SOA MATURITY ASSESSMENT FRAMEWORK

Document history

Revision history

Version	Author	Date	Revision

Read by

	Reader	Date
1		
2		
3		
4		
5		

Approved by

	Subject Matter Experts Name	Date
1		
2		
3		
4		
5		

INDEX

SOA MATURITY ASSESSMENT FRAMEWORK 1

ASSESSMENT FRAMEWORK: CONTEXT 4
1 Introduction 5
1.1 What is SOA? 6
1.2 Overview 6
1.3 Defining concepts 7
1.4 Principles 7
1.5 Implementation approaches 9
1.6 Organisational benefits 11
1.7 Criticism 12
1.8 Extensions and variants 13

ASSESSMENT FRAMEWORK: QUESTIONNAIRE 15
2 Business Dimension 16
2.1 Vision and drivers 16
2.2 Business Process Architecture 17
2.3 Business Process Agility 18
2.4 Cost Model 18
2.5 Business & IT Partnership 18
2.6 Metrics 18
2.7 Enterprise Architecture 19
2.8 Information Model 19
3 Organization & Governance Dimension 20
3.1 Skills 20
3.2 IT Governance and SOA 20
3.3 IT Cost Model 21
3.4 Breadth of SOA Solutions 21
4 Method Dimension 22
4.1 Methodology 22
4.2 Modelling Techniques 22
4.3 Tooling 22
4.4 SOA Design Techniques 23
4.5 Processes for Software Development, IT Project Management, Service Development and QA 23
5 Application Dimension 24
5.1 Reuse 24
5.2 Integration 24
5.3 Technologies 24
5.4 Separation of Concerns 25
5.5 Other Parameters 25
6 Architecture Dimension 26

6.1 General 26
6.2 Reference Architecture 26
7 Information Dimension 27
7.1 Data Models 27
7.2 Mapping Rules 27
7.3 Mapping Definition 27
7.4 Business Object Models 27
7.5 Data Model Representation 28
7.6 Data Object Directory 28
7.7 Data Transformation between applications 28
7.8 Data Migration 28
7.9 Other Parameters 29
8 Infrastructure and Management Dimension 30
8.1 Usage Guidelines 30
8.2 Quality of Service (QoS) 30
8.3 Security 30
8.4 Monitoring 31
8.5 Infrastructure 31

ASSESSMENT FRAMEWORK: CONTEXT

1 Introduction

Through this framework, we are able to assess the level / maturity of Service Orientation of a company. Service Oriented Architecture, as a concept, sets the baseline for digital success, serving as a fundament for an organisation's digital eco-system. Don't get lost in the API (management) versus SOA discussion, as there has been a lot of talk about APIs vs. SOA, and there is still a lot of confusion about whether APIs are different or similar to SOA. The needle keeps swinging from one side to another, with API purists trying to detach themselves completely from SOA and the SOA die-hards claiming that APIs are just an extension of SOA.

From the perspective of SOA Software, having served hundreds of Fortune 1000 customers that are either using our products for only SOA, APIs and a growing number that are now using for both, the real answer is probably somewhere in the middle. APIs share essentially all of the basic architectural principals as SOA. SOA stands for "Service Oriented Architecture" and it fosters business through linked services. APIs embrace essentially the same goals, but are more open, easily consumable and support human-readable formats (JSON). APIs are also relatively more amenable for external consumption and have strong business affinity in that they can be branded and marketed as products.

For API programs to be truly successful, they need to be planned and designed right, have the right level of operational and policy controls and should be effectively monitored, analyzed and governed. The last word ("governed"), often scares away developers, but it is essential that you build your services to meet your business requirements, ensure that these services meet the security and compliance standards set by your respective industry/company and that changes in your infrastructure do not break your APIs or the thousands or potentially millions (wow!) of apps that are using your APIs.

Here are 6 points to keep in minds when assessing the similarities or differences between APIs and SOA and choosing an API Management solution:

1. While APIs are generally associated with REST/JSON and SOA is associated with XML and SOAP, SOA is more than just a protocol. SOA stands for "Service Oriented Architecture" and is an architectural best practice around building de-coupled applications and fosters service re-use. The API Economy is also all about creating services and making them available in an open fashion.
2. APIs can be used for both external and internal use-cases. The key difference between APIs and SOA being that APIs are more open, well documented and can be often self-provisioned, with little or no guidance, making them better suited for mass developer/partner consumption. SOA or XML services are pre-dominantly used for internal use-cases, though they are prevalent in a number of external B2B scenarios.
3. While APIs are currently associated with REST and JSON, there are other protocols like web-sockets, MQTT etc. that are gaining prominence for specific use-cases. Just as SOAP has it merits and limitations, REST will be replaced by something else. API providers should look at deploying API Management platforms that future proof (and past-proof with SOA support) their deployment with an unified infrastructure, rather than taking a tiered approach. You do not want to deploy another tiered infrastructure once another protocol surfaces
4. Both APIs and SOA services have to be secured, monitored, orchestrated, mediated and audited. These operational aspects of APIs and Services are often not associated with the respective message protocols, but are characteristics of the specific business service or operation they represent. In the services parlance, the common terminology applied to

these operations is Policies. You should look at deploying a unified policy management across all your services, irrespective of the protocol they are tied to, in this case whether they are APIs or SOA.

