ZVAX – A Microservice Reference Architecture for Nation-Scale Pandemic Management

Oliver Cvetkovski[1,2], Carlo Field[1], Davide Trinchi[1], Christof Marti[1], and Josef Spillner[1]

[1] Zurich University of Applied Sciences, Winterthur, Switzerland
[2] University of St. Cyril and Methodius, Skopje, North Macedonia

Abstract

Domain-specific Microservice Reference Architectures (MSRA) have become relevant study objects in software technology. They facilitate the technical evaluation of service designs, compositions patterns and deployment configurations in realistic operational practice. Knowledge about MSRA is predominantly confined to business domains with usually not more than few thousands of users per application. Due to the ongoing digital transformation of society, people-related online services in e-government, e-health and similar domains must be designed to be highly scalable at entire nation level at affordable infrastructure cost. With ZVAX, we present work in progress in the e-health domain. Specifically, ZVAX is an MSRA for pandemic-related processes such as vaccination registration and passenger locator form submission, with emphasis on selectable levels of privacy. We argue that ZVAX is valuable as study object for the training of software engineers and for the debate on government-to-citizen services.

1 Introduction

Microservice-based software applications are expected to have many advantages. They are supposed to be easier to develop with distributed teams of software engineers, to allow for more customisation through flexible service composition, and to be more aligned with business-critical runtime properties such as high scalability and resilience.

In the first years of research on microservices, practical applications to prove architecture-related hypotheses and to implement new and innovative concepts were lacking. A first applications overview was assembled in 2017 [4]. Since then, the knowledge on microservice architectures has increased. This is especially true for microservice reference architectures (MSRAs) that serve as blueprint for other applications in the same domain. Multiple studies have been supported by the growing insight into such architectures, the coupling of the affected services, and the messaging patterns between them, as well as the resulting runtime characteristics. For instance, six reference architectures and use cases were analysed for service dependencies and interchangeability in software product lines, including a detailed study of the demonstration microservices of the Google Cloud Platform [2]. A well-known reference architecture for e-commerce is the Sock Shop, despite not having been formally introduced in the literature, due to its popularity in diverse software modernisation trainings. It consists of six polyglot backend services and one queue and is publicly available on the web[1]. Sock shop has been used among other works by the authors of MicroRCA, a root cause analysis framework to spot performance issues in microservices [13].

More focused and domain-specific studies become feasible when more MSRAs are investigated and published. In recent years, this has increasingly been the case. For instance, MSRA

[1]Sock Shop website: https://github.com/microservices-demo/microservices-demo

for measurement systems and enterprise measurement infrastructures were designed and evaluated [12]. Such systems collect business metrics along with business insights and feed them into enterprise dashboards. Advanced measurement systems include semantic measurement models with distributed data management [3]. The SAMSP platform for self-adaptive microservices has been similarly published. It supports instance overload, unreachable services and other events to adjust service delivery [8]. A multi-tenant SaaS hosted in a cloud based on microservices has been validated by transforming Microsoft MusicStore [11]. A platform based on ProteomicsDB has been built to investigate the use of microservices for proteomics and personalised medicine [10]. Elasticity research in document management systems yielded another MSRA [9].

While the existing MSRAs cover many domains, they are generally limited to business applications with a small or undefined set of users. Following the increasing digitalisation and digital transformation in society, nation-scale applications are emerging and warrant further analysis concerning their realisation based on microservices. During the COVID-19 pandemic, pandemic management services addressing the entire population were provided but often at high cost for raw data centre resources, without reconsidered and rethought software architecture. In other instances, scalability problems such as downtimes and long delays occurred and made it into the national press, contributing to the population's anger about the political management of the pandemic[2]. Public cloud services are promising high scalability and appear to be a solution. They are nevertheless only a part of a solution. First, public cloud services are often also limited to few thousand concurrent instances, which is insufficient for the problem field where tens of thousands of parallel requests may arrive. Second, they provide the heavy lifting but require domain-specific glue logic and interaction patterns, an area where much of the mistakes in architecture design may occur. The blueprint knowledge on how to build nation-scale microservice applications is thus limited to few approaches on vaccination passports [1, 7, 5].

