# A Conceptual Reference Model for Smart Services

*Whitepaper*

**Authors:**
**Dr. Andreas Riegg**
**Michael Klose**
**Adriana Matei**
**Lukas Reinfurt**
**Dr. Michael Ditze**
**Dr. Sebastian Hudert**

**Published in 2016 by SOA Innovation Lab e.V.**

# 1. Table of Contents

## 2. Introduction and Problem Statement

Recent developments in almost all areas of technologies have brought up a plethora of new so called “intelligent“ or “smart“ devices and environments. Ranging from smart products such as smartphones, smart watches and smart TVs to smart environments often aligned with smart cities, smart homes and smart factories, such concepts promise a more capable ecosystem of digital business models and respective services providing added values to enterprises and their solutions, to customers and their products and to developers and their development environments. They shall be capable of increasing the efficiency, robustness and safety in enterprises, they shall foster the possibility of offering additional business through new dimensions of comfort and assistance in products and they shall provide all the means to support the developer during implementation. But what smartness really means, how it can it be classified in order to derive levels of smartness and what implications smartness will have on enterprise IT architectures opens up new questions which have not been answered yet.

The purpose of this paper is to present a reference model for smart services and to provide a definition for evaluating the degree of smartness of such a service. Therefore, this work will first summarize relevant definitions of smartness and derive capabilities necessary for considering a service as being smart. This is followed by an illustration of a reference model, which classifies different capabilities, aspects and levels of smart services. The meaning of the identified dimensions in the reference model is clarified by matching the model against several use cases contained in a Smart City scenario. Finally, the technical implications of smart services on common enterprise IT systems are elaborated by identifying common technologies, frameworks and paradigms for implementing and realizing smart services.

# 3. Conceptual Foundations

In order to provide a generic definition of smartness to be used in the proposed reference model for smart services, this chapter summarizes surveyed research work from various areas. All these definitions lack significant properties of smart services or smartness in technical environments like enterprise IT. However, they provide valuable input for the reference model by setting the scope of which properties to be considered or neglected.

Given the current technological trends, smart cities are illuminated first:

> *A smart city denotes an **instrumented**, **interconnected** and **intelligent** city. “Instrumented” refers to the capability of capturing and integrating live real-world data through the use of sensors, meters, appliances, personal devices, and other similar sensors. “Interconnected” means the integration of these data into a computing platform that allows the communication of such information among the various city services. “Intelligent” refers to the inclusion of complex analytics, modelling, optimization, and visualization services to make better operational decisions. (Harrison et al., IBM Research)*

This definition stresses the point of data acquisition through sensors, communication of data and its analysis. Taking into account technologies which are already deployed in the domain of service-oriented architectures (SOAs), this focus on the data management and connectivity does not seem to be sufficient for a definition of smart services. In addition, it seems questionable that the mere existence of sensors and data communication systems can already be considered smart.

A generalized version of this view is expressed when defining smart environments:

> *A Smart Environment is one that is able to **acquire** and **apply knowledge** about the environment and its inhabitants in order to improve their experience in that environment. (G. Youngblood, 2005)*
>
> *A Smart Environment is a small world where different kinds of smart devices are continuously working to make inhabitants' lives more comfortable. (Cook and Das, Smart Environments: Technology, Protocols and Applications)*

Again, the focus here is on the existence and integration of sensors in a given environment. However, this does not entirely pay credit to the human understanding of intelligence or smartness. Environment sensing is just one (necessary) characteristic of intelligence, but clearly there are known much more aspects of human intelligence.

Smart devices, such as smartphones or smart watches are defined from a different perspective:

> *Smart Devices are electronic devices, which are **cordless**, **mobile**, **connected** and **equipped with various sensors** (such as geolocation or temperature sensors). (Translated from the original German source at Fraunhofer Institut für Materialfluss und Logistik)*
>
> *A smart device is an electronic device that is **cordless** (unless while being charged), **mobile** (easily transportable), always **connected** (via WiFi, 3G, 4G etc.) and is capable of **voice** and **video communication**, data, internet browsing, "geo-location" (for search purposes) and that can **operate** to some extent **autonomously**. (Eastern Connecticut State University, Information Technology Services)*

Such devices apparently bring in a new aspect to being smart: the mobility of the device. Another interesting aspect to mention here is the autonomy as claimed for smart devices in the second quotation. Being capable of autonomous actions is generally considered one of the most important aspects of life and intelligence in general.

After this rough idea of how new technological trends define smartness, more classical fields of science are considered to get a more general idea; especially since most of the defining aspects of smart new devices and environments until now are simply defined on the basis of some technical equipment being present. This did not grasp the concept of smartness or intelligence in its entirety.

Intelligence in the humanities is defined as follows:

*(Human) intelligence is the mental quality that consists of the abilities to **learn** from experience, **adapt** to new situations, **understand** and **handle concepts**, and use knowledge to **manipulate one‘s environment**. (Encyclopaedia Britannica)*

*Intelligence is a very general mental capability that, among other things, involves the ability to **reason**, **plan**, **solve problems**, **think abstractly**, **comprehend complex ideas**, **learn** quickly and **learn from experience**. (L. Gottfredson, Wall Street Journal)*

These definitions combined with the ability to learn from experience stress the service capability to adapt to new situations as one of the main aspects of intelligence. This results in one technical restriction that will be elaborated in more detail later in this paper: a smart system or smart service should be capable of storing its experience in the first place. This is not trivial in current service computing settings.