5. API and SOA are both services. They depend on other application and services, which are often managed by other developers, organizations and even completely different providers. Changes to these dependent services need to be managed and governed, as they could impact your API for SOA services. The architectural principles around security, compliance, policy management, monitoring and analytics are same or similar.
6. Both API and SOA services have a lifecycle, which spans beyond just versioning. You need to manage how they are planned, have tools that support the development cycle and integrated your SDLC tools, can be migrated with from different environments (ie from dev. to stage to production) and can be retired without adversely impacting your or your customers' business.

So while APIs have their own unique characteristics, at the core they are not that different from SOA and enterprises should try to use a common infrastructure to manage and govern the commonalities between them, rather than deploying redundant or "tiered" infrastructure.

1.1 What is SOA?

A service-oriented architecture (SOA) is a style of software design where services are provided to the other components by application components, through a communication protocol over a network. The basic principles of service-oriented architecture are independent of vendors, products and technologies. A service is a discrete unit of functionality that can be accessed remotely and acted upon and updated independently, such as retrieving a credit card statement online.

A service has four properties according to one of many definitions of SOA:

1. It logically represents a business activity with a specified outcome.
2. It is self-contained.
3. It is a black box for its consumers.
4. It may consist of other underlying services.

Different services can be used in conjunction to provide the functionality of a large software application. So far, the definition could be a definition of modular programming in the 1970s. Service-oriented architecture is less about how to modularize an application, and more about how to compose an application by integrating distributed, separately-maintained and deployed software components. It is enabled by technologies and standards that make it easier for components to communicate and cooperate over a network, especially an IP network.

1.2 Overview

In SOA, services use protocols that describe how they pass and parse messages using description metadata. This metadata describes both the functional characteristics of the service and quality-of-service characteristics. Service-oriented architecture aims to allow users to combine large chunks of functionality to form applications which are built purely from existing services and combining them in an ad hoc manner. A service presents a simple interface to the

requester that abstracts away the underlying complexity acting as a black box. Further users can also access these independent services without any knowledge of their internal implementation.

1.3 Defining concepts

The related buzzword service-orientation promotes loose coupling between services. SOA separates functions into distinct units, or services, which developers make accessible over a network in order to allow users to combine and reuse them in the production of applications. These services and their corresponding consumers communicate with each other by passing data in a well-defined, shared format, or by coordinating an activity between two or more services.

A manifesto was published for service-oriented architecture in October, 2009. This came up with six core values which are listed as follows:

1. Business value is given more importance than technical strategy.
2. Strategic goals are given more importance than project-specific benefits.
3. Intrinsic inter-operability is given more importance than custom integration.
4. Shared services are given more importance than specific-purpose implementations.
5. Flexibility is given more importance than optimization.
6. Evolutionary refinement is given more importance than pursuit of initial perfection.

SOA can be seen as part of the continuum which ranges from the older concept of distributed computing and modular programming, through SOA, and on to current practices of mashups, SaaS, and cloud computing (which some see as the offspring of SOA).

1.4 Principles

There are no industry standards relating to the exact composition of a service-oriented architecture, although many industry sources have published their own principles. Some of these include the following:

- Standardized service contract

Services adhere to a standard communications agreements, as defined collectively by one or more service-description documents within a given set of services.

- Service reference autonomy (an aspect of loose coupling)

The relationship between services is minimized to the level that they are only aware of their existence.

- Service location transparency (an aspect of loose coupling)

Services can be called from anywhere within the network that it is located no matter where it is present.

- Service longevity

Services should be designed to be long lived. Where possible services should avoid forcing consumers to change if they do not require new features, if you call a service today you should be able to call the same service tomorrow.

- Service abstraction

The services act as black boxes, that is their inner logic is hidden from the consumers.

- Service autonomy

Services are independent and control the functionality they encapsulate, from a Design-time and a run-time perspective.

- Service statelessness

Services are stateless, that is either return the requested value or give an exception hence minimizing resource use.

- Service granularity

A principle to ensure services have an adequate size and scope. The functionality provided by the service to the user must be relevant.

- Service normalization

Services are decomposed or consolidated (normalized) to minimize redundancy. In some, this may not be done, These are the cases where performance optimization, access, and aggregation are required.[15]

- Service composability

Services can be used to compose other services.