In this extended abstract accompanying the Microservices 2022 conference talk, ZVAX [6] shall serve as exemplary nation-scale application for the domain of digital pandemic management. We consider ZVAX to benefit from a reference architecture for such applications due to the careful design and realisation of the underlying microservices. Although work in progress, it is a functional prototype covering multiple pandemic management fields with separation of concerns along service boundaries. Hence, we argue that ZVAX can serve as practical study object in microservice-related education, such as software architecture courses. In the next sections, we derive the MSRA methodologically, present the microservice concepts of ZVAX and explain the realisation of the underlying MSRA on the implementation level. Finally, we motivate further research directions.

2 Concepts

We condense the problems raised in the introduction into a single research problem: Which software design and architecture adequately fits the e-health domain and more specifically the task of nation-scale pandemic management? In order to address the problem, we first contribute four novel concepts as generic, domain-independent architectural design foundation.

1. Nation-scale services. They are built to serve large parts of a population in a short amount of time. In quantitative terms, we assume a lower bound of many 10,000s of short-lived requests per second. This exceeds current public cloud offerings by an order of magnitude even when combining regions. We note that the target concurrency can often be influenced

[2] Swiss Radio & Television, Espresso, January 15, 2021

by political means outside the technical scope, for instance by population segmentation by age group. Such segmentation may however not be applicable in emergency situations. Under no circumstances should users perceive downtime or unresponded hanging requests.

2. Harmonic scalability. A distributed system is harmonically scalable if all of its constituent parts scale along the critical request paths. For an architecture based on microservices, this entails the ability to scale elastically in each service, but also in attached middleware services. In geometric terms, the system representation may overall shrink or grow but the proportions remain the same.

3. Selective decentralisation. The scalability is influenced by user preferences on where to store and process data. Hence, decentralisation is used to reduce microservice invocation load at neuralgic points while at the same time offering stronger privacy guarantees on demand. From an architectural perspective, the selective decentralisation leads to a selectively externalised statefulness, referring to the state as output of one microservice that determines the follow-up behaviour of another microservice.

4. "Flatten the curve". This term originating in the domain-specific goals of pandemic management also applies to the prevention of microservice overload. Queues and other asynchrony mechanisms are used to facilitate quick responses to service requests, leaving the bulk of the work for less loaded periods of time while granting a rapid responsive behaviour to users navigating the user interface.

Next, we derive a domain-specific reference architecture for the domain of pandemic management. The architecture must support the functional requirements implied by the domain. A representative application covering the main governmental interests in pandemic management consists of the following four classes of services to the population.

1. Appointments. In order to prevent residents from queueing unnecessarily, appointments help to "flatten the people curve" in real-world situations such as testing and vaccination points.

2. Certificate creation and signing. Various schemes exist, with most having settled on a QR code representation digitally signed by a single authority or, for better fraud protection, multiple authorities.

3. Certificate checking. The inverse process, combining signature validity checks with expiration policies.

4. Information solicitation. This encompasses mandatory solicitations such as passenger locator forms and contact quarantine tracing, but also voluntary information provided to assist the tracing.

In a third methodological step, we combine the conceptual requirements and functional scope, and match them against recent technological progress. This means that all the management services of the four classes are instantiated and delivered with high elastic scalability and low latency in order to achieve the desired volume of requests. Two main technological advances are increasingly available in managed microservice environments and are considered favourable from the reference architecture perspective.

1. Event-driven asynchronous function execution. Microservice instances, in particular when designed as event-triggered functions, scale almost proportional to the request rate. Factors such as cold start (for initial invocation) or spawn start (for concurrent invocation) have been thoroughly investigated in recent years and are now well understood, and with methods such as prewarming and idle time optimisation, further scalability gains can be achieved.

2. Preparedness for extreme edge deployments. New hardware allows for deploying microservices directly on network interfaces (e.g. SmartNICs) in edge data centres, allowing for unprecedented scaling of stateless services by geographic distribution close to the points of use and their low-latency delivery. This can be exploited for instance to distribute the work-intensive QR code creation and signing processes that require no state other than a set of input parameters. Moreover, federated deployments become possible to map political hierarchies and pandemic management responsibilities (e.g. federal-level, state-level) to delivery locations. At the same time, the system needs to remain deployable on a single off-the-shelf device to achieve full elasticity from single developer to nation-scale.

The concrete management system reference architecture results as a fourth step of the methodology. It is synthesised along with constituent microservices in Fig. 1, starting from the user perspective of either personal access through a mobile device or computer, or a stationary device such as a QR code scanner located at the entrance of a location that mandates such checks – such as restaurants, public authorities or university campuses. The architecture reflects the ability to choose, on a per-user basis, whether to opt into more government-managed or more self-managed data records, with corresponding responsibilities for backups and access protection. Moreover, it aligns with the increasing availability of micro datacentres and other edge deployments for scalable service delivery close to the users.

