Modeling of Event-based Communication in Component-based Architectures

Samuel Kounev [3,4]

Karlsruhe Institute of Technology
Karlsruhe, Germany

Christoph Rathfelder[1], Benjamin Klatt[2]

FZI Research Center for Information Technology
Karlsruhe, Germany

Abstract

Event-based communication is used in different domains including telecommunications, transportation, and business information systems to build scalable distributed systems. Such systems typically have stringent requirements for performance and scalability as they provide business and mission critical services. While the use of event-based communication enables loosely-coupled interactions between components and leads to improved system scalability, it makes it much harder for developers to estimate the system's behavior and performance under load due to the decoupling of components and control flow. We present an overview on our approach enabling the modeling and performance prediction of event-based system at the architecture level. Applying a model-to-model transformation, our approach integrates platform-specific performance influences of the underlying middleware while enabling the use of different existing analytical and simulation-based prediction techniques. The results of two real world case studies demonstrate the effectiveness, practicability and accuracy of the proposed modeling and prediction approach.

Keywords:

1 Motivation

The event-based communication paradigm is used increasingly often to build loosely-coupled distributed systems in many different industry domains. The

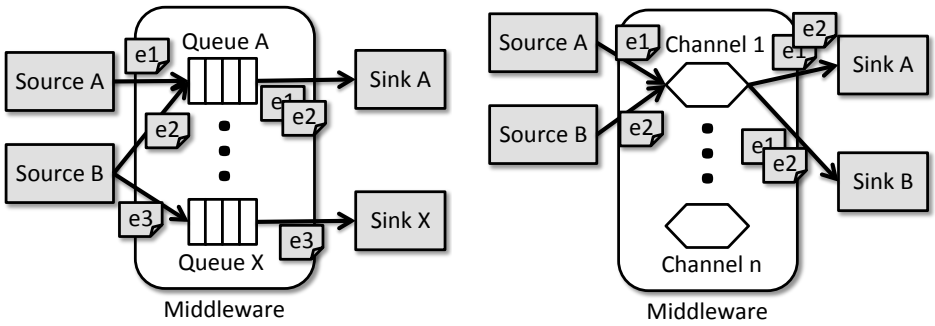
[1] Email: rathfelder@fzi.de
[2] Email: klatt@fzi.de
[3] Email: kounev@kit.edu
[4] This work was partially funded by the German Research Foundation (grant No. KO 3445/6-1)

This paper is electronically published in
Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

application areas of event-based systems range from distributed sensor-based systems up to large-scale business information systems [6]. Compared to synchronous communication using, for example, remote procedure calls (RPC), event-based communication among components promises several benefits such as high scalability and extendability [7]. Being asynchronous in nature, it allows a *send-and-forget* approach, i.e., a component that sends a message can continue its execution without waiting for the receiver to react on the message. Furthermore, the loose coupling of components achieved by the mediating middleware system leads to an increased extensibility of the system as components can easily be added, removed, or substituted.

With the growing proliferation of event-based communication in mission critical systems, the performance and scalability of such systems are becoming a major concern. To ensure adequate Quality-of-Service (QoS), it is essential that applications are subjected to a rigorous performance and scalability analysis as part of the software development process. In today's data centers, software systems are often deployed on server machines with over-provisioned capacity in order to guarantee highly available and responsive operation [9], which automatically leads to lower efficiency. Although the event-based communication model promises to improve flexibility and scalability, the complexity compared to RPC-based communication is higher since the application logic is distributed among multiple independent event handlers with decoupled and parallel execution paths. This increases the difficulty of modeling event-based communication for quality predictions at system design and deployment time. Thus, the evaluation of event-based systems requires specialized techniques that consider the different characteristics and features of event-based communication.

Performance modeling and prediction techniques for component-based systems, surveyed in [12], support the architect in evaluating the system architecture and design alternatives regarding their performance and resource efficiency. However, most general-purpose performance meta-models for component-based systems provide limited support for modeling event-based communication. Furthermore, existing performance prediction techniques specialized for event-based systems (e.g., [16,5,26]) are focused on modeling the routing of events in the system as opposed to modeling the interactions and message flows between the communicating components. In the following, we present an overview on our approach enabling the modeling of event-based communication at the architecture-level combined with platform-aware performance predictions. For more details on our approach we refer to [17,21]. Finally, we present the evaluation of our approach based on two case studies.



