Quality-based Methodology for Assessing the Applicability of Microservices Architecture

Andrés Nebel y Laura González

Instituto de Computación, Facultad de Ingeniería
Universidad de la República
J. H. y Reissig 565, Montevideo, Uruguay
{andres.nebel, lauragon}@fing.edu.uy

Resumen Microservices architecture is an approach for developing a system as a set of «small» independent services. Although this architecture has advantages (e.g. in terms of scalability, maintainability), it is not a general solution applicable to all scenarios. This paper proposes a quality-based methodology for assessing the applicability of a microservice architecture to build a system. The methodology includes an impact analisys of microservices on quality attributes that, together with the quality objectives of the system to be developed, constitute the basis for the assessment. The methodology is applied in a real-world case study for a middleware platform in the area of e-government, which provides an example of its use, as well as a first validation of the proposal.

Keywords: Microservices architecture · Quality attributes · e-government.

1. Introducción

La arquitectura de microservicios es un enfoque para construir sistemas que está ganando popularidad [22], y que proporciona ventajas en cuanto a escalabilidad y mantenibilidad, que resultan interesantes en varios escenarios. Aun así, sus propias características hacen que no sea aplicable a todos los contextos [3].

Por otro lado, reportes de la industria muestran que en ocasiones se decide utilizar microservicios sin un análisis adecuado [3,12,14], en particular, respecto al impacto sobre la calidad del sistema y a las consecuencias sobre los objetivos del negocio. Esto puede ocasionar que se adopte este enfoque incluso cuando no se adecua a las necesidades. Es de interés entonces contar con una forma sistemática de evaluar la aplicabilidad de microservicios para un contexto determinado.

Este artículo propone una metodología basada en atributos de calidad para evaluar la aplicabilidad de una arquitectura de microservicios para construir un sistema. La metodología se apoya en un estudio del impacto sobre atributos de calidad, elaborado también en este trabajo, que se contrasta con los objetivos de calidad del sistema a desarrollar. La metodología se aplica en un caso de estudio para una plataforma de middleware en el área de gobierno electrónico, permitiendo ejemplificar su uso y brindar una primera validación de la propuesta.

Nuestra propuesta se diferencia de trabajo existente por su metodología enfocada en atributos de calidad para evaluar la aplicabilidad de microservicios, así como por el número de atributos de calidad sobre los cuales se analiza el impacto de esta arquitectura. En [7] se utiliza ATAM (*Architecture Trade-off Analysis Method*) para evaluar la aplicabilidad y guiar las decisiones de migración hacia microservicios. Dicho trabajo reporta un esfuerzo considerable de costo y tiempo, mientras que nuestra propuesta acota el esfuerzo a identificar los objetivos de calidad y contrastarlos con los resultados de la Sección 3. En [23] el SEI (*Software Engineering Institute*) presenta un análisis donde clasifica el grado de satisfacción que provee SOA (*Service-oriented Architecture*) y tecnologías relacionadas sobre los atributos de calidad. En contraste, nuestra propuesta se orienta a microservicios, basándose en el concepto de preocupaciones de calidad y el impacto sobre éstas. Por otro lado, si bien existen trabajos [9,16] que realizan un análisis de impacto de microservicios sobre atributos de calidad, estos no abordan la misma cantidad de atributos y no consideran algunos trabajos relevantes en materia de desempeño [2,10,15,19,20,25] y testabilidad [9,17,26]. Otros trabajos [5] se enfocan en el impacto percibido y no en el real, como nuestra propuesta.

El resto del artículo se organiza de la siguiente manera. La Sección 2 presenta la metodología propuesta. La Sección 3 analiza el impacto de microservicios sobre la calidad de un sistema. La Sección 4 describe la aplicación de la metodología en un caso de estudio. La Sección 5 presenta conclusiones y trabajo a futuro.

2. Metodología de Evaluación de Aplicabilidad

Esta sección describe la metodología propuesta, la cual permite evaluar la aplicabilidad de una arquitectura de microservicios para construir un sistema. Para este trabajo se considera que dicha arquitectura es aplicable, cuando los beneficios sobre los objetivos de calidad del sistema resultan favorables aún considerando los posibles desafíos implicados.

