Protocol for Mapping Study on Software Architectured-based Self-adaption in Robotics

Elvin Alberts[1,2], Ilias Gerostathopoulos[1], Ivano Malavolta[1], and Carlos Hernández Corbato[2]

[1] Vrije Universiteit Amsterdam, Amsterdam, The Netherlands
{e.g.alberts,i.g.gerostathopoulos, i.malavolta}@vu.nl
[2] Technical University of Delft, Delft, The Netherlands
{e.g.alberts, c.h.corbato}@tudelft.nl

1 Background

The number of robots in use in our society is continually increasing. As this number grows, so too does the breadth of its applications. As the diversity in applications grows the expectations of robots, and by extension the software which drives them increases. A potential solution for meeting these expectations in their software is by endowing it with *self-adaptive* capabilities. Weyns [1] defines a self-adaptive system as a system which fulfills two principles: handling changes and uncertainties autonomously, and comprising of a managed system responsible for domain-specific concerns and a managing system responsible for adaptation concerns. We are specifically interested in architectural self-adaptation which adapts based on a system represented through an architectural model [2]. Depending on the objectives a robotic system is tasked with, it is safe to say (and established in research [3,4,5]) that a multitude of (architectural) self-adaptation approaches exist and have been applied to robotic systems. However, we cannot say that there is a consistency in the nomenclature of these approaches and the venues at which they are published. Neither can one speak of a consensus or taxonomy surrounding the approaches for endowing a robot with self-adaptivity. In the absence of these an established nomenclature or taxonomy the possibility to re-use and extend existing work is impeded. The community interested in this multidisciplinary research area then risk producing overlapping work in disconnected clusters due to a lack of awareness, ultimately hampering their collective progress.

Although there is no comprehensive overview of self-adaptation techniques and abstractions in robotic systems there do exist related studies covering lateral topics. Muccini et al. [6] have done a systematic literature review (SLR) on architectural self-adaptation techniques for cyber-physical systems. However, the authors do not make any significant distinction between the types of systems to the extent that conclusions can be drawn specifically about robots. Garcia et al. [7] recently did an SLR specifically on service robots and their software variability. Self-adaptation techniques are there mentioned as techniques to manage potential variability. However, the authors specifically seek to determine *drivers* (causes) for this variability rather than uncovering the self-adaptation techniques that may account for them, as we seek to. In another study, Albonico et al. [8] have done a systematic mappingspecifically on the state-of-the-art of software engineering in the Robot Operating System ROS. Our study will not limit itself to ROS-based systems, but does seek to similarly establish the state-of-the-art within robotics systems for self-adaptation which is a sub-category of software engineering.

2 Goal and Research Questions

Note: We reorganized the RQs to be more focused after the data extraction phase, for transparency the original RQs can be found in the appendix.

We formulate the goal of this study through GQM (Goal-Question-Metric) perspectives as established by Basili et al. [9].
Purpose: Identify and classify
Issue: the approach for architectural self-adaptation in robotics software
Object: in research on robotics software
Viewpoint: from a researcher's viewpoint.

- RQ1 – What are the key characteristics of approaches for architectural self-adaptation in robotics software?
  - RQ1.1 – What are the key characteristics of the entire self-adaptive system in approaches for architectural self-adaptation in robotics software? In other words, what are the characteristics of the system independent of the distinction between managing and managed system.
  - RQ1.2 – What are the key characteristics of the managing systems of architectural self-adaptation approaches in robotics software? In other words, the characteristics of the logic/algorithms being used by developers to adapt the behavior of their system at runtime.
  - RQ1.3 – What are the key characteristics of the managed systems of architectural self-adaptation approaches in robotics software? In other words, what are the characteristics of the robots being adapted, both in terms of being robots and as being system to be adapted.
- RQ2 – What are the evaluation strategies for approaches for architectural self-adaptation in robotics software? It is not readily obvious how to quantify the impact of introducing self-adaptation. The studies uncovered should show what the state-of-the-art is for doing so.

