zapthink white paper

THE MAINFRAME AS A FIRST CLASS SOA PARTICIPANT

ENHANCING THE LEGACY VALUE PROPOSITION WITH SOA

ZapThink, LLC • 108 Woodlawn Road • Baltimore, MD 21210 • info@zapthink.com • www.zapthink.com

THE MAINFRAME AS A FIRST CLASS SOA PARTICIPANT

ENHANCING THE LEGACY VALUE PROPOSITION WITH SOA

February 2007

Analyst: Jason Bloomberg

Abstract

Today's business imperative for IT is to do more with less, and to respond to ever-changing business requirements efficiently and inexpensively. It is vital, therefore, for organizations to get the most value possible out of legacy technology. The movement to Service-Oriented Architecture (SOA) demands that we reconsider the value of legacy and enhance its continued benefit to the business.

The business imperatives of agility and efficiency are core motivations for SOA, an approach for organizing existing IT resources to support changing business requirements in a flexible manner. For those organizations with legacy technologies like mainframes, SOA becomes a strategic approach for extending the value of of existing assets.

Simply incorporating the mainframe as a passive participant in a SOA implementation, however, does not fully leverage the inherent strengths of the mainframe—reliability, performance, scalability and security, to name a few. To fully exploit the mainframe with newer technologies and architectures, it is important to select an integration approach that provides industry standard interconnectivy without adding complexity, risk and cost.

A *Mainframe Service Bus* like DataDirect's Shadow can provide such an integration foundation. Shadow provides a unified architecture for mainframe transformation supporting direct SQL access, real-time events, Web enablement, and the capability to both publish and consume Web Services. This type of middleware support can change the role of the mainframe and enable it to become a full-fledged, active participant in a SOA implementation.

I. Transforming the Role of Legacy with Service-Oriented Architecture

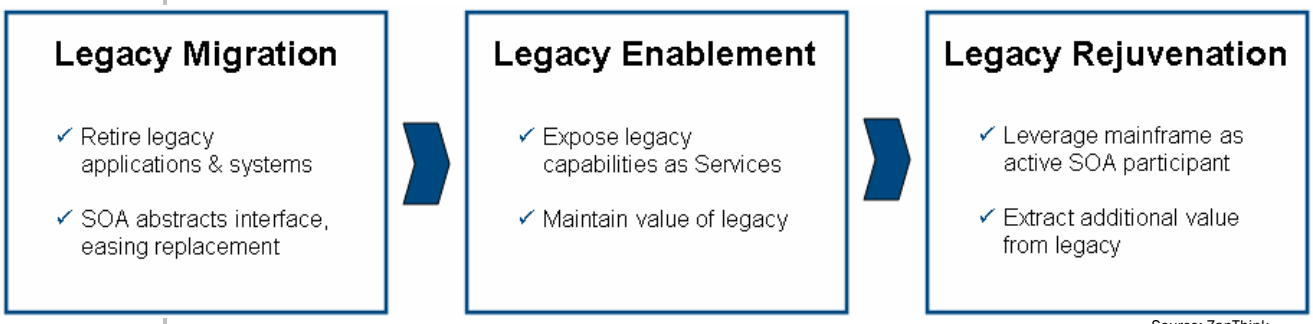
Legacy comes in many forms: custom-coded applications with long-lost source code, unsupported packaged applications, or mainframe-based programs with proprietary interfaces. The paradox of legacy is that for most enterprises, these systems are too old to justify ongoing improvements, but are nevertheless too important to retire. As a result, companies often feel that they must continue to maintain their investments even though the returns they receive from these older systems may diminish over time. Actually, legacy systems wouldn't be such a cause for consternation, if it weren't for the fact that so much business value resides on these systems—both in the form of essential data as well as critical business logic. Migrating these complex systems is costly and time consuming, and in cases where the mainframe is mission critical to the business, is simply not an option for some organizations. After all, if it were easy to retire these systems, there wouldn't be nearly as many of them around.