The science of Artificial Intelligence picked up these definitions, aiming at creating artificial devices capable of acting like humans (the most prominent test for the quality of an artificial intelligence is the famous Turing Test):

*It is the science and engineering of making intelligent machines, especially intelligent computer programs.*

*Artificial intelligence is the branch of computer science concerned with making **computers behave like humans.***

*(Both quotations from John McCarthy, Stanford/MIT)*

*Artificial Intelligence is the ability (of a computer or computer-controlled robot) to **reason**, **discover meaning**,* generalize, *or **learn from past experience**. (Encyclopaedia Britannica)*

A distinctive branch of Artificial Intelligence developed this concept even further in the form of intelligent (Software) Agents:

*An intelligent agent is an intelligent system which **perceives its environment** by sensors and which uses that information to **act upon the environment**. (Prof. Truemper, Univ. of Texas at Dallas)*

*A software agent is a long-term operating program whose function can be described as **autonomous execution of tasks or tracing of goals via interaction with his environment**. (Fensel and Toma, STI International)*

*Notion of Agency (Wooldridge and Jennings, Intelligent Agents: Theory and Practice):*

***Autonomy** (independently follows its goals)*

***Social ability** (cooperating with a human or other agents)*

***Pro-activeness** (takes initiative) and*

***Reactivity** (perceives its environment and reacts to changes).*

These definitions combine the capability of perceiving an environment, and hence the context, interpret the gathered data and subsequently act in the environment. In addition, the autonomy and pro-activeness of intelligent agents is stressed.

Another interesting aspect of intelligence deals with emergent behavior, i.e. a behavior of a system which is not a property of its individual parts but which arises from the relationships among the parts:

*Swarm intelligence is the discipline that deals with natural and artificial systems composed of many individuals that coordinate using **decentralized control** and **self-organization**. In particular, the discipline focuses on the collective (emergent) behaviors that result from the local interactions of the individuals with each other and with their environment. (M. Dorigo and M. Birattari, Université Libre de Bruxelles)*

*Emergence is the existence of a* ***coherent pattern*** *that arises out of interactions among simpler objects. (J. Odell, Agents and Emergence)*

This definition focusses on the interaction of individual entities (e.g. services or mobile devices in an enterprise setting) and the possibility of emergent patterns in the overall system. Such patterns could lead to a more efficient system behavior as compared with the original design.

A definition and reference model for smart services should incorporate several of these aspects. However, it should also be noted that most of these definitions reflect the capability of agency, learning and context-consideration. The capability of a service to describe itself is not discussed in detail. Nevertheless, according to the authors and well-known best practices in the area of SOA (service oriented architectures) this capability plays also an essential role in defining a service's smartness, as will be detailed further in the following.

## 4. Reference Model Design

The reference model for smart services derives from the definitions “smartness” and “intelligence”. This entails the identification of restrictions stemming from current service computing or the concept of a digital enterprise service in general and integrating those to the identified definitions.

The most crucial aspect of a smart service and the respective evaluation of its smartness is the **perspective** through which it is assessed. Perspectives in this case either refer to how a service acts (functional perspective) or towards whom it behaves smart (customer or “service consumer” experience).

Each of these perspectives do not provide a linear means of measuring smartness, but rather expose a set of levels a service can achieve. It is also conceivable that a given service achieves a certain level of smartness in one perspective whereas it appears non-smart in the other perspective(s). This concept hence already hints at how to deal with **context-specific smartness** which will be elaborated on later.

Based on these considerations a first draft of a frame of reference for smartness in digital services can be deduced as shown in figure 1.

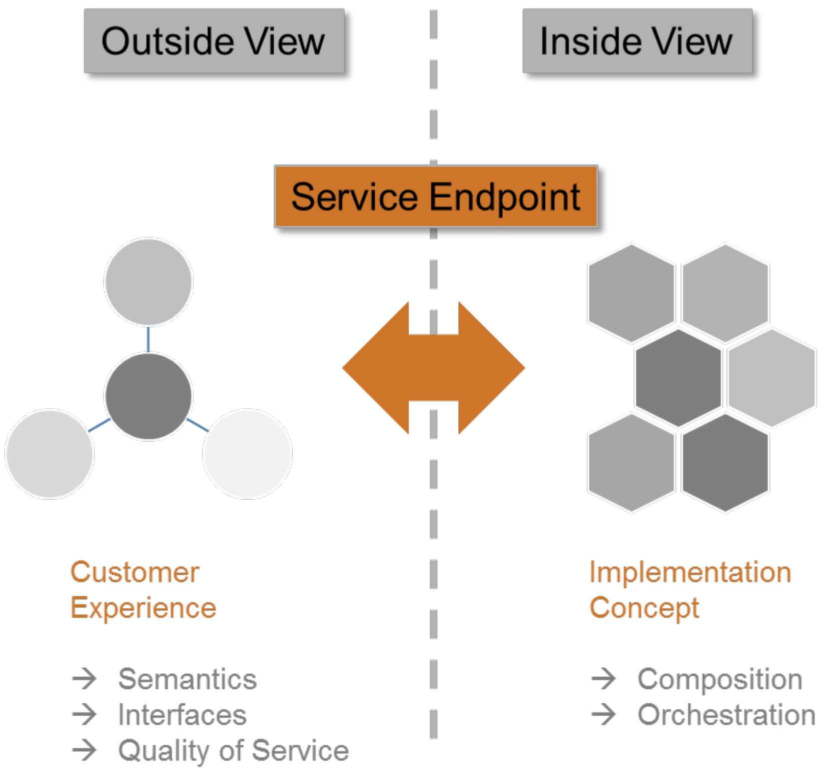

**Figure 1: Different Perspectives on a Smart Service**

The central concept in this frame of reference is the **Service Endpoint**. It denotes the access point to a given (composite) service and allows for an abstraction of internal vs. external aspects of a service. This generic view also matches very well the black-box metaphor of services computing which denotes that each service (no matter how complex internally or of how many sub-services and components it is composed) is just a service endpoint transparently encapsulating all the functionality needed to fulfil the offered interface to the outside world. It even makes at this abstraction level no assumption at all about the kind and nature of the internal implementation of the smart service.