- Service discovery

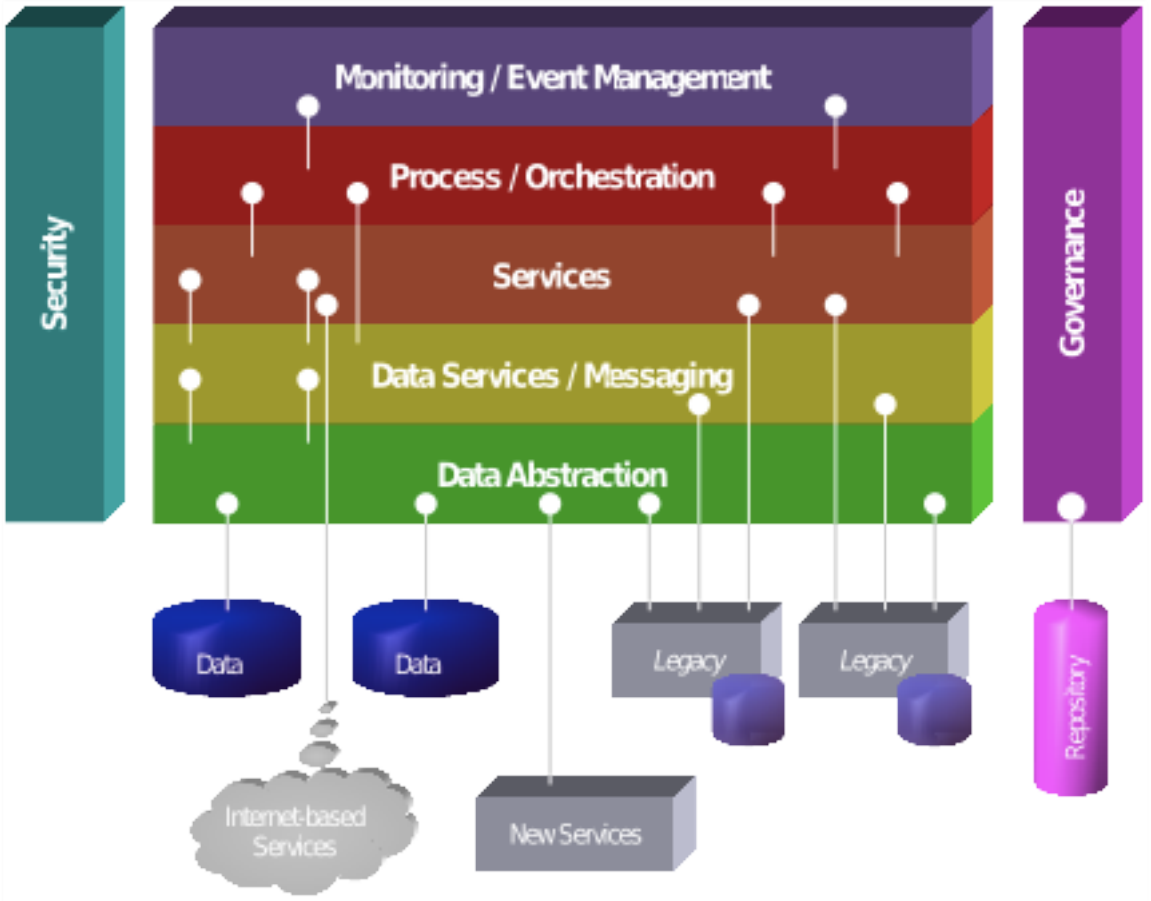
Services are supplemented with communicative meta data by which they can be effectively discovered and interpreted.

- Service reusability

Logic is divided into various services, to promote reuse of code.

- Service encapsulation

Many services which were not initially planned under SOA, may get encapsulated or become a part of SOA.



SOA Meta Model

1.5 Implementation approaches

Service-oriented architecture can be implemented with Web Services. This is done to make the functional building-blocks accessible over standard Internet protocols that are independent of platforms and programming languages. These services can represent either new applications or just wrappers around existing legacy systems to make them network-enabled.

Implementers commonly build SOAs using web services standards. One example is SOAP, which has gained broad industry acceptance after recommendation of Version 1.2 from the W3C[(World Wide Web Consortium) in 2003. These standards (also referred to as web service specifications) also provide greater interoperability and some protection from lock-in to proprietary vendor software. One can, however, also implement SOA using any other service-based technology, such as Jini, CORBA or REST.

Architectures can operate independently of specific technologies and can therefore be implemented using a wide range of technologies, including:

- Web services based on WSDL and SOAP
- Messaging, e.g., with ActiveMQ, JMS, RabbitMQ

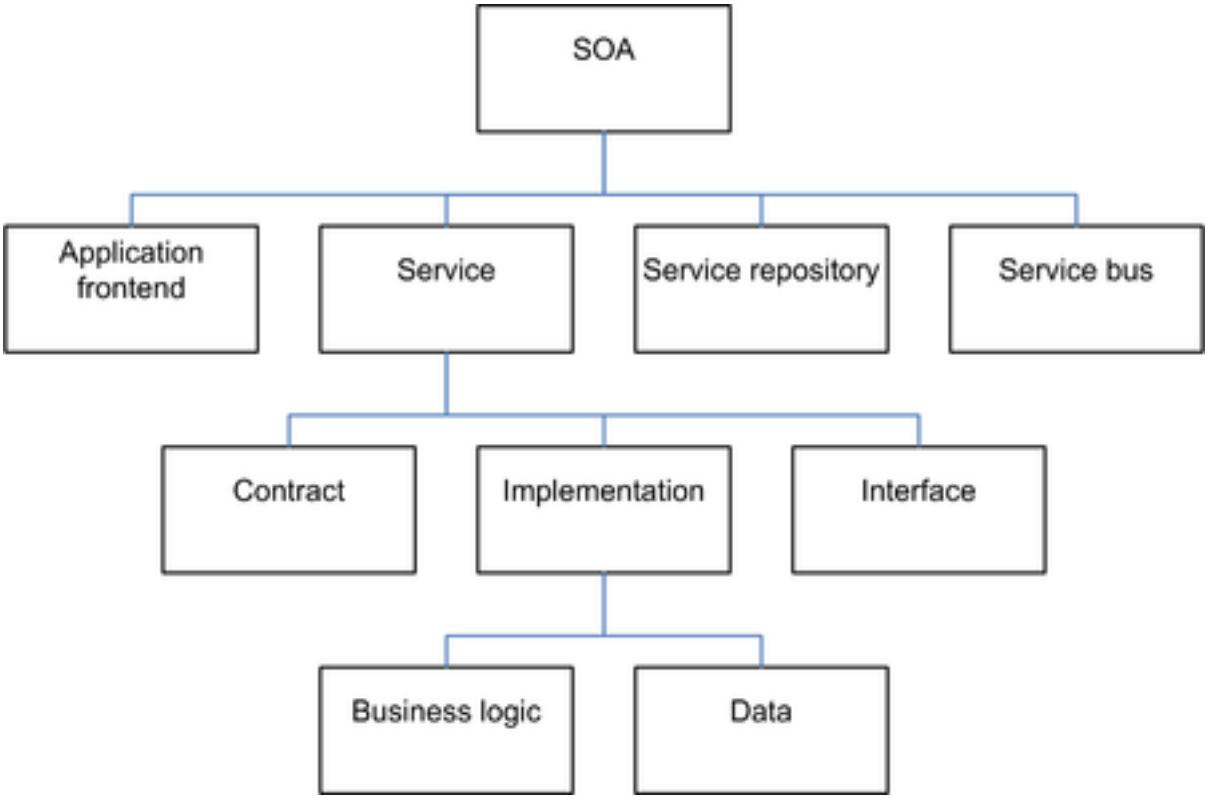
- RESTful HTTP, with Representational state transfer (REST) constituting its own constraints-based architectural style
- OPC-UA
- WCF (Microsoft's implementation of Web services, forming a part of WCF)
- Apache Thrift
- SORCER

