SlideWiki microservice architecture for collaborative online system development

ABSTRACT

SlideWiki is an online system for supporting educators and students in collaborative use of OpenCourseWare (OCW). We combine state of the art feature development and academic research to achieve this. Our online, distributed, and open source development of SlideWiki in an academic and multidisciplinary context introduces challenges related to separation of concerns, community development, the coordination of geographical distributed researchers and developers, and the variety of preferred and necessary programming languages and run-time environments. We present the SlideWiki microservice architecture that we use for collaborative online system development, and discuss how it alleviates above challenges. Our study exemplifies how a microservice architecture can alleviate challenges in developing and maintaining large-scale socio-technical systems in other academic, open source, multidisciplinary, online, and geographically distributed contexts.

CCS CONCEPTS

• **Information systems → World Wide Web**; *Collaborative and social computing systems and tools*; • **Software and its engineering → Collaboration in software development**;

KEYWORDS

Microservice Architecture, Online Collaboration, Distributed Development, Webdevelopment, OpenCourseWare platform

ACM Reference format:
. 2018. SlideWiki microservice architecture for collaborative online system development. In *Proceedings of ACM Web Science Conference, Amsterdam, The Netherlands, May 2018 (WebSci18),* 2 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

SlideWiki was conceived as an web-based OpenCourseWare (OCW) authoring system to allow educators to reuse, adapt, and share slide decks. The first version of SlideWiki 1.0 was launched in 2012 [2]. While this version attracted thousands of slides and acquired a user-base in the open educational community, it required further development to reach its full potential and become a sustainable open-source platform for online education.

In 2016 an EU H2020 grant was awarded for SlideWiki redevelopment to address scalability and usability issues and better serve the needs of users with more and modern features including state-of-the-art research prototypes. This collaboration introduces new challenges (marked in **boldface** text), as we organize development in an online open source project, in which many researchers and developers work together from several countries and time-zones.

In order for SlideWiki to be sustained in the long-term, a critically large **community of developers** and users needs to be established [3]. To support use-cases of large groups of users, many different types of features are required. These features should be easily **re-used** from existing software built in various programming languages and run-time environments.

Moreover, developing the SlideWiki system requires coordination and collaboration between researchers and developers from multiple domains and disciplines, e.g., natural language processing, pedagogy, didactics, education, semantic web, learning management systems, accessibility, usability, etc. The project and community members have varying degrees of proficiency with different programming languages and run-time environments. Certain state-of-the-art research prototypes are only available in (libraries for) specific programming languages and run-time environments.

To work effectively, the researchers, developers, and community contributors thus need a way to develop and integrate different parts of the SlideWiki system that are built in **different programming language and run-time environments**. Moreover, these parts need to have **minimal interdependencies** between each other, to ensure effective collaboration. I.e., there needs to be a strict **separation of concerns** between parts of the SlideWiki system that people will work on, for people to work independently.

The use of microservice architecture [5] has recently gained increased attention from industry and academia, as it provides benefits from allowing easy integration of different technologies in a system, minimized code dependencies, separation of concerns, and independent development of system parts [4]. We present the microservice architecture of SlideWiki that we use for collaborative online OCW system development. In the next section we explain in detail how the challenges (marked in **boldface**) are alleviated.

2 CHALLENGES ALLEVIATED BY MICROSERVICE ARCHITECTURE

The new SlideWiki microservice architecture is depicted in Figure 1 (The underlying code of SlideWiki can be seen on *Github* [1]). In a microservice architecture an application is separated into multiple stand-alone servers which each offer specific functions and data (compare [5]). The stand-alone servers that each host a microservice can run on separated dedicated hardware to scale up the system and ensure high performance, scalability, availability, reliability, and quality of service. In practice the user interface of SlideWiki (see 'frontend web-service' in Figure 1) connects to several back-end microservices to store and retrieve various forms of data, e.g., the deck service for data of decks and slides[2].