La idea general de la metodología consiste en obtener los objetivos de calidad del sistema a construir, y analizar cómo son afectados por la utilización de microservicios. Dichos objetivos, se especifican como preocupaciones sobre atributos de calidad del estándar ISO/IEC 25010:2011. Un subatributo constituye una «preocupación» cuando es de interés particular de los *stakeholders*, o si se requieren consideraciones especiales para satisfacerlo.

Notar que el análisis requiere conocer cuál es el impacto de aplicar microservicios sobre los atributos de calidad. Dicho estudio constituye parte de los aportes de este trabajo y se describe en la Sección 3.

La metodología consiste de cuatro actividades principales:

1. Identificar Preocupaciones: En esta actividad, se deben realizar reuniones con los *stakeholders*, para identificar las preocupaciones de calidad en base a los objetivos de calidad del sistema a construir. El resultado es un listado de los subatributos de calidad que constituyen una preocupación para el sistema.

2. Determinar Impacto de Microservicios sobre Atributos de Calidad: En esta actividad se debe determinar el impacto de aplicar una arquitectura

Tabla 1: Impacto de microservicios sobre calidad del sistema

Atributos de Calidad del sistema		Arq. de Microservicios	Referencias consultadas	Preocupación Caso Estudio
Atributo	**Subatributo**			
Eficiencia de Desempeño	Comportamiento del Tiempo	??	[2,10,15,17,19,20,25,27,28,29]	P
	Uso de Recursos	A+(*)	[11,15,27]	P
	Capacidad	A+(*)	[11]	P
Compatibilidad	Interoperabilidad	A+	[6,8,11,22]	P
Mantenibilidad	Modularidad	A+	[6,8,11,22]	P
	Reusabilidad	A+	[6,8,11,22]	P
	Evaluabilidad	P	[6,8,11,22]	P
	Modificabilidad	A+	[6,8,11,22]	P
	Testabilidad	P	[9,11,17,26]	P
Portabilidad	Adaptabilidad	A+	[6,8,11,22]	P
	Instalabilidad	—	[4,6,8,11,22]	P
	Reemplazabilidad	A+	[6,8,11,22]	P
Confiabilidad	Madurez	—	[6,8,11,22]	P
	Disponibilidad	A+	[6,8,11,22,24]	P
	Tolerancia a Fallas	A+	[6,8,11,18,22,24]	P
	Recuperabilidad	A+	[6,8,18,24]	P

A+: Afecta positivamente. P: Implica una preocupación —: No afectado ??: No determinado (*): Solo si se cumplen ciertas condiciones

de microservicios sobre los atributos de calidad. El resultado de esta actividad constituye parte de los aportes de este trabajo y se describe en la Sección 3, indicando qué impacto tiene la arquitectura sobre cada subatributo de calidad.

3. Analizar Impacto de Microservicios sobre Preocupaciones: En esta actividad se toman los resultados anteriores para analizar si la arquitectura de microservicios beneficia o no a las preocupaciones de calidad planteadas. El resultado de esta actividad es un listado que combina y contrasta los resultados de las dos primeras actividades, mostrando para cada preocupación, si el subatributo es afectado positivamente o no al aplicar la arquitectura.

4. Determinar Aplicabilidad de Microservicios: En esta actividad se debe determinar la aplicabilidad de una arquitectura de microservicios para construir el sistema, en base al resultado de la tercera actividad. El resultado de esta actividad muestra en qué medida esta arquitectura es aplicable para el sistema.

3. Impacto de Microservicios sobre Atributos de Calidad

En esta sección se analiza el impacto teórico en base al estado del arte actual, de una arquitectura de microservicios sobre los atributos de calidad de un sistema, extendiendo lo presentado en [21]. La Tabla 1 resume los resultados del análisis, brindando referencias a trabajos consultados.

El análisis se basa en los atributos del estándar ISO/IEC 25010:2011, de los cuales algunos como *Usabilidad* se omiten, por no ser afectados por microservicios, y *Seguridad* no se incluye por cuestiones de alcance. Los subatributos pueden: i) no ser afectados, ii) ser afectados positivamente o iii) ser una preocupación, y iv) ser afectados si cierta condición se cumple.

