Architecting Service-Oriented Systems

featuring Grace Lewis interviewed by Shane McGraw

Shane McGraw: Welcome to the SEI podcast series, a production of the Carnegie Mellon Software Engineering Institute. The SEI is a federally funded research and development center at Carnegie Mellon University in Pittsburgh, Pennsylvania. To learn more about the SEI, please visit our web site at www.sei.cmu.edu. My name is Shane McGraw, and today I'm pleased to introduce you to Grace Lewis, a senior member of the technical staff in the Research, Technology, and System Solutions initiative at the SEI. Grace's recent work includes architecture and design of service-oriented systems, cloud computing in tactical environments, and architectures for context-aware, mobile applications. In today's podcast, Grace will be discussing her latest research on architecting service-oriented systems, the results of which were published in August 2011, in the SEI Technical Note, Architecting Service-Oriented Systems. Welcome, Grace.

Grace Lewis: Thanks, Shane.

Shane: Grace, it's hard to go anywhere in software circles without hearing about service-oriented architecture, or SOA. I know that many people have an idea of what it is, but please explain what a service-oriented system is, and why it matters in the field of software.

Grace: Okay, to talk about service-oriented systems we first have to talk about service orientation. So, what service orientation is, is a way of designing, developing, and deploying software systems similar to, for example, to object-orientation, which is another way of designing, deploying, and developing systems. This particular way of designing systems, or paradigm, or you could even call it an architectural style, is characterized by basically three components. The first component is the services that represent reusable business functionality that is exposed via standardized service interfaces. The second part of this architectural style is the service consumers, which are the systems or applications that use the functionality provided by these services as part of the processes that they implement.

Finally, the third element is there's an SOA infrastructure that connects service consumers to service. So a service-oriented system is basically a system that is built following this particular paradigm.

Shane: Grace, before we take a deeper dive into your research, I'd like you to help us clear something up for us. A lot of the claims that we've heard in the past, and you and I have had this discussion in different webinars, is that SOA is dead. What are your thoughts on this claim?

Grace: Well, that claim was made some time ago, but you still hear about it. It was made in a blog post by a very well respected person in the field. Unfortunately, her blog post was completely misinterpreted, because what happened was that some folks only read the title and just assumed what the content would be. So, those that had made investments in service orientation were saying, "Okay, now what do I do?" Those that had not made the investments basically said "I told you so." Really, what the blog post was stating is that we need to stop thinking about SOA as a set of technologies and start thinking about service orientation as a development paradigm, which is completely consistent with what I just said earlier when you asked me to define service-oriented systems.

Shane: Thank you for that, Grace. Now, to turn our attention back to your latest SEI technical note, please tell us, what does it mean to architect a service-oriented system?

Grace: That is a perfect follow up to the previous question. So, going back to the SOA-is-dead blog post, it was triggered by the fact that many organizations would simply buy an SOA infrastructure product, such as an enterprise service bus (ESB), and immediately say that they had bought an SOA or implemented SOA. When, in reality, all that they had done was acquire an infrastructure product (like I said before), such as an ESB, that supports a particular set of standards, such as web-service standards. Really architecting a service-oriented system is so much more, because if you go back to my initial definition, something like an enterprise service bus only represents the SOA infrastructure piece. The architecting process starts when you identify the quality requirements that had to be met by the system, such as security, availability, performance, et cetera, and creating scenarios that provide concrete examples of what it means to satisfy each one of those requirements.

Implementing each scenario in the service-oriented world is most probably going to translate into the identification and design of services that truly represent reusable business or operational functionality, and the implementation of an infrastructure that is going to support those system qualities, like we said before, in terms of performance security, interoperability, availability, scalability, and any other quality attribute that you can think of. All this process does not happen just because you acquired an ESB product. In practice, the infrastructure ends up being a set of infrastructure components that go beyond an ESB and that together are what support the system quality attribute requirements. Really, keeping in mind that if you make an architectural decision

that promotes interoperability or modifiability, this can have a negative impact on other qualities, such as availability, reliability, security, or performance. Making these trade-offs is one of the hardest parts of architecting and designing any system. In the end, it could be possible that service-orientation is just not the right solution.

Shane: Okay. Next we'd like to ask, are there trends in SOA adoption that are making architectural considerations more important?

Grace: Absolutely. In the beginning, the use of service orientation was more internal to organizations. So, organizations were adopting service orientation as a way to integrate applications, to integrate data, or to optimize certain key business processes within the organization. Nowadays, there's a growing trend that has been documented in multiple surveys to use service orientation as an enterprise integration pattern that crosses organizational boundaries. For example, an organization could integrate services offered by the external organizations that form its supply chain in order to create some form of a dashboard to provide real-time data. What happens is that as service-oriented systems start becoming multi-organizational service-oriented systems, an architect has to realize that there will be parts of the system that will be outside of his or her control, such as service availability. This is going to become of even greater importance when you start thinking about cloud computing and third-party services that are offered as software-as-a-service. The architect, therefore, has to build in system elements to deal with this lack of control and make informed decisions on whether the use of these external systems is even feasible.

Shane: Okay, great. So what do you see as the practical application of this research?

Grace: So, the outcome of the research that we did was the report that you referred to, Shane, at the beginning. The main goal was to present the effect that the service orientation principles and their implementation have on system quality attributes. The first part of the report is a discussion of how common service-oriented principles, such as standardization, loose coupling, reusability, and composability affect system quality, such as interoperability, performance, and security.

The second part of the report focuses on common service-oriented systems elements, such as ESB, service registry and repository, messaging systems, business process engines, and monitoring tools, and the effect that these components have on system qualities.

The bottom line is that as an architectural style, SOA may be an appropriate solution in some situations, but there will be other situations where it is not appropriate, or it has to be used in conjunction with other technologies to achieve the desired system qualities. The architect is often at a conflict, because on one hand there are business and mission goals that dictate the quality attributes that are important for the system, and on the other hand, the architect that is using service orientation wants to adhere to service-orientation principles and leverage service

orientation to its advantage. Most often, we'll find out that it makes it difficult to achieve quality goals. We really hope that all this information is going to help architects of service-oriented systems make better architectural systems and understand the tradeoffs.

Shane: Grace, thank you very much for taking the time to join us today. As we mentioned earlier, the technical note is available on the SEI web site, www.sei.cmu.edu/library/reportspapers.cfm.

Thank you for joining us. This recording and a downloadable transcript are available at sei.cmu.edu/podcasts. If you have any questions, please e-mail us at info@sei.cmu.edu.