Implementations can use one or more of these protocols and, for example, might use a file-system mechanism to communicate data following a defined interface specification between processes conforming to the SOA concept. The key is independent services with defined interfaces that can be called to perform their tasks in a standard way, without a service having foreknowledge of the calling application, and without the application having or needing knowledge of how the service actually performs its tasks. SOA enables the development of applications that are built by combining loosely coupled and interoperable services.

These services inter-operate based on a formal definition (or contract, e.g., WSDL) that is independent of the underlying platform and programming language. The interface definition hides the implementation of the language-specific service. SOA-based systems can therefore function independently of development technologies and platforms (such as Java, .NET, etc.). Services written in C# running on .NET platforms and services written in Java running on Java EE platforms, for example, can both be consumed by a common composite application (or client). Applications running on either platform can also consume services running on the other as web services that facilitate reuse. Managed environments can also wrap COBOL legacy systems and present them as software services.

High-level programming languages such as BPEL and specifications such as WS-CDL and WS-Coordination extend the service concept by providing a method of defining and supporting orchestration of fine-grained services into more coarse-grained business services, which architects can in turn incorporate into workflows and business processes implemented in composite applications or portals.

Service-oriented modeling is an SOA framework that identifies the various disciplines that guide SOA practitioners to conceptualize, analyze, design, and architect their service-oriented assets. The Service-oriented modeling framework (SOMF) offers a modeling language and a work structure or "map" depicting the various components that contribute to a successful service-oriented modeling approach. It illustrates the major elements that identify the "what to do" aspects of a service development scheme. The model enables practitioners to craft a project plan and to identify the milestones of a service-oriented initiative. SOMF also provides a common modeling notation to address alignment between business and IT organizations.



Elements of SOA

1.6 Organisational benefits

Some enterprise architects believe that SOA can help businesses respond more quickly and more cost-effectively to changing market conditions. This style of architecture promotes reuse at the macro (service) level rather than micro (classes) level. It can also simplify interconnection to—and usage of—existing IT (legacy) assets.

With SOA, the idea is that an organization can look at a problem holistically. A business has more overall control. Theoretically there would not be a mass of developers using whatever tool sets might please them. But rather they would be coding to a standard that is set within the business. They can also develop enterprise-wide SOA that encapsulates a business-oriented infrastructure. SOA has also been illustrated as a highway system providing efficiency for car drivers. The point being that if everyone had a car, but there was no highway anywhere, things would be limited and disorganized, in any attempt to get anywhere quickly or efficiently. IBM Vice President of Web Services Michael Liebow says that SOA "builds highways".

In some respects, SOA could be regarded as an architectural evolution rather than as a revolution. It captures many of the best practices of previous software architectures. In communications systems, for example, little development of solutions that use truly static bindings to talk to other equipment in the network has taken place. By embracing a SOA approach, such systems can position themselves to stress the importance of well-defined, highly inter-operable interfaces. Other predecessors of SOA include Component-based software engineering and Object-Oriented Analysis and Design (OOAD) of remote objects, for instance, in CORBA.

A service comprises a stand-alone unit of functionality available only via a formally defined interface. Services can be some kind of "nano-enterprises" that are easy to produce and improve. Also services can be "mega-corporations" constructed as the coordinated work of subordinate services. A mature rollout of SOA effectively defines the API of an organization.