Por otro lado, el impacto de microservicios se estudia como un conjunto de enfoques que incluye: i) el uso de contenedores, ii) la aplicación de DevOps y iii) el despliegue en una plataforma de orquestación de contenedores (p. ej. *Openshift*).

El motivo es que microservicios no se aplica generalmente en forma aislada, sino como una serie de enfoques que abordan problemáticas relacionadas.

3.1. Eficiencia de Desempeño

Comportamiento del tiempo: Se observa que en teoría podría ser una preocupación debido a: i) la redundancia introducida por la virtualización de los contenedores y redes superpuestas [19,20] (aunque contradiciendo [10] y [27]), y ii) debido a las implicancias de los sistemas distribuidos, ocasionadas entre otros por el aumento de las comunicaciones de red [17]. En [2,28,29] se presentan pruebas de rendimiento similares, que comparan aplicaciones análogas de microservicios y monolíticas, donde se muestran tiempos de respuesta levemente mejores para la arquitectura monolítica. Sin embargo, dichos resultados no coinciden con los de [15] y [25] donde se obtienen mejores tiempos de respuesta con microservicios en pruebas similares. En resumen, los resultados de trabajos actuales no permiten concluir que este subatributo sea una preocupación.

Uso de recursos: Es afectado positivamente cuando se requiere escalamiento, ya que se escalan solo los microservicios requeridos, y no todo el sistema [11]. El impacto es mejor al utilizar patrones de diseño de microservicios [15] y al usar contenedores [27] y plataformas de orquestación de contenedores, que proporcionan escalamiento elástico. Si no se requiere escalamiento, la arquitectura de microservicios podría utilizar más recursos que la monolítica, debido a la redundancia de desplegar dependencias compartidas y componentes usados en sistemas distribuidos (p. ej. balanceadores). Esta redundancia se compensa al escalar, dado que los sistemas monolíticos escalan todo el sistema.

Capacidad: Tiene un impacto análogo al de Uso de Recursos, ya que si éste se beneficia se podrá satisfacer de mejor forma los requerimientos de capacidad.

3.2. Compatibilidad y Mantenibilidad

Interoperabilidad, **Modularidad**, **Reusabilidad**, **Modificabilidad:** Se afectan positivamente por las características que definen a microservicios [6,8,11,22].

Evaluabilidad: Constituye una preocupación. Si la cantidad de microservicios es alta, puede ser complejo determinar el impacto de una modificación, por la dificultad para determinar qué microservicios son afectados por el cambio. Existen herramientas como *DeployHub* que disminuyen en parte la complejidad al permitir visualizar las interacciones entre servicios y sus versiones.

Testabilidad: Constituye una preocupación. Si bien se simplifican las pruebas unitarias por la modularidad (y usualmente por automatización), las pruebas integrales son más complejas por haber varias partes involucradas [9]. Además, la distribución dificulta identificar dónde se originan errores no triviales. En [26] se muestran informes de la industria que reportan el testing como una preocupación de los desarrolladores respecto al enfoque, lo cual es consistente con lo anteriormente mencionado. Fowler advierte que el testing es más complejo [11], aunque alega que existen algunos enfoques para satisfacer el atributo [17].

3.3. Portabilidad

Adaptabilidad: Es afectada positivamente. Los contenedores aíslan las problemáticas de dependencias, temas de *hardware* y sistemas operativos, mejorando la adaptabilidad. Además, en el estándar ISO/IEC 25010:2011 el subatributo incluye la escalabilidad, la cual es beneficiada [11]. El impacto se puede mejorar si el sistema se despliega en una plataforma de orquestación de contenedores, donde el escalamiento elástico es manejado automáticamente.

Instalabilidad: Si bien la separación de un sistema en varias partes agrega complejidad operacional, existen enfoques y herramientas para satisfacer el subatributo, por lo cual no se lo considera una preocupación de calidad. Aún así, no puede afirmarse que la instalabilidad sea afectada positivamente.

