A Test-driven Approach for Life Cycle Management of Internet of Things enabled Services

Ralf TÖNJES¹, Eike Steffen REETZ¹, Klaus MOESSNER², Payam BARNAGHI²

¹University of Applied Sciences Osnabrück, P.O. Box 1940, 49009 Osnabrück, Germany
Tel: +49 (541) 969 3453, Fax: +49 541 969 13453
Email: {r.toenjes, e.reetz}@hs-osnabrueck.de

²University of Surrey, Guildford, Surrey, GU2 7XH, UK
Tel: +44 (1483) 683468, Fax: +44 (1483) 686011
Email: {k.moessner, p.barnaghi}@surrey.ac.uk

Abstract:
To date implementations of Internet of Things (IoT) architectures are confined to particular application areas and tailored to meet only the limited requirements of their narrow applications. To overcome technology and sector boundaries this paper proposes a dynamic service creation environment that employs i) orchestration of business services based on re-usable IoT service components, ii) self-management capable components for automated configuration and testing of services for things, and iii) abstraction of the heterogeneity of underlying technologies to ensure interoperability. To ensure reliability and robustness the presented approach integrates self-testing and self-adaptation in all service life cycle phases. The service life cycle management distinguishes the IoT service creation phase (design-time) and the IoT service provision phase (run-time). For test-friendly service creation (1) semantic service descriptions are employed to derive semi-automatically services and related tests, (2) and testing is systematically integrated into a Service Creation Environment. For reliable and robust service provisioning the presented system (3) forces validation tests in a sandbox environment before deployment and (4) enables run-time monitoring for service adaptation. The system under test is modelled by finite state machines (FSM) that are semi-automatically composed of re-usable test components. Then path searching algorithms are applied to derive automatically tests from the FSM model. The resulting tests are specified in the test control notation TTCN-3 and compiled to run the validation tests.

Keywords: Internet of Things enabled Services, Service Life Cycle Management, Service Creation Environment, Test Automation

1. Introduction
To date implementations of Internet of Things (IoT) architectures are confined to particular application areas and tailored to meet only the limited requirements of their specific applications. These silo solutions used in these individual sectors hinder the uptake and penetration of specific tailored services for Internet of Things applications, in particular for innovative business processes. Hence the provisioning of IoT enabled business services (in short IoT services) is a time- and cost extensive process. As a result, a lot of opportunities remain often unused in today’s Internet-connected objects platforms. To overcome technology & sector boundaries and therefore dynamically design and integrate new types of services and generate new business opportunities requires a dynamic service creation environment that gathers and exploits data and information from sensors and actuators that use different communication technologies/formats.
There are several sophisticated Service Creation Environments (SCE) which support the application developers in rapid service creation and deployment in different platforms. However, these SCE have not been developed for the IoT with a large number of resources and the need of automated interpretation of environmental and context information. To accelerate the introduction of new services an effective dynamic service creation environment architecture needs to bridge the gap between various business services and the heterogeneity of networked sensors, actuators and objects. It has to provide: 1) orchestration, i.e. composition, of business services based on re-usable IoT service components, 2) self-management capable components for automated configuration and testing of services for things, and 3) abstraction of the heterogeneity of underlying technologies to ensure interoperability. Moreover, the dynamics of the IoT environment make the development and maintenance of services an error prone challenge. Therefore the described approach integrate self-testing and adaptation capabilities in the service life cycle management from the beginning. The presented concepts are part of the IoT.est\(^1\) project, which aims at developing an IoT service creation environment, bridging the gap between various business services and the heterogeneity of networked sensors, actuators and objects. The approach distinguishes four service life cycle phases belonging either to design- or run-time. Each phase of the service life cycle is supported by the corresponding test and monitoring phase as described below and shown in Figure 1(a).

