

# What is: Service-Oriented Architecture (SOA)?

**AUTHOR:** BRUCE THOMAS Principal Consultant *Optimization, Wellington*

**REVIEWED BY:** PETER BURGGRAAF Chief Information Officer *Farmers Trading Company, Auckland*

Do you use different billing systems for your time and materials work, and your product sales? Do you use different payroll systems for contractors, shift workers and permanent nine-to-fivers? Do you face interoperability problems when you replace your back-office application systems because proprietary interface standards and APIs are being used? If so, you may need SOA.

Service-Oriented Architecture (SOA) is an approach to IT systems design that facilitates the creation of discrete, generic services that can be easily reused by differing business processes. Take, for example, a nut-tightening machine. If Henry Ford had a ¼-inch nut-tightening machine, he might have used it in his factories to tighten the nuts on the bolts of door hinges of the Model T. However, there were probably other places on the Model T where ¼-inch nuts were needed. By making the nut-tightening machine (service) available to other processes, perhaps by putting wheels on it, Henry could have achieved greater production efficiencies and lower cost.

By design, SOA services are *loosely coupled*, meaning each service has very little dependency on other services to perform the task(s) required of it. They are also *event-driven*, meaning they react to input of a predefined format. Each service should use standards-based interfaces for receiving input and transmitting output. Those interfaces should send and receive *messages* that contain enough information for a service to perform its required task. As far as possible, a message shouldn't describe how a service is to perform its task. Nevertheless, it must specify the task that is required and contain whatever information that task requires. This constraint ensures that the business logic of the service stays within that closed system, enabling the service to remain the discrete expert of the task that it performs.

Consider the nut-tightening machine again. If such a machine existed, it would probably have a single button to push to tell it that you had a nut that needed tightening. That button on the control panel and the ¼-inch socket are standards-based interfaces. This means that if the machine was changed at any time, neither the operator nor the bolts would need to be recalibrated. The machine should know how to tighten the nut and when to stop. It shouldn't need the operator to push the control button for every turn, or half turn, of the socket that is required.

The nut-tightening service is also *extensible*, meaning that service enhancements can be added by increasing the range of inputs and outputs that can be processed. For example, control buttons and sockets could be added to enable the machine to also process ½-inch and ¾-inch nuts. This would allow the nut-tightening service to be used in a far greater range of business processes and potentially even

in an aggregation of processes that produce a completely new product – such as the Model A Ford.

SOA depends upon the consistent application of widely accepted standards for interfaces and messaging. Each service should have a set of interfaces available to be invoked by any other service. The interfaces need to accept messages that conform to a predetermined schema of call/request and content. Often, but not always, these interfaces and schemas are created in a web services environment.

## SOA using web services

Web services SOAs use the XML, SOAP, WSDL and UDDI standards. The use of these common standards means that services will be accessible to, and widely interoperable with, other services. It also means that a service can be replaced without the need for excessive re-engineering of the other services that interact with it.

XML stands for eXtensible Mark-up Language and is a standard for encoding information within a meaningful structure. XML allows tags to be described to define data structures and enable, for instance, the validation of data against predetermined rules. XML tags are enclosed in angle brackets, for example <heading>My Heading</heading>. XML is the foundation for information exchange between SOA systems in a web services context.

SOAP stands for Simple Object Access Protocol. It is an extensible XML messaging or communications protocol that allows applications to communicate using XML over HTTP (HyperText Transfer Protocol). XML firewalls can be used to inspect and filter unwanted SOAP and XML messages based on the tags used and their content.

WSDL, pronounced "wizzdil", stands for Web Services Description Language. It is the standard for describing SOA web services and interfaces. WSDL definitions describe how to access a service and what tasks it can perform: WSDL uses XML tags to prescribe the overall structure of requests and the names and data types of the data elements to be exchanged. When services are described using WSDL, organisations can choose to advertise those services.

UDDI stands for Universal Description, Discovery and Integration and is a directory

of web-service interfaces described by WSDL. UDDI also communicates using the SOAP protocol.

Let's use the nut-tightening machine analogy again. Henry owns a number of nut-tightening machines of various sizes and torques. If he publishes a UDDI standard for nut-tightening machines, he can register their services into a UDDI directory. Applications can then search the UDDI directory to find the organisation's nut-tightening interfaces. When the interface for a nut-tightening machine of the right specification is found, the application can instantly send requests to it because it uses a standard and well-defined schema and communications protocol.

Similarly, Henry can advertise that service (alone or packaged with other services) to other businesses through the UDDI directory. This could result in entirely new markets. He could sell nut-tightening and many other services to other businesses, including his car manufacturing competitors.