Referring to the abovementioned discussions on internal vs. client perspective, this abstraction also allows defining the smartness of a service either as its “customer experience”, and hence its intelligent behaviour in the client or **“outside”** view as opposed to its intelligent composition or orchestration of internal modules, and hence its **“inside”** view.

Especially from the outside view, the assessment finally depends on the subjective perception of the service client. Such an assessment is only possible when a human triggers the service execution; in a fully automated environment, the notion of observable (outside view) intelligence is not easily applicable. However, even when a human user is the initial starting point, its assessment is highly subjective.

On one hand, a service can be considered smart by a user for simply answering the request correctly. On the other hand, a highly flexible and adaptable service with semantic technology etc. could be deemed not smart because the user simply specified a bad request with the expectation of a correct result, even though the service fulfilled the request perfectly. Another possible scenario could correspond to a service optimizing the overall system instead of individual requests (for example, a routing service which targets optimizing the overall traffic flow in a complete geographical area, but partially proposes suboptimal routes for an individual driver to support the “higher goal” of traffic system optimization). Such a service would also be perceived as not smart although it acts highly smart on a system level. Therefore, the assessment of service smartness requires considering different perspectives, and along with the perspectives, several capabilities a service should offer with respect to its customer experience and its implementation concept.

One of these capabilities is the learning skill of a service, finally resulting in **self-adaption** over time. Especially in the context of human intelligence, the capability of learning from experience is considered crucial. Transferred to digital services this aspect, however, poses several conceptual challenges. First, service learning from its experience will contradict one of the most crucial axioms of SOAs and services computing: statelessness. A stateless service will never be able to gain experience, and finally exploit it for adaptation. Ultimately, this also implies a feedback mechanism allowing a service from adapting, and subsequently evaluating this adaption by means of a user feedback. This directly leads to the assumption that a service can only appear smart over a given amount of invocations (“learning curve”). It will never be considered (outside view) smart based on just one individual invocation. In addition, the process of automated intelligent forgetting gets more and more crucial in the context of digital services due to privacy considerations.

Another aspect of the smartness of a service is its capability to work within a given **context**, either by observing the context itself or by processing submitted context information (as currently done within SOA orchestration engines dealing with stateless services). Processing context information hereby denotes a situation where the context, which is provided to a service, and the event, which triggers the service implicitly, define the state of the service. This state, however, is transient and therefore not remembered by the service.

One final characteristic of a service or service composition smartness is its **semantic** capability (regarding self-description). Especially when operating under an open world assumption as is the case in the Internet of Things, a service must be capable of describing its formal semantics. This need becomes apparent latest when thinking about complex orchestrations for creating composite services.

All these aspects culminate in the assumptions that a smart service must be measured from different perspectives (inside vs. outside view) and can significantly contradict some of the SOA axioms such as statelessness or self-containedness (when assuming the services interact to learn / adapt, as commonly applied in the software agent community). In addition, the semantic aspect plays an important role, especially when assessing a service’s smartness from the outside view.

Based on these considerations, a preliminary reference model of a smart service can be deduced, incorporating all of the abovementioned aspects as smartness aspects. Figure 2 shows this model.

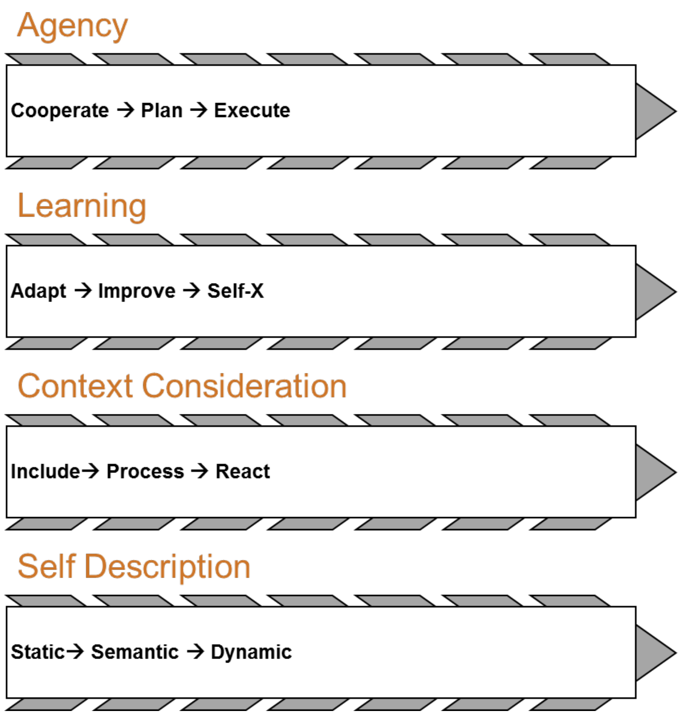

**Figure 2: Initial Reference Model for Smart Services**

The reference model defines a set of four capabilities on a service's smartness:

1. **Agency**: Following the software agent paradigm, this capability describes the interactive and pro-active behavior of a service in order to autonomously reach its goals.
2. **Learning**: This capability reflects the service ability of changing its behavior over time to adapt to new invocation contexts based on information it is supplied with.
3. **Context Consideration**: This capability deals with the ability of a service to include, process or even react to a given context.
4. **Self-Description**: This capability deals with the service ability to describe its own behavior and make it available e.g. to higher-order orchestrations.