(a) Point-to-Point Communication (b) Publish/subscribe Communication

Fig. 1. Event-based Communication Styles

2 Approach

Our integrated approach for modeling event-based interactions in component-based architectures for quantitative system evaluations [17] combines two aspects: i) the identification and implementation of meta-model elements required for modeling event-based communication at the architecture level, and ii) the design and realization of a two-step model-to-model transformation integrating platform-independent and platform-specific aspects of event-based communication into the prediction model. As an example of a representative mature meta-model for component-based software architectures, we consider the Palladio Component Model (PCM) [3]. PCM is accompanied with different analytical and simulative analysis techniques, e.g., [3,14,13] enabling quality predictions at system design time. Similarly to most component meta-models for component-based architecture, PCM, in its original version did not provide support for modeling event-based communication. Performance predictions are only possible using workarounds as demonstrated in [18]. The modeling effort incurred by this workaround approach is very high and provides limited flexibility to evaluate different design alternatives.

2.1 Modeling Event-based Interactions

Modeling event-based communication at the architecture level requires new meta-model elements. We identified the required elements enabling the modeling of event-based interactions. We initially started our work with elements enabling the modeling of direct point-to-point connections (illustrated in Figure 1(a)). These first extensions [23,22] include event ports as well as connectors between those ports and elements to describe the processing of events and the event itself as a first class entity. Enabling the modeling of publish/subscribe communication requires additional elements representing objects like the event channel (see Figure 1(b)). Our final model, presented in [21,17], supports point-to-point as well as publish/subscribe communication

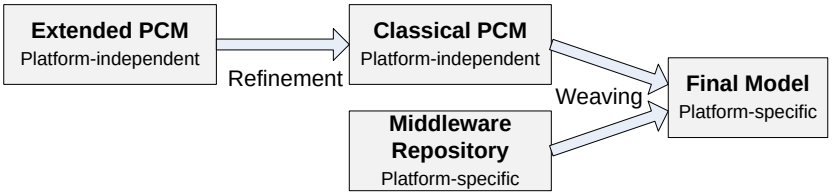


Fig. 2. Two-step Refinement Transformation

in component-based systems. We exemplary implemented these meta-model extensions based on PCM, as a mature and representative meta-model for component-based architectures.

2.2 2-Step Refinement Transformation

The two-step transformation (illustrated in Figure 2) refines the event-based connections between components as envisioned in [20] . The transformation is partitioned into a platform-independent and a platform-specific part. In the first part, the new elements are transformed to a set of elements, following a generic event processing chain. In the second step, platform-specific components located in a separate middleware repository are woven into the final model. Due to this separation, the influence of using different middleware systems can be analyzed by simply selecting another middleware repository and the system itself can be modeled independent of the underlying middleware.The transformation allows the modeling of event-based communication at the architetcure-level using the introduced meta-model elements while still supporting all existing prediction techniques such as simulation [3], LQNs [13] or QPNs [14].

Figure 3 illustrates the result of the refinement transformation when applied to a direct point-to-point connection. An platform-independent event processing chain in form of several components is integrated. These components represent different steps in the middleware's event processing:

- **SourcePort** Interaction between source component and middleware.
- **DistributionPreparation** Processing within the middleware done once per event (e.g., marshaling before the distribution).
- **EventDistribution** Splitting the control flow and distributing the events to all sinks.
- **EventSender** Processing within the middleware that per connected sink (e.g., communication handshake).
- **EventReceiver** Receiver-side event processing within the middleware library (e.g., receiving from the middleware server).
- **SinkPort** Pass event to the receiving component.