### What are the benefits?

SOA provides reusability for business services. SOA principles suggest that data can (and sometimes should) be separated from the business processes that the service delivers. This separation makes many operational benefits possible.

For example, data storage and related activities can be isolated into a database service that is made available to a number of other services. This would allow the underlying systems that deliver the database service to be optimised for only that specific function, which could provide cost and performance benefits. It would also permit the database system to be replaced with another SOA-enabled database system without any changes needed to any of the services that rely upon it. Similarly, standardisation of interfaces enables the underlying systems of any other service, such as billing, to be swapped out in a plug-and-play fashion.

An organisation can mitigate many of the interoperability and backward-compatibility risks of its technology roadmap by adopting SOA standard interfaces. It can expect reduced integration costs for new systems and services if its SOA is based upon common open standards such as those available for web services.

The key business advantages of SOA relate to the organisation's ability to reconfigure its core business processes. A prerequisite for this is a solid understanding of the business processes that an organisation has in place and the IT services that support them. Undertaking an SOA development provides the opportunity to explore and document business processes that may not have been well understood previously. This can provide opportunities to implement internal controls to satisfy the expectations of legislation and regulations like Sarbanes Oxley, Basle and the Privacy Act.

Once organisations understand their business processes, they can rationalise the services that support these by reducing the degree of "single-task orientation", making the services more generic and easier to reuse. These generic services can be used to address new opportunities and challenges.

Organisations can recombine SOA services to create new business processes and customer offerings. For example, consider a car parts warehouse business that has SOA-enabled product-ordering, billing and product catalogue services. It could combine these with a customer web portal and other services to achieve a degree of vertical integration in its value chain by linking ordering processes directly to the distribution process, eliminating the need for warehouse logistics.

Downstream retailers could also be bypassed, as the business could sell direct to consumers over the internet. The generic nature of the SOA-enabled services could also make it possible for them to be recombined to create a bookselling operation, a

computer sales operation or any other business activity that can be derived from the organisation's portfolio of services. In that respect, the portfolio of SOA services can be viewed as the organisation's IT core competency.

### What are the challenges?

There are some difficulties and challenges associated with the use of SOA. Designing a service isn't easy. Organisations wishing to develop SOA services must acquire the skills needed to develop services for easy reuse – these must be constructed in a way that is appropriately abstracted to provide the greatest flexibility. A framework must be carefully planned to ensure that SOA services are only created where they will provide real business value – through reuse and interoperability. The goal of SOA is not to create as many services as possible, but to reuse services as much as possible. For this reason, effective governance is necessary to ensure that new IT projects leverage the existing SOA infrastructure.

Another set of challenges relate to isolating functionality into discrete SOA services, as this puts increased demand on the network infrastructure. Processing that once occurred within a single "does everything" application is delegated to generic services around the network. This has the effect of generating additional network traffic and can introduce process latency. Also, the translation of process information into and out of XML can introduce a performance overhead. Network links may not have sufficient capacity to cope with the demands of SOA traffic. Numerous and verbose XML messages can place a heavy load on networks. Organisations need to design SOA with an eye on network capacity and the end-to-end performance of business processes.

Careful interface design is also essential. Although the changes to a service can be simplified by SOA because of the standardisation of platform-neutral service interfaces, tweaks to the design of a service's interfaces can cause problems. For example, interface changes result in the need to change all of the systems and services that interact with that interface. Accordingly, interfaces need to be correctly designed from the outset to ensure they are appropriately scalable and flexible.

Another problem relates to the open standards that are used. Standards can sometimes be underdeveloped or unclear. This can lead to differences in interpretation and implementation by developers and vendors, resulting in compatibility issues. Organisations should be careful to select standards carefully and diligently test for interoperability to avoid this risk.

The above notwithstanding, the potential benefits could encourage businesses to engage in large and complex whole-of-organisation SOA projects. Undertakings of this magnitude can create significant project and business risks. As we have seen, care must be taken in the design and development of SOA services. Poorly designed services and interfaces can result in performance issues and the need for rework. Because they are designed to be reused, unreliable and poor performing SOA services can affect multiple business processes, upsetting staff and customers. Consequently, the usual project management disciplines should be employed to ensure that SOAs are deployed cautiously and incrementally.

### Conclusions

Service rationalisation with SOA can be particularly advantageous to organisations that have a number of systems performing similar but unrelated tasks. For example, an organisation that has multiple payroll systems to manage shift workers, contractors and permanent staff might see benefit from creating an abstracted payroll service that can perform all those tasks. That payroll service could also be

combined with other services to create new business processes.