Reasons for treating the implementation of services as separate projects from larger projects include:

- Separation promotes the concept to the business that services can be delivered quickly and independently from the larger and slower-moving projects common in the organization. The business starts understanding systems and simplified user interfaces calling on services. This advocates agility. That is to say, it fosters business innovations and speeds up time-to-market.
- Separation promotes the decoupling of services from consuming projects. This encourages good design insofar as the service is designed without knowing who its consumers are.
- Documentation and test artifacts of the service are not embedded within the detail of the larger project. This is important when the service needs to be reused later.
- SOA promises to simplify testing indirectly. Services are autonomous, stateless, with fully documented interfaces, and separate from the cross-cutting concerns of the implementation. If an organization possesses appropriately defined test data, then a corresponding stub is built that reacts to the test data when a service is being built. A full set of regression tests, scripts, data, and responses is also captured for the service. The service can be tested as a 'black box' using existing stubs corresponding to the services it calls. Test environments can be constructed where the primitive and out-of-scope services are stubs, while the remainder of the mesh is test deployments of full services. As each interface is fully documented with its own full set of regression test documentation, it becomes simple to identify problems in test services. Testing evolves to merely validate that the test service operates according to its documentation, and finds gaps in documentation and test cases of all services within the environment. Managing the data state of idempotent services is the only complexity.

Examples may prove useful to aid in documenting a service to the level where it becomes useful. The documentation of some APIs within the Java Community Process provide good examples. As these are exhaustive, staff would typically use only important subsets. The 'ossjsa.pdf' file within JSR-89 exemplifies such a file.

1.7 Criticism

SOA has been conflated with Web services; however, Web services are only one option to implement the patterns that comprise the SOA style. In the absence of native or binary forms of remote procedure call (RPC), applications could run more slowly and require more processing power, increasing costs. Most implementations do incur these overheads, but SOA can be implemented using technologies (for example, Java Business Integration (JBI), Windows Communication Foundation (WCF) and data distribution service (DDS)) that do not depend on remote procedure calls or translation through XML. At the same time, emerging open-source XML parsing technologies (such as VTD-XML) and various XML-compatible binary formats promise to significantly improve SOA performance. Services implemented using JSON instead of XML do not suffer from this performance concern.

Stateful services require both the consumer and the provider to share the same consumer-specific context, which is either included in or referenced by messages exchanged between the provider and the consumer. This constraint has the drawback that it could reduce the overall scalability of the service provider if the service-provider needs to retain the shared context for each consumer. It also increases the coupling between a service provider and a consumer and makes switching service providers more difficult. Ultimately, some critics feel that SOA services are still too constrained by applications they represent.

A primary challenge faced by service-oriented architecture is managing of metadata. Environments based on SOA include many services which communicate among each other to perform tasks. Due to the fact that the design may involve multiple services working in conjunction, an Application may generate millions of messages. Further services may belong to different organizations or even competing firms creating a huge trust issue. Thus SOA governance comes into the scheme of things.

Another major problem faced by SOA is the lack of a uniform testing framework. There are no tools that provide the required features for testing these services in a service-oriented architecture. The major causes of difficulty are:

- Heterogeneity and complexity of solution.
- Huge set of testing combinations due to integration of autonomous services.
- Inclusion of services from different and competing vendors.
- Platform is continuously changing due to availability of new features and services.

1.8 Extensions and variants

1.8.1 Web 2.0

Tim O'Reilly coined the term "Web 2.0" to describe a perceived, quickly growing set of web-based applications. A topic that has experienced extensive coverage involves the relationship between Web 2.0 and service-oriented architectures.

SOA is the philosophy of encapsulating application logic in services with a uniformly defined interface and making these publicly available via discovery mechanisms. The notion of complexity-hiding and reuse, but also the concept of loosely coupling services has inspired researchers to elaborate on similarities between the two philosophies, SOA and Web 2.0, and their respective applications. Some argue Web 2.0 and SOA have significantly different elements and thus can not be regarded "parallel philosophies", whereas others consider the two concepts as complementary and regard Web 2.0 as the global SOA.

The philosophies of Web 2.0 and SOA serve different user needs and thus expose differences with respect to the design and also the technologies used in real-world applications. However, as of 2008, use-cases demonstrated the potential of combining technologies and principles of both Web 2.0 and SOA.

1.8.2 Microservices

Microservices are a modern interpretation of service-oriented architectures used to build distributed software systems. Services in a microservice architecture are processes that communicate with each other over the network in order to fulfill a goal. These services use

technology agnostic protocols, which aid in encapsulating choice of language and frameworks, making their choice a concern internal to the service. Microservices are a new realisation and implementation approach to SOA, which have become popular since 2014 (and after the introduction of DevOps), and which also emphasize continuous deployment and other agile practices.

There is no single commonly agreed definition of microservices. The following characteristics and principles can be found in the literature:

- fine-grained interfaces (to independently deployable services),
- business-driven development (e.g. domain-driven design),
- IDEAL cloud application architectures,
- polyglot programming and persistence,
- lightweight container deployment,
- decentralized continuous delivery, and
- DevOps with holistic service monitoring.

ASSESSMENT FRAMEWORK: QUESTIONNAIRE

2 Business Dimension

2.1 Vision and drivers

To what extend is architecture supported by the senior management?

☐ Architecture is strictly an exercise by some individuals, typically in the IT department.

☐ Management supports the architecture effort with budget and mandate. But limited to a single operating units or project.

☐ Architecture is seen as a structural contribution and senior management treats it as such. Bringing architecture in the loop early on in changes.

☐ Architecture is strongly supported at the executive level as a key contribution to the business direction and communicated as such.

☐ An architecture driven approach permeates throughout the management level within the value chain. Senior management actively participates in the architecture process as stakeholder.

What are the major business drivers for this initiative? Please indicate the level of importance for each business benefit.

