Refining Reliability Estimation of Mobile Software Systems

Roshanak Roshandel
Computer Science & Software Engineering
Seattle University,
Seattle, WA, 98122-1090, U.S.A
roshanak@seattleu.edu

Sam Malek
Department of Computer Science
George Mason University
Fairfax, VA 22030-4444 U.S.A.
smalek@gmu.edu

1. INTRODUCTION AND BACKGROUND

Over the last decade, emergence of small, resource-constrained, and highly-mobile computing platforms has presented numerous new challenges to the engineers. The development in this setting is known as *programming-in-the-small-and-many (Prism)* [11][10]. Software systems in the Prism setting are required to cope with the unpredictability of available resources (e.g., availability of network connectivity, lack of battery power, etc.). Yet they are often depended upon to deliver the functionality that is expected of them.

Studies have shown that a promising approach to resolve the challenges of developing software in the prism setting is to employ the principles of software architectures [10][11]. Software architectures provide abstraction for representing the structure, behavior, and key properties of a software system. They are described in terms of software *components* (computational elements), *connectors* (interaction elements), and their *configurations*. Specialized architectural styles, architecture description languages (ADLs), and middleware platforms have been developed to address the challenges of development in the prism setting [6][7][8][10]. These architectural artifacts have been leveraged to analyze, verify, and validate both functional and non-functional properties of software systems.

While several previous works have focused on predicting the reliability of software systems using the software architecture artifacts [4][5][9], there has been a lack of previous works on estimating the reliability of mobile software systems. Note that reliability prediction serves two goals: (1) It can help evaluate the quality of the software architecture itself. Corresponding analyses can be used to compare the quality of competing architectural designs. (2) It can serve as a prediction of the future quality of the software under the development.

The architecture-based reliability models leverage the knowledge about the system's behavior in terms of the behavior of communicating components and incorporate data from various available information sources to provide a prediction of the components and system reliability. While component interaction is embodied in software connectors, the reliability modeling approaches do not leverage connectors directly. Instead components play the central role in these reliability models. Modeling and analyzing the software reliability by focusing on various type of software connectors [12] and considering their critical role in component interaction, not only makes the reliability model more comprehensive, but also it enables addressing the uncertainties associated with the prism setting.

In particular, mobility imposes specific challenges on component interactions that are associated with connectivity issues, as well as the dynamic nature of the software system. For example, host mobility could result in network disconnection, which impacts the interaction of software components. Similarly, as a result of a component migrating from one hardware host to another, the component's interaction behavior and peers may change significantly. This could impact the engineer's original prediction of the component and system reliability, which depends on the accuracy of the interaction protocols between software components.

In this paper, we propose a connection-centric approach to reliability modeling, which account for the uncertainties associated with mobility and thus provide a better analysis of system's reliability in the face of changes to the software system's structure and behavior. The approach extends our previous work on component-level [1] and system-level reliability analysis [13]. The details of the approach, challenges, and the preliminary work are described in this short paper.

2. APPROACH

Our ongoing research offers a compositional approach to system reliability prediction at the software architecture-level. First, we model the reliability of individual components in the system using a Markov-based reliability model. The model addresses the uncertainty associated with early reliability prediction by incorporating a variety of available information sources. In cases where prototype implementation of components is available, the model leverages the failure behavior obtained at runtime. When no runtime failure behavior is available the model relies on the defects detected via architectural analyses and uses Hidden Markov methodology to obtain component reliability values. The component reliability values are then used in building a compositional Dynamic Bayesian Network (DBN) that is used to analyze the reliability of software systems in terms of the interactions among components which may result in failures.

Our proposed research will leverage the component reliability prediction described above, but relies on runtime monitoring of the system (as well as the system-level architectural models) to refine the estimation of the system's reliability. In the context of mobile software systems, the interaction among components is highly dynamic. By monitoring this runtime interaction, we are able to create dynamic models of components interaction. These models along with architectural models of the system will serve as the basis of a DBN-based reliability model. The result is a refinement of the initial architecture-based reliability prediction in the face of dynamic changes to the system's architecture in the distributed and mobile setting. Other factors such as the reliability of software connectors (e.g., those provided by middleware platforms such as CORBA, RMI, or Prism-MW [10]), and the reliability of the underlying network connectivity can be also incorporated into the reliability model.

