Misunderstanding SOA

Vladimir Dimitrov

Misunderstanding SOA: *The term Service-Oriented Architecture (SOA) was introduced in 1996 by Yefim V. Natiz – Gartner's analyst. SOA means different thing for different people. This paper is an attempt to explain what not SOA is and what is.*

Key words: *Service Oriented Architecture.*

"SOA IS SYNONYM OF WEB SERVICES"

Common error is that SOA means application of Web services technologies. So called "Web services" received its name from the fact that their methods are accessible via http. Services could be offered in many other ways not only as Web services.

A set of Web services is not SOA; it does not differ from Remote Procedure Call (RPC) via some technology like CORBA, DCOM etc. So, to that SOA uses Web services simply not true and Web services do not define SOA.

"APPLICATION THAT USES WEB SERVICES IS SERVICE ORIENTED"

The problem here comes from the way in which SOA is defined. SOA is an abstract model that can be applied to Web services, but Web services themselves could be used in non service oriented way. In the same way, SOA could be applied to any distributed architecture to be implemented service oriented solution, but this does not mean that the target architecture is service-oriented.

As abstract model SOA has many variants – every application of SOA is such a one. SOA benefits with Web services could be achieved only if Web services are designed and positioned in accordance with the principles of service orientation. This statement depends of our expectations from the application: even traditional distributed architecture can be called service oriented if we do not expect from it SOA benefits.

"APPLICATION THAT USES WS-* EXTENSIONS IS SERVICE ORIENTED"

WS-* extensions are the driving force for common acceptance of SOA, but application of these extensions as a part of some architecture does not make this architecture service oriented. Web services functionality does not play role for the orientation to services; what makes a solution service oriented is its architectural design.

Could be expected that most of applications that use WS-* extensions are service oriented. Today SOA depends on WS-* extensions support in the development of middleware software.

"IF YOU HAVE KNOWLEDGE ON WEB SERVICES, THEN YOU HAVE NO PROBLEMS IN SOA DEVELOPMENT"

Technical and conceptual knowledge on SOA are useful, but the real knowledge on SOA is the fundamental principles of service-orientation.

Service-orientation requires change in the way in which business logic and applications are viewed, separated and automated. Service-orientation means that Web services have to be designed and used following certain principles. Web services could be embedded in traditional distributed architectures. There they can be positioned to play significant central role in the processing or could be peripheral end points in the solution.

The way in which Web services are used in SOA is very distinct. The accent here is on the business logic capsulation and creation of abstraction levels of services. This requires expertise in technology and business analyze. SOA implementation requires much more knowledge than Web service technology.

"SOA IS ENTERPRISE ARCHITECTURE INTEGRATION (EAI)"

Another widespread definition of SOA is that it is EAI without all expensive EAI vendors. EAI is an attempt to be solved the enterprise integration problem. EAI tries to solve the problems of interoperability and data transfers among the applications using standardized tools. EAI has no success because:

- It data but not process centered;
- It cannot be synchronized with business process changes;
- EAI solutions are technically very expensive, special technical skills are needed, and exploitation costs are very high.

EAI is nothing new – it is simply a new technology following an existing way of thinking (architectural approach).

SOA is defined term and bounding it to different and contradictive ideas does not make it clearer.

"WITH SOA REUSE IS EASY"

Services encapsulate consistent business capabilities that are integrated in business processes to satisfy some business needs. The chance some service to be used somewhere else is very little. SOA is flexible and permits service changes to be easy done, but not out of service context.

"SOA IS 'BIG BANG' APPROACH"

SOA is pragmatic step-by-step approach. SOA permits legacy systems to be removed one-by-one and all available functionality to be replaced with new SOA interfaces to the underlying system.

"WITH SOA INTEGRATION IS EASY"

SOA does not change the main rules. Good service integration needs contracts to be modeled for use in business processes. This task is difficult and SOA does not solve it. When the designer has resolved this most difficult task, SOA can help the integration to be done easier, but not is SOA that make the integration easy.