Migration, Enablement, and Rejuvenation of Legacy Systems

Many technologists have found it difficult to build or run new applications on legacy platforms, and many times older operating systems, enterprise applications, or middleware fail when made to perform new tasks they were not originally designed for. There's no denying that in some cases, the legacy system poses significant problems that lead to reduced business value because the software no longer meets business needs or is too inflexible to change. Some organizations also find themselves in the situation where certain legacy systems duplicate existing functionality, either with other legacy or with newer systems or applications. In those cases, they will likely seek to develop a legacy migration and retirement strategy.

Extracting the value out of legacy has many facets. The adoption of *Service-Oriented Architecture* (SOA) can simplify the process of legacy migration, as well as legacy enablement and legacy rejuvenation. An illustration of the relationship among legacy migration, enablement, and rejuvenation within the context of SOA is shown in the figure below.

Increasing Value of Legacy within SOA



Source: ZapThink

Legacy migration, while it can reduce the ongoing costs of maintenance, can also present substantial risks to the enterprise. Rarely is an older system entirely superfluous; far more often it still serves some valuable purpose at the time of its retirement. In those situations, it is essential to transition the consumption of

its capabilities to a replacement system in as seamless a way as possible. More frequently, however, organizations find that they don't actually require the retirement of legacy systems. Instead, they face issues relating to the usability of the legacy applications, or the applicability of those applications to evolving business needs.

In such cases, it makes more sense to put together a *legacy enablement* strategy that seeks to improve the value of the system in question without requiring its retirement. Web Services provide the flexibility to abstract mainframe functionality for immediate reuse or as a path for easier migration. For example, one migration scenario would be to wrap legacy transactions as a Web Service and in turn expose the Service via a Web interface, supporting an online application. It might also be feasible to migrate a duplicate instance of the legacy Web Service implementation to a distributed environment, enabling the eventual decommissioning of the original legacy transaction without any disruption to the business.

Such legacy enablement can clearly provide new business value, but in today's environment of rapid change, increased competition, and limited budgets, the business is calling upon IT to squeeze even more value out of legacy technology than migration or even enablement can provide. To meet today's IT challenges, it's necessary to *rejuvenate* the legacy environment, in other words, to obtain more value out of older applications than their original programmers intended. The greatest challenge to legacy rejuvenation involves making changes to the legacy environment itself; in fact, in many cases, such invasive changes are impractical or impossible. The secret to successful legacy rejuvenation, therefore, is to employ a non-invasive approach that extracts new value out of older systems without requiring risky changes to the systems themselves.

It's necessary to rejuvenate the legacy environment to obtain more value out of older applications than their original programmers intended.

The New Role of Legacy in the Service-Oriented Enterprise

The movement to legacy rejuvenation requires rethinking the way in which existing technology assets can meet new business requirements, and this rethinking is transforming legacy, not just as a technology but as a concept. Leading the movement to rethink the position of legacy in the enterprise is the rise of *Service Orientation*. Fundamental to Service Orientation is a separation between the business requirements and logic on the one hand, defined in the form of business processes, and the technology on the other, consisting of the infrastructure that underlies the Services layer of abstraction, including all legacy applications and the business logic they have locked up inside them.

In a properly architected SOA implementation, business Services represent the data and processes available to the business and the core functionality of the underlying systems. People then compose Services into *Service Oriented Business Applications* (SOBAs) that implement Service-Oriented Processes, configure SOBAs based upon the applicable business rules and policies, and then expose the SOBAs to allow other users to compose new SOBAs with them.

Service Orientation offers a way for companies to think differently about legacy in the first place.

The only way to satisfy an organization's increasing need for business agility is by making this shift to an environment that meets requirements in a flexible and adaptable manner. This change in thinking will transform the concept of legacy in the Service-Oriented enterprise, because Service Orientation offers a way for companies to think differently about legacy in the first place, by leveraging mainframe computing resources for more flexible consumption.

Organizations spend precious little of their time and budget building new applications. Part of the reason for the lack of emphasis on new application development is the fact that they must create each new application in isolation from the previous applications they've built, resulting in a new piece of the IT puzzle that they must integrate in turn with other components. It's clearly important to build new applications in such a way that not only reduces the cost of development, but also maintenance, over time.

One of the most important benefits of SOA is the ability to create new SOBAs from existing Services. In other words, Service reuse becomes the central theme, rather than application integration. As they create new Services that they can reuse within new SOBAs, organizations can realize significant return from their SOBA investment, in terms of both cost efficiencies as well as other sources of business value. As a result, the economics of SOBA improve over time, as companies build and reuse an increasing number of Services.