IoT service creation (design-time):

- **Modelling Phase**: Knowledge based methods derive semi-automatically services and related tests from semantic service descriptions based on standard service interfaces and re-usable service and test components.
- **Composition Phase**: A test-aware IoT Service Creation Environment supports incremental service evolution by regression tests. When adding new functionalities, the service components and system tests are included to ensure backward compatibility with previous service releases.

\(^1\)FP7 EU ICT IoT.est project, http://ict-iotest.eu
IoT service provision (run-time):

- Deployment Phase: The framework forces service validation tests in a sandbox environment before deployment in the service providers infrastructure, including automated deployment procedures based on semantics for service resource requirements and network capabilities.
- Execution Phase: Run-time monitoring mechanisms enable service adaptation to environment changes and adjustment of network parameters. This adaptation can result in reselection of involved components at run-time.

The paper is organised as follows. After a review of the State of the Art the concept is presented in section 3. The following sections discuss the key issues of the corresponding service life cycle phases, i.e. service modelling (section 4), service creation (section 5), and service deployment and execution (section 6).

2. State of the Art and Open Issues

2.1 IoT Services and the resource concept

Different to the concept of web services IoT services have a strong connection to real world resources with interaction capabilities. The SENSEI\textsuperscript{2} Project follows this concept by introducing of denoting services as resources. Sensor and Actuator Networks can be described as resources. The resource model enables to model functionalities and how they can be accessed. The representation are embodied in an ontology form. Models for self-descriptive meta-data have been investigated by a number of publications (e.g. [1]) but most of them do not provide models for observation and measurement of data. One well known approach is the Observations & Management and SensorML\textsuperscript{3} specification by the OGC Sensor Web Enablement (SWE). Another attempt for an ontology describing sensors has been developed by the W3C Semantic Sensor Network Incubation Group (SSN-XG)\textsuperscript{3}. It focusses on modelling and describing sensor devices rather than addressing the feature of interest as well as spatial and temporal attributes. None of the mentioned concepts and specifications addresses the integration of testing in the design phase. Our proposed concept will assure that the semantic description of IoT services will include meta-information which enables a test-driven service creation.

2.2 Service Creation and Deployment

Middleware architecture approaches play an important role for the service creation and deployment of IoT services. Middleware architecture proposals in the IoT domain are often inspired by Service Oriented Architectures (SOA). SOA solutions for IoT need to address abstraction device/resource functionalities, communication capabilities, provision of service components and service composition. Several research projects have been focused on an easy composition of services and not on testability. The SOCRADES project\textsuperscript{3} middleware architecture enables applications to interact with and consume data from a wide range of networked devices and sensors. It realises a web-service interface which abstracts low-capacity devices interfaces. The middleware supports composition of IoT services based on Business Process Execution Language (BPEL) extensions with run-time selection of actual service endpoints.

\textsuperscript{2}The FP7 ICT SENSEI project, http://www.sensei-project.eu/
\textsuperscript{3}http://www.w3.org/2005/Incubator/ssn/
Discovery of services in a dynamic environment, where devices move and may dis-appear with no guarantee on re-appearance at any time, make the service discovery and provisioning a very challenging tasks in the domain of IoT. SENSEI’s Resource Lookup Interface provides among resource inspired lookup also a semantic oriented resolution method using SPARQL. In contrast to the mentioned approaches our service creation platform will automatically acquire knowledge about services and determine performance of services under different conditions (network, load, environment). This introduces a new view on the service testing process and intelligent deployment.

2.3 Business Process Management

Explicit process design enables unambiguous interpretation of the process by different stakeholders by utilising formal specifications. One example is the Business Process Execution Language for Web Services (BPEL4WS) [4] which is an XML-based language for web service composition. While BPEL offers specifications for structured activities (e.g. while, sequence, switch, etc) as programming constructs there is currently no standard way to visualise the representation graphically. Current business process descriptions have to be extended and enhances to support IoT use-cases and scenarios. Therefore, in our approach IoT service descriptions and relevant attributes will be designated to associate and relate the service description to business process requirements.

