Semantic Data Mediator: Linking Services to Websites

Dennis Wolters, Stefan Heindorf, Jonas Kirchhoff, and Gregor Engels

Department of Computer Science, Paderborn University, Paderborn, Germany
{dennis.wolters@, heindorf@, jonaskir@mail., engels@}
uni-paderborn.de

Abstract. Many websites offer links to social media sites for convenient content sharing. Unfortunately, those sharing capabilities are quite restricted and it is seldom possible to share content with other services, like those provided by a user's favorite applications or smart devices. In this paper, we present *Semantic Data Mediator (SDM)* — a flexible middleware linking a vast number of services to millions of websites. Based on reusable repositories of service descriptions defined by the crowd, users can easily fill a personal registry with their favorite services, which can then be linked to websites by SDM. For this, SDM leverages semantic data, which is already available on millions of websites due to search engine optimization. Further support for our approach from website or service developers is not required. To enable the use of a broad range of services, data conversion services are automatically composed by SDM to transform data according to the needs of the different services. In addition to linking web services, various service adapters allow services of applications and smart devices to be linked as well. We have fully implemented our approach and present a real-world case study demonstrating its feasibility and usefulness.

Keywords: Services; Semantic Data; Mediation; Data Conversion; Interface Adaptation

1 Introduction

Many websites enable users to share the presented content by integrating services of social media sites, e.g., the service to share content on Facebook or pin it on Pinterest. Mobile websites might even allow to share content via locally installed messengers like WhatsApp. Sharing content presented on a website with other services than those which are already linked to the respective site is up to end users, because website developers cannot accommodate all services relevant for users, especially since those might differ from user to user. Unfortunately, directly linking services to websites is a difficult task for end users, because it requires extensive technical knowledge. Thus, end users usually settle for manually copying content to other services, which is often very tedious. For instance, when transferring a recipe found on a website to a cookbook app, all relevant information like title, ingredients, instructions, and associated images need to be copied to the respective fields of the cookbook app's input form. If the app does not run on the same device as the browser presenting the recipe, it is even more complicated.

In this paper, we explain how our approach Semantic Data Mediator [2] can be used to link additional services to websites. On the one hand, we demonstrate how regular end users without any programming knowledge can make use of SDM to link their favorite services to millions of websites. For this, SDM analyzes the semantic data embedded into websites for search engine optimization and typed over ontologies like schema.org[1]. Thereupon, it determines which additional services can be offered for the found data. On the other hand, we explain the different building blocks of SDM and how power users can extend our approach. Moreover, we describe how we implemented SDM and provide a video demonstration how to use it. To the best of our knowledge, SDM is the first comprehensive approach enabling end users to bridge the gap between websites and services. Existing approaches are either restricted to a few selected services, e.g., share buttons for social media sites, or browser extensions for single services such as Skype.

2 System Overview

In this section, we give an overview of our approach *Semantic Data Mediator (SDM)*. The different buildings block of SDM visualized in Figure 1 are explained in the following.

In the center of SDM is a *personal registry*, which manages all services relevant to a user. Services, which include both web services and services provided by locally installed applications, can be added to the registry by providing their OpenAPI descriptions[2]. Those descriptions can either be referenced via an URL or can be retrieved from service repositories (see below). The personal registry also contains services for data conversion, since the format of the semantic data embedded into websites usually differs from the data format expected by services. The input and output formats of such converter services are used to create a graph representing all possible conversion options. Based on this graph, the personal registry can efficiently compute output formats from given input formats. This information is used to determine which services can be offered to users.

A *service repository* contains OpenAPI specifications of services that can be added to a user's personal registry. Upon adding a service, credentials for using the service can be permanently stored or access tokens can be obtained, e.g., when OAuth is used. Additionally, user-specific parameters, e.g., an API key required to access a service, can be defined by using a custom property in the service description.

[1] http://schema.org [2] https://github.com/OAI/OpenAPI-Specification

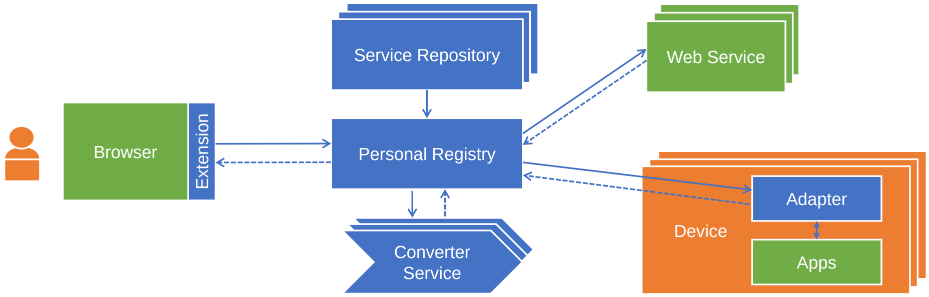

Fig. 1. Linking services to websites: Architecture of Semantic Data Mediator

The *browser extension* extracts semantic data from a website and queries the personal registry for services that are able to process this data. These services are presented to the user either in an embedded menu attached to the visual representation of a data item on the website, or in a browser-based menu, which lists all data items embedded into the respective site. When a user chooses to invoke a service, the extracted data item is transmitted to the personal registry, which applies all necessary conversions using the previously determined converter services and afterwards transmits the (converted) data to the chosen service. A service can be a web service or it can be a service provided by an application installed on one of the user's devices. This does not necessarily has to be the device on which the browser runs, since SDM supports the cross-device integration of services via adapters.