II. Extending the Legacy Benefit with SOA

Leveraging mainframe integration as part of a SOA implementation offers important capabilities, including the opportunity to inherit mainframe application attributes like predictable performance, reliability, and availability. A secondary consideration is the abstraction of mainframe capabilities by leveraging interfaces that support industry standards, helping to alleviate the skills gap relevant to mainframe environments. In fact, the transformation of legacy data and application functionality enables multiple approaches for leveraging mainframe-based capabilities in SOBAs.

SOA and Mainframes

Any mainframe-based organization who is implementing SOA will be particularly interested in how well they can leverage their mainframe-based capabilities in their SOA infrastructure.

Any mainframe-based organization who is implementing SOA will be particularly interested in how well they can leverage their mainframe-based capabilities in their SOA infrastructure, as they plan and build SOBAs. For example, System z and z/OS attributes include five-nines availability, support for high volumes of concurrent users and transactions per second (tps), as well as high reliability, performance, and security—all capabilities that a SOA implementation can and should leverage. Furthermore, the System z hardware platform offers Logical Partitions (LPARs) that enable the distribution of the physical mainframe into multiple virtual mainframes, and sysplex that enables multiple physical mainframes to appear as a single unit,

z/OS mainframes are well-suited to support the SOA infrastructure because they typically dispatch work through Task Control Blocks (TCBs) and special workloads as Service Request Blocks (SRBs) for very high performance, Mainframes can also contribute to the security of the SOA environment through the Security Authentication Facility (SAF), which provides a common API for mainframe applications to use the services of an extended security manager (including RACF, ACF/2, and TopSecret. In addition, Resource Recovery Services (RRS) provide the foundation for the mainframe XA 2 Phase Commit protocol.

Service Intermediaries To Facilitate SOA

Service Orientation represents a movement toward the decentralization of application capabilities, as enterprises enable a broad range of users to consume Services and compose them into SOBAs. Such a trend, however, isn't new. The transition from the centralized mainframe environment to client/server in the 1980s followed an earlier IT decentralization pattern, which was then followed by the Internet's return toward centralized application logic that leveraged the thin clients of the Web.

The mainframe remains a constant, providing much of the core IT value in the organizations that depend upon them to run their businesses.

Throughout these frequent swings of the pendulum, however, the mainframe remains a constant, providing much of the core IT value in the organizations that depend upon them to run their businesses. As a result, there has always been a strain between centralized, host-based capabilities and decentralized, distributed computing architectures. As we move to SOA, we face the same issues all over again, only now our technology and business practices are that much more mature.

Before the advent of SOA, the mainframe was at best a reluctant participant in the distributed computing environment. Early terminal emulation and screen scraping techniques were inflexible and tightly coupled. With the advent of more sophisticated screen scraping, API-based connectors, and now Web Services, however, the mainframe is increasingly able to participate as a first-class citizen in the distributed computing world.

Typically, however, this participation is in the role of a server, and the various value propositions of legacy systems within SOA depend upon the role of the mainframe as a server. But once mainframe applications can consume as well as provide Web Services, in addition to providing additional infrastructural capabilities essential to SOA including transformations, management, and security functionality, it's now becoming possible to consider the mainframe as a Service intermediary.

It's now becoming possible to consider the mainframe as a Service intermediary.

Service intermediaries provide the infrastructure necessary to ensure loose coupling between Service providers and consumers at the endpoints of each Service interaction. Intermediaries offer transformations at the wire protocol, message protocol, and semantic layers, as well as security, management, and business process management capabilities. Enterprise Service Buses (ESBs) are examples of distributed Service intermediaries that provide a messaging middleware infrastructure.

III. The Mainframe Service Bus

It's now possible for the mainframe itself to take this intermediary role, what might be called a *Mainframe Service Bus* (MSB). A MSB is a mainframe transformation engine which makes mainframe assets available to distributed systems via standards-based interfaces. Key attributes of MSBs include comprehensive support for data sources, (databases, business and screen logic), scalability, security, and other mainframe capabilities as listed above.