The platform-independent components do not include any platform-

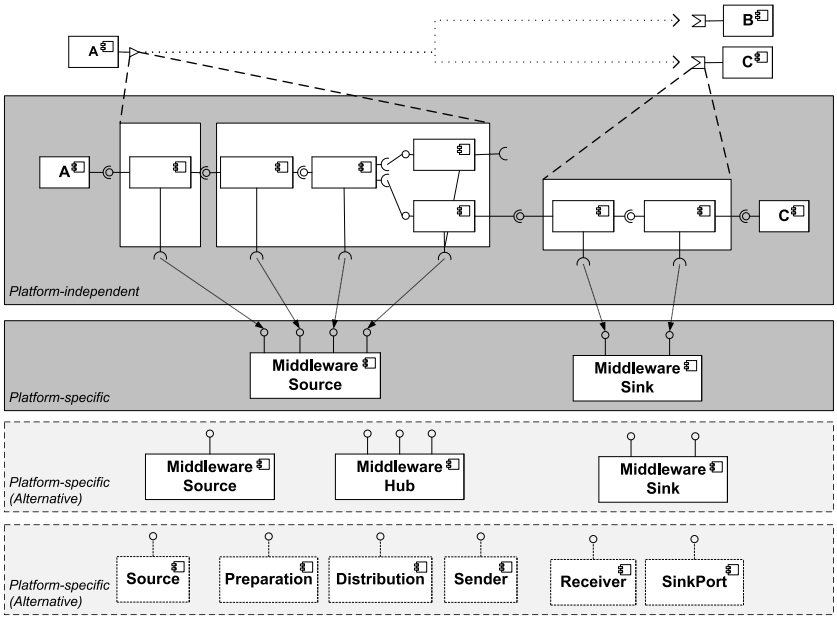


Fig. 3. Transformation Result

specific resource demands. In order to enable the integration of platform-specific resource demands and behavior descriptions, each component includes a required interface that can be used to connect additional components described as part of a platform-specific middleware repository. The second transformation step interweaves these components into the model and automatically deploys them on the same container the belonging platform-independent component is deployed on. The transformation is implemented as QVT-O script [10] and fully integrated into the automated performance prediction process within the PCM workbench [19].

3 Evaluation

In order to provide a comprehensive evaluation of our approach, we selected two representative real-world system from different application domains and covering most aspects of event-based systems. The first case study, described in [24], is based on a distributed traffic-monitoring system developed at the University of Cambridge [2] and built on top of the distributed peer-to-peer middleware SBUS [8]. In addition to the prediction accuracy, we evaluated the adaptability of the model to reflect architectural changes and deployment variations, which are typical for distributed event-based systems. The SPECjms2007 benchmark, our second case study [21], is a supply chain management system representative of real-world industrial applications built on top of a centralized message-oriented middleware. The different interactions exercise a complex transaction mix including point-to-point and pub-

lish/subscribe communication [25]. The results of our evaluation [17] demonstrate that our approach reduces the modeling efforts by more than 80% compared to the use of workarounds. The effort to reflect system variations and different deployment options in the architecture level models is less than 30 minutes. In both case studies, the detailed evaluation of the prediction accuracy shows that the prediction error, compared to measurements on the running system, is less than 20% in most cases. This is a more than acceptable accuracy for design-time performance analysis [15].

4 Conclusion and Outlook

We presented an overview on our research results enabling the modeling of event-based interactions at the architecture-level for performance predictions. These results cover i) a modeling approach for event-based communication at the architecture level exemplarily implemented based on PCM, ii) a two-step transformation approach enabling the performance prediction of the system including the consideration of platform-specific middleware influence factors, and iii) a detailed evaluation of our approach based in two real-world case studies representing different domains of event-based systems. Our research results form the basis for several areas of future work. The presented approach requires the existence of a platform-specific middleware repository. The Performance Cockpit approach [27] uses automated experiments to derive parameterized resource demands for components. As a next step, we plan to define a set of experiments that allow us to automatically derive the middleware repository using the Performance Cockpit. Our current and future research focuses on the idea to make architecture-level performance models usable at run-time. The Descartes Research Group [1] is working on enhancing design-time models to capture dynamic aspects of the environment and making them an integral part of the system [11]. Currently, we work on the integration of our extensions into the Descartes meta-model [4] to enable the runtime management of event-based systems.