Business benefit	Importance (1 = Low, 2 = Medium, 3 = High)
Improving the time to market and IT responsiveness	
Lowering the Total Cost of Ownership (TCO)	
Reducing vendor lock-in by having standards based interoperatibility	
Lowering software development and maintenance cost	
Incremental roll-out to better control spend and risk	
Creating a flexible platform for future expansion	

To what extent is there a high level desired end situation regarding SOA?

☐ Local-heroes create a desired end-state at the start of a project

☐ Informal architects create a desired end-state based for a project based on previous work

☐ The organization-wide desired end-state has been defined

☐ The organization-wide desired end-state has been defined and is maintained

☐ The organization-wide desired end-state has been defined, is maintained and is continuously refined and improved

To what extent is there a plan to reach the desired end situation regarding SOA?

☐ Only during projects, a plan is made to align to a desired end-state

☐ Architects try to align project to the desired end state in an informal way

☐ An organization-wide roadmap of projects is defined in which actions and deliverables are plotted

☐ The organization-wide roadmap of projects is maintained by updating the actions and deliverables periodically

☐ The organization-wide roadmap of projects is actively improved by defining and updating current and future actions and deliverables

2.2 Business Process Architecture

Please select the answer that best applies to your current situation.

☐ The existing Business Process Architecture has not been documented.

☐ The existing Business Process Architecture has been defined, documented and managed.

☐ The existing Business Process Architecture has been documented and is completely up to date.

☐ The existing Business Process Architecture has been documented, is completely up to date and the ROI of the initiative is being measured.

☐ Other: _____

2.3 Business Process Agility

How agile are the existing business processes?

Business Process	Agility
	Low / Medium / High
	Low / Medium / High
	Low / Medium / High
	Low / Medium / High
	Low / Medium / High
	Low / Medium / High
	Low / Medium / High
	...

2.4 Cost Model

Please select one or more answers that best apply to your current situation.

- ☐ Current funding practices are not always transparent.
- ☐ The cost of the process, application and service portfolio is transparent.
- ☐ There is a charge-back agreement to bill the consumers of IT assets and services.
- ☐ The Total Cost of Ownership is clear, including software, hardware and maintenance.
- ☐ Other: _____

2.5 Business & IT Partnership

Please select one or more answers that best apply to your current situation.

- ☐ The IT organization is seen as merely a supplier.
- ☐ There is no trust between the business and IT organizations.
- ☐ There is a clear Business-IT partnership
- ☐ Other: _____

2.6 Metrics

Please select one or more answers that best apply to your current situation.

- ☐ The business SLAs are not related to the IT SLAs.

☐ The business SLAs are translated into IT SLAs (‘top down’).

☐ The IT SLAs are translated into business SLAs (‘bottom up’).

☐ The business and IT SLA’s mutually impact each other.

☐ Other: _____

2.7 Enterprise Architecture

Please select one or more answers that best apply to your current situation.

There is a formal Enterprise Architecture: Yes - No
The Enterprise Architecture is formally governed: Yes - No

2.8 Information Model

Are there different Lines of Business (LOBs) within the firm: Yes – No
If Yes, do these LOBs have their own business processes: Yes - No

Please select one or more answers that best apply to your current situation.

☐ There is a basic information model.

☐ The LOBs use different information models.

☐ The LOBs use a joint information model.

☐ Other: _____

Do the different LOBs share any vendors or partners: Yes - No

3 Organization & Governance Dimension

3.1 Skills

Please indicate the level of SOA experience within the team.

☐ Hardly any to none

☐ Academic

☐ Selected applications

☐ Applications on a program level

☐ Enterprise-wide

Please indicate which types of SOA training are available within the organization

☐ No training materials.

☐ SOA training is available, but limited to IT personnel.

☐ Training programs have been made available and have been adapted to both business and IT audiences.

3.2 IT Governance and SOA

Please select one or more answers that best apply to your current situation.

☐ SOA is not being governed formally (organization, roles & responsibilities, processes, standards, etc.)

☐ The SOA organization and roles & responsibilities have been defined.

☐ SOA processes and procedures have been defined.

☐ SOA standards and good practices have been defined.

☐ There is SOA governance on a program level.

☐ The SOA governance has been implemented enterprise-wide.

To what extend is the architecture department seen as the go-to-guys for problems?

☐ Limited or no communication between business and IT. Architects are seen as IT guys and are not considered as valuable counterpart.

☐ Architects have standards lists and keeps business to this basic equipment list. Business will try to by-pass architecture when not fit-for-purpose.

☐ Architects have an open door policy and will actively try to accommodate business requests. Business appreciates this and does leverage the knowledge of the architects.

☐ Architects actively works with the business to look for value add by understanding the needs and suggesting solutions that address them. Fitting within the overall architecture when possible,

but changing it when new knowledge requires. Architects are not seen as solely IT guys, but as business counterparts.

☐ There is a continuous interaction between business and IT with help of the architects on the most effective way to move forward. Both respect the strengths and weaknesses of one another and strive for a single goal.

The SOA governance model fits in with a broader IT governance model: Yes - No

SOA governance processes are being used for services at design time and at run time: Yes - No

3.3 IT Cost Model

Please select one or more answers that best apply to your current situation.