3 Search Strategy

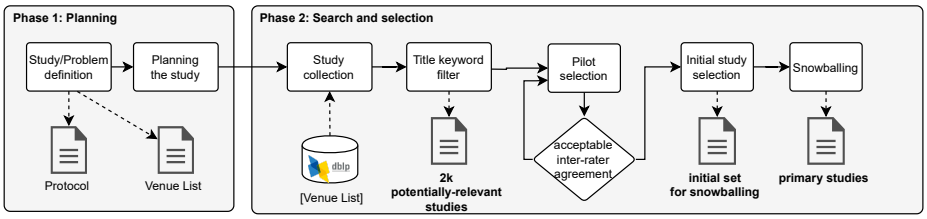

Fig. 1. Search and Selection Strategy

3.1 Planning

To find the potentially-relevant studies for our mapping study we initially gathered a list of venues from which these studies would emerge. We intentionally avoided a keyword search into a database as we know from experience that there is no consistent nomenclature for the topic at hand. The list of venues and their corresponding research area can be found in Table 1 The entire process we follow is also depicted in Figure 1.

3.2 Search and Selection

Study Collection We initially queried studies from the chosen venues through dblp [3] within the time-frame of 2011-2022 (inclusive). The time-frame is chosen as it corresponds with SEAMS the primary conference on self-adaptive systems' first edition as such. This can be justified by the fact that a study's existence in research area of self-adaptive systems is considered a key in identifying potentially-relevant studies for this mapping study and ergo reflected in the inclusion criteria to follow. Due to the magnitude of potentially-relevant studies, particularly the robotics conferences ICRA and IROS which accounted for 2000 studies per year, it was necessary to introduce a keyword filter on the titles of the studies.

Title keyword filter For the filtering by title, we contrived a set of keywords dependent on the research area. To determine if a paper is related to software architecture we searched for 'architect' in the title, for self-adaptive systems 'self-' OR 'adapt', and for robotics 'robot'. Then, we applied an OR between the sets of keywords and searched for those in the titles of studies which did not belong to a venue in that research area. For example, for a robotics study to make it through the filter its title should include either 'architect' or one of 'self-' or 'adapt'. It should be noted that if we filter more strictly e.g. an AND between the multiple keywords in adaptive, or between the categories of keywords, this results in little (magnitude of 1) to no studies satisfying the filter.

Pilot selection For the pilot we have selected 120 studies, 40 per research area in a random seeded manner. In reality, the potentially-relevant studies originate with an overwhelming majority from robotics venues. However, as we target consensus among the authors we artificially bias the pilot selection to feature an equal number of potentially-relevant studies per research area. The script we use to do so as well as extract the date can be found in our GitHub repository [4]. The pilot consists of the first author applying the selection criteria to all 120 studies to determine their inclusion, the remaining three authors are allocated 40 of the studies at random. Afterwards, we calculate the inter-rater agreement between the first author and the other three for the pilot selection using Cohen's kappa. We repeat this pilot process until we have achieved a Cohen's kappa score of > 0.81 between the first author and each of the other authors respectively as this signifies near-perfect agreement [10].

[3] https://dblp.uni-trier.de
[4] https://github.com/S2-group/rsass-mapping-study-rep-pkg

Initial study selection Once this has been established, we can proceed to selecting the primary studies from the remaining potentially-relevant studies with the same distribution of labor, a 3:1:1:1 split while assured there is be no significant discrepancy between each author's selections. From the result of these selections, we then use snowballing to collect any remaining potentially-relevant studies and select from this set as well. Only then is the final set of primary studies to be candidates for data extraction established.

Snowballing Lastly, we gather more potentially-relevant studies to do a selection from through snowballing both backward and forward. This entails gathering all the studies which cite our initial set or are cited by studies in it. Then, they are selected from in the exact same manner in which the initial studies were. Notably, this can be done recursively (once again snowballing from the newly selected studies through snowballing). We leave the decision for the number of iterations of snowballing to be determined from the number of primary studies that result from it.

