Applying Reconfiguration Cost and Control Pattern Modeling to Self-Adaptive Systems

Michael Matthé
Chair of Information Systems II, University of Mannheim, Germany
michael.matthe@uni-mannheim.de

ABSTRACT

Self-adaptive systems have become a popular research topic to overcome challenges of developing highly complex, interconnected, and heterogeneous systems and networks. These systems aim to autonomously adapt to a changing environment by adapting system behavior or composition to improve performance. Many self-adaptive systems are designed in a purely reactive way and without considering costs that may be incurred by performing adaptation. This thesis therefore aims to develop an approach for proactive self-adaptive systems and evaluate the impact of reconfiguration cost and proactive adaptation in an edge computing system. Additionally the autonomous adaptation of different control patterns for centralized or decentralized control will be explored and evaluated. This thesis proposes to extend feature models, as used in dynamic software product lines, with modeling for reconfiguration cost and of uncertainty in the system's environment.

CCS CONCEPTS

• **Software and its engineering → Abstraction, modeling and modularity**;

KEYWORDS

self-adaptive systems, context-aware systems, edge computing

ACM Reference Format:
Michael Matthé. 2022. Applying Reconfiguration Cost and Control Pattern Modeling to Self-Adaptive Systems. In *44th International Conference on Software Engineering Companion (ICSE '22 Companion), May 21–29, 2022, Pittsburgh, PA, USA.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3510454.3517056

1 INTRODUCTION

Many modern information systems are composed of interconnected, distributed, and heterogeneous devices operating in an environment with fluctuating network resources and availability [11]. Users expect high quality of service from these devices with potentially varying performance goals over time. The process of manually adapting systems in order to achieve these performance goals is often error prone and time consuming [20]. Therefore, self-adaptive systems aim to alleviate these challenges by allowing systems to perform adaptations autonomously. The system monitors its environment and system state in order to analyze the need for adaptations, which are either necessary to keep the system functioning properly or are able to improve system performance in regard to a certain performance goal [3]. The managed resource itself is extended by a managing subsystem (adaptation logic) which performs the monitoring of the environment, as well as the decision making about necessary adaptations [11].

Coordination between different systems in a distributed network is an important task in order to support global optimization efficiently. In distributed and mobile systems the utilization of a central entity for controlling and coordinating all adaptations of the participating systems is not always feasible [21]. A network of systems without central control where each system strives for, potentially greedy, optimization of local goals may not lead to a globally optimized system. A variety of decentralized control patterns have been proposed by Weyns et al. [21]. The control patterns range from fully centralized control, where one adaptation logic manages all resources, to fully decentralized control, where each managed resource has its own adaptation logic. Additionally, the adaptation logic may be constructed in a hierarchical hybrid pattern.

The traditional method of implementing self-adaptive systems is a reactive approach, adapting to changes based on event triggers in the monitored environment [11]. Each adaptation, however, may come at some cost, e.g. an energy cost on a battery powered device or a delay in communication.

Proactive systems on the other hand are equipped to anticipate or predict changes in the system's environment. By predicting the future environment and state of the system over a longer time horizon, a proactive self-adaptive system is able to reduce the number of non-optimal adaptations performed and thereby reduce the incurred cost of a potential reconfiguration. Additionally, the adaptation of one system may have an influence on the performance of another system operating in the same network. Therefore, coordinating the adaptations of a network of systems cooperatively and proactively increases the global utility and its stability.

2 RESEARCH PROBLEM AND EXPECTED CONTRIBUTIONS

Self-adaptive system are typically realized with closed loop control, e.g., with the well established MAPE-K loop [10]. The MAPE-K loop consists of four phases: (i) monitoring of the system and its environment, (ii) analyzing the collected information and the need for adaptation, (iii) planning the adaptation if it is needed, and (iv) executing the adaptation. Additionally, the knowledge component (K) contains a model of the system's state and properties, as well as its environment (context). Feature models [8] are one method of modeling the configuration space of a system. Additionally, feature

This work is licensed under a Creative Commons Attribution International 4.0 License.

