
# Composing Web Services: A QoS View

**Daniel A. Menascé** • *George Mason University* • *menasce@cs.gmu.edu*

An Internet application can invoke several services — a stock-trading Web service, for example, could invoke a payment service, which could then invoke an authentication service. Such a scenario is called a *composite Web service*, and it can be specified statically or established dynamically.

Dynamic composition of Web services requires service consumers to discover service providers that satisfy given functional and nonfunctional requirements, including cost and QoS requirements such as performance and availability.[1] In previous columns, I've examined how quality of service (QoS) comes into play for service providers, consumers, and parallel transactions.[2,3] Here, I'll show how it fits into composite Web services.

## Basic Model

Let's say that each Web service $i$ is characterized by the tuple $(F_i, Q_i, C_i)$, where $F_i$ is a description of the service's functionality, $Q_i$ is a specification of its QoS attributes, and $C_i$ is its cost specification. Although the Web Services Description Language is customary for describing a Web service's functional aspects, there's no universally accepted manner for specifying a service's QoS or cost structure.

A Web service's QoS attributes include metrics such as response time, throughput, security, and availability.[1,4] The exact definition and measurement process for each metric must be well-defined to give service consumers and providers a common understanding. Response time, for example, could be measured as an average over the past 15 minutes, as a 90th percentile, or as an array of average times for each 15-minute interval during the day. E.M. Maximilien and M.P. Singh proposed an ontology-based framework for dynamic Web service selection that provides a starting point for a QoS lingua franca.[1] However, they didn't address the fact that some QoS metrics, such as response time, depend on workload intensity level, which means a single value is not appropriate.

The measurement process for each QoS metric should consider

- what to measure (aggregates or percentiles);
- how to measure (frequency, intervals, and tools);
- who does the measuring (service providers or external independent agents); and
- where the measurements are taken (Web service access point, network edge, or at the consumer).[5]

A Web service's cost is often related to its quality. Faster, reliable, secure services will be more expensive, for example, but there could also be penalties associated with not meeting certain QoS goals or service-level agreements (SLAs).

## Composite Web Services Flow

Figure 1 illustrates some Web service composition scenarios. Each service is represented with an oval and connected via directed arrows (an arrow from Web service A to Web service B indicates that A invokes B). The figure shows five scenarios:

- *Probabilistic invocation.* A probability value $p$ on an outgoing arrow from A to B indicates that A invokes B with probability $p$ (see Figure 1a). If no value is indicated, the probability is assumed to be 1.
- *Parallel invocation (fork).* A Web service can require that a set of its successors be invoked in parallel (see Figure 1b).
- *Sequential activation.* A Web service is activated as a result of the completion of one of the set of mutually exclusive predecessor Web services (see Figure 1c).
- *Fastest-predecessor-triggered activation.* The first predecessor to complete activates a Web service (see Figure 1d).