2.4 Test environments with automated test composition

Current solutions do not consider the automated test-friendly development of services. Some approaches like the UML 2.0 Testing Profile [5] provides concepts for designing and developing black-box-tests but does not provide guidance how to use it. The abstract U2TP notation have to be transformed into a test specific programming language like TTCN-3 [6] or JUNIT [7]. TTCN-3 specify tests and how they have to be executed. Several tools provide an environment for test creation and execution. Nevertheless, a drawback of TTCN-3 is the lack of simplicity in the generation of tests since the users need to learn a new language. Our approach tries to overcome this drawback by automated test case creation and execution during service development.

3. Concept

Most of the current Internet of Things architectures provide environments for specific applications and domains. To overcome the current silo solutions and different technologies used in the individual sectors, the paper proposes an IoT service creation environment, bridging the gap between various business services and the heterogeneity of networked sensors, actuators and objects. Figure 1(b) depicts an architectural overview of the approach. The service creation and provisioning Environment (SCPE) distinguishes three layers: Starting from the bottom the abstraction layer provides a common interface from the heterogeneity of sensors, actuators and objects to the service components. Hence when a new sensor interface has to be included, the mapping has to be provided only once in the abstraction layer. In the middle layer, to ease service creation, re-usable components provide common functionalities for IoT services such as data access, pattern recognition, context processing, streaming data processing and mining, decision support, and distribution and aggregation of information. The service components have to support adaptability to dynamic environments and business application requirements. They provide self-managed and interoperable building
blocks. The top layer takes care of the service composition. To describe the business workflow for the services, a standardized technique, i.e. BPEL, is used to compose and control the sequence of the re-usable service components. In BPEL it is possible to define synchronous and asynchronous processes. Thus it is very well suited to be used in combination with IoT services. The architecture empowers a test-driven approach for service life cycle management. During design time the service description and service requirements are exploited to derive tests to validate functionality and robustness (see section 5.). Complementary, during run-time, knowledge about the services behaviour dependent on the environment is used to start monitoring procedures for service networks and environment and adapt the services to changed situations as defined in the modelling process (see section 6.). To support the service developer a knowledge based approach is proposed for service creation and provision. Semantic description models are employed to guide and automate the service modelling, development and adaptive execution during the whole service life cycle (see section 4.).

4. Semantic Modelling of Services

The first step in creation of IoT services is to define a model for the IoT enabled services. The model has to map IoT resources in the real world to services in the digital world. Services may be simple services or complex services utilising several other services to perform the task. Hence the modelling has also to support the composition of complex services based on reusable service building blocks. To ease the service modelling a structured, machine-processible approach is needed to access the things in the real world and compose complex service models. The model of the basic service components has to represent the input and output, its effects and the pre-conditions for invoking the service. The OWL-S specification has been designed as a description framework for Semantic Web Services and provides efficient mechanisms to represent the basic service components, including the resource service, in a machine-interpretable way. In the service creation process the IoT models and service building block models can be exploited to select suitable service components and to configure them automatically. It is currently investigated how this reasoning process can be efficiently represented employing knowledge based technologies. IoT based services have to cope with resources that have limited energy and processing power and high dynamics of the environment, i.e. physical world. Therefore the services should be tested for possible known scenarios in design time and adapted to events and changes during run-time. For test and adaptation purposes the service descriptions have to be associated to domain knowledge and related to IoT resource and use case scenario descriptions. In [8], we have explained a service model that extends the OWL-S service description model and have related the IoT service description to IoT resources and physical and logical world entities. To relate the descriptions to domain knowledge, in [9] we have described a linked data approach to use community driven ontologies and meta data to annotate the IoT resources. The service modelling aspect in this work focuses on integration of the works in [8], [9] and extending them to include links to possible evaluation metrics that can be used to identify suitable test modules for the services in design time. This will create a test-aware IoT service description framework. The defined description framework will enable associations of IoT services to resource, environment, network capabilities and