- ☐ There are no shared services deployed.
- ☐ Shared services are under development between some of the LOBs.
- ☐ The use of shared services is widely recognized within the firm's business and IT models.
- ☐ IT charge back mechanisms are in use.
- ☐ Other: _____

3.4 Breadth of SOA Solutions

Please select one or more answers that best apply to your current situation.

- ☐ There is no cross-organizational coordination relating to SOA.
- ☐ The use of SOA is widely recognized within the firm's business and IT models.
- ☐ SOA solutions cross organizational boarders: they extend to external parties/partners
- ☐ Other: _____

4 Method Dimension

4.1 Methodology

Which development methods and good practices are currently in use?

- ☐ Structured methodologies
- ☐ Rational Unified Process
- ☐ Agile Methodology
- ☐ Extreme Programming
- ☐ Other: _____

4.2 Modelling Techniques

Please indicate the relevant modelling techniques:

- ☐ Structured analysis and design
- ☐ Object-oriented analysis and design
- ☐ Service-oriented modelling with low business involvement
- ☐ Service-oriented modelling with high business involvement
- ☐ Model-driven architecture
- ☐ Other: _____

4.3 Tooling

Which design tools are currently in use?

SDLC phase	Environment			
	Mainframe	Client/Server	Web	Web Services
Analysis				
Design				
Development				
Test				
• Functional				
• Regression				
• Performance				
• Load/Stress				
Roll-out				

4.4 SOA Design Techniques

Are SOA design techniques in use? Yes - No

Is the development of SOA methods and practices being managed by an active community? Yes - No

Has the organization developed a repository for SOA methods and practices? Yes - No

4.5 Processes for Software Development, IT Project Management, Service Development and QA

Please select one or more answers that best apply to your current situation.

- ☐ Documentation of processes is still ongoing.
- ☐ Processes are documented and used repeatedly.
- ☐ Processes are known to all, contained within standards, procedures, tools and methods
- ☐ Processes help to effectively control and measure the software development effort
- ☐ The IT organization focuses on continuous improvement of processes by incremental change and innovative improvements.
- ☐ Other: _____

5 Application Dimension

5.1 Reuse

What types of reuse are being applied? Please select one or more answers that best apply to your current situation.

☐ Opportunistic – when a new project starts, the team realizes that existing components can be reused

☐ Planned – a team develops strategic components to be used in future projects

☐ Internal reuse – teams primarily reuse their own components

☐ External reuse – teams can license third party components (which typically are 1 to 20% of the cost of developing an internal component, plus the integration learning curve cost).

Are there any metrics regarding component reuse? Yes - No

5.2 Integration

How are the applications/systems within the enterprise integrated? Please select one or more answers that best apply to your current situation.

☐ Application architectures and topologies are monolithic with minimal separation of concerns between architectural layers or application tiers.

☐ Applications use minimal integration between other systems. Integration is usually implemented using point-to-point techniques.

☐ The use of SOA-enabling technologies – such as an ESB – is inconsistent across the enterprise.

☐ Service integration is achieved using an ESB in some but not all business units.

☐ Applications and systems are integrated Enterprise-wide

5.3 Technologies

Please indicate which SOA enabling technologies are in use within the enterprise.

☐ XML

☐ Web Services

☐ Enterprise Service Bus

☐ Shared Data Environment

☐ Service Registries

☐ Virtualization

☐ Other: _____

Please provide an overview of development languages and integration technologies within the enterprise.

Languages	Integration Technologies

5.4 Separation of Concerns

Please select one or more answers that best apply to your current situation.

☐ There is no distinction made between representation, business, data and technical layers.

☐ Most application architecture topologies have a separation of concerns both physically and logically in presentation, business logic, and data tiers.

☐ Service components of application architectures employ SOA patterns such as separation of concerns between logical and physical layers of the presentation and business logic.

☐ Application architecture is decoupled from infrastructure components.

☐ Application architecture supports dynamically reconfigurable business and infrastructure services and SOA solution for internal or external partner consumption.

☐ Other: _____

5.5 Other Parameters

Please select one or more answers that best apply to your current situation.
How would you rate the robustness of business-critical applications? Fragile – Robust – Very Robust

How would you describe the pace of change for the organization? Linear – Exponential

For the critical business applications, please give an estimate Time to Market:

_____

6 Architecture Dimension

6.1 General

How elaborate are the SOA initiatives? Please select one or more answers that best apply to your current situation.

☐ No SOA methods or practices.

☐ Limited use of formal SOA methods and practices can be observed.

☐ Formal SOA methods and practices are employed by multiple groups within the enterprise.

☐ Formal SOA methods and practices are employed across the enterprise supported by a formal governance process.

☐ Enterprise frameworks and practices supported using a formal SOA method and reference architectures across the enterprise.

☐ Service components are designed using formal methods, practices, and frameworks that promote the re-use of assets.

☐ Service components are designed using formal SOA methods, principles, patterns, frameworks, or techniques.

6.2 Reference Architecture

Please select one or more answers that best apply to your current situation.

☐ There is no reference architecture.

☐ The reference architecture is partly complete (limited layers).

☐ The reference architecture is complete.

To what extent are the current projects and activities aligned to the reference architecture?

☐ The projects we perform often have an ‘ad-hoc’ nature and targeted at solving the problems we have.

☐ In the first phase of our projects, an architect is often asked for information or to deliver some comments.

☐ Parts of the formal target architecture are followed which helps to reach the project goal.