On the other hand, SOA is not the panacea for every problem. Because SOA offers businesses the ability to more easily repackage services, market advantages may be gained by first converting customer-facing services to SOA. This agility may not be as useful in back-office services, so those should be treated on a case-by-case basis. For example, if an organisation could not benefit from regularly altering or reusing its procurement processes, then procurement is an unlikely candidate for SOA.

Organisations that don't have a clear need for business process agility and service reuse are unlikely to benefit from SOA. Those organisations that *can* gain advantage from SOA must be careful to avoid the problems and pitfalls that can occur. As with all major IT initiatives, the best mitigation is through the use of skilled and experienced staff and careful management. Any SOA development should be based upon a sound business case that balances the expected advantages against the potential for problems.

The business case for converting any service to SOA should stand on its own merits. Services should not be universally migrated to SOA because of an organisational policy or philosophy that promotes SOA. Instead, there should be measurable business benefits that can be derived from the migration of each service. A good way to avoid the problems associated with "Big Bang" SOA projects is to incrementally develop and deploy services. These could be either entirely new services or existing services that can be immediately reused.

The principles and disciplines associated with SOA have been developing over a number of years. Consequently, there are a number of skilled and experienced practitioners who can help organisations down the SOA path. Organisations should seek help from those experts when determining whether SOA is appropriate for them and to devise an SOA strategy that mitigates the risks that exist.

### The bottom line

- SOA facilitates the creation of discrete, generic services that can be easily reused by differing business processes.
- Businesses can respond quickly to new opportunities by repackaging SOA services to create new business processes and customer offerings.
- SOA can overcome interoperability issues by introducing widely accepted open standards.
- Organisations that don't have a clear need for business process agility and service reuse are unlikely to benefit from SOA.
- Careful planning is essential – because they are designed to be reused, unreliable and poor performing SOA services can affect multiple business processes, upsetting staff and customers.

#### About the contributors:

BRUCE THOMAS is a Principal Consultant for Optimization. His primary focus is helping customers to extract the most value from their IT investment and to create new capabilities and opportunities. He has worked in IT leadership roles in government and the private sector and holds numerous IT credentials and an Executive MBA degree.

Contact: 04-470 5848, bruce.thomas@optimization.co.nz

PETER BURGGRAAF, the Chief Information Officer at Farmers Trading Company, has extensive experience in IT strategy development and execution. He shares his experience at conferences and publications around the world. The Farmers IT team has developed a SOA platform over the last few years, delivering great benefits to the organisation.

Contact: 09-272 6946, peter.burggraaf@farmers.co.nz

## Are you a neophiliac?

It sounds like a terrible disease. Lewd, too. But, if you are a regular reader of *IT Brief*, chances are you have it. You are a gadget junkie or techno freak. Possibly an early adopter and a marketer's dream. Neophiliacs are people who love everything new or novel.

Take a look around the office – those living life through a Blackberry or PDA are almost certainly afflicted. Researchers in Japan say they have identified a genetically determined enzyme associated with this novelty-seeking tendency. So, biologically speaking, this enzyme is the joy juice which keeps you vigilant for the new, different and innovative.

While most people have some element of this trait in their personality, there are those among us who are almost unstopably drawn to every new electronic gizmo. If you are one of these, you now have a great excuse – you can't help yourself.

According to a report from the Yamagata University School of Medicine in Japan, people produce different variations of a mitochondrial enzyme called monoamine oxidase A. The researchers found that one form of this enzyme was "significantly associated with higher scores of novelty seeking". In other words, people who produce that form of the enzyme are more likely to have novelty-seeking traits in their personality than others.

But before you neophiliacs start trotting out this new excuse, be aware that not everyone agrees that novelty-seeking is likely to have a genetic link. Colin Campbell, a professor of sociology at the University of York and an authority on consumerism, believes that the existence of novelty-seeking is a relatively new phenomenon, so it can't by definition be genetic.

Campbell dates the emergence of novelty-seeking to the beginnings of the industrial revolution in the mid-18th century. His theory is that with the industrial revolution there was a sudden shift toward rapid fashion changes, and he believes that it has accelerated ever since.

Whether or not people with neophilia are genetically predisposed, one thing is certain. There is a general desire for the new and different, and it has become a key driver in our culture and our economy. In other words, while some people may consider neophilia a disorder, if it didn't exist our economy wouldn't either. Cure neophilia and our capitalist economy will have one big headache.

Imagine if you bought new clothes only when you wore out the old ones. Or if you kept your car, television, stereo, laptop and camera until they died and could not be revived – and then and only then did you buy a replacement which had only the limited functionality which you really needed. Everything you bought was utilitarian.

Think about it – how much of our income do we actually need to live, and how much is spent keeping up with (or ahead of) the Joneses?