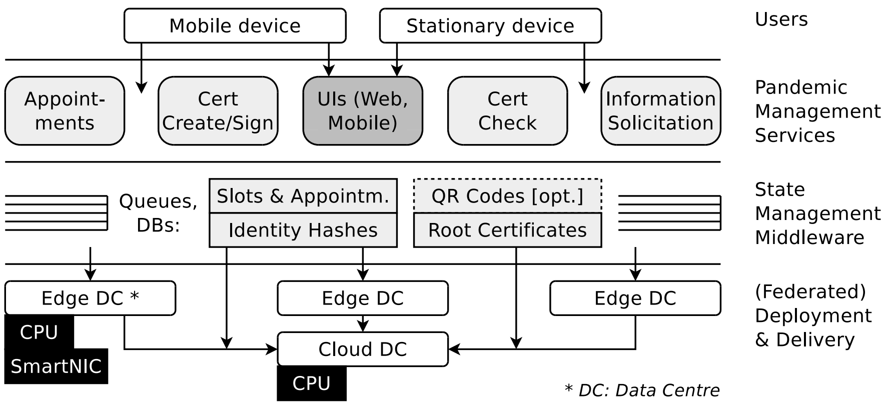

Figure 1: Reference architecture; components marked [opt.] are optional and only activated on demand depending on user preferences due to being offloadable to client-side devices

3 Implementation

ZVAX is a demonstration system of the MSRA for nation-scale pandemic management. Its development started in the second year of the COVID-19 pandemic, and its functional extensions were driven by digital management processes introduced by governments around the world, with emphasis on the services on different administrative levels of the Swiss confederation. Hence, apart from implementing the MSRA and thus overcoming scalability and privacy issues, it also serves as testbed for solving challenges that emerged on a societal level in Switzerland, often with extensive local or national press coverage. These challenges relate to many cases of certificate fraud (solved in ZVAX by multi-signatures, leading however to larger QR codes), interoperability (solved by multi-QR code schemes), different expectations on privacy (solved by selective decentralisation), and flexible federated deployment (solved by an adaptive user interface connected to configurable sets of backend services). ZVAX can furthermore serve as testbed for improved client-side QR code detection, for instance on black background that usually causes problems in today's mobile applications, and coloured or animated codes to represent portrait photos for safer identification.

Fig. 2 gives an impression of the user-facing functionality of the current implementation. Users are able to select the desired level of privacy, in turn determining the degree of centralised versus decentralised data storage. Decentralisation is achieved by a combination of in-browser storage and file downloads, leading to no trace of any health-related activities when full decentralisation is selected by the user. Subsequently, users apply for a test or vaccination appointment, retrieve certificates, enter locator form information, or manage contact tracing. Correpondingly, doctors and health officials are able to sign (and counter-sign) as well as verify certificates.

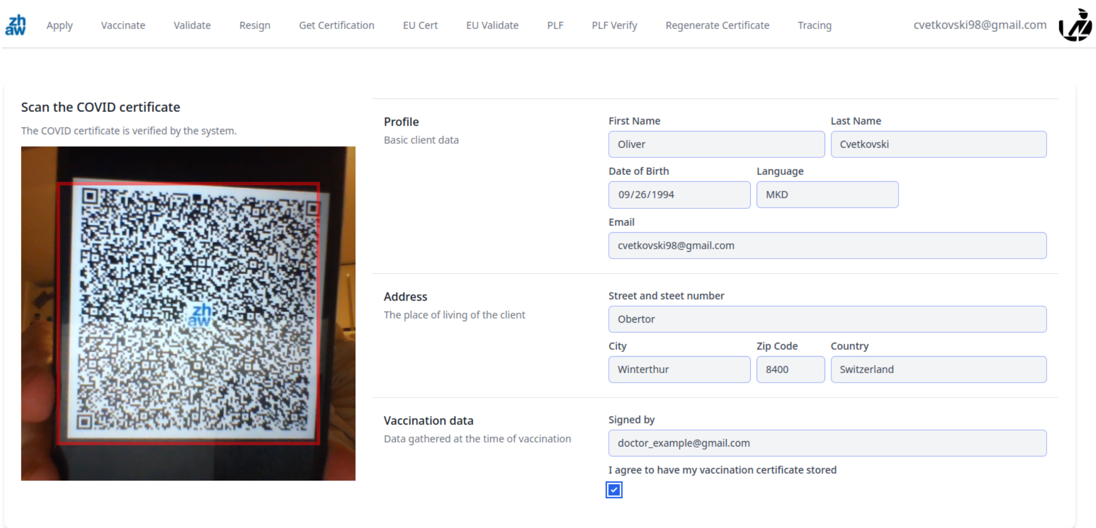

Figure 2: Exemplary screenshot of the QR code verification within the ZAX implementation