Leveraging mainframes as Service intermediaries might limit the dependency on adapters for point-to-point integration, as well as reducing the overall complexity and risk of the SOA implementation. Fundamentally, the Service-Oriented Architect must no longer consider the mainframe to be merely a back-end system in need of rejuvenation. Instead, it's possible for the mainframe to be a full participant in the plans for the SOA infrastructure.

It's possible for the mainframe to be a full participant in the plans for the SOA infrastructure.

MSBs provide connectivity between mainframes and distributed application platforms and middleware including ESBs by leveraging the transformation of mainframe data, business logic, and screen-based applications via industry standards, including Web Services, XML and SQL standards. MSBs also offer enterprise operational characteristics including predictable performance, scalability, reliability and manageability, integration with existing security infrastructures, and enterprise management and change management capabilities.

The MSB must also enable the transformation of a comprehensive range of data, business logic, and screen-based assets into reusable industry standard components, as shown in the table below:

Mainframe Service Bus Transformation Capabilities

	Data Logic Integration	Business Logic Integration	Screen Logic Integration
Web Services	X	X	X
SQL Access	X	X	X
XML-Based Events	X		
XML Web Enablement			X
Compatible Environments	DB2 IMS/DB| IDMS VSAM Adabas	CICS/TS IMS/TM IDMS ADS/O Natural	CICS/TS IMS/TM IDMS ADS/O 3270

Source: DataDirect

MSBs provide for interoperability and reusability of mainframe data and processes as well as business and screen logic in support of the SOA infrastructure by leveraging industry standards, including WSDL and SOAP for application transformation, in addition to SQL and XML events to provide the ability to treat mainframe data as a virtual relational database. MSB interoperability should also be bidirectional, enabling the mainframe to act both as a provider and consumer of Services.

Enterprise Operations and Security on the MSB

MSBs also enable mainframes to offer their core capabilities to operational SOA environments by offering predictable performance, scalability, manageability, and reliability. MSBs can exploit the Workload Manager (WLM) on z/OS, enabling granular control over quality of service (QoS) and provide the foundation for scalability and workload distribution throughout a sysplex with thread technologies including TCBs and SRBs, MSBs should also provide real-time operational visibility to support problem determination and problem resolution procedures.

MSBs also enhance reliability by supporting two-phase Commit (2PC) transactions for data and application integrity, as well as failover and load balancing. In addition, MSBs require advanced security capabilities that integrate into the existing infrastructure. Any MSB implementation must also include the procedural support for basic change management and version control.

IV. DataDirect Shadow

As shown in the table above, there are three basic approaches for using a MSB to transform legacy data sources for use in SOA implementations:

- *Data Logic Integration* – accessing the data locked inside legacy systems directly via direct-to-database approaches such as SQL to SOAP transformations and client drivers - ODBC/JDBC/JCA or using asynchronous XML events for real-time mainframe data change capture.
- *Business Logic Integration* – accessing business logic through interfaces such as SOAP for Web Services, or client drivers like ODBC/JDBC as virtual database stored procedures.
- *Screen Logic Integration* – accessing screen-based logic through interfaces such as Web Services or XML-based Web enablement using JSP or ASP.

The first step to leveraging legacy within SOA is to use a Service intermediary to provide for interoperability.

The first step to leveraging legacy within SOA is to use a Service intermediary to provide for interoperability. The choice of which intermediary approach to take depends on the specific requirements and technical limitations at hand. DataDirect's Shadow can act as a Service intermediary or work in combination with an ESB or other middleware suite to provide interfaces to the mainframe.

Shadow supports multiple industry standards, including SOAP/Web Services, SQL, and XML, to provide a secure, reliable and scalable foundation for mainframe integration within the context of a distributed computing environment. A key benefit of such an MSB is the reduction in integration complexity associated with layers of interconnecting points that supporting multiple vendors and integration technologies requires. Reduced complexity translates into a reduction in cost and the elimination of numerous points of potential failure.

When using Shadow directly as a Service Intermediary to an SOA requires wrapping the legacy applications as Web Services. Service-enabled mainframe applications support synchronous responses to individual requests, but sometimes the legacy application can be the source of an event, that requests a external to the mainframe Web Service, therefore, the Service intermediary must be able to support the mainframe as both a *Service provider* and a *Service consumer*.