4OWL-S: Semantic Markup for Web Services, http://www.w3.org/Submission/OWL-S/
5. **Test-driven IoT Service Creation**

Developing and provisioning of IoT services is a time consuming, cost extensive and an error-prone process. The service creation composes IoT services based on re-usable service components. The knowledge-based service composition employs the semantic service description to compose reusable service components. In parallel the service description is used to infer the system under test (SUT) model, using Finite State Machines (FSM) and other explicit representations. From the SUT model an algorithm generates the service specific test cases by identifying every possible path through the FSM. After the generation is done, every identified test case is converted to TTCN-3 within the Test Case Generation process. Finally the TTCN-3 test cases are compiled and run on the test environment. The results are feedback to the SCE. The test cases are stored in the SCE to be reused for backward compatibility tests of future releases.

Figure 2 emphasizes the required steps to build test cases from a IoT-aware business process description. Figure 2(a) shows a created business process based on BPEL. The intention of the example business process is to change the wake up time according to other domain knowledge in service platforms.
the weather conditions. Therefore, the created service has to invoke external services for the user location and weather condition. The creation of a TTCN-3 test cases are accomplished by three main steps: i) identification of the interfaces of the SUT components. This step is shown in Figure 2(b). Afterwards ii) a message template has to be build to describe the expected messages received and send it to each identified interface (Figure 2(c)). At the last stage iii) a FSM has to be build from the business process description (Figure 2(d) shows a simplified FSM for our example business process). With this abstracted models it is possible to build the functional test cases, which are automatically compiled to TTCN-3 and executed [10]. Nevertheless, in practise current business process descriptions do not cover the required information for automated testing and it is part of our current work to identify missing information (for example the complete description of external interfaces of a invoked service including expected complex data types and internal relevant states).

6. Knowledge based Service Provision and Monitoring

In the presented concept it is foresee that the services are tested in a sandbox before deployment in the service provider’s infrastructure. The service deployment utilises high-level semantic descriptions of IoT services, resources, and network capabilities for fail-safe deployment of services in large scale environments. This data is processed and interpreted in order to get a full description of the service (hardware and software). This description is provided using the OVF (Open Virtualization Format, DMTF standard) format, in which it is possible to specify from hardware resources and configurations such as network interfaces and their configuration, to the software used in the service. Thereafter a control mechanisms can automatically generate and configure the virtual machines that conform to the service.

In the execution phase the service networks and applications need to be significantly flexible to operate and evolve in highly dynamic environments. This requires the services to adequately identify and react to various changes in the environments using automated mechanisms. Therefore monitoring methods will identify, detect, and foresee critical situations and changes that may occur in service networks and environments.

7. Conclusions

The service creation environments and service oriented architectures are adopted for many business applications. However, the current service creation environments have not been designed to respond to the requirements of the dynamic and heterogeneous environments and business applications in the IoT field. The network dependency and interoperability together with mobility and spontaneous disconnection of IoT resources and physical world objects make the development and management of IoT-enabled service an error prone process. Service testing is widely neglected or is considered in later stages; i.e. after deployment in the service provider’s infrastructure. This paper describes a test-driven approach to address the requirements of IoT-enabled services in dynamic environments. The framework proposes a composition of IoT-enabled business services based on re-usable service components and integration of testing in the service life cycle management. Although testing will significantly improve IoT services at runtime it will be part of further investigations to determine if it will pay off the additional test effort during the design-phase. The semantic description of the services, domain
knowledge and technical aspects of services, network and execution environment helps to automated the knowledge-driven service selection, test, composition and integration that considers dynamicity of the IoT environment at different stages of the service life cycle. This extends the conventional services development process and addresses rapid changes in IoT business environments and also enables adaptable and trustworthy service provisioning, deployment, and execution for IoT platforms.

8. Acknowledgement

The research leading to these results has received funding from the European Union Seventh Framework Programme under grant agreement n° 257521.

References