 Published by the IEEE Computer Society 1089-7801/04/$20.00  

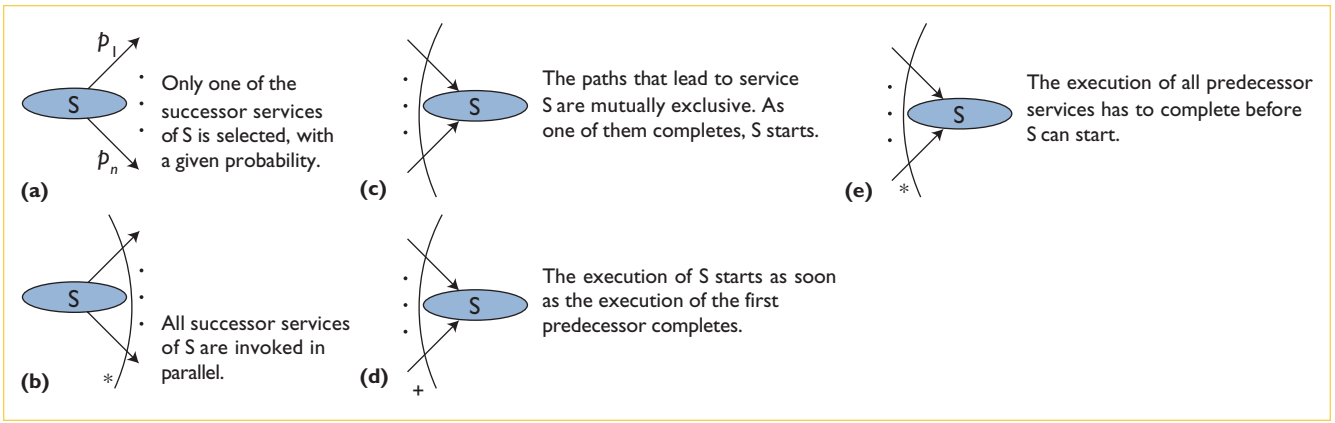


*Figure 1. Graphical notation for Web service composition. Ovals represent the actual Web service; arrows connect the services: (a) probabilistic service invocation, (b) parallel service invocation, (c) service invocation after one of several mutually exclusive predecessors finishes, (d) service execution after the first predecessor completes, and (e) service execution after all predecessors complete.*

- *Synchronized activation (join).* A Web service is activated only when all of its predecessor Web services have completed (see Figure 1e).

Figure 2 illustrates an example of a service composition using this notation. Web service A invokes service B with probability $p_1$ and C with probability $p_2$ ($p_1 + p_2 = 1$). Service C invokes service D with probability $p_3$ and service E with probability $p_4$ ($p_3 + p_4 = 1$). Finally, service F is activated when B finishes if A activated B, or when D or E complete if A activated C.

Let $t_i$ and $c_i$ be the execution time of Web service $i$ and the cost to execute that service, respectively. If $T$ and $C$ are the total execution time and total cost of the composite Web service, then for Figure 2,

$$T = t_A + p_1 t_B + p_2(t_C + p_3 t_D + p_4 t_E) + t_F,$$
$$C = c_A + p_1 c_B + p_2(c_C + p_3 c_D + p_4 c_E) + c_F.$$

For Figure 2, consider the following modification: if service A invokes B and C concurrently, then B and either D or E are predecessors of F. In Figure 3, service F is activated when its fastest predecessor completes.

The execution time and cost equations for Figure 3 become

$$T = t_A + \min\{t_B, t_C + p_3 t_D + p_4 t_E\} + t_F,$$
$$C = c_A + c_B + c_C + p_3 c_D + p_4 c_E + c_F.$$

The min term in the time equation comes from the fact that F is activated when the fastest of B and C plus either D or E complete. The cost equation does not have the min term because both branches are executed in parallel.

Figure 4 (next page) shows another modification of the Web service in Figure 3. Here, service F only starts when all active predecessors complete, thus B and either D or E have to complete before F can start. The execution time equation then becomes

$$T = t_A + \max\{t_B, t_C + p_3 t_D + p_4 t_E\} + t_F.$$

The max term comes from the fact that all predecessors must complete. The cost equation is the same as in the previous case.

## Numeric Example

Consider that each of the six Web services, A through F, has three different possible providers; Table 1 shows the execution time and cost per request for each provider per service. Imagine that a system for dynamic Web service selection would select the fastest provider for each service. Then, assuming that $p_1 = 0.3$, $p_2 = 0.7$, $p_3 = 0.25$, and $p_4 = 0.75$, and using the equations

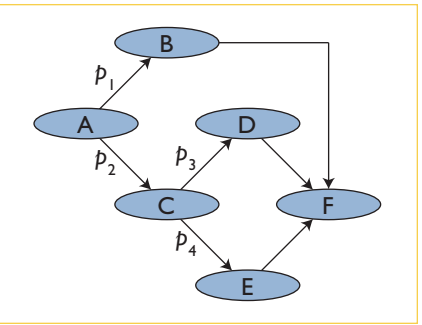


*Figure 2. A Web service composition. Web service A invokes B with probability $p_1$ and C with probability $p_2$. C invokes D with probability $p_3$ and E with probability $p_4$. F is activated when B finishes if B was activated, or when D or E completes if C was activated.*

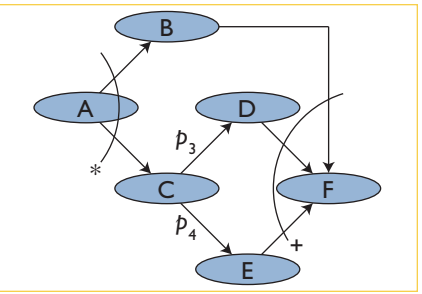


*Figure 3. Web service composition with parallel activation. If service A invokes B and C concurrently, then B and either D or E are predecessors of F. Service F is activated when the fastest predecessor completes.*