The notion of service "capabilities" was chosen to provide an alignment with other capabilities that are considered in the area of capability models for digital services. The level of importance of each of these capabilities corresponds to the order by which they are listed, i.e. the capability of "Agency" is considered to have a higher impact on the perceived smartness of a service than the capability "Learning". For each of these capabilities individual levels can be identified, depicting different levels of smartness of a service:

**Agency**:

1. **Cooperate**: This basic level denotes services being able to cooperate with other systems/services in order to achieve their goal.
2. **Plan**: A service with such capabilities is able to deduct a plan from given information on internal steps to achieve its goal within a given activity room, i.e. a set of known services which it is allowed and designed to interact with.
3. **Execute**: The highest level describes services, which are able to further explore and extend their activity room so as to allow them to cooperate with services which are not known beforehand, but which are supposed to assist them in achieving their goals.

**Learning**:

1. **Adapt**: This basic level describes services being able to adapt their behavior to a given context.
2. **Improve**: The second step depicts services being able to learn and consequently make proposals to improve its behavior over time, either from experience or from earlier invocations. Usually, the user or the system has the possibility to intervene.
3. **Self-X**: The final level describes services being able to expose self-x capabilities such as e.g. self-optimization or self-healing for handling complex situation without any user or system intervention.

**Context Consideration**:

1. **Include**: A service on this level is able to pick-up and include context information as received from other services or applications. Only parts of the contextual information, however, are considered during processing.
2. **Process**: This level describes services, which are able to fully use the context as defined in their current mode of operation. Contextual information is only considered once.
3. **React**: The final level denotes services, which are not only able process context information continuously, but also change the context itself in a given environment.

**Self-Description**:

1. **Static**: A service on this basic level is able to provide a static description about itself and its capabilities.
2. **Semantic:** The second level describes services, which are able to provide their semantic description.
3. **Dynamic**: In this final stage services are not only able to semantically describe themselves, but also to dynamically update this information, e.g. to reflect new or adapted technologies (such as a result of a learning process).

The separation between these capabilities, however, is not considered as selective since some of these skills have mutual dependencies. For example, the final level “React” of the capability “Context Consideration” usually requires a certain level of “Agency” in order to change the context of an environment. Similarly, the level “Plan” under the capability “Agency” will usually rely on a certain level of “Learning” in order to propose a reasoned approach.

Furthermore, three additional levels that summarize the degree of smartness for all capabilities are proposed to simplify the individual alignment: These levels should reflect different levels of smartness and are denoted as **“Handled”**, **“Reasoned”** and **“Autonomous”**.

In addition, two different major aspects of smartness (as orthogonal description capabilities to the ones just listed) were identified: **Functional** and **Non-functional**. These major aspects are complemented by a set of characteristic non-functional aspects such as Security, Development and Legal, which are also outlined further down in this paper.

The service model provides aspects (in form of four different capabilities) on a service that are applicable in different dimensions. It will thus be possible in principal for a service to exhibit certain agency aspects within its functional aspect, but none within its non-functional aspect.

Figure 3 shows the resulting three dimensional service model.