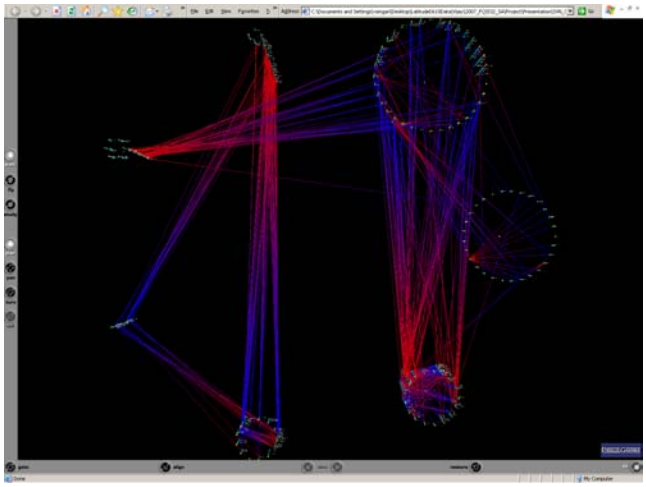

Figure 1. Graphical Representation of Interactions

Our preliminary work on runtime interaction monitoring and visualization allows us to recover procedure calls, data access, and types of interactions from a software system. The results are visualized using a directed graph (Figure 1). The hot/cold color designation represents the direction of the interaction (e.g., caller vs. callee).

We plan to use a variety of graph-theoretic algorithms to analyze the *cohesion* among interacting (mobile) nodes (e.g., density and centralization), the *clustering* of interactions (e.g., cliques), and the *centrality* of interacting nodes (e.g., degree, closeness, betweenness) [1][3][14]. The results of these analyses are then incorporated into the DBN-based model to offer a more refined reliability estimation of the system.

3. CONCLUSION

Architecture-level reliability prediction of software systems offers useful insights for the ongoing activities associated with the design and development of software systems. In the context of distributed and mobile software systems however, structural dynamism results in significant level of uncertainty in components interactions. Reliability predictions based on architectural models thus need to be refined with runtime behavioral analysis of the system (or its prototype) with a focus on communications and interactions among components. We propose one such approach that extends our past research on architectural middleware and reliability analysis.

4. ACKNOWLEDGMENTS

The authors wish to thank Jafar Adibi for stimulating discussions on social networks. Also thanks to Suresh Batta, Hans Sebastian, and Vijay Singari for the preliminary work on interaction monitoring.

5. REFERENCES

[1] Borgatti S.P. and Everett M.G., A Graph Theoretic Perspective on Centrality. *Social Networks,28 (4) 466-484,* 2006.

[2] Cheung L., Roshandel R., Medvidovic N., Golubchik L., Early Prediction of Software Component Reliability, in proceedings of the 30th International Conference on Software Engineering (ICSE 2008), Leipzig, Germany, May 2008 (to appear).

[3] Everett, M. G. and Borgatti, S. P. Analyzing clique overlap. *Connections, 21(1): 49-61*, 1998.

[4] Gokhale S., Architecture-Based Software Reliability Analysis: Overview and Limitations. IEEE Transactions on Dependable and Secure Computing: 4(1), Jan 2007.

[5] Goseva-Popstojanova K. et al., Architecture-Based Approaches to Software Reliability Prediction, International Journal Computer & Mathematics with Applications, Vol. 46, Issue 7, October 2003.

[6] Gruhn V., Schäfer C. Architecture Description for Mobile Distributed Systems Using Typed pi-Calculus. Electr. Notes Theor. Comput. Sci. 150(1): 51-60 (2006).

[7] V. Issarny, F. Tartanoglu, J. Liu, F. Sailhan, Software Architecture for Mobile Distributed Computing, 4th Working IEEE/IFIP Conference on Software Architecture (WICSA'04), 2004.

[8] Hwa-Young Jeong, Young-Jae Song, Prism-WM Based Connector Interaction for Middleware Systems, in the proceedings of International Conference on Embedded Software and Systems (ICESS 2004), Hangzhou, China, pp 258-265.

[9] Immonen A., and Niemela E. Survey of reliability and availability prediction methods from the viewpoint of software architecture. Software and Systems Modeling, Jan 2007.

[10] Malek S., Mikic-Rakic M. , and Medvidovic N. "A Style-Aware Architectural Middleware for Resource Constrained, Distributed Systems." IEEE Transactions on Software Engineering, vol. 31, no. 3, March 2005.

[11] Medvidovic N., Mikic-Rakic M., Mehta N., and Malek S.. Software Architectural Support for Handheld Computing. *IEEE Computer – Special Issue on Handheld Computing, vol. 36, no. 9, pages 66-73,* September 2003

[12] Mehta N. R., Medvidovic N., and Phadke S. "Towards a Taxonomy of Software Connectors." In Proceedings of the 22nd International Conference on Software Engineering (ICSE 2000), pages 178-187, Limerick, Ireland, June 4-11, 2000.

[13] Roshandel R., Medvidovic N., Golubchik L., A Bayesian Model for Predicting Reliability of Software Systems at the Architectural Level, in proceedings of 3rd International Conference on Quality of Software Architectures (QoSA 2007), Boston, MA, July 2007.

[14] White, D. R., & Borgatti, S. P. Betweenness centrality measures for directed graphs. *Social Networks, 16(4): 335-346,* 1994