☐ The target architecture is leading in the current projects; however, not all targets are reached because of deadlines.

☐ All IT project and activities are reviewed and need to be approved by architects in respect to the contribution to the target architecture before the they can be started. The architect is actively reviewing the results during the projects.

7 Information Dimension

7.1 Data Models

Please select one or more answers that best apply to your current situation.

☐ Information is replicated and redundant. Conceptual enterprise information model is absent.

☐ Information is shared across some applications using Extraction, Transformation, Load, Manipulate (ETLM) or message-oriented technologies. Initial data vocabularies are beginning to emerge.

☐ Business data vocabularies have emerged but are application or system-specific.

☐ Business data vocabularies are standardized within a business unit or process area.

☐ Business data vocabularies are standardized for use across the enterprise.

☐ Business data vocabularies can be expanded and enhanced as required to support new services, external partners, and business process reconfiguration.

7.2 Mapping Rules

Please select one or more answers that best apply to your current situation.

☐ There are no mapping rules in place to convert different data models.

☐ Mapping rules are in place, but are difficult to understand and/or badly maintained.

☐ Mapping rules are maintained by the business.

☐ Mapping rules are maintained by IT.

☐ Mapping rules are maintained by the infrastructure provider.

7.3 Mapping Definition

How are the joint data model or data model mappings defined? Please select one or more answers that best apply to your current situation.

☐ Based on programming objects within the API.

☐ Based on XSD schemes.

☐ Based on written documentation.

☐ Based on other computer based modelling tools.

☐ Based on other non-computer based modelling tools.

7.4 Business Object Models

Please select one or more answers that best apply to your current situation.

☐ Business Object Models are understood and managed by the business.

☐ Business Object Models are actually IT Object Models and therefore under the ownership of the IT teams.

7.5 Data Model Representation

Please select one or more answers that best apply to your current situation.

- ☐ Data models are represented based on taxonomies.
- ☐ Data models are represented based on ontologies.
- ☐ Data models are represented based on other high level logical representations.

7.6 Data Object Directory

Please select one or more answers that best apply to your current situation.

- ☐ There is no global directory or database of data objects.
- ☐ A global directory or database of data objects is maintained based on global identifiers.
- ☐ There are manual mechanisms for mapping data objects between different databases and directories.
- ☐ There are automated mechanisms for mapping data objects between different databases and directories.
- ☐ There are mechanisms to search global objects based on characteristics.

7.7 Data Transformation between applications

Please select one or more answers that best apply to your current situation.

- ☐ Transformations are performed by an Enterprise Service Bus (ESB).
- ☐ Transformations are performed by custom adapter as required.
- ☐ Transformations are performed based on an elaborate set of APIs.
- ☐ Transformations are performed by calling a service.
- ☐ Other: _____

7.8 Data Migration

Please select one or more answers that best apply to your current situation.

- ☐ Data migration between applications is a tedious process.
- ☐ Data migration is possible only between some applications.
- ☐ Data migration is possible between all applications.

7.9 Other Parameters

Is there a master data service? Yes - No

Does the organization have or are you developing a Business Information Model to standardize data and message formats and concepts across the enterprise? Yes - No

8 Infrastructure and Management Dimension

8.1 Usage Guidelines

What are the current guidelines regarding infrastructure use? Please select one or more answers that best apply to your current situation.

- ☐ Little or nonexistent operating support for the deployment of services.
- ☐ Service management and service security are partially implemented.
- ☐ Processes for service management and security have been published and are in use for the business unit or enterprise.
- ☐ Operating environment supports enterprise-wide service deployment. Identities of distributed users across departmental, organizational, and enterprise boundaries can be administered and managed.

8.2 Quality of Service (QoS)

Quality of Service implies non-functional requirements such as i.e. performance, scalability, availability, robustness and manageability. Please select one or more answers that best apply to your current situation.

- ☐ There is no real focus op QoS concerns.
- ☐ The SOA methodology addresses and requires compliance with QoS attributes.
- ☐ Tools and best practices are available to support QoS testing.
- ☐ Tools, processes and procedures are available to support QoS attributes and monitoring within the production environment.

8.3 Security

Please select one or more answers that best apply to your current situation.

- ☐ There is no SOA security policy.
- ☐ Transport security mechanisms exist.
- ☐ Service provider and consumer authentication security mechanisms exist.
- ☐ Message integrity and confidentiality mechanisms exist.
- ☐ Inter-agency security mechanisms exist.

8.4 Monitoring

What level of monitoring has been put in place? Please select one or more answers that best apply to your current situation.

- ☐ System monitoring (CPU load, RAM, disk space etc.)
- ☐ Dependency monitoring (web server processes, states, %CPU consumption etc.)
- ☐ Integration (third-party tracking)
- ☐ Business activity monitoring (KPIs, e.g. amount of successful transactions)
- ☐ Complex event processing (to gain insights from large amounts of data)
- ☐ Other _____

8.5 Infrastructure

Please select one or more answers that best apply to your current situation.

- ☐ Basic services (connectivity, messaging, routing, transformation, security) are available
- ☐ Advanced services (transaction management, metadata management, version resolution, orchestration, policy-based processing) are available.
- ☐ Typical management and monitoring components have been installed.
- ☐ There is a change management process and configuration management tools.
- ☐ The operational architecture supports the non-functional application and service requirements.