It's also clear that simply exposing legacy assets as Services—what you might call the bottom-up approach—does not by itself provide SOA. In fact, you need a high-level plan that can guide the whole SOA initiative, so that you can evolve your technology in a way that meets ongoing business needs, rather than heading off into the weeds. The best way to build SOA involves combining two intertwining approaches: top-down as well as bottom-up.

The best way to build SOA involves combining two intertwining approaches: top-down as well as bottom-up.

The top-down approach starts with the architects on the project putting together a long-term architectural design. It's important to have the right level of detail in this plan, since too much detail can slow down the project, and too little can lead to poor architecture. SOA, however, should be bottom-up, as well. If you only take a top-down approach, you're likely to recommend building Services that are too technically difficult or complex to implement. On the other hand, solely taking a bottom-up approach can yield unnecessary or redundant Services.

Shadow supports the integration specialist by supporting these approaches as well as the following:

- Viewing the mainframe as a virtual database, seeing data as relational database tables and business logic as stored procedures.
- Enabling complex event processing for mainframe–based, real-time business events.
- Web enablement to support the development of Web channels.
- Participation in Service-oriented environments as either a publisher or consumer of Web Services.

Shadow provides a wide set of options for the integration specialist. Beyond industry standards support is a set of infrastructure capabilities that support the deployment and maintenance of SOBAs that abstract mainframe data and functionality.

Minimizing New Legacy Troubles in SOA

Many organizations with SOA in place might indeed continue investing in legacy processes beyond their anticipated lifetime, shifting the idea of legacy from the underlying systems to the processes themselves. But then in this case, we're dealing with a different concept of legacy. Rather than the systems or implementations becoming legacy, it's the *processes* that may become difficult to change.

Effective approaches to SOA mean performing a balancing act between business requirements on the one hand and legacy assets on the other.

Effective approaches to SOA mean performing a balancing act between business requirements on the one hand and legacy assets on the other. Any SOA plan, therefore, must include the answers to certain key questions about your legacy assets and how you want to use them:

- Which legacy assets are within the scope of your SOA initiative? You should have a roadmap in mind for the Services you will need to fulfill your business requirements, where you identify certain assets for your pilot project, and bring additional assets into the initiative over time.
- What is the best way to extend each legacy asset? If the value of the asset lies in the data, then directly accessing the data using a SQL metaphor makes sense, as long as it's possible to do so. However, if your requirement includes accessing the legacy business logic, then look for an available API. If there is no practical way to access an existing API, then turn to session-based screen integration techniques.
- Rather than providing a single solution or API, use an MSB like Shadow that offers integration specialists the ability to match the API to the skills that the project requires, including SOAP, SQL, or XML.
- How scalable, robust, and secure do you want to be? You must plan early to address non-functional requirements of scalability, fault tolerance, and security. Identify service levels that your project must satisfy, and work through the infrastructure issues behind meeting those levels.
- Visibility into the system becomes critical for managing Quality of Service. Real-time in production visibility into the operation of SOBAs supports improved reliability, availability and problem determination. A single, unified view across all mainframe integration operations is one of the benefits of a MSB and this capability for operational support is provided through the *Shadow Instrumentation Server*.

Shadow's bidirectional capability enables mainframes to both provide and consume Web Services, allowing them to participate in SOA implementations as full-fledged participants.