ICSE '22 Companion, May 21–29, 2022, Pittsburgh, PA, USA
© 2022 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9223-5/22/05.
https://doi.org/10.1145/3510454.3517056

models may be extended by a context branch which includes the modeling of the systems environment [17]. The resulting context feature model of a self-adaptive system may then consist of mandatory and optional system and context features as well as constraints between features from different branches of the model.

Context feature models may be used to express the currently selected configuration of the system and environment or the configuration space of all valid configurations. Therefore, they provide a useful method for evaluating the validity of configurations but lack expressiveness for future environment behavior and the cost of adaptation between two valid configurations. This thesis therefore proposes to extend these feature models to model a future time horizon of the system's environment as well as the incurred costs when switching between different configurations. When a high reconfiguration cost results from adapting the system, a more stable configuration that regards the future development of the system's environment may perform better over a longer time interval.

Firstly, this thesis aims to utilize Markov decision processes to model the uncertainty in the system's environment over a longer time horizon, as previously proposed in [13]. This enables proactive adaptations to be performed by the system. RQ1 - How can we extend feature model based self-adaptive systems to allow proactive system adaptation?

Secondly, this thesis aims to extend feature models by annotating them with the reconfiguration cost from one configuration to another. In some applications frequent switching between configurations without regarding the incurred cost of the adaptation may not lead to optimal performance. RQ2 - How can we extend feature models to express the costs of switching between different configurations?

Finally, we aim to incorporate the adaptation of control patterns into the approach. In the edge computing application we aim to use for the evaluation the choice of control pattern has a significant impact on the performance, such as the latency of task execution. Therefore, an adaptive approach of the control pattern, depending on system load and application requirements, is able to perform better.

3 RELATED WORK

In [19] we developed an approach modeling the system and environment state with context feature models, in addition to performance goals and performance-influence models, as part of the knowledge component to find optimal system configurations for a wireless sensor network. Pfannemüller et al. [16] propose a model-based runtime environment for implementing self-adaptive capabilities in communication systems by using Clafer models for modeling adaptation options and using a target system specification for defining the solution space of the system. The system does not take into account cost for a reconfiguration of the system and performs adaptations reactively.

Moreno et al. [14] propose proactive latency-aware adaptation based on Markov decision processes for probabilistic modeling of the uncertainty in the system and its environment. There are a large variety of application specific proactive approaches, e.g. in auto-scaling for cloud computing [7], [1], [9], that employ Markov decision processes and reinforcement learning for enabling proactive system behavior.

Van der Donckt et al. [4] propose a cost-benefit analysis at runtime for self-adaptive systems in an Internet of Things application. The approach uses weighted utilities for modeling benefits and domain-specific properties for defining costs of adaptation decisions. The authors note that a further analysis of different types of potential costs that may apply in self-adaptive systems is required as well as methods for modeling different types of costs.

The authors of [15] estimate the volatility of tactics for adapting a system by using time series forecasting. In [12] a MAPE specification language is extended with annotations to allow for the evaluation of adaptation costs such as latency. Additionally, Filieri [6] proposes an approach for run-time efficient verification of nonfunctional properties and gives outlook of future improvements by mentioning the addition of reward models for supporting cost metrics. This thesis proposes to use annotated feature models to allow the modelling of different systems and potential costs during a reconfiguration in a consistent and reproducible manner.

4 EVALUATION OF RESULTS

We plan to initially evaluate the results of the developed model enhancements in an edge computing system, namely the Tasklet system [18]. The system consists of a network of resource providers, providing their computational resources to other participants, and resource consumers, offloading computational tasks to providers for remote calculation. The system can operate using a central broker for task scheduling, or using a decentralized approach [5], allowing consumers to chose a producer for performing the task calculation themselves. Different applications offloading tasks can have differing non-functional requirements such as latency or reliability. We aim to evaluate our extension to the DSPL based self-adaptive approach by employing it in the autonomous selection of decentralization mechanism, taking into account reconfiguration costs of switching between different centralized or decentralized scheduling strategies.

