# Reliability of Service-Based and Agent-Based Systems

**Prof. Michael N. Huhns**
**University of South Carolina**
**huhns@sc.edu**

The use of services in *static* SOA-based systems has been successful in many application domains. However, the use of *dynamically* discovered, configured, deployed, engaged, and maintained services has not been successful. The problem is that current service standards, which are necessary for widespread usage of services, are unable to describe anything other than the simple syntax and formatting of service invocations; they are thus insufficient for characterizing the rich usage and interactions required throughout the *lifetimes* of service-based applications. Moreover, service-based architectures will need to become more flexible and accommodate peer-to-peer interactions, as well as client-server interactions.

The Web has been successful largely because its founding principles and protocols are simple and minimal. Also, when uncertainties arise, they are overcome by relatively simple indexing, ranking, and redundancy. None of these techniques has been exploitable for services. In addition, the simplicity of services applies only to their structure, and not to their function and behavior, which have mostly been ignored in service engineering.

Agents exacerbate the problems, while--surprisingly--also providing the only reasonable solutions to them. The autonomy of agent-based services makes them less predictable, but also enables them to self recover and to avoid deadlocks and livelocks, thereby making them more reliable. Their ability to learn can increase their robustness by being able to adapt to changing interaction environments, but also can increase their unpredictability. Their abilities to negotiate and reconcile semantics can enable them to reestablish connections and relationships among services and ameliorate uncertain execution environments. The peer-to-peer interactions of agents can improve the efficiency of agent-based services, particularly when they are deployed in clouds. Finally, agents can exploit redundancy.

The objective of service-oriented computing (SOC) is to construct software applications out of appropriate services available and executing in place anywhere across the Web. The applications, because they are naturally distributed, could be for any of the domains listed above. To achieve this objective at all requires that techniques for discovering and engaging services be developed. Doing this well requires that additional techniques be developed for ensuring desired quality of service (QoS) metrics.

**Google for Services.** Imagine that there are a large number of Web services, each described by a WSDL (Web Service Description Language) file. One might think that these could be indexed in the same way that Web pages are, so that search algorithms could help a developer find the services needed for a service-oriented computing application. That is, services could be clustered by applying text-mining techniques to WSDL files. This works well for Web searches, because keywords describe the partial semantics of Web pages and the PageRank algorithm helps in determining the most appropriate pages. However, this would not be successful for services, because WSDL files describe the inputs and outputs of a service, and provide only minimal information about the semantics of their function and behavior. Moreover, there is no equivalent to the PageRank algorithm for services, because services do not have references or pointers to each other.

All is not lost, however, because services could still be clustered into categories and middle agents could then manage and use the clusters to help locate possible services to satisfy requests from users and developers.  A procedure such as unit testing could then be used as a behavioral query tool to test candidate services and select ones that have the desired behavior.  A negotiation between the service requestor and providers could then ensue to establish an agreed upon QoS and formalize a contract.  The requestors and providers would commit to honor the resultant contracts.  These are agent abilities.