Transforming the Mainframe into a Full-Fledged SOA Participant

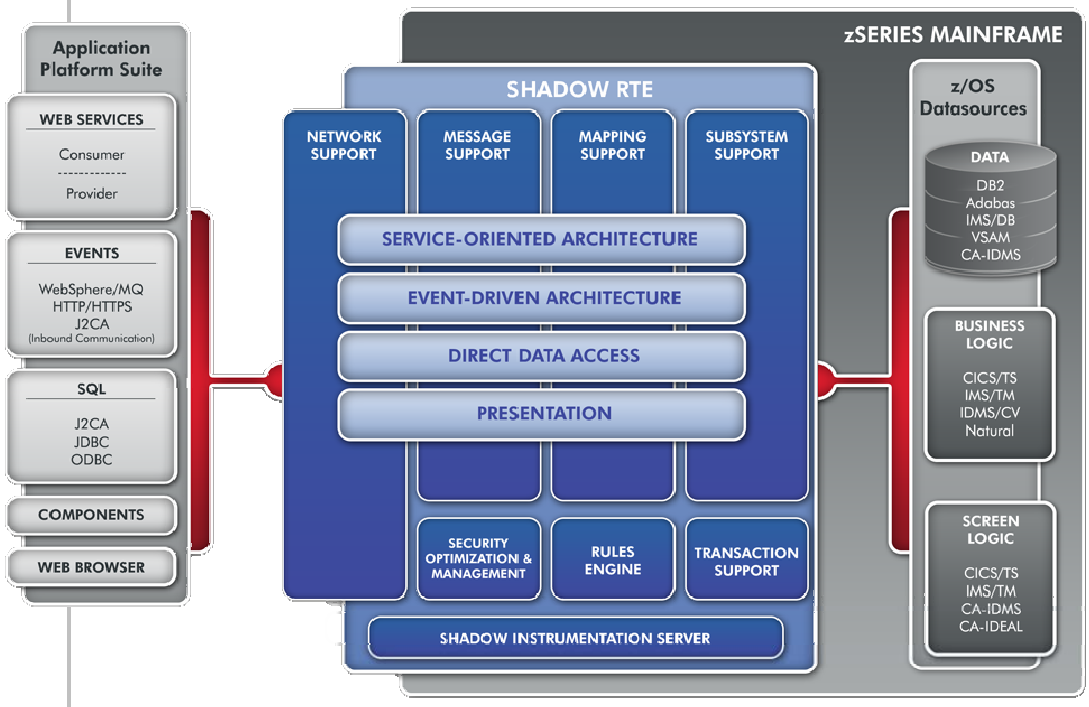
One of the distinguishing characteristics of Shadow is its ability to enable bidirectional use of existing legacy screens, programs and mainframe data. This bidirectional capability enables mainframes to both provide and consume Web Services, allowing them to participate in SOA implementations as full-fledged participants, rather than simply as back-end data sources.

Furthermore, Shadow offers a non-invasive development approach which protects existing legacy logic, and also supports a variety of development environments, both within the Eclipse framework as well as for .NET. It automatically generates Web Services from mainframe screens, business logic, and mainframe data sources, as well as automating the generation of starter programs which enable the testing of newly created Services.

Shadow also improves the management of Web Services within mainframe security protocols. Shadow's *Security Optimization Management* ensures the quality of mainframe-based Web Services by reducing the repetitive overhead of authenticating loosely-coupled interactions.

The Shadow Mainframe Services Bus architecture is illustrated in the figure below:

DataDirect Shadow - Mainframe Service Bus Architecture



Source: DataDirect

Shadow offers a variety of development options, enabling the integration with various J2EE and .NET application development environments. *Shadow Studio* is a graphical interface tool and IDE built on the Eclipse framework. Shadow Studio provides integration within the IDE to expose mainframe applications and data as Web Services, real-time business events or as direct SQL calls, enabling a

single development environment to handle any integration requirement without requiring detailed knowledge of the mainframe or distributed technologies.

Shadow also provides the flexibility of multiple deployment options, including a native mainframe server, an option to deploy within CICS or a distributed, off-host option. Shadow's MSB architecture doesn't force organizations to adapt their IT infrastructure to a particular integration method. Shadow also supports a broad range of mainframe data sources, specifically mainframe databases, Adabas, DB2, IAM, IMS/DB, IDMS, and VSAM; mainframe business logic, CICS, IMS/TM, CA-IDMS, and Natural; and mainframe screen logic, CICS, IMS/TM, CA - IDMS-CV, and CA-Ideal. From a developer perspective, Shadow provides expanded client-side support for SQL, HTML, WSDL, XML and SOAP.

Shadow allows the mainframe to produce and consume events as part of its bi-directional Web Services capabilities, further enabling real-time business that leverage SOBAs based upon SOA. This bi-directional Web Services capability allows the mainframe to be a participant in any SOA implementation.

The key MSB capabilities of Shadow are in the box below.