Appointments are possible for groups of persons, reducing the need to fill out forms considerably for families. The appointments-related invocation flow on the microservice level, not including the middleware components, is shown in Fig. 3. All microservices can be scaled with instance selection through the Traefic load balancer. First, the authentication service grants a time-limited token. Then, the appointments service is invoked with the token. In case an appointment can not be obtained immediately or the system is overloaded, an asynchronous re-invocation is scheduled ("flatten the system load curve"). Next, the appointment is expressed as a signed QR code, re-using the same services that will also perform the same task for vaccination certificates. All personal information apart from the time slot blocking and an identity hash are then removed from the system, and the QR code is sent via e-mail for the person(s) having received an appointment.

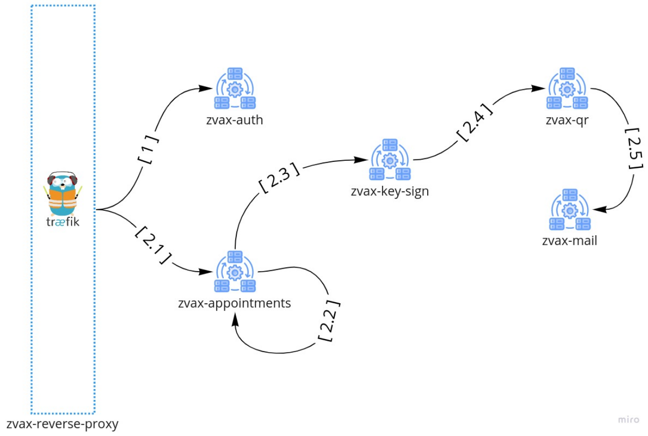

Figure 3: Exemplary microservice invocation flow consisting of two subflows and six steps

The ZVAX implementation is easy to bootstrap on new machines due to its complete containerisation. The codebase is prepared to be scaled horizontally with orchestrators such as Docker Compose, Docker Swarm and Kubernetes. Almost all microservices ship with a single implementation. In order to stress the substitution principle and to compare polyglot implementations, the QR code microservice ships with two implementations, in Python and in Rust. This has not only performance implications, but also affects the scaling behaviour due to differences in size and resource requirements of the resulting container images.

4 Conclusions and Future Work

With ZVAX, we have developed a pandemic management system that improves upon current government-provided services. It is designed to accomodate surge requests and sustained requests at a higher scale, and adjusts to user preferences concerning data handling through selective decentralisation. ZVAX builds upon a domain-specific MSRA that takes current technological progress into account and lets researchers explore more flexible externalised state management beyond the conventional stateful/stateless distinction. Moreover, ZVAX is positioned as study object for the training of software engineers who learn the construction of societally relevant digital services.

Future work is divided into domain-specific and general architectural directions. Within the domain of pandemic management systems, we aim at reducing the need for complementing the QR code with a passport by embedding photographic identity information within a static (coloured) or animated code. On the architectural level, we aim at conducting large-scale and nation-scale performance tests to prove the proposed approach at all levels: services, queues, composition and deployment. This will encompass the acceleration gained by edge deployments, and thus contribute to the debate on digital souvereignty of administrative levels achievable through a wider digital transformation enabled by flexibly deployable microservices.

References

[1] Masoud Barati, William J. Buchanan, Owen Lo, and Omer F. Rana. A privacy-preserving distributed platform for COVID-19 vaccine passports. In Luiz F. Bittencourt and Alan Sill, editors, *UCC '21: 2021 IEEE/ACM 14th International Conference on Utility and Cloud Computing, Leicester, United Kingdom, December 6 - 9, 2021 - Companion Volume*, pages 16:1–16:6. ACM, 2021.

[2] Benjamin Benni, Sébastien Mosser, Jean-Philippe Caissy, and Yann-Gaël Guéhéneuc. Can microservice-based online-retailers be used as an spl?: a study of six reference architectures. In Roberto Erick Lopez-Herrejon, editor, *SPLC '20: 24th ACM International Systems and Software Product Line Conference, Montreal, Quebec, Canada, October 19-23, 2020, Volume A*, pages 24:1–24:6. ACM, 2020.