Por un lado, *DevOps* beneficia este subatributo, uniendo responsabilidades sobre las mismas personas, y apoyándose en herramientas de entrega continua. Por otro lado, las plataformas de orquestación de contenedores mejoran la instalabilidad, al automatizar despliegues y gestión de componentes utilitarios.

Reemplazabilidad: Es afectada positivamente. Los microservicios son fáciles de sustituir, permitiendo el reemplazo sin desplegar el sistema entero [11].

3.4. Confiabilidad

Tolerancia a Fallas: Es afectada positivamente. Si bien los sistemas distribuidos agregan problemas de arrastre de fallas y más comunicaciones de red que pueden fallar, existe una amplia cobertura sobre el tema [6,8,24] y soluciones varias que permiten satisfacer el subatributo.

Los sistemas basados en microservicios no poseen un único punto de falla, lo cual permite aislar y controlar mejor las mismas al usarse patrones como *Circuit Breaker* y *Bulkhead*, los cuales suelen complementarse con políticas de expiración de pedidos (i.e. *timeouts*) y de reintentos, beneficiando el subatributo [18]. En el aspecto tecnológico bibliotecas como *Netflix Hystrix* y *Polly* aportan funcionalidades para tolerancia a fallas y latencia en sistemas distribuidos.

Recuperabilidad: Análogo al subatributo anterior, es afectada positivamente, ya que existen enfoques y tecnología para satisfacerla. Los enfoques ya mencionados en tolerancia a fallas, aportan también a la recuperabilidad: i) el patrón *Circuit Breaker* permite recuperar la operatividad normal, ii) el uso de *timeouts* y reintentos brinda recuperabilidad frente a fallas de latencia o de red.

Por otro lado, el patrón *health check* puede usarse para detectar si los servicios están disponibles [24] y abortar en caso necesario la operación; también para medir si el servicio está operativo, reiniciando en caso contrario su contenedor. Esto es posible debido a que: i) los microservicios no suelen poseer estado [18] y ii) el tiempo de inicio de los microservicios y contenedores es rápido [18].

Disponibilidad: Por las razones vistas para los dos subatributos anteriores, la disponibilidad es afectada positivamente. Además, el patrón *health check* contribuye a mantener la alta disponibilidad (i.e. durante el despliegue permite saber cuándo la nueva instancia está disponible, para no remover la versión anterior hasta ese momento y no tener tiempos de baja). Esto es usado por plataformas de orquestación de contenedores, como *OpenShift*.

4. Caso de Estudio: Plataforma Middleware e-Gobierno

La Plataforma de Interoperabilidad (PDI) apunta a facilitar y promover la implementación de servicios de gobierno electrónico en Uruguay y fue puesta en marcha por la agencia de gobierno electrónico de este país (AGESIC). Uno de sus componentes es la Plataforma de Middleware (PM) [13], que se utiliza por los organismos para publicar y consumir servicios. La PM se implementó inicialmente con un *Enterprise Service Bus* (ESB) [13] con arquitectura monolítica y se migró recientemente a microservicios, lo cual resultó beneficioso [1].

La aplicación de la metodología para la PM consistió en (cf. Sección 2):

1. Identificar Preocupaciones. Se elaboró un listado inicial de atributos de calidad que se entendió constituyen preocupaciones para la PM, en base a: i) las características de la PM, ii) el conocimiento de los autores sobre la PM, y iii) documentación [13]. El listado se presentó y enriqueció con referentes de la PM, dando como resultado lo presentado en la última columna de la Tabla 1.

Por ejemplo, el Comportamiento del Tiempo y la Disponibilidad constituyen preocupaciones dado el tipo de sistemas que se apoyan en la PM (p. ej. Historia Clínica Electrónica Nacional). La Capacidad y Adaptabilidad (que incluye escalabilidad) también se identificaron como preocupaciones, dado que la PM procesa todas las interacciones entre servicios de los organismos, los cuales pueden tener picos de utilización. La Modularidad e Interoperabilidad también se identificaron como preocupaciones, dado que la PM incluye componentes (p. ej. servicio de ruteo) que pueden ser implementados por distintos proveedores así como con distintas tecnologías y pueden tener que interactuar entre sí.

