Service Integration - A Web of Things Perspective

W3C Workshop on Data and Services Integration

Simon Mayer
Institute for Pervasive Computing
ETH Zurich, Switzerland
simon.mayer@inf.ethz.ch

The augmentation of everyday things by embedding computation and communication abilities enhances their utility beyond their traditional use and generates substantial added value for individuals as well as companies. While isolated "smart things" are already able to provide useful services to human users (e.g., Nike+[1], smart plant care[2], etc.), the real potential of this development lies in interconnecting the capabilities and services of such augmented devices and thus creating an Internet of Things (IoT). Based on this concept, the Web of Things (WoT) [3] aims at connecting devices by making them prime citizens of the World Wide Web. The main advantage of this approach is that the use of widely deployed and accepted Web standards and protocols like the Hypertext Transfer Protocol (HTTP) or Uniform Resource Identifiers (URIs) allows for high scalability, increased interoperability, and a low barrier of entry to use the system (via one's browser).

Within the WoT, we advocate the use of Representational State Transfer (REST)-driven [1] resource-oriented architectures, meaning that every concept that needs to be used or addressed receives a URI. This has advantages with respect to the exchange of information between devices and the use of services provided by them, as all communication takes place via well-defined interfaces using basic HTTP verbs (e.g., `GET`, `POST`, `PUT`, `DELETE`) with defined semantics (cf. Figure 1). When used according to standards and together with the content negotiation feature of HTTP, this system already provides a very basic service integration scheme whose features are similar to what the Web Services stack (WS-*) offers [6].

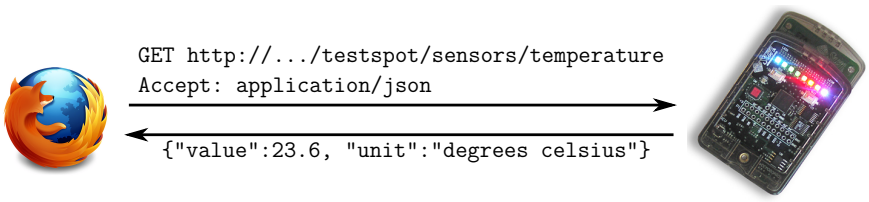

Figure 1: Request to `GET` the current temperature from a sensor.

[1] See `http://www.apple.com/ipod/nike/`
[2] See `http://www.koubachi.org`

Service Integration in the Web of Things

To make Web-enabled devices widely usable, the interaction with such smart things must be facilitated for both, other devices and human end-users. Current research in the WoT domain therefore targets the development of a discovery and look-up mechanism for Web-enabled devices. To allow the semantic discovery of resources and to enable users of the system to *find* appropriate services, our approach is to enrich smart things' representations with metadata like names, brands, tags, or geographical information. Our main goal when designing this markup is to create a scheme that is simple to use not only for users of annotated resources but also for their creators: The annotation format should be simple enough to be generated and embedded not only by programmers but also by Web developers to annotate their Web services (and Web-enabled things). In addition to the annotations required for making smart things findable, information on a device's program interface must be provided for enabling *device-device interaction* and *service integration across devices*. To this end, smart things representations must specify, at the least, which URL to use to invoke the service, what parameters to supply, and what response format to expect. Knowledge of this information already allows for the semi-automatic construction of physical or virtual mashups, i.e., applications that combine the functionality of services provided by smart things and programs: Based on the information about which devices could in principle interact with each other, a system could recommend possible interconnections and thereby support users in mashing up smart things.

In our projects and prototypes, we have been gathering experience with using different lightweight markup languages like Microdata and Microformats. We have also looked at how much information can be possibly deduced by simply crawling the representations of Web-enabled things. Finally, we have investigated approaches of how to integrate descriptions provided using different annotation formats and languages. In the following, we give an overview of our findings.

Crawling Web-enabled Smart Things

With the term *crawling*, we refer to the recursive analysis of the REST interface of resources and of their outgoing links. This can be useful to extract meaningful metadata from smart things that do not provide any explicit semantic description. The crawling of smart things' Web representations has been evaluated within the context of the Social Access Controller (SAC) application that allows the sharing of smart things via social networks [2]. When a user registers a smart thing to be shared, the thing's resource structure is crawled to identify its resources and capabilities and a list of the resources is compiled and presented. The user can then select which of his friends can interact with what resource.

During crawling, the application makes use of the `HTTP OPTIONS` and `HTTP GET` requests as well as content negotiation to get an image of the interface of a smart thing. Adhering to the REST architectural style during the design of our resource prototypes thus allows to deduce meaningful information (i.e., allowed HTTP methods, response types, authentication mechanisms) in this way. Crawling also has benefits with respect to the extensibility of the Social Access Controller as all our prototypes created after the implementation of SAC

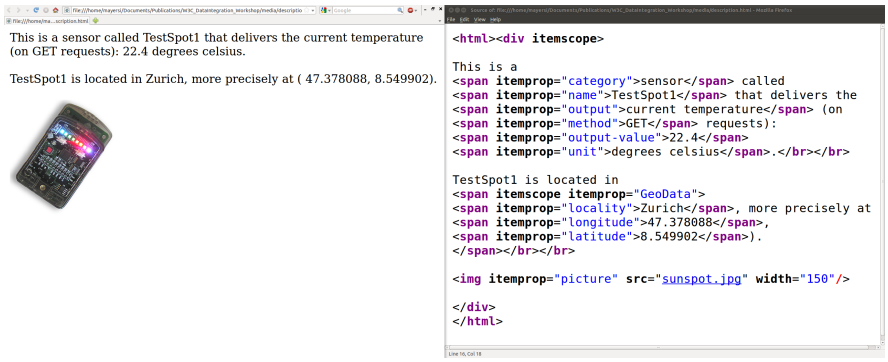

Figure 2: Microdata-based annotation for a sensor node: Browser view and HTML markup with annotations.

were compatible with the application. However, the data generated by crawling smart things does only represent very rough information and, specifically, does not enable automated service integration for smart things.