Name	Type	Focus
International Conference on Robotics and Automation (ICRA)	Conference	Robotics
International Conference on Intelligent Robots and Systems (IROS)		
Robotics Science and Systems (RSS)		
IEEE Robotic Computing (IRC)		
International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS)		Self-adaptation
International Conference on Self-Adaptive and Self-Organizing Systems (SASO)		
International Conference on Autonomic Computing (ICAC)		
Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)		
International Conference on Software Architecture (ICSA)		Software Architecture
European Conference on Software Architecture (ECSA)		
Working IEEE/IFIP Conference on Software Architecture (WICSA)		
International Symposium on Component-Based Software Engineering (CBSE)		
International Conference on Quality of Software Architectures (QoSA)		
IEEE Transaction on robotics (T-RO)	Journal	Robotics
IEEE Robotics and Automation Letters (RAL)		
The International Journal of Robotics Research		
Science Robotics		
Journal of Software Engineering for Robotics (JOSER)*		
ACM TAAS		Self-adaptation
Elsevier's JSA		Software Architecture

*indicates the venue is not indexed in dblp but instead sourced externally

Table 1. List of Venues

4 Selection/Criteria

Before listing the selection criteria it is necessary to define key terms which will feature in them. This definition serves not only to provide transparency for potential reproduction of the study but also to explicitize a common understanding among the authors who will apply them to the potential studies. To wit, our glossary of terms:

- **self-adaptive system**: We use the definition found in the book 'An Introduction to Self-Adaptive Systems' by Weyns [1]. A self-adaptive system is one which handles changes and uncertainties autonomously and has two distinct sub-systems, a managed system responsible for domain concerns, and a managing system responsible for adaptation concerns.
- **software architecture** We use the definition found in the book by Bass et al. [11]. They define a software architecture as "the set of structures needed to reason about the system. These structures comprise software elements, relations among them, and properties of both.".
- **robotic system**: We determine whether a system can be considered a robot using the 'Springer Handbook of Robotics' by Siciliano and Khatib [12]. Besides the clear notion that a robot is a hardware entity interacting with the physical world, the robotic systems we consider are those which conform to the designs described in the Handbook.

Inclusion Criteria:

I1: Studies focusing on robotic systems.

I2: Studies that involve software architecture (as software structures that enable system reasoning) at runtime.

I3: Studies focusing on self-adaptation and/or self-adaptive systems.

I4: Peer-reviewed scientific publications (journal papers, workshop papers, book chapters, conference papers).

I5: Studies published between 2011 and 2022 (inclusive)

Exclusion Criteria:

E1: Studies on self-adaptation in which the managed system is not changed based on an architectural model/representation of it. [5]

E2: Studies written in languages different than English.

E3: Studies not available as full-text documents.

E4: Secondary/tertiary literature studies.

E5: Studies in the form of theses.[6]

[5] Added during selection phase

[6] Added during snowballing phase

5 Data Extraction

To answer RQ1-3 we extract data from our primary studies as in any mapping study. We describe our planned extraction below categorized by RQ.

5.1 RQ1.1 Self-adaptation Techniques in Managing Systems

Some characteristics we extract will be general to the whole system. Here we seek to get data describing aspects about how self-adaptive systems in robotics are designed, and not just implemented. It is an important distinction, as while managed and managing systems may be somewhat interchangeable, distinct instances of each are integrated together in every approach we identify. It is then useful to characterize how this integration is accomplished.

5.2 RQ1.2 Self-adaptation Techniques in Managing Systems

We need to gain an understanding of the particular technique used in each primary study. Practically we can expect the system to consist of a managed system i.e. the robot and a managing system. Here we want to understand the underlying logic that the managing system uses in making adaptation decisions dependent on the knowledge afforded to it by the environment and the managed system. This logic may make use of machine learning, control-theoretic or other approaches. Depending on the data a suitability of particular approaches to the common type of managed system (a robot) may be discovered.

5.3 RQ1.3 Integration of Self-adaptation into managed systems

Realistically, the robots used for research are not all purpose-built and designed for each particular project. Therefore, manufacturers are tasked with creating general robots with their own general-purpose software to meet the needs of their customers i.e. researchers. All this to say, supplied robotics software is highly unlikely to come with self-adaptation in mind, as this is only potentially solution to a problem which is not faced in every, and likely neither the majority, of use cases. Some implementation is then required on the end of the researcher to translate self-adaptive behavior to the software controlling the robot. In our previous work we have used the Robot Operating System 2 (ROS2) to serve as this intermediary. ROS2 takes care of controlling the robot and we develop components within it to encode self-adaptivity. From our primary studies we want to understand what approaches, like ROS2, are in use in the state-of-the-art and what they may have in common. .

