



“Service oriented architecture with LogicBlaze FUSE”

Analyst Report #017

Released 2006/07/06

<http://www.entivagroup.com/>

+1 800 613 8783

Contents

- **Introduction and Background:**

Provides introductory and background information about the subject of this report.

- **Term Sheet:**

List of important definitions and terms with which readers should become familiar.

- **Technology Management:**

Topics that are important when considering how a given technology will be handled and some potential implications.

- **Business Value:**

How a given technology is ready to provide value *today*, from a high level business perspective.

- **Analysis and Conclusions**

The closing statements and analyst perspective related to the future and potential of the highlighted open source technology.

Introduction and Background

Service oriented architecture (SOA) has recently emerged as a hot topic in the IT industry. With a dizzying maze of products, services, platforms and literature associated with the term, it is challenging to sort through the hype and the market-speak. Products that yesterday were categorized as anything from enterprise messaging to Web services anything are now being 're-branded' and marketed as a SOA offering. The truth is, service oriented architecture is more than a single product or even an application suite. Rather, it is the alignment of business and IT through both the conceptual and technical integration of data, processes and the people who use them, brought together in a loosely coupled yet fully integrated fashion. Still, there is a tremendous diversity amongst the different technologies under the SOA family umbrella, a fact which can be attributed to their inherent complexity.

The convergence of a set of components that fit under the SOA umbrella can be defined as a platform. All SOA implementations involve a platform in one form or another. Version 1.1 of LogicBlaze FUSE is currently available and is capable of serving as the technical underpinnings for the individual SOA infrastructure pieces that FUSE merges together. Even with the scores of SOA products available, there are very few integrated platforms offered. The general reluctance to assemble SOA platforms as a product can be partially linked to their architectural complexity. Another reason for the lack of SOA platform offerings is that despite the push for the adoption of commonly accepted integration standards, most individual SOA products do not use these standards to integrate with each other. Thus, integration of disparate SOA and Web services solutions is often a difficult prospect, as proprietary products are not typically designed for interoperability.

The emergence of open source in the SOA arena is marked by a new wave of projects that focus on different aspects of service-based architectures. LogicBlaze FUSE is the first complete SOA runtime platform that is distributed as open source. One of the primary benefits of an open source platform is that users are granted increased control over the platform as a whole. They can address issues that a commercial vendor might not even be aware of or might not deem meaningful enough to fix. An open source platform can be integrated with systems and run on platforms that might not be officially supported. Furthermore, there are a greater range of options in terms of who can perform the work: in-house, OSS community members or specialized system integrators. Oftentimes with closed source products, the vendor is the sole resource. In addition, open source solutions remove the additional licensing overhead associated with extending the architecture across an entire enterprise.

Last but not least, open, community-developed source code allows system architects, developers and other IT professionals to work with SOA technologies using a hands-on approach. Since proprietary solutions do not offer access to source code, they are not capable of providing value past their packaged, out-of-the-box state. In contrast, open source products transfer more authority into their owners' hands through the open, flexible access to source code.

Term Sheet

Business Integration: Capabilities associated with increasing the efficiency and value of the processes that are critical to the operation of a business, including improvement of information and service delivery.

Software Architecture: (According to IEEE 1471-2000) Software architecture is the fundamental organization of a system, embodied in its components, their relationships to each other and the environment and the principles governing its design and evolution.

Business Architecture: The supporting functions and processes that enable a business to carry out its specified line of work. This term includes both internal and external operations that execute strategy and contribute to its agenda. It is an interpretation of how an enterprise actually goes about accomplishing its mission.

Service oriented architecture (SOA): An approach to integration architecture based on the concept of a service and on the service as an architectural building block. Business and IT functionality can be represented as services within a distributed system; all of which can be pieced together in order to deliver composite functionality to either user applications or other services. Communication mechanisms between the services should be loosely coupled while supporting the use of explicit interfaces. As a methodology it applies successful concepts that are proven by Component-Based Design, Enterprise Application Integration, and Object-Oriented development.

Service Consumer: A requesting application or service that executes a service after sending a request that meets the requirements of a service provider's interface contract.

Service Provider: An application or service that advertises the availability of a service within a registry along with defined interface(s), so that a service consumer can discover, connect to and interact with the service.