Another aspect of task offloading in the Tasklet system is data placement for certain applications, such as image rendering or face recognition, which require additional data, such as a learned model, in order to perform the offloaded computation. When offloading a task to a resource provider, it first needs to obtain the required data before starting the calculation and returning the result. By proactively placing the required data set on one or multiple resource providers, the round-trip time from offloading a task to receiving the result can be significantly reduced [2]. Currently this is data-placement is not performed adaptively. We aim to evaluate the proactive capabilities of our approach by using it to add proactive self-adaptation in the computational offloading application. The participating systems should be able to adapt their specific data-placement mechanisms depending on the current and future predicted system load, performance, and composition.

ACKNOWLEDGMENTS

This work has been funded by the German Research Foundation (DFG) as part of project A4 within the Collaborative Research Center (CRC) 1053 - MAKI.

REFERENCES

[1] JV Bibal Benifa and D Dejey. 2019. Rlpas: Reinforcement learning-based proactive auto-scaler for resource provisioning in cloud environment. *Mobile Networks and Applications* 24, 4 (2019), 1348–1363.

[2] Martin Breitbach, Dominik Schäfer, Janick Edinger, and Christian Becker. 2019. Context-aware data and task placement in edge computing environments. In *2019 IEEE International Conference on Pervasive Computing and Communications (PerCom*. IEEE, 1–10.

[3] Rogério de Lemos, Holger Giese, Hausi A. Müller, and Mary Shaw et al. 2010. Software Engineering for Self-Adaptive Systems: A Second Research Roadmap. In *Software Engineering for Self-Adaptive Systems II - International Seminar, Dagstuhl Castle, Germany, October 24-29, 2010 Revised Selected and Invited Papers (Lecture Notes in Computer Science, Vol. 7475)*. Springer, 1–32. https://doi.org/10.1007/978-3-642-35813-5_1

[4] M. Jeroen Van Der Donckt, Danny Weyns, M. Usman Iftikhar, and Ritesh Kumar Singh. 2018. Cost-Benefit Analysis at Runtime for Self-adaptive Systems Applied to an Internet of Things Application. In *Proceedings of the 13th International Conference on Evaluation of Novel Approaches to Software Engineering, ENASE 2018, Funchal, Madeira, Portugal, March 23-24, 2018*, Ernesto Damiani, George Spanoudakis, and Leszek A. Maciaszek (Eds.). SciTePress, 478–490. https://doi.org/10.5220/0006815404780490

[5] Janick Edinger, Mamn Breitbach, Niklas Gabrisch, Dominik Schäfer, Christian Becker, and Amr Rizk. 2021. Decentralized low-latency task scheduling for ad-hoc computing. In *2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, 776–785.

[6] Antonio Filieri. 2011. QoS verification and model tuning @ runtime. In *SIGSOFT/FSE'11 19th ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE-19) and ESEC'11: 13th European Software Engineering Conference (ESEC-13), Szeged, Hungary, September 5-9, 2011*, Tibor Gyimóthy and Andreas Zeller (Eds.). ACM, 408–411. https://doi.org/10.1145/2025113.2025176

[7] Indu John, Aiswarya Sreekantan, and Shalabh Bhatnagar. 2019. Auto-scaling Resources for Cloud Applications using Reinforcement learning. In *2019 Grace Hopper Celebration India (GHCI)*. IEEE, 1–5.

[8] Kyo C Kang, Sholom G Cohen, James A Hess, William E Novak, and A Spencer Peterson. 1990. *Feature-oriented domain analysis (FODA) feasibility study.* Technical Report. Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst.

[9] Sara Kardani-Moghaddam, Rajkumar Buyya, and Kotagiri Ramamohanarao. 2020. ADRL: A Hybrid Anomaly-Aware Deep Reinforcement Learning-Based Resource Scaling in Clouds. *IEEE Transactions on Parallel and Distributed Systems* 32, 3 (2020), 514–526.

[10] Jeffrey O Kephart and David M Chess. 2003. The vision of autonomic computing. *Computer* 36, 1 (2003), 41–50.