References

[1] *Descartes Research Group ; http://www.descartes-research.net.*

[2] Bacon, J., A. Beresford, D. Evans, D. Ingram, N. Trigoni, A. Guitton and A. Skordylis, *Time: An open platform for capturing, processing and delivering transport-related data*, in: *Consumer Communications and Networking Conference, 2008. CCNC 2008. 5th IEEE*, 2008, pp. 687 – 691.
URL http://dx.doi.org/10.1109/ccnc08.2007.158

[3] Becker, S., H. Koziolek and R. Reussner, *The Palladio component model for model-driven performance prediction*, Journal of Systems and Software **82** (2009), pp. 3–22.
URL http://dx.doi.org/10.1016/j.jss.2008.03.066

[4] Brosig, F., N. Huber and S. Kounev, *Descartes Meta-Model (DMM)*, Technical report, Karlsruhe Institute of Technology (KIT) (2012), to be published.
URL http://www.descartes-research.net

[5] Castelli, S., P. Costa and G. P. Picco, *Modeling the communication costs of content-based routing: the case of subscription forwarding*, in: *DEBS '07: Proceedings of the 2007 inaugural international conference on Distributed event-based systems* (2007), pp. 38–49.
URL http://doi.acm.org/10.1145/1266894.1266902

[6] Hinze, A., K. Sachs and A. Buchmann, *Event-based applications and enabling technologies*, in: *Proceedings of the Third ACM International Conference on Distributed Event-Based Systems*, DEBS '09 (2009), pp. 1:1–1:15.
URL http://doi.acm.org/10.1145/1619258.1619260

[7] Hohpe, G. and B. Woolf, "Enterprise integration patterns : designing, building, and deploying messaging solutions," The Addison-Wesley signature series, Addison-Wesley, Boston, Mass., 2008, 11. print. edition.

[8] Ingram, D., *Reconfigurable middleware for high availability sensor systems*, in: *Proceedings of the Third ACM International Conference on Distributed Event-Based Systems*, DEBS '09 (2009), pp. 20:1–20:11.
URL http://doi.acm.org/10.1145/1619258.1619285

[9] Kaplan, J. M., W. Forrest and N. Kindler, *Revolutionizing data center energy efficiency*, Technical report, McKinsey&Company (2008).

[10] Klatt, B., C. Rathfelder and S. Kounev, *Integration of event-based communication in the palladio software quality prediction framework*, in: *Proceedings of the joint ACM SIGSOFT conference – QoSA and ACM SIGSOFT symposium – ISARCS on Quality of software architectures – QoSA and architecting critical systems – ISARCS*, QoSA-ISARCS '11 (2011), pp. 43–52.
URL http://doi.acm.org/10.1145/2000259.2000268

[11] Kounev, S., *Engineering of Next Generation Self-Aware Software Systems: A Research Roadmap*, in: *Emerging Research Directions in Computer Science. Contributions from the Young Informatics Faculty in Karlsruhe*, KIT Scientific Publishing, 2010 ISBN: 978-3-86644-508-6.
URL http://uvka.ubka.uni-karlsruhe.de/shop/isbn/978-3-86644-508-6

[12] Koziolek, H., *Performance evaluation of component-based software systems: A survey*, Elsevier Performance Evaluation **67** (2010), pp. 634–658.
URL http://portal.acm.org/citation.cfm?id=1808359.1808729

[13] Koziolek, H. and R. Reussner, *A Model Transformation from the Palladio Component Model to Layered Queueing Networks*, in: *Performance Evaluation: Metrics, Models and Benchmarks, SIPEW 2008*, Lecture Notes in Computer Science **5119** (2008), pp. 58–78.
URL http://www.springerlink.com/content/w14m0g520u675x10/fulltext.pdf

[14] Meier, P., S. Kounev and H. Koziolek, *Automated Transformation of Component-Based Software Architecture Models to Queueing Petri Nets*, in: *Proceedings of the 2011 IEEE 19th Annual International Symposium on Modelling, Analysis, and Simulation of Computer and Telecommunication Systems*, MASCOTS '11 (2011), pp. 339–348.
URL http://dx.doi.org/10.1109/MASCOTS.2011.23

[15] Menascé, D. A., V. A. F. Almeida and L. W. Dowdy, "Performance by Design," Prentice Hall, 2004.