Enterprise Service Bus (ESB): Frequently an abstraction of a standardized enterprise messaging system that enables messaging functionality to be used without writing explicit code to do so. It is beginning to emerge as a standard for integrating enterprise applications in a decoupled manner, at a coarse-grained service level, by leveraging key principles of service oriented architecture.

Java Business Integration (JBI): Essentially an approach for accomplishing business integration using a message-based, pluggable architecture. JBI is intended for use with an enterprise service bus with a standardized packaging of applications that consist of service consumers and service providers.

Business Process Execution Language (BPEL): An orchestration language focused on business processes, which uses Web Service Description Language (WSDL) 1.1 to

describe both outgoing and incoming messages. BPEL is extended from WSFL and XLANG, BPEL is an XML based language.

Loosely-Coupled: A relationship between components (applications, services, systems, etc.) that is capable of withstanding changes to participants. Loose coupling is based on the concept of 'near zero assumption,' which attempts to minimize the number of assumptions made about the interfaces of any given module(s). Thereby, diminishing the possibility that changes to either end will force changes in another component.

Service Orchestration: Consists of the ability to stipulate a sequence of services and execute that sequence on behalf of a [user or system] request.

Technology Management

A Complete Approach to SOA

At the core of LogicBlaze FUSE is the design principle of a single, accessible, packaged SOA solution. This principle is the bonding element for the integration of the separate software components taken from separate project communities. More complete than just a Web Services stack, it is a certified collection of software that can be assembled and employed as needed. LogicBlaze FUSE is a *comprehensive SOA runtime platform*. As such, it is directed at meeting the requirements for a variety of composite applications and event-driven processes, across a tightly managed platform of loosely-coupled services.

Because service oriented architecture, aside from being a technical entity, is also an umbrella concept, there are certain requirements that must be met in order for any SOA solution to be considered complete. Some of the main requirements are:

- **Performance:** consistent performance levels that guarantee things such as message delivery and the persistence of interrupted processes.
- **System and process level transparency:** visibility into the composition, operational status and events at both the message level and the business process layer.
- **Security:** covers authentication, validation and identity management for the application and service layers.
- **Reliability:** the ability to complete pre-defined processes and execute instructions in spite of network interruptions and/or service unavailability.
- **Manageability:** the ability to maintain control over components while they are in various runtime states.

LogicBlaze FUSE addresses the above by packaging an integrated platform that ensures users only have to deal with install and deployment procedures. In their disparate form, the different components that compose the FUSE platform are not directly related. Neither are they designed to fit smoothly into an integrated baseline. It is here that LogicBlaze applies a working knowledge of SOA principles and technology. Through the fusion of each part into a single package, the necessity for the end user to become an expert with the individual technical details is eliminated. Instead it is sufficient to learn how to manage each piece within the larger context of the platform.

The support model for FUSE also covers each part of the platform. Chairman and CEO of LogicBlaze Winston Damarillo describes the support model for FUSE as providing,

“...a single point of information and support for enterprise users of the FUSE distribution. We do this through a business and technical infrastructure called the CoRE Network. CoRE stands for ‘Community-oriented Real-time Engineering’, and it enables end user IT professionals to work closely with our developers in an environment that combines the best practices of open source development – transparency and collaboration

– with the requirements of enterprise IT – service level agreements, customer confidentiality, etc.”

The Enterprise Service Bus & SOA

One of the promises of SOA is that implementations will leverage existing data and systems, enabling the realization of the *agile enterprise*. An agile enterprise is one that is capable of responding to rapidly changing business and technical environments. Since there is no global standard shared by *all* applications, legacy systems, middleware and services, integration with existing as well as prospective application and service providers is critical. It is not sufficient enough to simply support Web Services. As much as they are sometimes billed as the panacea for connectivity and compatibility, Web Services are not capable of covering every real world requirement. There is a need to interact with messaging systems, C/C++ applications *and* Web Service endpoints.

At the core, LogicBlaze FUSE is based on the ServiceMix Enterprise Service Bus (ESB). In addition to supporting the Java Business Integration (JBI) specification, the ServiceMix ESB supports communication over standardized protocols such as HTTP, JMS and FTP among others. Generally, an ESB serves as a middle tier between service consumers and providers by mediating connectivity between the two. From a service consumer’s perspective, the only connection that is made is to the bus itself; nothing is known about the actual service implementation or the provider. Capabilities such as security and performance guarantees are implemented centrally in the bus, not in the individual applications and services that use the bus. Using this approach, a new level of manageability is achieved through the ability to maintain functionality in the bus instead of requiring service consumers and providers to do so.

