F5 White Paper

Architecting Highly Available ESB Services—A Healthier, Happier SOA

Many Enterprise Service Bus (ESB) products can load balance the services they orchestrate; however, just because you can, doesn't mean you should.

By Lori MacVittie
Technical Marketing Manager, Application Services

Contents

Introduction

Many service oriented architecture (SOA) implementations include a messaging backbone known as the enterprise service bus (ESB). This messaging backbone not only integrates services, but also orchestrates, enriches, and secures them as part of an application or business process.

The nature of SOA is such that reuse is highly valuable, and this is also true of services managed by an ESB. A single service might be—and many would argue should be—consumed by multiple applications and processes. This makes services inherently more critical to the day-to-day business operations than their single, application-bound logical counterparts that are hidden within traditional applications. If a single service upon which multiple applications are dependent is unavailable, every application that relies upon that service is affected.

While the ESB is adept at providing reliable messaging, transformation, message enrichment, and service orchestration, it is not so adept at ensuring the availability of services it may orchestrate or integrate within a SOA. The ESB provides some measures such as load balancing and rudimentary failover to assist with maintaining service reliability, but these measures are largely afterthoughts.

In order to ensure that services are highly available and reliable it may be necessary to adapt a strategy that includes technology designed specifically to address the availability, reliability, and scalability issues: an Application Delivery Controller (ADC).

Availability and Reliability

Assuring Availability

An ESB generally includes the ability to provide load balancing for services on which it calls. The load balancing algorithms offered by most ESBs are rudimentary, generally lack more advanced load distribution options, and include only the most basic support for availability assurance.

Given the likely dependence of multiple applications on a single service, the availability of that service in an SOA becomes even more crucial. If a single service becomes unavailable or unresponsive, every application relying upon that service becomes unavailable. A successful SOA implementation must be able to assure availability to every service—and every application that depends on that service.

Availability assurance is an important facet of load balancing. While one of the primary reasons load balancing is used in any application scenario centers around performance and capacity, application availability has become equally important when architecting an application infrastructure. It's no longer enough that a service is merely available; it's also important for the application to consistently return correct responses within the required response times that meet the expectations of its business owners and users.

Over time, Application Delivery Controllers have evolved to extend load balancing functionality to include availability assurance. Not only are intelligent ADCs capable of distributing requests across a pool—or cluster—of applications and service instances, but they are also able to understand responses and take corrective action when an error occurs. For example, if a service returns a SOAP fault or some other error indication, an ADC can re-direct the request to another instance of the service to ensure that the transaction can be completed and the applications relying upon that service remain available.

Advanced Health Monitoring

Application Delivery Controllers are now able to help assure Service Level Agreements (SLAs) by determining, in real-time, which instance of a service best meets the performance requirements of a given application. An ADC can monitor performance characteristics and choose the process best suited to meet requirements. The resulting process can be based on factors such as number of oustanding requests, average transaction response time over a defined interval, and even individual server CPU and memory utilization.

An intelligent ADC such as the F5 BIG-IP® Local Traffic Manager™ (LTM) can further take advantage of the F5 iRules™ scripting language to customize the choice of services based on the unique needs of any given SOA. iRules provides a dynamic, flexible platform that extends the parameters through which services are selected to respond to a request.

By employing the capabilities of an ADC like BIG-IP LTM, the ESB is assured of the availability of services it orchestrates. The resources previously used to load balancing those services can instead be spent on its core competencies.

Scalability

Capacity

Another concern about availability should be the overall capacity of an ESB to handle application and user concurrency.

ESBs suffer many of the same high resource-usage issues as other XML-focused applications. Memory and processor usage is higher than many other applications for XML-focused applications such as services. This is due to the additional resources required to parse and interpret XML before it is executed. While streaming-based parsers have alleviated some of the issues with high memory consumption, the use of these is contraindicated in scenarios where message transformation is required. The result is that many ESB deployments cannot support the required concurrent user base for either mission-critical applications or high-volume, public-facing web applications.

The general answer to the inherent capacity constraints placed on any software is to architect a solution using multiple instances of the software in a clustered or load-balanced environment. You can increase capacity by distributing the load across a number of instances. Furthermore, when capacity needs be increased again, a load balancing solution such as an ADC can make the addition of a new instance of the ESB a seamless process requiring little service interruption and often, no interruption whatsoever.

While you can certainly take advantage of the load-balancing features offered by your ESB, the process of scaling up services using the ESB's load balancing options can be time consuming and can result in a sub-optimal implementation. Generally speaking, adding services to a pool of servers that are already providing services requires the ESB orchestration to be modified, which then requires the ESB to be re-tested and redeployed. Worse, retesting and redeployment is then necessary every time a new server is added to the pool of available servers. This makes it at best more difficult, and at worst impossible, to quickly deal with capacity issues. An ADC removes these obstacles, as servers can be added to pools transparently without modifying applications or orchestrating the ESB.

Key Benefits

- Offloads SSL
- Reduces management time and costs with centralized certificate management
- Ensures availability through load balancing and advanced health-monitoring
- Decreases the burden of TCP connection management overhead for services with TCP multiplexing
- Provides more flexibility and better control over the definition of "availability" through advanced health monitoring
- Improves performance of all services and applications through optimization

Performance

As previously referenced, ESB performance is very dependent on its XML processing capabilities. While there is very little an ADC can do to improve the

ESB parsing and execution performance, it can aid in other ways by reducing the need to perform other compute-intensive processing.

Processing SSL-encrypted traffic is compute-intensive. A service that requires SSL therefore negatively impacts ESB performance, increases the response time for that service, and degrades the overall responsiveness of applications using the service. By offloading SSL functionality to an ADC, there is little impact on the ESB, thus improving overall performance. In load balanced scenarios, enabling the ADC to offload SSL functionality also removes the need for the ESB to manage certificates.

Another function that can consume resources and thus adversely affect performance is connection management. The more applications using a service, the more connections and connection processing are required between the ESB and that service. Services can quickly become bogged down with the need to manage the increasing number of connections as more users access the application or site. The more connections, the more time and resources must be spent managing those connections, which can ultimately degrade both the overall capacity of the server as well as its performance.

Application Delivery Controllers employ a technique called TCP-multiplexing that maintains connections to the services and reuses those connections when possible, thus reducing the overhead associated with TCP connection management. Server capacity can often be increased by 30 percent when an ADC employs this capability.

Conclusion

While all modern ESB products offer the ability to perform load-balancing duties, these offerings are not as robust or evolved as those offered by application delivery controllers. An Application Delivery Controller provides additional benefits beyond load-balancing such as optimization, acceleration and performance enhancing features, resulting in a more scalable, reliable architecture.

Just because you can use the load-balancing options in your ESB doesn't mean you should. For truly reliable, available, and secure services consider adding an ADC to your application infrastructure

F5 Networks, Inc.
Corporate Headquarters
401 Elliott Avenue West
Seattle, WA 98119
+1-206-272-5555 Phone
(888) 88BIGIP Toll-free
+1-206-272-5556 Fax
www.f5.com
info@f5.com

F5 Networks
Asia-Pacific
+65-6533-6103 Phone
+65-6533-6106 Fax
info.asia@f5.com

F5 Networks Ltd.
Europe/Middle-East/Africa
+44 (0) 1932 582 000 Phone
+44 (0) 1932 582 001 Fax
emeainfo@f5.com

F5 Networks
Japan K.K.
+81-3-5114-3200 Phone
+81-3-5114-3201 Fax
info@f5networks.co.jp

WP-LOAD-BALANCING 05/08