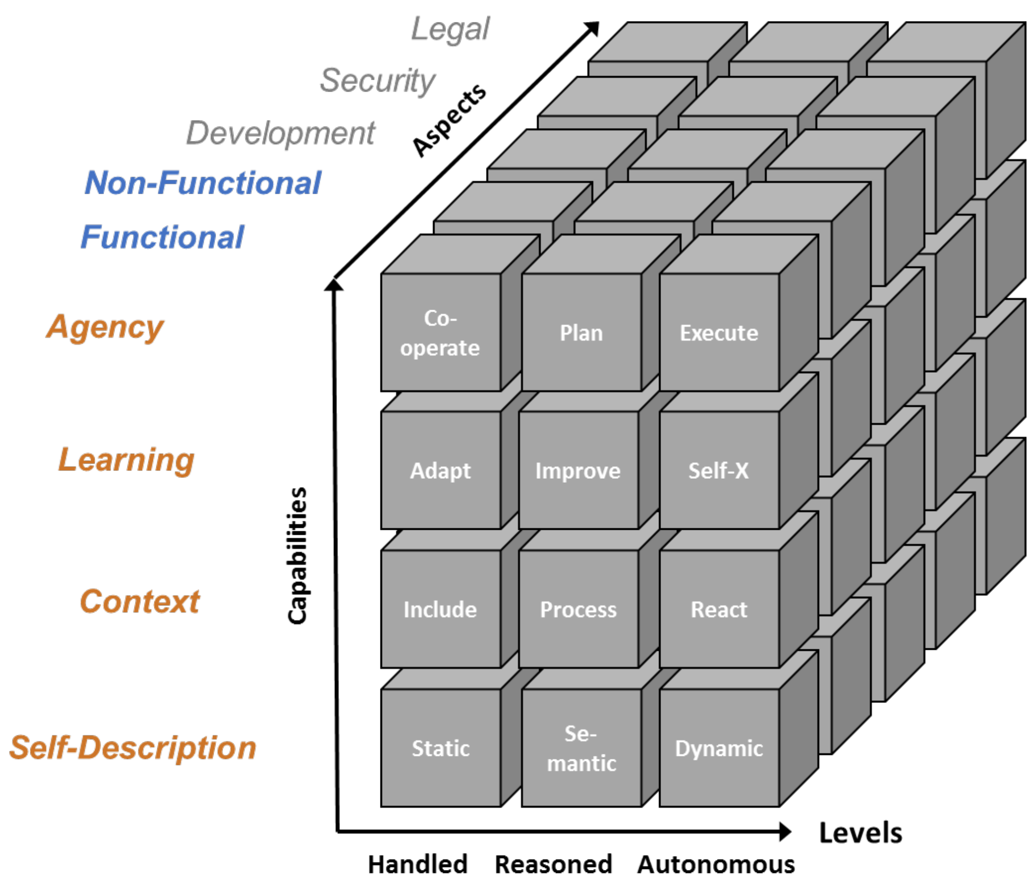

**Figure 3: 3D-Service Model**

Following the definition of the various capabilities, aspects and levels, now finally the following definition of a smart service can be derived from the reference model above:

**Definition Smart Service:**

A smart service is a digital connected service exposing at least one of the levels of smartness within the identified capabilities (Adaptation, Context, Agency or Self-Description) in at least one of the identified primary aspects (Functional, Non-functional). The degree of smartness derives from the consolidated value of the individual levels.

# 5. Aligning the Reference Model with a Smart City Scenario

This chapter will sketch a scenario in the area of smart city navigation and match individual use cases in this scenario against the reference model as defined in the last chapter. Whereas particular attention will be paid to the functional and non-functional aspects of the reference mode, the implications of security, development and legal will be also roughly sketched.

## 5.1 Scenario Description

The target scenario centers on a group of three people with the intention to navigate to a Point of Interest in a smart city starting at the respective city border. It is very important to note that in our scenario, the described kind of smart navigation service is provided (form the customer perspective) completely by the smart city itself and **NOT** by any of the individual transportation mode providers within the city that deliver concrete transportation steps along the route.

Each of the travelers does not live in the smart city (being thus a casual visitor) and has set up a user profile, which denotes personal preferences and handicaps for travelling. Furthermore, the group of travelers has roughly agreed upon a time frame by which the point of interest shall be reached and a budget available for travelling. Both, time frame and budget have a limited degree of freedom. Hence, budget, time frame and preferences can be considered as a target function against which to optimize. The dynamic navigation solution should give a list of routes from a certain location A at the city border to the point of interest at location B as an answer to the routing request, which fulfills or even optimizes the target function. The navigation solution is adaptable, i.e. it adapts the proposed route with respect to the current context the users find themselves in. An additional constraint for the route contained in a travel plan is that the means of transportation must be changed during travel. The overall navigation solution should be partitioned into four different phases, namely (1) **Planning**, (2) **Execution**, (3) **Billing** and (4) **Feedback**.