In order for any ESB to support the realization of SOA within the enterprise, it must, among other things:

- Support the integration and management of enterprise services
- Separate service implementation from the service view
- Provide both a technical and business level view of service interfaces

Decoupling the service view from the actual implementation significantly increases the flexibility of the architecture. Service providers can be substituted for one another without causing interruptions to consumers. If an amendment to a standard is put into place or a lower cost provider replaces another, changes can be made to the architecture without large scale, negative effects.

The benefits of service decoupling are also demonstrated by investigating the alternative of directly connecting service consumers and providers. Figure 1-1 demonstrates the pitfalls of such an approach. While it is not entirely difficult to directly connect consumers and providers on an individual basis, as the number of consumers and/or providers grows, the number of links also expands exponentially. A large number of

point-to-point connections leads to a difficult to manage set of interface coupling and security models. And since each consumer must implement a form of routing logic, there is quite a bit of duplication without any reuse. Plus, a change to a service provider affects each and every registered consumer, making the prospect of change both complex and impractical.

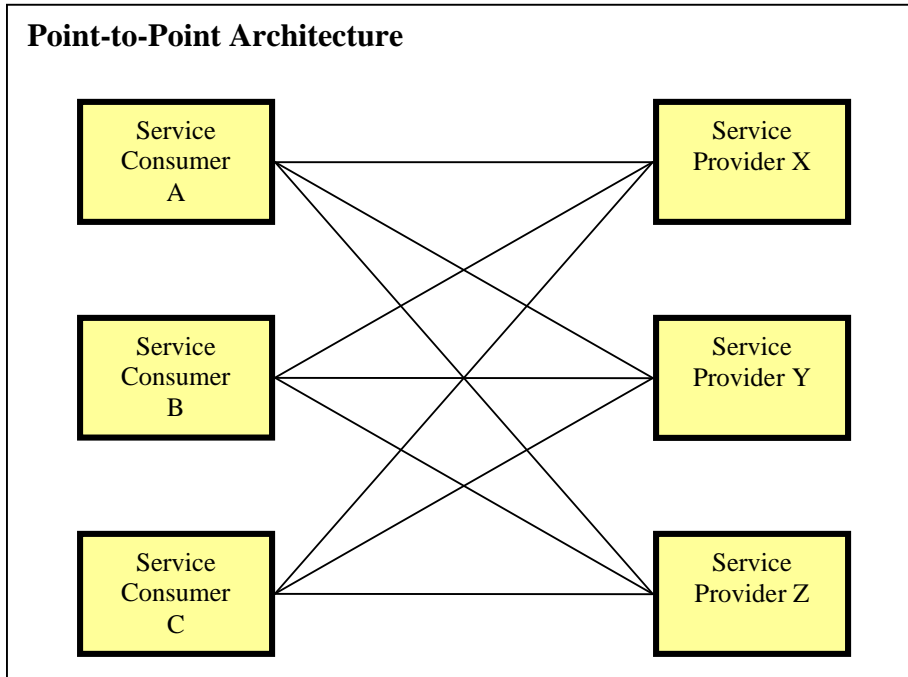


Figure 1-1

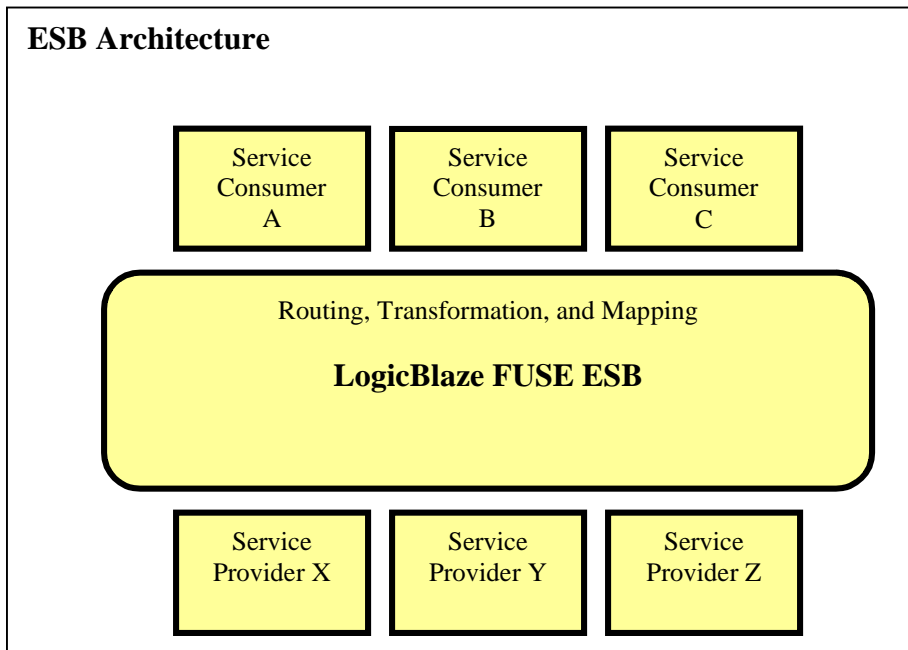


Figure 1-2

In the past, addressing the complexity of point-to-point architectures was done with the traditional Hub-and-Spoke architecture. The difference between the Hub-and-Spoke and an ESB-based architecture (as shown in Figure 1-2) like LogicBlaze FUSE, is that the ESB model is not a central, monolithic EAI stack but a conceptual unit that consists of a number of distributed entities.

