Modular Device Architecture. Master's Thesis. Tampere University of Technology, Degree Programme in Information Technology, Tampere

- [15] RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation 10 February 2004 (21.10.2009)
- [16] W3C's Resource Description Framework Activity (16.10.2009)
- [17] SPARQL Query Language for RDF. W3C Recommendation 15 January 2008 (25.10.2009) URL: <http://www.w3.org/TR/rdf-sparql-query/>
- [18] Ovaska, E, Salmon Cinotti, T. Design Principles and Practices of Interoperable Smart Spaces. To be published in "Advanced Design Approaches for Emerging Software Systems", Xiaodong Liu and Yang Li (eds)..
- [19] SIB Interface (2009), available at [www.sourceforge.org](http://www.sourceforge.org) in smart-m3 project
- [20] Smart-M3 software architecture (2009), available at [www.sourceforge.org](http://www.sourceforge.org) in smart-m3 project
- [21] Sofia project deliverable D5.21: Interoperable Service Architecture (2010) Sofia project, 79 p.
- [22] DIEMUI JavaScript/QtScript demos in SourceForge:, available at <http://sourceforge.net/projects/diemui>
- [23] LivelyQt: JavaScript-based UI platform, available at <http://lively.cs.tut.fi/qt>
- [24] Nokia Starlight, available at <http://www.opensource.nokia.com/Starlight>