"SOA IS SIMPLY NEW MARKETING BRAND FOR WEB SERVICES"

The term SOA is frequently used for marketing purposes. It is a buzzword in the raising wave of Web services application. The fact that SOA implemented with Web services puts under question validity of the term for some people. Even the "SOA support" is considered as relabeling of "Web services support".

SOA is not discovered by the media or marketing departments. It is a different architecture based on a set of certain principles. SOA is legitimated and to some extend mature term. SOA (SOA fundamental principles) could be implemented with technologies. Today Web services are the technological platform and that is why SOA and Web services are in so strong connection.

"SOA IS SIMPLY MARKETING TERM USED FOR DISTRIBUTED COMPUTING WITH WEB SERVICES"

Many people think so. The buzz around SOA hides its real meaning. Many migration paths, outlined by the vendors, start from traditional distributed computing with Web services. They are advertized as "SOA supported" and the result of this is confusing. SOA support in some cases is under question.

SOA is independent entity. It consists of a set of design principles. These principles are connected with are connected with the distributed computing platforms from the past, but they are different.

“SOA SIMPLIFY DISTRIBUTED COMPUTING”

SOA principles are relatively simple, but their application in the real life can be difficult task. SOA benefits potential can be achieved only through careful business analyses and application of service oriented principles during design time.

SOA implementations require more preliminary investigations than the solutions created with previous platform paradigms. This is partially influenced by the use of many Web service based technology platforms applied in SOA implementations.

When the service orientation is established and standardized in IT environment, simplicity can be achieved. Information integration with SOA can be achieved only when there are enough composing services and service orientation principles are well integrated in the organization.

“IN SOA EVERYTHING IS INTEROPERATING”

The marketing of SOA is responsible for this myth. Many people think that building service oriented solutions will transform their environments into federated enterprise. This could be achieved, but many investments, analyses and standardization are needed. Definition of communication levels in the framework of open Web services, and service-oriented (integration) architectures naturally abstract and hide everything private in the solution, its platform and its technology. The communication environment is predictable for all applications represented as Web services. This does not automatically standardize the information exchanged in the environment. SOA has good and not so good implementations. Quality SOA requires services to conform to common design standards, to be interoperable, to be reused etc; i.e. fully to be applied SOA principles.

DEFINITION OF SOA

There are business and technical points of view to SOA.

Business Point of View. Only business terms are used at enterprise architecture level. The focus has to be on the business needs – IT must serve the business, not the opposite. From the business point of view the main is service-orientation “SOA is a conceptual business architecture where business functionality, or application logic, is made available to SOA users, or consumers, as shared, reusable services on an IT network. ‘Services’ in an SOA are modules of business or application functionality with exposed interfaces, and are invoked by messages.” [1]

From business point of view, SOA analyses the business to define business domains and business processes. After that services representing these domains are defined. The services provide its functionality via message interfaces. They can be choreographed or orchestrated to implement business processes. SOA aim is to leverage business and IT to achieve flexibility – a capability for fast and efficient response to changes.

Technical Point of View. From technical point of view the focus is on the architecture. There is no standard definition of SOA. Broadly speaking SOA is architecture or architecture style build on loosely coupled, interoperating and compositing components or software agents called “services.” Services have well defined interfaces based on standard protocols (usually Web services, but in most definitions is mentioned that this is not the only possible implementation). Web services have QoS attributes (or politics) for interface usage by the service consumers. SOA definitions mention that the main communication template in SOA is “request/answer”, but many of them focus on asynchronous communications.

Definition of SOA. Now, it is time to join the technical and business points of view. First, we have to do distinction between an architecture style and its application. The SOA definition has to be applied at organizational level where SOA initiative encapsulates business logic into services. At project level SOA can be considered with technical details like service security and management.

We have to separate design goals (as loose coupling and architectural blocks) and limitations (as coarse grained services or politics based on interoperability). Architecture styles are defined in terms of components, their attributes, relations, rules and limitations applicable on them.