The ESB is responsible for pulling together different components into what seems like a centralized component. However, there really is not an actual central hub and all processing is done close to appropriate component. For example, routing is performed wherever the routing component exists, not in some predetermined location. The LogicBlaze ESB provides the fixed management and administration framework for an essentially distributed integration infrastructure, which includes the responsibility of providing the appropriate transformation, routing and correlation services as needed when processing incoming and outgoing messages.

A Family of Products

According to Mr. Damarillo, LogicBlaze serves three general market requirements:

- **High-Performance, pervasive business integration:** By providing a reliable, high-performance JMS-based messaging infrastructure as an open source offering, LogicBlaze removes the financial barrier that has prevented pervasive implementation of "enterprise-caliber" messaging across any number of endpoints.
- **SOA and ESB:** LogicBlaze FUSE is the first open source distribution that addresses the comprehensive range of requirements for SOA runtime in a pluggable architecture that facilitates integration with current and future enterprise technologies.
- **Transactional Web 2.0:** LogicBlaze FUSE enables scalable, reliable access to the enterprise infrastructure from Web 2.0 applications such as Ajax, Ruby, etc., effectively bridging SOA and Web 2.0.

LogicBlaze maintains that the components selected for the FUSE platform are all from open communities that demonstrate active member participation and vigorous development cycles. A basic rule is that the more activity surrounding a project, the more likely there is a healthy rate of development and innovation. Directed in the proper direction, this activity can result in more useful and stable features being added over an extended period of time. Amongst communities with steady participation, there is a high probability that end users are contributing feedback that might be incorporated as additional features, bug fixes, defect resolutions, etc. in the future.

Another key value proposition presented by FUSE is the fact that it combines a number of quality open source components into an integrated offering, with each of those components distributed under the Apache Software License 2.0. Below is a list of the

main technologies and a description of their place within the LogicBlaze FUSE SOA platform:

- **ServiceMix ESB:** Forms the heart of the LogicBlaze FUSE SOA runtime environment. Ensures connectivity to a number of internal and external interfaces, routes communications using mediation routines, provides support for, among other things, transactional functionality and guaranteed messaging. Since ServiceMix is fundamentally a distributed runtime environment made to appear like a centralized system, it also serves as a single point of access for management and monitoring activity. The management console can be accessed through a JMX based interface. Security is provided by the Java Authentication and Authorization Service (JAAS) which implements a Java adaptation of the Pluggable Authentication Module (PAM) and supports user authentication.
- **Apache Ode:** The basis of the FUSE orchestration engine, consists of contributions from the Apache development community as well as code from the Sybase workflow engine and the PXE BPEL 2.0 engine. LogicBlaze FUSE 1.1 does not provide BPEL design tools or simulation mode.
- **ActiveMQ:** A commercial grade, open source messaging system compliant with the Java Message Service (JMS) API that includes support for persistence, the XA protocol (a standard that covers two-phase commits), TCP and SSL. Like the ServiceMix ESB, it can be deployed within a managed J2EE container environment or as a standalone process. LogicBlaze FUSE uses ActiveMQ to gain connectivity over an impressive number of protocols including SOAP, Java Connector Architecture (JCA) and Representational State Transfer (REST).
- **Apache jUDDI:** Based on the Universal Description and Discovery of Information (UDDI) specification, it is used as a UDDI 2.0 compliant registry for service discovery and lookup.
- **Jetty:** An HTTP server that meets the high-end performance requirements of AJAX (Asynchronous Javascript and XML) applications, which often result in heavy operational loads being placed on the server due, in part, to a high number of connections and immediate re-connections.
- **WebSphere Community Edition:** An open source J2EE 1.4 compliant application server, provided by IBM, based on the Apache Geronimo project. As of April 2006, a WebSphere-CE configuration is being made available as an optional version of FUSE. The two separate versions enable deployment independent of any application server (using the XBean-based deployment kernel), while facilitating deployment within other J2EE environments (e.g. JBoss).

Business Value

Cost Savings & Flexibility

Besides the fact that service oriented architecture has grown into an IT buzzword, a valid reason for all of the surrounding hype is its potential to significantly reduce maintenance and development costs, while at the same time, bringing loosely coupled, flexible integration of IT systems closer to realization. Beginning with clear interface definitions, complex business systems can be broken down into components enabling improved enterprise-wide processing modeling and monitoring.

In order to realize this unbounded flexibility and cost savings, an integration architecture must be constructed and maintained throughout its life cycle. Since SOA is not a packaged application, organizations often have to go through the tedious and error prone process of purchasing, manually creating and customizing the pieces of the architecture. By removing the burden of the upfront license cost associated with closed source software, LogicBlaze FUSE mitigates a great deal of the initial risk involved with an investment in an enterprise-class SOA platform. Additionally, since the code base is open, customization and extension is less resource-intensive.

With LogicBlaze FUSE provided under the terms of the Apache Software License (ASL) 2.0, LogicBlaze has made the effort to exclude any components which are distributed under conflicting licenses, such as the General Public License (GPL) or the Lesser General Public License (LGPL). As a result, users can be assured that they retain the right to modify FUSE source code and redistribute it as they see fit without restriction. Also, the ASL 2.0 frees organizations from having to submit to dual licensing terms, which mandate that any incorporation of source code into closed source products require the purchase of an enterprise license.

While the flexibility of open source solutions are cited as one of the main competitive advantages over closed sourced competitors, this characteristic takes on an added importance with an SOA suite. This is because, despite being powerful solutions, SOA platforms, such as LogicBlaze FUSE, are not a ready-made cure-all. The fact that solutions like FUSE are open source *does* reduce cost and development effort. However, it does not mean there is *no* effort involved with realizing the integration architecture. What it *does* mean is that within the scope of that effort, a flexible open source platform provides the ideal groundwork on which to proceed. Thus far, the combination of flexibility and cost is a value proposition that no proprietary solutions can match.

Ease of Integration

Powered by a commodity collection of runtime open source solutions, LogicBlaze FUSE offers the continued value of cost-effective and simplified integration with both commercial and proprietary elements alike. The transition of any IT asset into an existing organization takes resources in the form of time and energy with the main challenge of making sure existing pieces work with the newly introduced asset(s). Integrating FUSE

with existing systems is aided by the design of the platform, one that facilitates agility through a component-based, pluggable design. The end result is that FUSE components can be worked into other frameworks and platforms as an entire solution or in incremental small parts and vice versa.

The value of open source has already been proven within the enterprise infrastructure layer, so it is only natural that SOA, which depends heavily on infrastructure and middleware software, can benefit from the commoditized infrastructure. The LogicBlaze FUSE solution marries open source with open standards, eliminating the vendor-lock in that can make large scale integration such nightmare in the first place. The establishment of more industry-wide open standards has been driven by the recognition of their value in heterogeneous enterprise environments, where there needs to be common ground amongst a rapidly expanding set of technologies. Open standards enable vendor-neutral specifications to which a standing implementation must adhere, helping resolve integration by ensuring that other technologies adhere to the same criteria.