2. Determinar Impacto de Microservicios. Se utiliza el resultado del estudio descripto en la Sección 3 (cf. tercera columna de Tabla 1).

3. Analizar Impacto de Microservicios sobre Preocupaciones. Para esta actividad se contrastan la tercer y última columna de la Tabla 1, con el fin de conocer el impacto que tiene la arquitectura de microservicios sobre las preocupaciones de calidad de la PM. Se observa que microservicios tiene un impacto positivo para la mayoría (once de dieciseis) de las preocupaciones de calidad de la PM. Además para dos de las preocupaciones (Madurez e Instalabilidad) el atributo no se ve afectado y para una de ellas (Comportamiento del Tiempo) el impacto no está determinado. Sólo dos de las preocupaciones de la PM son también una preocupación al utilizar una arquitectura de microservicios.

4. Determinar Aplicabilidad de Microservicios. Si bien existen dos preocupaciones de la PM que son también preocupaciones para microservicios, se entiende que los beneficios a la calidad de la PM son ampliamente favorables. En particular, la arquitectura de microservicios afecta positivamente once de las preocupaciones de la PM, varias de las cuales son de alta prioridad. En base a esto, se determina que la arquitectura de microservicios es aplicable para la PM.

Resumen de la Aplicación de la Metodología en el Caso de Estudio. La aplicación de la metodología permitió ejemplificar su uso y determinar que la arquitectura de microservicios es aplicable para la PM. También permitió comprobar que para el caso de estudio los resultados concuerdan con lo reportado por la AGESIC [1], y con lo reportado en las reuniones con referentes de la PM.

El caso de estudio también identificó que el impacto teórico de microservicios (cf. Sección 2), se corresponde con el impacto real que la arquitectura tuvo sobre varios de los atributos de calidad de la PM. En particular, en base a la reunión con referentes de la PM se supo que microservicios afectó positivamente el Uso de Recursos, la Modularidad y la Adaptabilidad (en particular, la escalabilidad).

5. Conclusiones y Trabajo Futuro

Este artículo propone una metodología basada en calidad, que permite evaluar la aplicabilidad de una arquitectura de microservicios para un sistema. La metodología define actividades, que buscan contrastar las preocupaciones de calidad de los *stakeholders* con el impacto de microservicios. La metodología se utiliza en un caso de estudio real, lo cual permite ejemplificar su aplicación y brindar una primera validación de la propuesta.

Los principales aportes de este trabajo son la metodología y su aplicación a un caso de estudio real, el estudio del impacto de microservicios sobre los atributos de calidad, y la presentación unificada de trabajos en esta área.

Como conclusiones generales, percibimos interés en metodologías para evaluar la aplicabilidad de microservicios y en su impacto sobre la calidad. El caso de estudio permitió comprobar la adecuación de la propuesta a contextos reales y contar con una primera validación.

Como trabajo a futuro se plantea incorporar el atributo de seguridad, soportar la priorización y ponderación de preocupaciones en el análisis de impacto, e incorporar aspectos económicos. Se apunta también a aplicar la metodología en otros contextos reales y a generalizarla para otro tipo de evaluaciones.

Agradecimientos. Los autores agradecen a AGESIC por las reuniones mantenidas, las cuales permitieron el desarrollo del caso de estudio.

Referencias

1. Actualización tecnológica de la plataforma de interoperabilidad. https://centroderecursos.agesic.gub.uy/web/arquitectura-de-gobierno/noticia-actualizacion-tecnologica-de-la-plataforma-de-interoperabilidad-publico.
2. Al-Debagy, O., Martinek, P.: A comparative review of microservices and monolithic architectures. CoRR **abs/1905.07997** (2019)
3. Balalaie, A., Heydarnoori, A., Jamshidi, P.: Migrating to cloud-native architectures using microservices: An experience report (2015)
4. Balalaie, A., Heydarnoori, A., Jamshidi, P.: Microservices architecture enables devops: Migration to a cloud-native architecture. IEEE Software **33**(3), 42–52 (2016)
5. Bogner, Fritzsch, et al.: Microservices in industry: Insights into technologies, characteristics, and software quality. In: IEEE ICSA-C (Mar 2019)
6. Carneiro, C., Schmelmer, T.: Microservices From Day One : Build Robust and Scalable Software From the Start. Apress (2016)
7. Cruz, Astudillo, Hilliard, Collado: Assessing migration of a 20-year-old system to a micro-service platform using atam. In: IEEE ICSA-C (Mar 2019)