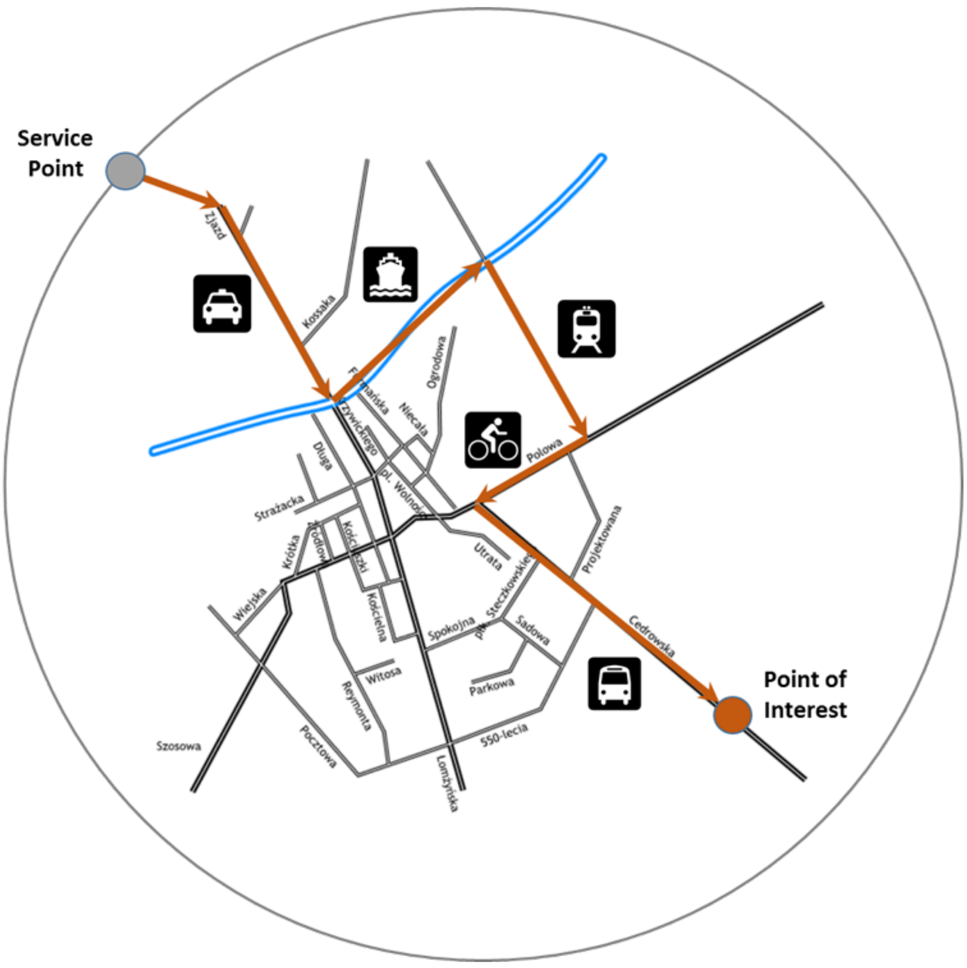

**Figure 4: Smart City Use Case**

## Planning

Once a travel plan is supposed by the navigation solution, the following actions can be triggered: 1) Selection of travel plan, 2) Booking/Reservation of tickets and 3) Ticketing. Selection hereby refers to the process of picking one of the suggested solutions, which fulfil the target functions. Booking/Reservation means pre-booking all the necessary tickets along the route. Ticketing refers to the process of making the tickets available to the user.

The navigation solution acts dynamically. In case of a non-working route, e.g. because a travel connection has been missed by the travelers or is temporarily not available or does not fulfil the target function, the system should inform the user and it should make suggestions for alternative connections which still fulfill the target function.

## Execution

As transportation modes during travel execution, the following options were identified with additional use cases and constraints, respectively:

- **Pedestrians** have a specific pedestrian navigation with additional information about e.g. how to get to the next platform in a station or how to navigate to the next transport mode.
- **Cars** should continuously display the route to the driver. The navigation also has the option to download the necessary route and card material for navigation. Furthermore, there is an option to reserve the car in case a car-sharing model is used.

- **Bike-to-Go** solutions contain the option to reserve a bike in advance.
- **Buses, Subways, Suburban Trains and Trams** offer the option to display the next station for change of transport.
- **Taxis** display the route to the user, thus allowing monitoring of the driven route.
- **Lift** solutions display selected routes and consider the reputation of the 3rd party offers.

Among the contexts to be considered by the navigation solution are current weather information, news from various sources and the state of charge of the accumulator / state of fuel tank in the device (with the idea that the final destination should be reached before the accumulator / fuel tank runs empty).

**Billing**

During the billing phase, the user gets an individual invoice, which includes the service fee and the transportation and ticket costs. This phase should work on a pay-per-use strategy and employ various payment models, such as digital payments (via for example PayPal) or actual bills to the user to be paid later on.

**Feedback**

Complimentary to the planning, execution and billing phase, the feedback phase gives the user the possibility to evaluate the experience of the received smart navigation service. The user has the possibility to give feedback to the system about the planned and executed route. Furthermore, feedback about the travel experience (e.g. cleanliness of the used transport services or the city areas he came across) helps to improve the environment of the city. Last, an evaluation of the service providers is supposed to have an impact on the reputation of the provider.

## 5.2 Aligning the Scenario with Functional Smartness Aspects

### 5.2.1 Agency

- **Cooperate:** The smart service cooperates with other services, e.g. it calls the service for suburban trains in order to reserve a seat for the user or the group of users on the train or in order to get updated travelling or context information, which it could use.
- **Plan:** The smart service plans independently within its own activity room, e.g. it chooses the next means of transportation through cooperation with other services such as a calendar service. The calendar service indicates the latest permissible arrival time of the group as a consequence of an appointment, which cannot be delayed. Therefore, points of interest which are in the vicinity and which have been declared as preferences for a visit by the group are proposed to be neglected if the latest arrival time obtained from the calendar cannot be met. Whatever the service plans it still requires the acknowledgement of the user.
- **Execute:** The smart service may act and extend its activity room autonomously. As an example, it has detected a new private lift service, which exhibits the necessary level of reputation as given in the user or group preferences and chooses, reserves and books it without any further intervention from the user or group.

### 5.2.2 Learning

- **Adapt:** The smart service learns user or group preferences once and subsequently uses them. For example, the user or group preferences may indicate to call a taxi to reach location B instead of proposing to use the tram if the latter means of transportation does not operate on time and the user or group has indicated that they do not prefer riding on trams.
- **Improve:** The smart service optimizes its operation based on learning approaches in the long run and offers respective options, e.g. it detects through data mining approaches that the user or groups generally prefer taking the bus during winter while they prefer taking the bike-to-go or even walk as pedestrians in summer. The smart service proposes to use these means of transport for the ride to location B. If, however, the state of charge of the smart device hosting the service may indicate that the battery is about to run empty, it may consider to offer a quicker

option for navigation since the smart device may hold tickets for travelling onwards which cannot be used if the smart device has run empty. Afterwards, the smart service may ask the user or the group of users for feedback about the proposal.
- **Self-X:** The smart service optimizes itself without user intervention and can handle complex situations with the provided information, e.g. it makes predictions about potential degrees of capacity utilisation for several means of transport and exploits this perception during planning. As an example, the service predicts that there will be congestion on the route to location B because of an accident, and therefore decides to book tickets for the subway during execution instead of selecting the shared e-car. The reservation of the shared e-car is in this case cancelled without any further user intervention.