[11] Christian Krupitzer, Felix Maximilian Roth, Sebastian VanSyckel, Gregor Schiele, and Christian Becker. 2015. A survey on engineering approaches for self-adaptive systems. *Pervasive Mob. Comput.* 17 (2015), 184–206. https://doi.org/10.1016/j.pmcj.2014.09.009

[12] Raffaela Mirandola, Elvinia Riccobene, and Patrizia Scandurra. 2019. Self-accounting in architecture-based self-adaptation. In *Proceedings of the 13th European Conference on Software Architecture, ECSA 2019, Paris, France, September 9-13, 2019, Companion Proceedings (Proceedings Volume 2),*, Laurence Duchien, Anne Koziolek, Raffaela Mirandola, Elena Maria Navarro Martínez, Clément Quinton, Riccardo Scandariato, Patrizia Scandurra, Catia Trubiani, and Danny Weyns (Eds.). ACM, 14–17. https://doi.org/10.1145/3344948.3344957

[13] P Read Montague. 1999. Reinforcement learning: an introduction, by Sutton, RS and Barto, AG. *Trends in cognitive sciences* 3, 9 (1999), 360.

[14] Gabriel A. Moreno, Javier Cámara, David Garlan, and Bradley R. Schmerl. 2018. Flexible and Efficient Decision-Making for Proactive Latency-Aware Self-Adaptation. *ACM Trans. Auton. Adapt. Syst.* 13, 1 (2018), 3:1–3:36. https://doi.org/10.1145/3149180

[15] Jeffrey Palmerino, Qi Yu, Travis Desell, and Daniel E. Krutz. 2019. Improving the Decision-Making Process of Self-Adaptive Systems by Accounting for Tactic Volatility. In *34th IEEE/ACM International Conference on Automated Software Engineering, ASE 2019, San Diego, CA, USA, November 11-15, 2019*. IEEE, 949–961. https://doi.org/10.1109/ASE.2019.00092

[16] Martin Pfannemüller, Martin Breitbach, Christian Krupitzer, Markus Weckesser, Christian Becker, Bradley R. Schmerl, and Andy Schürr. 2020. REACT: A Model-Based Runtime Environment for Adapting Communication Systems. In *IEEE International Conference on Autonomic Computing and Self-Organizing Systems, ACSOS 2020, Washington, DC, USA, August 17-21, 2020*. IEEE, 65–74. https://doi.org/10.1109/ACSOS49614.2020.00027

[17] Karsten Saller, Malte Lochau, and Ingo Reimund. 2013. Context-aware DSPLs: model-based runtime adaptation for resource-constrained systems. In *Proceedings of the 17th International Software Product Line Conference co-located workshops*. 106–113.

[18] Dominik Schafer, Janick Edinger, Justin Mazzola Paluska, Sebastian VanSyckel, and Christian Becker. 2016. Tasklets:" better than best-effort" computing. In *2016 25th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 1–11.

[19] Markus Weckesser, Roland Kluge, Martin Pfannemüller, Michael Matthé, Andy Schürr, and Christian Becker. 2018. Optimal reconfiguration of dynamic software product lines based on performance-influence models. In *Proceeedings of the 22nd International Systems and Software Product Line Conference - Volume 1, SPLC 2018, Gothenburg, Sweden, September 10-14, 2018*. ACM, 98–109. https://doi.org/10.1145/3233027.3233030

[20] Danny Weyns. 2017. Software engineering of self-adaptive systems: an organised tour and future challenges. *Chapter in Handbook of Software Engineering* (2017).

[21] Danny Weyns, Bradley R. Schmerl, Vincenzo Grassi, Sam Malek, Raffaela Mirandola, Christian Prehofer, Jochen Wuttke, Jesper Andersson, Holger Giese, and Karl M. Göschka. 2010. On Patterns for Decentralized Control in Self-Adaptive Systems. In *Software Engineering for Self-Adaptive Systems II - International Seminar, Dagstuhl Castle, Germany, October 24-29, 2010 Revised Selected and Invited Papers (Lecture Notes in Computer Science, Vol. 7475)*, Rogério de Lemos, Holger Giese, Hausi A. Müller, and Mary Shaw (Eds.). Springer, 76–107. https://doi.org/10.1007/978-3-642-35813-5_4