SOA is architecture style for building of systems with interoperable, coarse grained, autonomous components called “services”. Services provide processing (behavior) via message contracts on discoverable addresses called “end points”. Service behavior is governed by politics set outside the service.

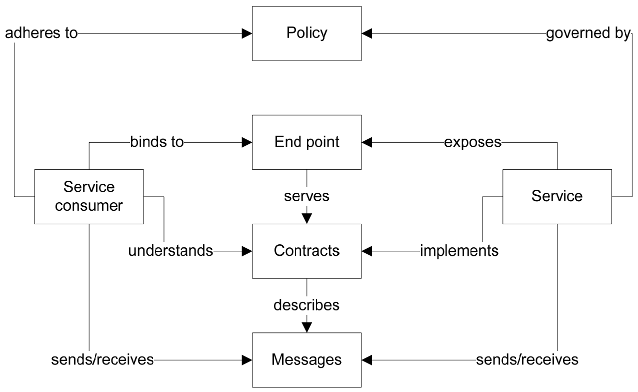

Figure 1 SOA components [2].

Service. The main SOA term is the service. In Marrian Webster service is defined as “a facility supplying some public demand.” The service is tightly coupled and supplies some functionality. Services have to be grain coarse parts of logic. The service has to implement at least all the functionality promised by the contracts that expose. The service has to be autonomous to certain level and to be self healing.

Contract. The set of all messages supported by the service is called “service contract”. The contract is closed set of messages supported by the service. The contract can be defined in advance for a group of participants. The contracts are interfaces to the services like the interfaces in object-oriented systems.

End Point. End point is an address, URI, particular place where the service can be found and consumed. Particular contract is provided at particular end point.

Message. Message is the main communication unit in SOA. Messages can be of type http, Get message, SOAP message, JMS message and even SMTP message. The main distinction between messages and the other communication forms (i.e. RPC) is that the message has header and body. The header is usually more common and can be recognized by the infrastructure and the framework components. This permits infrastructure components to send back answer messages and better to manage security.

Policy. One of the biggest differences between object-orientation (component-orientation) and SOA is the policy availability. The interface (contract in SOA) separates specification from implementation and policy specification dynamic specification static/semantic specification. Policy represents availability conditions of semantic specification for service consumers. The policy can be changed during execution time – it is external to the business logic. Policy specify dynamic features like security (cryptography, authorization), audit, SLA etc.

Service Consumers. The service is nothing if there is no one or nothing to consume it. So, the SOA picture needs of service consumers. Service consumer is software that interoperates with the service exchanging messages. Service consumers can be applications-clients or other services; the only requirement is conform to service contract.

CONCLUSION

SOA definition accent is on the interface: starting from the messages that are part of the interface; contract that is a set of messages; end point in which the contract is provided; and the policy, at the end, that governs end point behavior. In such a way SOA has four different components that are interfaces – OO has only one. Focus on the interfaces permits SOA to support loose coupling, composite components, reuse, and easy to be achieved the different design goals. This definition of SOA can be used from both technical and business perspectives.

Most of the mismatches with SOA are connected with the way of use of this term in the media and the marketing.

Most frequently SOA is mismatched with Web services application in distributed Internet architectures.

The most dangerous assumption about SOA is that service-oriented solutions are simple by nature, they are easy built, and their interoperability is automatic.

ACKNOWLEDGMENT

This work was supported by National Ministry of Science and Education of Bulgaria under contract VU-MI-109/2005: "Creation and development of Grid infrastructure for research and education at University of Sofia".

REFERENCES

[1] Marks E. A., M. Bell, Service-Oriented Architecture (SOA): A Planning and Implementation Guide for Business and Technology, Wiley, 2006.
[2] Rotem-Gal-Oz A., SOA Patterns, ISBN: 1-933988-26-6, Manning Publications Co., June 2007.

За контакти:

Доц. д-р Владимир Димитров, Катедра “Компютърна информатика”, Факултет по математика и информатика, Софийски университет “Св. Климент Охридски”, Тел.: 02 8161 594, E-mail: cht@fmi.uni-sofia.bg.

Докладът е рецензиран.