5.4 RQ2 Evaluation of Self-Adaptation

Ultimately, having a any system behave self-adaptively including robots is done to solve a problem, typically addressing an uncertainty found at runtime. We seek to determine to what extent self-adaptation solves this problem for that system described in each primary study. Further, it is of interest to determine what the baselines are for these comparisons. Depending on the quality of the evaluations of the primary studies, we can then determine whether self-adaptation provides a unique solution to what we know is a real problem that is meant to address, or whether the state-of-the-art favors sees other solutions perform better. By extension if the latter were discovered a gap in research reveals itself as to improving the performance of self-adaptation in robotics.

References

1. D. Weyns, *An Introduction to Self-adaptive Systems: A Contemporary Software Engineering Perspective.* John Wiley & Sons, 2020.
2. D. Garlan, B. Schmerl, and S.-W. Cheng, "Software architecture-based self-adaptation," *Autonomic computing and networking*, pp. 31–55, 2009.
3. B. H. Cheng, R. J. Clark, J. E. Fleck, M. A. Langford, and P. K. McKinley, "Ac-ros: assurance case driven adaptation for the robot operating system," in *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*, 2020, pp. 102–113.
4. S. Gerasimou, R. Calinescu, S. Shevtsov, and D. Weyns, "Undersea: an exemplar for engineering self-adaptive unmanned underwater vehicles," in *2017 IEEE/ACM 12th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS).* IEEE, 2017, pp. 83–89.
5. A. Nordmann, R. Lange, and F. M. Rico, "System modes-digestible system (re-) configuration for robotics," in *2021 IEEE/ACM 3rd International Workshop on Robotics Software Engineering (RoSE).* IEEE, 2021, pp. 19–24.
6. H. Muccini, M. Sharaf, and D. Weyns, "Self-adaptation for cyber-physical systems: a systematic literature review," in *Proceedings of the 11th international symposium on software engineering for adaptive and self-managing systems*, 2016, pp. 75–81.
7. S. García, D. Strüber, D. Brugali, A. Di Fava, P. Pelliccione, and T. Berger, "Software variability in service robotics," *Empirical Software Engineering*, vol. 28, no. 2, pp. 1–67, 2023.
8. M. Albonico, M. Dordevic, E. Hamer, and I. Malavolta, "Software engineering research on the robot operating system: A systematic mapping study," *Journal of Systems and Software*, vol. 197, p. 111574, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121222002503
9. V. R. B. G. Caldiera and H. D. Rombach, "The goal question metric approach," *Encyclopedia of software engineering*, pp. 528–532, 1994.
10. M. L. McHugh, "Interrater reliability: the kappa statistic," *Biochemia medica*, vol. 22, no. 3, pp. 276–282, 2012.
11. L. Bass, P. Clements, and R. Kazman, *Software architecture in practice.* Addison-Wesley Professional, 2003.
12. B. Siciliano, O. Khatib, and T. Kröger, *Springer handbook of robotics.* Springer, 2008, vol. 200.

A Original Research Questions

- **RQ1**: What architectural self-adaptation techniques are being used in the state-of-the-art of robotics software? In other words, the logic/algorithms being used by developers to adapt the behavior of their system at runtime.
- **RQ2**: How do state-of-the-art approaches integrate architectural self-adaptation into the development of robotics software? Given an existing robot software system or during the development of a new one a developer makes architectural decisions which ensure self-adaptation is possible, what are these?
- **RQ2.1**: Is it possible to create a unified model of these approaches? Is it possible to determine tactics/techniques which generalize across these systems as applied by developers?

– **RQ3**: How is the impact of self-adaptation measured among these robotics systems? It is not readily obvious how to quantify the impact of introducing self-adaptation. The studies uncovered should show what the state-of-the-art is for doing so.