8. Daya, Duy, V., et al.: Microservices from Theory to Practice: Creating Applications in IBM Bluemix Using the Microservices Approach. IBM Redbooks (2016)
9. Dragoni, Giallorenzo, et al.: Microservices: yesterday, today, and tomorrow. CoRR (2016)
10. Felter, W., Ferreira, A., et al.: An updated performance comparison of virtual machines and linux containers. In: IEEE ISPASS (Mar 2015)
11. Fowler, M., Lewis, J.: Microservices. 2014
12. Fritzsch, Bogner, et al.: Microservices migration in industry: Intentions, strategies, and challenges. CoRR **abs/1906.04702** (2019)
13. González, Ruggia, et al.: A service-oriented integration platform to support a joined-up e-government approach: The uruguayan experience. In: EGOVIS/EDEM. Lecture Notes in Computer Science, vol. 7452, pp. 140–154. Springer Berlin Heidelberg (2012)
14. Gouigoux, J., Tamzalit, D.: From monolith to microservices: Lessons learned on an industrial migration to a web oriented architecture. In: 2017 IEEE International Conference on Software Architecture Workshops (ICSAW). pp. 62–65 (Abr 2017)
15. Guaman, D., Yaguachi, L., et al.: Performance evaluation in the migration process from a monolithic application to microservices. In: 13th Iberian Conference on Information Systems and Technologies (Jun 2018)
16. H. Zhang, S. Li, J.S.Z.J.Z.L.: Understanding quality attributes in microservice architecture. Tech. rep., Nanjing University, Nanjing, China (Ene 2018)
17. Hassan, S., Bahsoon, R.: Microservices and their design trade-offs: A self-adaptive roadmap. IEEE International Conference on Services Computing (SCC) **00** (2016)
18. Indrasiri, K., Siriwardena, P.: Microservices for the Enterprise : Designing, Developing, and Deploying. Apress (2018)
19. Kratzke, N.: About microservices, containers and their underestimated impact on network performance. CoRR **abs/1710.04049** (2017)
20. Lloyd, W., Ramesh, S., et al.: Serverless computing: An investigation of factors influencing microservice performance. In: 2018 IEEE International Conference on Cloud Engineering (IC2E) (Abr 2018)
21. Nebel, A.: Arquitectura de microservicios para plataformas de integración (Dic 2018), tesis de Maestría en Ingeniería de Software
22. Newman, S.: Building Microservices: Designing Fine-Grained Systems. O'Reilly Media (2015)
23. O'Brien, Bass, Merson: Quality attributes and service-oriented architectures. Tech. Rep. SEI-2005-TN-014, SEI, Carnegie Mellon University, Pittsburgh, USA (2005)
24. Richardson, C.: Pattern: Microservice Architecture. [En línea] Disponible en: http://microservices.io/patterns/microservices.html [Accedido: 01-06-2017]
25. Singh, V., Peddoju, S.K.: Container-based microservice architecture for cloud applications. In: 2017 ICCCA (May 2017)
26. Soldani, J., Tamburri, D., Heuvel, W.J.: The pains and gains of microservices: A systematic grey literature review. Journal of Systems and Software **146** (Sep 2018)
27. Soltesz, Pötzl, et al.: Container-based operating system virtualization: A scalable, high-performance alternative to hypervisors. SIGOPS Oper. Syst. Rev. **41**(3) (Mar 2007)
28. Ueda, T., Nakaike, T., Ohara, M.: Workload characterization for microservices. In: IEEE International Symposium on Workload Characterization (Sep 2016)
29. Villamizar, Garces, et al.: Evaluating the monolithic and the microservice architecture pattern to deploy web applications in the cloud. In: 10th Colombian Computing Conference. UNIANDES (2015)