Adapters are used to create external service interfaces for applications that only provide interfaces limited to other applications within the device on which they are installed. For instance, we use our Cross-Device Application Integration approach for Android apps (XDAI-A) [3,4] to enable the usage of services offered by Android apps, e.g., sharing data with a messenger app, adding data like an event, a recipe, or a note to apps like Android Calendar, MyCookbook[3], or Google Keep[4], respectively. XDAI-A provides a domain specific language to create adapters to support further Android apps. Additionally, we provide a customizable command-line interface (CLI) adapter that can be used to enrich command-line tools with a RESTful HTTP interface. To configure the adapter, an OpenAPI specification needs to be created, which describes the different endpoints provided by the service. For each endpoint, a custom property is used to configure which command-line tool is executed upon incoming requests. Data from an HTTP request can easily be mapped to arguments of the invoked tool and the tools output can again be mapped to the HTTP response. We also provide a Cross-Device Application Integration approach for Windows applications (XDAI-W), which extracts information about installed applications from the Windows registry and offers an HTTP interface to use these applications in combination with SDM.

Converter services are specially tagged services that are used by the personal registry to convert the semantic data extracted by the browser extension. SDM can automatically compose converter services, and thereby, enables the use of services that only accept a data format different to the one of the extracted data. The data conversion takes both the semantic type, e.g., event or recipe, as well as the data format, e.g., JSON-LD or RDF/XML into account. Details about the data conversion can be found in [2]. To accelerate the creation of converter services, we provide a template which only requires a conversion function and some meta information, like a description and input/output types. Additionally, our CLI adapter allows to simply reuse existing command-line conversion tools as converter services. For instance, we use this adapter to provide a converter service based on the universal document converter Pandoc[5].

In combination, the previously described building blocks of SDM allow to link services to the millions of websites that provide semantic data [1]. Further support for SDM by website developers or service providers is not required. SDM is simple enough to be used by average end users, while power users can also extend SDM by adding new (converter) services or adapters.

[3] http://mycookbook-android.com/ [4] https://google.de/keep/ [5] https://pandoc.org/

3 Implementation Details

The browser extension is available for Google Chrome. Similar extensions can be developed for all major browsers since no features unique to Chrome have been used to implement the extension. The converter service template, the CLI adapter, the Windows adapter, the server-side part of the Android adapter, and the personal service registry (see Section 2) are developed using JavaScript and the Node.js runtime. The client-side part of the Android adapter consists of two interconnected apps, one responsible for providing the external interface and the other one for adapting service request. Further details on the Android adapter are given in [3,4]. The personal registry uses a document-oriented database (CouchDB) for storing OpenAPI specifications of all registered services along with some additional parameters, like authentication tokens or credentials to use the services. Furthermore, it uses a graph database (Neo4j) to efficiently find conversion options and relevant services for a given website. SDM natively supports OpenAPI service descriptions. Other service description languages, like RAML[6] or API blueprint[7], are supported by converting them upfront to OpenAPI.[8] To enable easy deployment, the registry as well as all converters based on our converter service template or CLI adapter can be deployed as Docker containers.

4 Demonstration

The demonstration consists of two parts: (i) Usage of SDM and (ii) technical details.[9] In the first part, we show how SDM links services to websites and we demonstrate the invocation of web services as well as services provided by applications installed on different devices. For instance, we demonstrate how a recipe found on a website on a desktop computer can be exported to Word or can be added to a cookbook app on a smartphone. Moreover, we explain the addition of services to the personal registry by utilizing services repositories. In part two, we explain the technical details of SDM. We show how the personal registry maintains the information about possible data conversions and how this information is exploited to find services that require data in a specific format. Further, we present the different types of adapters and explain how they can be used to create new services that are accessible for SDM.

References

1. Bizer, C., Meusel, R., Primpeli, A.: Web Data Commons - RDFa, Microdata, and Microformat Data Sets, `http://webdatacommons.org/structureddata/`
2. Wolters, D., Heindorf, S., Kirchhoff, J., Engels, G.: Linking services to websites by leveraging semantic data. In: ICWS 2017. pp. 668–675. IEEE (2017)
3. Wolters, D., Kirchhoff, J., Gerth, C., Engels, G.: Cross-Device Integration of Android Apps. In: ICSOC 2016, pp. 171–185. No. 9936 in LNCS, Springer (2016)
4. Wolters, D., Kirchhoff, J., Gerth, C., Engels, G.: XDAI-A: Framework for Enabling Cross-Device Integration of Android Apps. In: ICSOC 2016 Workshops and Satellite Events. LNCS, Springer (2016), (to appear)

[6] http://raml.org [7] http://apiblueprint.org/ [8] http://github.com/LucyBot-Inc/api-spec-converter

[9] A demo video is available at: `http://sdm.dwolt.de/demo/`