Modelling and Analysis of Microservice Based Systems by Composing Petri Nets

Görkem Kılınç Soylu[1] and Luca Bernardinello[2]

[1] Izmir University of Economics, Turkey
gorkem.soylu@ieu.edu.tr
[2] University of Milano-Bicocca, Italy
luca.bernardinello@unimib.it

1 Microservice-based systems

The rapid advancements in cloud technology and the growing demand for continuous digital transformation have led to the rise of modern applications that are responsive, flexible, and reliable. To meet these needs, new software design paradigms have emerged, notably Microservice-based Architecture (MSbA) [9, 4]. MSbA involves breaking down an application into loosely coupled, independently deployable services, known as microservices. This modular approach enhances agility, scalability, and fault tolerance in applications [1, 5].

While the Microservice-based Architecture (MSbA) offers numerous advantages for software engineering, such as flexibility and scalability, it also introduces significant challenges due to its complex and distributed nature. These challenges, particularly in areas like modelling, testing, and verification, are highlighted in [10, 11].

2 Utilising Petri nets for modelling microservice-based systems

Formal methods, such as Petri nets [8, 7], offer a robust solution to these challenges by providing a rigorous and systematic approach to modelling and analyzing microservice-based systems. These methods are beneficial because they enable the identification of potential errors or vulnerabilities in the system design before implementation, thus saving time and resources. Formal methods ensure system correctness, provided the application adheres to the model, and assist in generating test cases by computing reachable states and identifying critical ones.

In [6], a case study is performed to demonstrate the suitability and usefulness of Petri nets to model and analyse microservice based systems. In the study, a Banking-as-a-Service (BaaS) application was modelled via 1-safe Petri nets, a class of Petri nets in which each place can have at most 1 token. The model was then analysed to verify some structural and behavioural properties. The study follows an approach based on abstraction and composition. A microservice based software system can include many microservices working together and

communicating in different ways. However, the internal details of a service are not known by the other services. Thus the internal details of a specific service can be abstracted from the overall view while focusing on the communication and collaboration among the services in the whole system.

The case study draws a sketch for a compositional method while showing the use of Petri nets for modelling and analysis of microservice based systems to verify some structural and behavioural properties. While the case study provides valuable insights and demonstrates the practical applicability of 1-safe Petri nets in this context, our objective is to extend and refine this work and provide a more detailed and sound framework for the abstraction and composition of microservice models.

3 A compositional approach based on regions

The approach we propose here rests on a notion of abstraction and composition of Petri nets, based on regions. A *region* of a labelled transition system is a subset of states, r such that, for each label, there is a fixed crossing relation: for a given label x, either all arcs labelled by x enter r, or all those arcs leave r, or no arc crosses the border of r (see [2]).

The marking graph of a 1-safe Petri net is a labelled transition system, where markings correspond to states. The set of markings in which a given place is marked is a region; the marking graph can have regions which do not correspond to actual places; those regions are *potential places*, in the sense that one can add new places to the net, corresponding to such regions, without changing the overall behaviour of the net.

In particular, there can be regions corresponding to the logical disjunction of existing places, seen as propositions. Exploiting this fact, one can define a notion of abstraction of a net, or of a part of a net, by keeping some "coarse" places, and masking some more detailed ones.

Abstraction and composition based on regions, for elementary net systems, are defined and studied in [3], where it is shown that some behavioural properties are preserved, or reflected, by abstraction or composition, in particular with respect to invariants and bisimulation relations. We plan to extend those results to 1-safe Petri nets, which are a generalization of elementary net systems.

The approach presented in [6] can then be made more general and flexible. A specific microservice can be modelled in detail by a 1-safe Petri net, where any assumptions about the behaviour of the environment of the service are modelled without details, but still respecting behavioural features of implementations. This "environment" may include other microservices, users, and other components of the real system. It is then possible to explore the behaviour of one, or more, microservices with respect to different assumptions about the operating environment, and formalize precisely when a given implementation of the environment satisfies those assumptions.

References

1. Alshuqayran, N., Ali, N., Evans, R.: A systematic mapping study in microservice architecture. In: 2016 IEEE 9th International Conference on Service-Oriented Computing and Applications. pp. 44–51. IEEE (Nov 2016). https://doi.org/10.1109/SOCA.2016.15
2. Badouel, É., Bernardinello, L., Darondeau, P.: Petri Net Synthesis. Texts in Theoretical Computer Science. An EATCS Series, Springer (2015). https://doi.org/10.1007/978-3-662-47967-4, https://doi.org/10.1007/978-3-662-47967-4
3. Bernardinello, L., Monticelli, E., Pomello, L.: On preserving structural and behavioural properties by composing net systems on interfaces. Fundam. Informaticae **80**(1-3), 31–47 (2007), http://content.iospress.com/articles/fundamenta-informaticae/fi80-1-3-03
4. Bonér, J.: Reactive Microservices Architecture. O'Reilly Media, Inc. (2016)
5. Di Francesco, P., Lago, P., Malavolta, I.: Architecting with microservices: A systematic mapping study. Journal of Systems and Software **150**, 77–97 (2019). https://doi.org/https://doi.org/10.1016/j.jss.2019.01.001, https://www.sciencedirect.com/science/article/pii/S0164121219300019
6. Kılınç Soylu, G., Demirörs, O.: An exploratory case study: Using Petri nets for modelling microservice-based systems. In: 49th Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2023, Durres, Albania, September 6-8, 2023. pp. 254–261. IEEE (2023). https://doi.org/10.1109/SEAA60479.2023.00047, https://doi.org/10.1109/SEAA60479.2023.00047
7. Murata, T.: Petri nets: Properties, analysis and applications. Proc. IEEE **77**(4), 541–580 (1989). https://doi.org/10.1109/5.24143
8. Petri, C.A.: Kommunication mit Automaten. Ph.D. thesis, Technischen Hoschule Darmstadt (1962)
9. Thönes, J.: Microservices. IEEE Software **32**(1), 116–116 (2015). https://doi.org/10.1109/MS.2015.11
10. Ünlü, H., Bilgin, B., Demirörs, O.: A survey on organizational choices for microservice-based software architectures. Turkish J. Electr. Eng. Comput. Sci. **30**(4), 1187–1203 (2022). https://doi.org/10.55730/1300-0632.3843, https://doi.org/10.55730/1300-0632.3843
11. Ünlü, H., Kennouche, D.E., Kılınç Soylu, G., Demirörs, O.: Microservice-based projects in agile world: A structured interview. Inf. Softw. Technol. **165**, 107334 (2024). https://doi.org/10.1016/J.INFSOF.2023.107334, https://doi.org/10.1016/j.infsof.2023.107334