[16] Mühl, G., A. Schröter, H. Parzyjegla, S. Kounev and J. Richling, *Stochastic Analysis of Hierarchical Publish/Subscribe Systems*, in: *Proceedings of the 15th International Euro-Par Conference on Parallel Processing*, Euro-Par '09 (2009), pp. 97–109.
URL http://dx.doi.org/10.1007/978-3-642-03869-3_13

[17] Rathfelder, C., "Modelling Event-Based Interactions in Component-Based Architectures for Quantitative System Evaluation," Ph.D. thesis, Karlsruhe Institute of Technology (2012), to be published.

[18] Rathfelder, C., D. Evans and S. Kounev, *Predictive Modelling of Peer-to-Peer Event-Driven Communication in Component-Based Systems*, in: A. Aldini, M. Bernardo, L. Bononi and V. Cortellessa, editors, *Computer Performance Engineering*, Lecture Notes in Computer Science **6342** (2010), pp. 219–235.
URL http://www.springerlink.com/content/3418104340042412/

[19] Rathfelder, C. and B. Klatt, *Palladio workbench: A quality-prediction tool for component-based architectures*, in: *Proceedings of the 2011 Ninth Working IEEE/IFIP Conference on Software Architecture*, WICSA '11 (2011), pp. 347–350.
URL http://dx.doi.org/10.1109/WICSA.2011.55

[20] Rathfelder, C., B. Klatt, S. Kounev and D. Evans, *Towards middleware-aware integration of event-based communication into the palladio component model*, in: *Proceedings of the Fourth ACM International Conference on Distributed Event-Based Systems*, DEBS '10 (2010), pp. 97–98.
URL http://doi.acm.org/10.1145/1827418.1827437

[21] Rathfelder, C., B. Klatt, K. Sachs and S. Kounev, *Modeling event-based communication in component-based software*, Journal on Software and Systems Modeling; Theme Issue on Models for Quality of Software Architecture (2012), under review.

[22] Rathfelder, C. and S. Kounev, *Model-based Performance Prediction for Event-driven Systems*, in: *Proceedings of the Third ACM International Conference on Distributed Event-Based Systems*, DEBS '09 (2009), pp. 33:1–33:2.
URL http://doi.acm.org/10.1145/1619258.1619300

[23] Rathfelder, C. and S. Kounev, *Modeling Event-driven Service-oriented Systems using the Palladio Component Model*, in: *Proceedings of the 1st international workshop on Quality of service-oriented software systems*, QUASOSS '09 (2009), pp. 33–38.
URL http://doi.acm.org/10.1145/1596473.1596482

[24] Rathfelder, C., S. Kounev and D. Evans, *Capacity Planning for Event-based Systems using Automated Performance Predictions*, in: *26th IEEE/ACM International Conference On Automated Software Engineering (ASE 2011)*, Oread, Lawrence, Kansas, 2011, pp. 352–361.
URL http://dx.doi.org/10.1109/ASE.2011.6100073

[25] Sachs, K., S. Kounev, J. Bacon and A. Buchmann, *Benchmarking message-oriented middleware using the SPECjms2007 benchmark*, Performance Evaluation **66** (2009), pp. 410–434.
URL
http://www.elsevier.com/wps/find/journaldescription.cws_home/505618/description

[26] Schröter, A., G. Mühl, S. Kounev, H. Parzyjegla and J. Richling, *Stochastic performance analysis and capacity planning of publish/subscribe systems*, in: *Proceedings of the Fourth ACM International Conference on Distributed Event-Based Systems*, DEBS '10 (2010), pp. 258–269.
URL http://doi.acm.org/10.1145/1827418.1827468

[27] Westermann, D., J. Happe, M. Hauck and C. Heupel, *The Performance Cockpit Approach: A Framework For Systematic Performance Evaluations*, in: *Proceedings of the 2010 36th EUROMICRO Conference on Software Engineering and Advanced Applications*, SEAA '10 (2010), pp. 31–38.
URL http://dx.doi.org/10.1109/SEAA.2010.24