[3] Eric Braun. *Microservice-based Reference Architecture for Semantics-aware Measurement Systems*. PhD thesis, Karlsruhe Institute of Technology, Germany, 2020.

[4] Antonio Brogi, Andrea Canciani, Davide Neri, Luca Rinaldi, and Jacopo Soldani. Towards a reference dataset of microservice-based applications. In Antonio Cerone and Marco Roveri, editors, *Software Engineering and Formal Methods - SEFM 2017 Collocated Workshops: DataMod, FAACS, MSE, CoSim-CPS, and FOCLASA, Trento, Italy, September 4-5, 2017, Revised Selected Papers*, volume 10729 of *Lecture Notes in Computer Science*, pages 219–229. Springer, 2017.

[5] Andreea Ancuta Corici, Tina Hühnlein, Detlef Hühnlein, and Olaf Rode. Towards interoperable vaccination certificate services. In Delphine Reinhardt and Tilo Müller, editors, *ARES 2021: The 16th International Conference on Availability, Reliability and Security, Vienna, Austria, August 17-20, 2021*, pages 139:1–139:9. ACM, 2021.

[6] Oliver Cvetkovski and Josef Spillner. A self-contained, decentralized and nation-scale approach to e-government services. 10th IEEE International Conference on Cloud Computing in Emerging Markets (CCEM) – Student Project Showcase, October 2021.

[7] Mauricio de Vasconcelos Barros, Frederico Schardong, and Ricardo Felipe Custódio. Leveraging self-sovereign identity, blockchain, and zero-knowledge proof to build a privacy-preserving vaccination pass. *CoRR*, abs/2202.09207, 2022.

[8] Peini Liu, Xinjun Mao, Shuai Zhang, and Fu Hou. Towards reference architecture for a multi-layer controlled self-adaptive microservice system. In Óscar Mortágua Pereira, editor, *The 30th International Conference on Software Engineering and Knowledge Engineering, Hotel Pullman, Redwood City, California, USA, July 1-3, 2018*, pages 236–235. KSI Research Inc. and Knowledge Systems Institute Graduate School, 2018.

[9] Manuel Ramírez López and Josef Spillner. Towards quantifiable boundaries for elastic horizontal scaling of microservices. In Ashiq Anjum, Alan Sill, Geoffrey C. Fox, and Yong Chen, editors, *Companion Proceedings of the 10th International Conference on Utility and Cloud Computing, UCC 2017, Austin, TX, USA, December 5-8, 2017*, pages 35–40. ACM, 2017.

[10] Marwin Shraideh, Patroklos Samaras, Maximilian Schreieck, and Helmut Krcmar. A microservice-based reference architecture for digital platforms in the proteomics domain. In Leona Chandra Kruse, Stefan Seidel, and Geir Inge Hausvik, editors, *The Next Wave of Sociotechnical Design - 16th International Conference on Design Science Research in Information Systems and Technology, DESRIST 2021, Kristiansand, Norway, August 4-6, 2021, Proceedings*, volume 12807 of *Lecture Notes in Computer Science*, pages 260–271. Springer, 2021.

[11] Hui Song, Phu Hong Nguyen, Franck Chauvel, Jens M. Glattetre, and Thomas Schjerpen. Customizing multi-tenant saas by microservices: A reference architecture. In Elisa Bertino, Carl K. Chang, Peter Chen, Ernesto Damiani, Michael Goul, and Katsunori Oyama, editors, *2019 IEEE International Conference on Web Services, ICWS 2019, Milan, Italy, July 8-13, 2019*, pages 446–448. IEEE, 2019.

[12] Matthias Vianden, Horst Lichter, and Andreas Steffens. Experience on a microservice-based reference architecture for measurement systems. In Sungdeok (Steve) Cha, Yann-Gaël Guéhéneuc, and

Gihwon Kwon, editors, *21st Asia-Pacific Software Engineering Conference, APSEC 2014, Jeju, South Korea, December 1-4, 2014. Volume 1: Research Papers*, pages 183–190. IEEE Computer Society, 2014.

[13] Li Wu, Johan Tordsson, Erik Elmroth, and Odej Kao. Microrca: Root cause localization of performance issues in microservices. In *NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium*, pages 1–9. IEEE, 2020.