[1]https://github.com/slidewiki/, deployed on e.g., http://slidewiki.org/

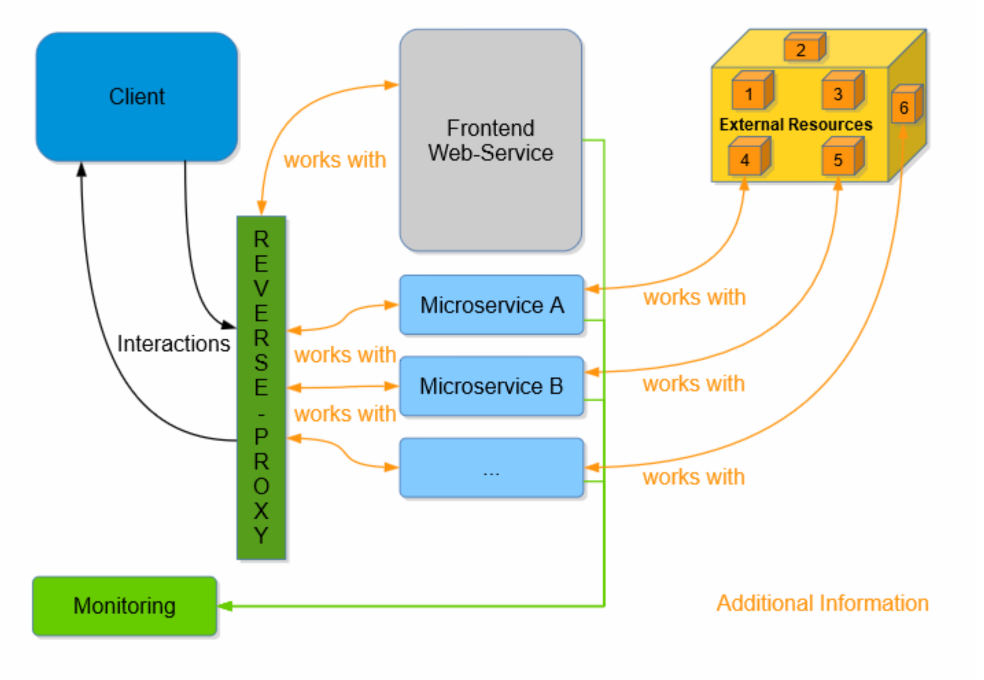

Figure 1: Architecture diagram of SlideWiki 2.0

The use of a microservice architecture alleviates challenges with **community development**: It gives freedom for developers to create microservices by **re-using** existing software or developing new software using **different programming languages and run-time environments**. The microservices connect to SlideWiki via well defined Web-based Application Programming Interfaces (APIs) [2] according to the REpresentational State Transfer (REST) principle [1] which ensures **interoperability**.

Examples of microservices in different programming languages and run-time environments are:

(1) Most SlideWiki microservices are built using Node.js (server- and client-side Javascript) with MongoDB as database
(2) The NLP microservice[3] is built in Java (hosted via Play)
(3) We integrated Unoconv (built in Python) as a microservice

Because the microservices expose RESTfull APIs covering the backend functionality of SlideWiki, the ability to connect to other systems is increased and made significantly easier. Thus **re-use** and integration of existing systems in SlideWiki, as well as interoperability between SlideWiki and other systems is improved.

oreover, microservices provides **separation of concerns**; each microservice addresses a specific concern, for example, storage of user data, storage of slide data, import of educational content in Powerpoint, et cetera. The separation of SlideWiki functions and data into microservices, connected via well-defined and backward compatible Web APIs, **minimizes code dependencies**, thereby increasing maintainability and collaboration; it allows researchers and developers to work in parallel, without having to synchronize between time-zones due to geographic distance, and without worrying about affecting or overwriting interdependent code. This also allows new developers to easily join our project and start working on code that is largely independent from other code and addresses a specific concern, thereby increasing **community development**.

[2] for example: https://deckservice.slidewiki.org/documentation
[3] https://github.com/slidewiki/nlp-service and https://nlpservice.slidewiki.org/docs/

3 CONCLUSIONS

We presented the new SlideWiki microservice architecture used for collaborative online OCW system development. Our study exemplifies how a microservice architecture can alleviate challenges in developing and maintaining large-scale socio-technical systems in other academic, open source, multidisciplinary, online, and geographically distributed contexts.

REFERENCES

[1] Roy Thomas Fielding. 2000. *Architectural Styles and the Design of Network-based Software Architectures.* Ph.D. Dissertation. University of California, Irvine.
[2] Ali Khalili, Sören Auer, Darya Tarasowa, and Ivan Ermilov. 2012. SlideWiki: elicitation and sharing of corporate knowledge using presentations. In *International Conference on Knowledge Engineering and Knowledge Management.* Springer, 302–316.
[3] Kumiyo Nakakoji, Yasuhiro Yamamoto, Yoshiyuki Nishinaka, Kouichi Kishida, and Yunwen Ye. 2002. Evolution patterns of open-source software systems and communities. In *Proceedings of the international workshop on Principles of software evolution.* ACM, 76–85.
[4] Sam Newman. 2015. *Building microservices.* O'Reilly Media, Inc.
[5] Eberhard Wolff. 2016. *Microservices: Flexible Software Architecture.* Addison-Wesley Professional.