Using Workflows to Build Compositions of Read-Write Linked Data APIs on the Web of Things

Tobias Käfer[1], Sebastian Lauber[1], and Andreas Harth[2]

[1] Institute AIFB, Karlsruhe Institute of Technology (KIT), Germany
`tobias.kaefer@kit.edu`   `mail@slauber.de`
[2] Friedrich-Alexander University Erlangen-Nuremberg, Germany
`andreas.harth@fau.de`

Abstract. We showcase an approach to use workflows for composing applications from components with Linked Data interfaces. Our approach consists in an ontology and corresponding operational semantics. We use the ontology to describe workflow models and workflow instances in Linked Data. To execute workflows, we developed operational semantics for the ontology using ASM4LD, a rule-based language for specifying computation in the context of Read-Write Linked Data. For our interactive showcase, we use as components: (1) networked devices with sensors and actuators from the Internet of Things / Web of Things, and (2) APIs from the Web that produce and consume RDF.

1   Introduction

The increasing distribution of information systems into decentralised components poses new challenges for integration. Consider, e. g. the increasing amount of devices on the Internet of Things (IoT) caused by the widespread availability of cheap networked hardware[3], the modularisation of software into microservices caused by the need for component re-use in rapidly changing business environments [6], and decentralised social networks driven by the demand of users to retain data ownership[4]. To equip components with Linked Data interfaces is a first step towards integrated systems, as the Linked Data principles advocate the use of web technologies in interfaces for large-scale integration: On the data level, RDF allows for simple merging of data from different sources, and RDFS allows for basic schema alignment using light-weight reasoning. On the interaction level, HTTP provides an uniform interface to functionality on the web.

Besides components with interfaces made for integration, we need a way to compose applications from the components. Workflows are a way to create applications that is highly suitable for intergration scenarios [2]. To use workflows in an environment requires a model of computation for the environment and

---

[3] `http://www.forbes.com/sites/oreillymedia/2015/06/07/`
`how-the-new-hardware-movement-is-even-bigger-than-the-iot/`
[4] "Putting Data back into the Hands of Owners", `http://tcrn.ch/2i8h7gp`

a corresponding workflow language with operational semantics [1]. In previous work, we have developed ASM4LD [3], a model of computation for Read-Write Linked Data. In this demo, we showcase WiLD, short for **W**orkflows **i**n **L**inked **D**ata, a workflow language with operational semantics in ASM4LD. The language is closely related to standard workflow languages such as BPMN. Basic assumptions around Linked Data are in contrast to established environments for workflow systems, which poses peculiar challenges when providing operational semantics of a workflow language for the environment of Linked Data:

Querying and reasoning under the open-world assuption Semantic Web ontology languages such as RDFS and OWL make the open-world assumption, whereas traditional workflow systems operate on relational data bases, which make the closed-world assumption. To check whether something holds for all parts of a workflow requires closedness.

The lack of event data in REST HTTP supports the CRUD operations (create, read, update, delete), but not the subscription to events. However, works in workflow management traditionally use events as notifications of change.

We elaborate in our paper "Specifying, Monitoring, and Executing Workflows in Linked Data Environments" [4] how we address those challenges in *WiLD*, provide the operational semantics, and evaluate *WiLD* empirically and formally.

The contribution of this demo is a working system in the form of an interactive demo[5] of *WiLD*, which brings workflow-based composition to Linked Data-based components, e. g. on the Web of Things. In contrast to rule-based composition typically found in IoT-based home automation scenarios, workflow-based composition allows to specify compositions that require a notion of application state. While the demo scenario is fairly small and the selection of components may seem a bit made-up, we present an analogy of the demo scenario to a more serious scenario of a digitised work environment in the conclusion.

2   Related Work

Unlike in other works around semantics, REST, and services, we do not study how to derive compositions from descriptions, e. g. using a planner or proof mechanism, but study the enactment of compositions. In other words, we study how to execute a composition, be it generated automatically or manually, as for this paper. For instance, in Semantic Web Services, on the Web of Things, or when using RESTdesc [9], a major focus is on descriptions of functionality, prominently inputs/outputs, and how to derive compositions from such descriptions.

Unlike in Web Services, which are based on a model of computation where service calls are first-class citizens, we work with REST, where state information is the first-class citizen. Pautasso proposes an extension to BPEL [7], a language to specify executable compositions of Web Services, to regard REST calls as BPEL service calls, thus retrofitting REST calls for state information as function calls into BPEL. Our approach however is based on ASM4LD, a model of

---

[5] Please find a video at `http://people.aifb.kit.edu/co1683/2018/iswcdemo/`
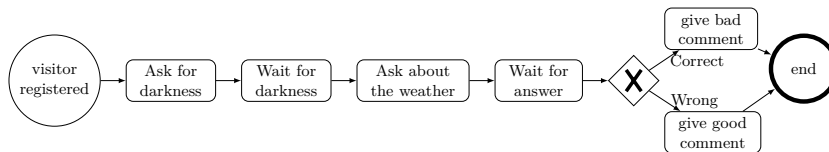
Fig. 1. The workflow of our demo, depicted in the notation BPMN.

computation that is built for state information. Similarly, scientific workflows, another approach to bridge semantic technologies and workflows, work on different models of computation, e. g. pipe-and-filter, or function calls [1]. Our work is based on a model of computation for REST.