#### 5.2.3 Context Consideration

- **Include:** The service receives simple preferences, but can also consider some parts of contextual information such as weather conditions, congestion notifications, etc. even if the service just processes the preferences. For example, besides preference information the service may receive contextual information such as humidity for its geo-location, but is not able to process it any further and to conclude on the chance of rain within the city border. The service continues to act according to the user or group preferences.
- **Process:** The service processes preferences along with all contextual information such as weather conditions, congestion notifications, etc., e.g. for selecting the best and potentially driest route for the users or the group of users. Contextual information is considered only once, but not continuously. As an example, if the user or group of users has indicated certain points of interest they would like to visit if they are in the vicinity, the service may propose to make a short stop-over if it can match its own location with the location of the points of interests and calculate the overhead for the visit.
- **React:** The service continuously reacts to changing preferences and contextual information such as weather conditions, congestion notifications, etc. and adjusts the navigation dynamically. As an example, the service receives a new congestion notification that will delay the journey with the shared electric car and therefore considers public transport such as subways or suburban trains as an alternative. Since the service continuously monitors any contextual information it received during execution, it may decide to change the travel plan for the user or group of users several times.

#### 5.2.4 Self-Description

- **Static:** The smart service offers a fixed description of its capabilities, e.g. it describes its support for subways, suburban trains and taxis as means of transportation.
- **Semantic:** The description of the smart service employs machine-readable semantics and other services are capable of understanding the description, e.g. other services derive the difference between classical taxis and private lift services or they understand the difference between urban and suburban trains. If available, existing standard ontologies are used for the description.
- **Dynamic:** The smart service adapts its description autonomously with updated information, e.g. in case a private lift service is no longer supported by the service since it does no longer operate or does not exhibit the necessary reputation level.

## 5.3 Aligning the Scenario with Non-Functional Aspects

##### 5.3.1.1 Agency

- **Cooperate:** The smart service cooperates with other services, e.g. it agrees with other services on the usage of a certain security protocol and/or minimum security level.
- **Plan:** The smart service may plan within its own activity room, e.g. it plans changing enquired services due to unsatisfying response times and asks for acknowledgements.

- **Execute:** The smart service may act inside and extend its activity room autonomously, e.g. it decides to look for alternative services with better response times and uses them without any further manual intervention.

### 5.3.2 Learning

- **Adapt:** The smart service can adjust its behaviour based on fixed rules, e.g. it may choose to deploy a better encryption protocol.
- **Improve:** The smart service optimizes its own cost function (e.g. a trade-off between deployed security mechanism and response time) based on rules and learning approaches. For example, the smart service may learn the response times of other services and select them to optimize its cost function.
- **Self-X:** The smart service optimizes itself without any interventions based on predictions. It can also handle unknown situations und define new rules, e.g. it reserves computation resources early based on predicted resource usage, and thus avoids resource overloads.

### 5.3.3 Context Consideration

- **Include:** The service can obtain various formats for context description, but is only capable of processing a part of them.
- **Process:** The service processes all obtained formats for context descriptions such as means of interaction, connectivity, service workload once, e.g. interaction and response time shall be optimized.
- **React:** The service reacts continuously to changing means of interaction, connectivity and service utilization, e.g. interactions and response times shall be optimized dynamically.

### 5.3.4 Self-Description

- **Static:** The smart service delivers a fixed description of its non-functional properties, e.g. in describes that it can a deliver an updated route for the navigation in less than 30 seconds.
- **Semantic:** The description of the non-functional properties is semantic and other services are capable of understanding them, e.g. they understand the difference between minimal, average and maximum response time. If available, existing standard ontologies are used for the description.
- **Dynamic:** The smart service adapts its description of non-functional properties with updated information, e.g. its prevailing average response time.

### 5.3.5 Security, Development and Legal as characteristic non-functional Aspects

Non-functional aspects primarily cover all aspects related to non-functional characteristics of a smart service or its surrounding activity room such as e.g. security or performance (commonly expressed as QoS – parameters). Three of the most prominent areas of non-functional aspects are shortly detailed in the following.
The **Development** aspects distinguish from the other characteristics as they do address the inside view of a service. They deal with processes and development steps that need to be considered during the development of a smart service such as the testing of smartness in addition to traditional module and integration tests. From a different perspective, also the development process and the respective technical infrastructure can be viewed as smart and assessed accordingly.

The **Legal** aspects mainly deal with requirements regarding all legal and compliance aspects of a service executed in an enterprise application. The aspects are complemented by certification or licensing issues which may come up during the development or execution of the smart services. From a holistic view, a multitude of explicit and implicit regulatory norms such as laws, certifications, best practices or contracts influence the legal aspects, a statement about which is commonly found as non-functional guarantee in service level agreements.

Similarly, **Security** aspects are concerned with the meeting of security standards by employing certain protocols (such as https etc.), certification standards or encryption algorithms. Security aspects are mostly seen as mandatory requirements for a service to be used or integrated with other services. Apart from such technical security measures, also strategical security considerations can become relevant, especially in massively distributed and open systems. Such aspects deal with strategic misbehavior of individual services to exploit the overall system, while complying with all security protocols, certificates etc. A common countermeasure for such behavior are electronic reputation systems, helping the collective of services to identify misbehaving services and excluding them from further interactions

# 6. Refinement of Smart Service Capabilities and Aspects and Alignment of Enterprise IT

## 6.1 Agency

The technical basis for all technology and frameworks implementing agency in current enterprise IT settings was created by the early research on distributed artificial intelligence. Under the acronym of Intelligent Software Agents a plethora of methods and algorithms was designed with the goal of achieving truly autonomous software modules. The main categories of such technologies are the internal artificial intelligence, allowing an agent to (re-)act within its environment, its capabilities to sense its context and the underlying communication infrastructure enabling the agents to interact.

The internal cognitive capabilities range from simple reflex-oriented, goal- or utility-oriented, learning or even BDI (Belief- Desire-Intention) agents, interpreting their surroundings by comparing their internal image of their context with their goals and chose activities (called "plans") accordingly to achieve their goals.

