Assured and Correct Dynamic Update of Controllers [⋆]

L. Nahabedian[⋆], V. Braberman[⋆], N. D'Ippolito[⋆], S. Honiden[+], J. Kramer[†], K. Tei[+] and S. Uchitel[†⋆]

[†] Department of Computing, Imperial College London, UK
[⋆] Departamento de Computación, FCEN, Universidad de Buenos Aires, Argentina
[+] National Institute of Informatics, Japan

In many application domains, continuous operation is a desirable attribute for software-intensive systems. As the environment or system requirements change, so the system should change and adapt without stopping or unduly disturbing its operation. There is, therefore, a need for sound engineering techniques that can cope with dynamic change. We address the problem of dynamic update of controllers in reactive systems when the specification (environment assumptions, requirements and interface) of the current system changes as in [5].

When is it safe to change a running system? One conservative answer to this question is "when components are not involved in any interactions"; this was formalised through notions of quiescence [5] and later tranquility [7].

Zhang et al. show that different domains and scenarios require distinct update conditions. Sometimes update may be allowed to occur cleanly at any point in time (usually as soon as possible), and in others obligations of the current specification must be honoured before switching to the new specification. Requirements to guide the system from one specification to another (as in [2]) or to ensure graceful update (as in [3]) may be appropriate in certain settings. These update strategies generally use manual verification to ensure correctness.

Automatic synthesis of controller update strategies for reactive systems has been proposed in [4,6,1] amongst others. However, in all cases it is assumed that the system being executed eventually reaches (what each technique considers) a safe state. This liveness assumption of eventually reach a safe state is very strong. Zhang et al. [8] recognize that a system may need to be guided to a safe updatable state; however the user have to produce the strategy to do so.

We present a general approach to specifying correctness criteria for dynamic update and a technique for automatically computing a controller that handles the transition from the old to the new specification, assuring that the system will reach a state in which such a transition can correctly occur. Indeed, using controller synthesis we show how to automatically build a controller that guarantees both progress towards update and safe update.

The purpose of our validation is to show applicability of the approach by resolving seven case studies taken from literature and also the generality of the approach with respect to existing work.

Keywords: Controller Synthesis, Dynamic Update, Adaptive Systems

[⋆] This work was accepted at International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS'16), Austin TX, 2016

References

1. S. An, X. Ma, C. Cao, P. Yu, and C. Xu. An event-based formal framework for dynamic software update. In *Software Quality, Reliability and Security (QRS), 2015 IEEE International Conference on*, pages 173–182, 2015.
2. J. Appavoo, K. Hui, C. A. N. Soules, R. W. Wisniewski, D. M. Da Silva, O. Krieger, M. A. Auslander, D. J. Edelsohn, B. Gamsa, G. R. Ganger, P. McKenney, M. Ostrowski, B. Rosenburg, M. Stumm, and J. Xenidis. Enabling autonomic behavior in systems software with hot swapping. *IBM Syst. J.*, 42(1):60–76, Jan. 2003.
3. W.-K. Chen, M. Hiltunen, and R. Schlichting. Constructing adaptive software in distributed systems. In *Distributed Computing Systems, 2001. 21st International Conference on.*, pages 635–643, Apr 2001.
4. C. Ghezzi, J. Greenyer, and V. Manna. Synthesizing dynamically updating controllers from changes in scenario-based specifications. In *Software Engineering for Adaptive and Self-Managing Systems (SEAMS), 2012 ICSE Workshop on*, pages 145–154, June 2012.
5. J. Kramer and J. Magee. The evolving philosophers problem: Dynamic change management. *IEEE Trans. Softw. Eng.*, 16(11):1293–1306, Nov. 1990.
6. V. Panzica La Manna, J. Greenyer, C. Ghezzi, and C. Brenner. Formalizing correctness criteria of dynamic updates derived from specification changes. In *Proceedings of the 8th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, pages 63–72. IEEE Press, 2013.
7. Y. Vandewoude, P. Ebraert, Y. Berbers, and T. D'Hondt. Tranquility: A low disruptive alternative to quiescence for ensuring safe dynamic updates. *Software Engineering, IEEE Transactions on*, 33(12):856–868, Dec 2007.
8. J. Zhang and B. H. C. Cheng. Specifying adaptation semantics. In *Proceedings of the 2005 Workshop on Architecting Dependable Systems*, WADS '05, pages 1–7, New York, NY, USA, 2005. ACM.