The inclusion of open standards within the FUSE platform supplies a number of options through which integration can be explored. The ActiveMQ messaging system itself supports a variety of clients from Java, C, C++, C#, Ruby, Perl, Python and PHP. SOAP support is covered by the inclusion of the JAX-WS, SAAJ, Xfire and Apache WSIF SOAP stacks. The concept of integration with external technologies is woven into the very essence of FUSE and is demonstrated by the various avenues by which the platform is exposed to the outside world.

Enabling Future Growth

SOA is an extremely complex subject area (design, management, implementation, etc.) that has only begun to take shape. A great deal of effort goes into creating the models and conceptualizations of the various levels of complexity involved. Both the proprietary and open source software industries have just begun to offer products and services that reflect this high level of complexity and are only beginning to understand the best manner in which to make this functionality available. Yet the lack of widespread, formal experience implementing SOA does not take away from the healthy number of well-built, enterprise-ready technologies, which also happen to be open source.

An important characteristic of LogicBlaze's SOA platform is that it allows users to leverage existing investments without tying an SOA approach to preconceived technologies/proprietary standards. The objective of FUSE as a platform is not to introduce an entirely new set of technologies to which users will be tied indefinitely. Instead the company's purpose is to create a solution that exhibits loosely coupled design, standards support and compatibility with other systems. By doing so, there is an underlying assurance that it can adapt no matter how large or small the overarching business and technological architecture around it grows or shrinks. Organizations are afforded the opportunity to 'start small' without having to resize efforts further down the line, because rapid growth has occurred. The applications and services built on the FUSE platform will grow in parallel to meet the needs of the owning enterprise, minus an

unbalanced cost structure that makes it expensive to remain aligned with business objectives.

The ability to scale and grow along with the change and growth of business operations can be referred to as *agile integration*. LogicBlaze realizes that, “The goal of SOA is to provide systems integration to support business requirements,” says Rob Davies, vice president of Product Development. Rob goes on to add that in order to avoid creating an inflexible solution that does not adapt well, LogicBlaze has taken, “the idea of agility a step further than many, because we think the SOA infrastructure itself should be agile. That is, we believe that middleware technologies should also be able to take advantage of unforeseen technologies or business opportunities and to adapt to changing requirements. Being based on open source is a big part of that agility.”

Analysis and Conclusions

The bottom line concerning LogicBlaze FUSE is that it is capable of meeting the objectives for most SOA plans while offering a lower total cost of integration. Any diligent company, developer or architect in the market for SOA solutions must consider FUSE as an option. Even without a long release history, the FUSE platform consists of a set of quality, open source technologies that have established their value individually, so their distribution as an integrated package does not involve an inordinate amount of risk. Recently, the company has garnered a fair share of attention as the first vendor to present a comprehensive runtime solution made available as an open source distribution. And far from being based on the pure novelty of the product, the notice paid LogicBlaze FUSE is warranted. This attention has shed light on the fact that open source software, in general, holds serious value within the SOA paradigm both now and in the future.

In general, the value proposition is too great for quality, community developed software, for it to be minimized within the context of SOA implementations. In the same way that commoditized infrastructure software, such as databases and application servers, has found a home in the enterprise IT landscape, so will technologies that are critical to service-based computing. By 2008, components such as ActiveMQ and ServiceMix will stand as cornerstones for enterprise open source SOA strategies. Likewise, integrated platforms such as LogicBlaze FUSE will continue to encapsulate the value of these technologies into a logical single point of integration that eases support, maintenance and continued use.

In the meantime, the course for LogicBlaze FUSE will continue with influence from: tightly coordinated cooperation from the project community, customer input regarding improvements/enhancements to the platform and the growth of an extended third party ecosystem. Areas of concern that LogicBlaze will address in its forthcoming release in the summer of 2006 include:

- Improved tooling
- Enhanced security
- Portal-based information delivery

Support for advanced enterprise level clustering and high availability architecture are slated to be addressed in future releases. Considering LogicBlaze's strong community participation and involvement with the efforts of a large number of talented developers, the FUSE platform has the opportunity to continue to mature into a stalwart open source product that can meet the needs of an emergent SOA market.

About Us

Entiva Group, Inc. is an independent analyst firm specializing in examining the ways in which Open Source Software can be used to improve the business operations of enterprises across the world. If you are interested in finding out how Entiva Group can empower your organization's SOA strategy using open source, contact us here: info@entivagroup.com.