Especially in the area of sensors and available context information recent technological advances in the area of cyber-physical systems, embedded sensors and ubiquitous smart devices have drastically increased the possibilities to acquire information about an agent's context.

One of the key characteristics of the Software Agent paradigm is its ability to achieve effective and efficient results by interacting with each other. For this, powerful communication mechanisms are employed, such as peer-to-peer protocols (e.g. Chord rings, allowing a robust communication even in systems with high fluctuation of participating agents). Especially in the wireless domain many protocols and technologies (new Bluetooth versions, LTE, WLAN, tethering etc.) have been invented in the past years enabling a near-failsafe communication network even when operating in an ad-hoc mode.

Software Agent research and related technologies have more or less ceased to exist under that name and have diverted into many other technology areas that are more relevant than ever, such as autonomous cars, cyber-physical systems and industrial internet / Industrie 4.0 not least because of the ever increasing amount of mobile smart devices that are ubiquitously connected and "always on".

## 6.2 Learning

The most prominent area of enterprise IT in which (machine) learning methods are currently employed is the more than ever growing area of data analytics. Following the huge impact of current megatrends in business information systems, such as Big Data, Smart Data and Internet of Things, a huge amount of frameworks and software systems emerged employing learning techniques for better data analytics results.

The majority of systems used in data analytics focus either on a) the analytics itself (such as finding patterns in a set of data or retrieving a given dataset sought for) or b) the data management allowing to process the huge amounts of data present in Big Data settings such as Apache Hadoop or Spark.

The most prominent analytics methodologies can be clustered into supervised and unsupervised learning techniques. Examples of the former are the many clustering approaches or Google's Page Rank algorithm; an example of the latter are the often-used Support Vector Machines.

Another set of learning technologies often employed in the Software Agent domain are evolutionary or genetic algorithms that mimic the real-world evolution by supplying agents with a "genome" of parameters defining the agents' behaviour that can be a) adapted by the agents or b) "mutate" randomly over time. By comparing the achieved results of a given behaviour (thus parameter set) to a goal function an agent adapts over time in order to achieve an optimal parameter set, thus "evolutionary path".

## 6.3 Context Consideration

Unfortunately[1] there has not been too much progress in context-based computing (especially illustrated by the context of context-based (web)-search) in the recent years. The same author identifies a quite extensive set of possible context information relevant for discovery engines, although this list even holds for general smart IT environments: personal history, demographic profile, interests, location, device used for accessing the system, date and time, weather conditions or the mood the user is in.

Mainly in the aspects of location and device there seems to be significant progress in the recent years, as can be seen in the area of location-based services, incorporating the users geographic location in their behaviour or the distinction between mobile and stationary devices while internet browsing. In the other areas, established standards, methods of representation or processing of context and even the semantic concepts to be considered are lacking, although potentially offering a significant potential for enterprise IT. This can for example be seen by current efforts for defining personal profiles and their usage within personalized services in autonomous cars.

## 6.4 Self Description

There has been a significant amount of research and industry adoption of various service description approaches. The most prominent originating in the Web Service domain is the Web Service Description Language (WSDL), allowing for the description of individual service instances and their interfaces. Together with its respective directory standard (Universal Description, Discovery and Integration, UDDI) it forms the basis of the WS* technology stack regarding service description and discovery.

In recent years the “microservice” software engineering paradigm has more and more displaced WS* technology in favor of more fine granular services mostly implemented as RESTful services, thus exposing their interfaces via the ubiquitous http protocol. New interface / service description technologies for RESTful APIs emerged, such as RSDL (RESTful Service Description Language) or Swagger.

A significant contribution to the self-description of non-functional parameters of a service was made by various works on electronic service level agreements, resulting in standards or de-facto standards such as Web Service Level Agreement (WSLA) by IBM or WS-Agreement, currently standardized by the Open Grid Forum. Such technologies provide document templates, protocols and process descriptions for specifying and negotiating machine-readable Quality-of-Service aspects. Such mechanisms are broadly used in infrastructure management and comply roughly with the ITIL Service Level Objectives (SLO), as they define technical, non-legal service parameters within the ITIL framework.

Additionally the area of Semantic Web provides several frameworks for incorporating semantic service descriptions such as the Web Ontology Language OWL or the Resource Description Framework RDF. Given such semantic service descriptions are in place and backed up with respective (industry-specific) ontologies smart services will be able to understand, and therefore reason about other services to be invoked, thus achieving a more flexible orchestration of services and more efficient result as .

[1] http://rossdawsonblog.com/weblog/archives/2011/01/the-8-kinds-of-context-that-will-define-contextual-search.html

## 7. Summary and Conclusion

The goal of this paper was to deduce a reference model for smart IT services, allowing for the evaluation of the degree of smartness of such a service. After reviewing relevant definitions of smartness from various research disciplines, we derived the various capabilities necessary for considering a service as being smart. Based on these considerations we were able to design a multidimensional reference model for the evaluation of smartness in IT services. The presented model is based on different capabilities, aspects and levels of smart services, able to cope with the complex nature of smartness as perceived by service users. A detailed description of a Smart City scenario, analyzed through the perspective of the designed reference model, was given to illustrate our approach.

With the designed reference model a comparable evaluation of smartness in IT services can be done, thus allowing users and potential service customers to measure smartness and therefore compare smart IT services in a much more comprehensive way. The model also addresses different aspects relevant for real-world service development and deployment, such as legal or security, thus allowing for a detailed service management throughout the overall service lifecycle.