Semantic Resource Markup

To allow smart things to give more specific information about their relevant properties regarding service integration, we have chosen to evaluate the use of lightweight annotation formats like Microformats or Microdata. The modeling of services in a resource-oriented style advocates the use of service descriptions embedded directly in the Web representations of resources. These concepts are especially interesting in the considered domain, as they allow to provide integrated descriptions for both, humans and machines (cf. Figure 2).

Microformats Microformats[3] are a set of formats that are used to embed semantic information directly in the HTML representation of a resource by re-using existing (X)HTML tags. Already well-known formats that are increasingly being exploited by search engines to index information on the Web are `geo`, `adr` (both part of the `hCard` Microformat), and `hProduct`. The most relevant Microformat for service integration is `hRESTS` [4] which specifies information on the RESTful services provided by a specific resource. The example (cf. Listing 1) illustrates how to use Microformats to describe the use of a service called *ACME Hotels* by sending a `GET` request to the endpoint at `http://example.com/h/{id}` where "*id*" is replaced by the appropriate hotel ID.

The main advantage of Microformats is that they lower the barrier of entry for people to contribute to the proliferation of semantic data on the Web. However, while for instance the `geo` Microformat allows for automatic processing of the embedded information, `hRESTS` still does not allow a machine to automatically deduce a meaningful interface to the service, for instance due to the lack of type information on the variables to embed.

[3]See `http://microformats.org`

```
<div class="service">
  <p class="label">ACME Hotels</p>
  <div class="operation">
     The operation is invoked using a
     <span class="method">GET</span> on
     <code class="address">http://example.com/h/{id}</
        code>, with
     <span class="input">the hotel ID replacing <code>id
        </code>.</span>
  </div>
</div>
```

Listing 1: hRESTS service description markup, adapted from [4].

Microdata Microdata is a HTML5 specification[4] by the Web Hypertext Application Technology Working Group (WHATWG) to embed semantic information within Web pages similar to the way this is handled with Microformats. The main difference to Microformats is that Microdata uses new tag attributes (e.g., `itemscope`, `itemprop`, etc., cf. Figure 2) to embed information while Microformats overload the HTML `class` tag, which causes problems for parsers to differentiate between semantic information and styling markup. The semantics of Microdata are provided by Microdata vocabularies such as `http://data-vocabulary.org` that includes definitions for *Person*, *Event*, *Organization*, etc. These terms are already supported by Google, while Google, Bing, and Yahoo! have created the `schema.org` initiative that defines the vocabulary for a more extensive collection of concepts. There are currently no efforts to standardize a Microdata-based description language for RESTful services. Still, this could be a practicable approach to creating lightweight, easily understandable, and expressive Web-based resource annotations.

Decoupling Representation and Interpretation

We have recently proposed a way to achieve a decoupling between the format a resource uses to describe its service interface and the format that is consumed by prospective clients [5]. Our approach was to create a semantic discovery service for Web-enabled smart things that is based on the application of multiple *Discovery Strategies* to a Web resource's representation: One strategy would handle Microformats-based annotations, another would search for Microdata markup, etc. To make this mechanism future-proof, we allow for the creation and updating of strategies at runtime. The service thus enables developers and users to create and submit new methods of parsing the semantic descriptions of Web resources on demand.

Conclusion

We believe that the creation of standardized interface descriptions and formal service contracts is of great importance for RESTful services, especially given the proliferation of RESTful APIs on the Web. In this paper, we have provided an overview of approaches to enable the integration of services provided by Web-enabled smart things within the Web of Things domain. While a well-designed

[4] See `http://www.whatwg.org/specs/web-apps/current-work/`

RESTful interface already allows for very basic service integration, we advocate providing further information on the program interface of a service using standardized markup embedded in the HTML representation of its resources. We note that, while in our opinion the `hRESTS` Microformat lacks expressiveness, creating standardized RESTful service descriptions based on Microdata is feasible and desireable.

References

[1] R. T. Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine, 2000.

[2] D. Guinard, M. Fischer, and V. Trifa. Sharing Using Social Networks in a Composable Web of Things. In *Proceedings of the 1st IEEE International Workshop on the Web of Things (WoT 2010) at IEEE PerCom 2010*, Mannheim, Germany, Mar. 2010.

[3] D. Guinard, V. Trifa, F. Mattern, and E. Wilde. From the Internet of Things to the Web of Things: Resource Oriented Architecture and Best Practices. In D. Uckelmann, M. Harrison, and F. Michahelles, editors, *Architecting the Internet of Things*. Springer, 2011.

[4] J. Kopecký, K. Gomadam, and T. Vitvar. hRESTS: An HTML Microformat for Describing RESTful Web Services. *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, 1:619–625, 2008.

[5] S. Mayer and D. Guinard. An Extensible Discovery Service for Smart Things. In *Proceedings of the 2nd International Workshop on the Web of Things (WoT 2011)*, San Francisco, USA, June 2011. ACM.

[6] L. Richardson and S. Ruby. *RESTful Web Services*. O'Reilly Media, Inc, Sebastopol, CA, USA, First edition, May 2007.