**Table 1. Execution time and cost per Web service per provider.**

| | Provider 1 | | Provider 2 | | Provider 3 | |
|---|---|---|---|---|---|---|
| Web service | Time | Cost | Time | Cost | Time | Cost |
| A | 2.0 | 1.0 | 0.5 | 0.7 | 0.8 | 0.5 |
| B | 3.0 | 2.5 | 0.9 | 0.6 | 4.1 | 2.8 |
| C | 2.5 | 1.7 | 3.0 | 2.0 | 2.8 | 1.9 |
| D | 4.0 | 2.8 | 4.2 | 2.5 | 3.0 | 4.0 |
| E | 3.5 | 1.5 | 1.8 | 0.8 | 2.0 | 0.7 |
| F | 3.0 | 2.0 | 3.5 | 1.5 | 2.9 | 2.4 |

**Table 2. Execution time and cost for the Web service compositions in Figures 2, 3, and 4.**

| | Figure 2 | Figure 3 | Figure 4 |
|---|---|---|---|
| Execution time | 6.9 | 4.3 | 8.0 |
| Cost | 5.6 | 7.0 | 7.0 |

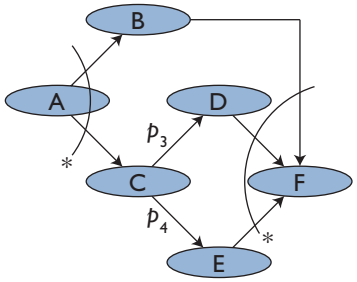


*Figure 4. Web service composition with synchronized activation. Service F only starts when all active predecessors complete, thus B, C, and either D or E must complete before F can start.*

previously presented, we can compute the execution time and execution cost for all three cases (Figures 2, 3, and 4).

As Table 2 indicates, the case in Figure 4 has a higher execution time because of the need to synchronize (that is, wait) for all active antecessors of F to complete. The case in Figure 3 provides the lowest execution time because F can be started as soon as one of its predecessors completes. The case in Figure 2 has a lower cost because services B and C are activated in a mutually exclusive way.

## Final Remarks

The simple example described here helps illustrate the importance of being able to select which services to use in a composite Web service. Other attributes, such as availability, security, and trust, must also be considered, but an important question is how to combine different QoS attributes.

Multiattribute utility theory could provide an answer.[1] Think of cost per request, availability, and response time per request as three attributes to be considered. We can assign a utility score between 1 and 5, with five representing the highest utility, to each attribute's value range. For instance, the scores for the availability attribute $A$ could be 5 ($A \geq 99.999$ percent), 4 (99.99 percent $\leq A <$ 99.999 percent), 3 (99.9 percent $\leq A <$ 99.99 percent), 2 (99 percent $\leq A <$ 99.9 percent), and 1 ($A <$ 99 percent). Scores for cost and response time could be assigned in a similar fashion. Then, we assign a weight between 0 and 1 to each attribute; the single combined score is computed as a weighted average of the scores for all attributes. The best composition of Web services can then be decided on the basis of the optimum combined score.

### References


1. E.M. Maximilien and M.P. Singh, "A Framework and Ontology for Dynamic Web Services Selection," *IEEE Internet Computing*, vol. 8, no. 5, 2004, pp. 84–93.
2. D.A. Menascé, "QoS Issues in Web Services," *IEEE Internet Computing*, vol. 6, no. 6, 2002, pp. 72–75.
3. D.A. Menascé, "Response Time Analysis of Composite Web Services," *IEEE Internet Computing*, vol. 8, no. 1, 2004, pp. 90–92.
4. D.A. Menascé and V.A.F. Almeida, *Capacity Planning for Web Services: Metrics, Models, and Methods*, Prentice Hall, 2000.
5. H. Ludwig, "Web Services QoS: External SLAs and Internal Policies: Or, How Do We Deliver What We Promise?", *Proc. 4th IEEE Int'l Conf. Web Information Systems Eng. Workshops*, IEEE CS Press, 2003, pp. 115–120.


### Acknowledgments

Daniel A. Menascé's work is partially supported by grant number NMA501-03-1-2022 from the National Geospatial-Intelligence Agency (NGA).

**Daniel A. Menascé** is a professor of computer science, codirector of the E-Center for E-Business, and director of the MS in E-Commerce program at George Mason University. He received a PhD in computer science from UCLA. Menascé is a fellow of the ACM. Contact him at menasce@cs.gmu.edu.