In a previous demo, we showed rule-based composition [5], whereas in this demo, we present workflow-based composition.

3 Demo Walk-Through

Visitors to our booth open their smartphone and find our WiFi network. They connect and are directed to our registration form where they register their name and home town. By registering, visitors create a workflow instance in a LDP container. The instance is immediately executed and guides the visitor in the following manner: At the first station, visitors are asked using a loudspeaker connected to an IoT device to make it dark by covering another IoT device's light sensor. The request for darkness shall get the visitors closer to the devices and familiarise them with the interactive scenario. Again using the loudspeaker, the system thanks the visitor, and next asks the visitor how the weather is at home. The visitors answer by placing a RFID card on RFID sensors, that correspond to nice / not so nice (☺/☹), connected to another IoT device. Next, the system evaluates the answer live using a weather API from the web and gives a comment to the visitor, again using the loudspeaker. The underlying workflow that our demo system executes is sketched in Fig. 1.

4 Demo Components

The demo consists of multiple IoT devices, all exposing data and functionality on a (Read-Write) Linked Data interface: RFID readers, a light sensor, and a loudspeaker. The devices are built using off-the-shelf IoT components such as Tessel 2[6]. In terms of Web APIs, we use interfaces to Weather Underground, a weather data service, and a registration form. We describe the workflow model for the system as Linked Data using WiLD[7], the ontology for our workflow language, and put the workflow model in an LDP Container. Moreover, we maintain workflow instances in the same LDP container. To execute the workflow instances, we deploy the operational semantics for the workflow language [4] on a rule engine for ASM4LD, Linked Data-Fu [8]. Fig. 2 shows an overview over the set-up.
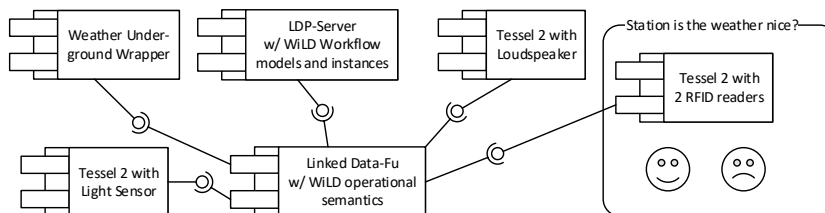
---

[6] `https://tessel.io/`
[7] `http://purl.org/wild/vocab`

Fig. 2. The set-up of our demo as UML Component Diagram. All interfaces are Linked Data interfaces.

5 Conclusion

We presented our approach to use workflows to compose components that expose Linked Data interfaces to an integrated application. Thus, our approach allows to do composition in the context of the fundamental technologies of the Web of Things. For the composition, we use a workflow language for Linked Data with a formal grounding. The formal grounding allows to apply formal techniques for workflow analysis, e. g. for liveness and deadlock properties. While the demo looks somewhat made-up, our approach allows to build serious applications: For instance, an assistance system for workers that work in an digitised environment: Imagine the loudspeaker as virtual assistant that gives instructions to an order picker. The tracking of the progress of the picker's work can be done using light and proximity sensors, or RFID sensors on shelf and tray.

Bibliography

1. Elmroth, E, Hernández-Rodriguez, F, and Tordsson, J: Three fundamental dimensions of scientific workflow interoperability: Model of computation, language, and execution environment. Future Generation Computer Systems 26(2) (2010)
2. Jablonski, S and Bussler, C: Workflow Management. International Thomson (1996)
3. Käfer, T and Harth, A: Rule-based Programming of User Agents for Linked Data. In: Proc. 11th LDOW (2018)
4. Käfer, T and Harth, A: Specifying, Monitoring, and Executing Workflows in Linked Data Environments. In: Proc. 17th ISWC (2018). Accepted
5. Keppmann, FL, Käfer, T, Stadtmüller, S, Schubotz, R, and Harth, A: High Performance Linked Data Processing for Virtual Reality Environments. In: Proc. Posters and Demos at the 13th ISWC (2014)
6. Newman, S: Building microservices - designing fine-grained systems. O'Reilly (2015)
7. Pautasso, C: RESTful Web Service Composition with BPEL for REST. Data and Knowledge Engineering 68(9) (2009)
8. Stadtmüller, S, Speiser, S, Harth, A, and Studer, R: Data-Fu: A language and an interpreter for interaction with R/W Linked Data. In: Proc. 22nd WWW (2013)
9. Verborgh, R et al.: Functional descriptions as the bridge between hypermedia APIs and the Semantic Web. In: Proc. 3rd WS-REST (2012)