Shadow as a Mainframe Service Bus

Key MSB capabilities of Shadow that support the transformation of mainframe assets and bidirectional Service implementations:

- **Transformation** – Comprehensive transformation capabilties from a diverse set of data, application, or screen logic into standards-based outputs.
- **Management –** Sophisticated problem determination and resolution, change management support, and visibility into real-time operations.
- **Quality** – Measurable and predictable performance, Support for IBM Work Load Management for granular contol of service levels, auditing and chargeback support, and a rules engine for specifying QoS policies.
- **Security** – Support for authentication and authorization of stateless protocols like SOAP to optimize security management, increase throughput and reduced MIPS usage via credential caching.
- **Operations** – Shadow instrumentation Server (SIS) provides real-time visibility into Shadow operations in support ofscalability, automated QoS support, auditing, and chargebacks.
- **Enhanced Reuse** – Accelerates development of mainframe Web Services publishing and external Web Services consumption via a simple Eclipse-based tool.
- **Standards Support** – Web Services, XML Web enablement, and direct SQL access
- **Mainframe Resource Support** – Data logic integration, business logic integration, and screen logic integration.

V. The ZapThink Take

To fully leverage mainframe assets, it's important to take advantage of MSB capabilities, including management, quality, security, operations, reuse, standards support, and mainframe resource support.

The maturation of Web Services standards combined with next-generation Mainframe Service Bus technologies are combining to provide an environment that enables the mainframe to become an active participant in Service Oriented Architecture implementations. SOA provides an unprecedented opportunity to marry new applications with legacy assets. To fully leverage mainframe assets, it's important to take advantage of MSB capabilities, including management, quality, security, operations, reuse, standards support, and mainframe resource support. Vendors such as DataDirect Technologies and its Shadow mainframe integration product provide options for organizations with significant investments in mainframe technologies to Service-enable their legacy data and applications, transforming them into reusable Services as part of a high-performance SOA implementation.

Shadow offers capabilities that specifically address the issues this white paper raises dealing with how to enable the mainframe to be a first class participant in a SOA implementation:

- A non-invasive Service intermediary in the form of an MSB for seamless transformation of legacy systems to enable participation in a SOA implementation.
- Middleware which complements and reinforces mainframe Quality of Service with a multi-threaded native server for high performance, scalability and reliability; advanced instrumentation and diagnostics to ensure mission critical availability; and security optimization and management support to ensure security, throughput and efficient CPU utilization.
- Bi-directional Web Services support to allow the mainframe to participate as both a Service provider and a Service consumer.

For most organizations, successful SOA initiatives will include the ongoing preservation of legacy for the foreseeable future.

It's all too easy to fall into the good-versus-evil pattern of touting the flexibility of SOA over the brittleness of legacy. The best path, of course, falls in between these extremes. We want stability without brittleness, and flexibility without chaos. For most organizations, then, successful SOA initiatives will include the ongoing preservation and transformation of legacy for the foreseeable future. And, more importantly, the realization that service-enabled mainframe technology can provide powerful, new capabilities to support an SOA implementation.

Companies are increasingly demanding the numerous benefits that SOA promises. Clearly, SOA is no magic cure-all, but if you take the correct approach to creating and rolling out the new architecture, then you can achieve dramatic business value, and one of the primary sources for such value is the legacy assets in the organization. Furthermore, by utilizing an MSB technology like Shadow, it's possible to do more than simply leverage existing assets—it's now possible to include the mainframe as a first-class citizen in Service-Oriented Architecture implementations.

Copyright, Trademark Notice, and Statement of Opinion

About ZapThink, LLC

ZapThink is an IT advisory and analysis firm that provides trusted advice and critical insight into the architectural and organizational changes brought about by the movement to XML, Web Services, and Service Orientation. We provide our three target audiences of IT vendors, service providers and end-users a clear roadmap for standards-based, loosely coupled distributed computing – a vision of IT meeting the needs of the agile business.

ZapThink helps its customers in three ways: by helping companies understand IT products and services in the context of Service-Oriented Architecture (SOA) and the vision of Service Orientation, by providing guidance into emerging best practices for Web Services and SOA adoption, and by bringing together all our audiences into a network that provides business value and expertise to each member of the network.

ZapThink provides market intelligence to IT vendors and professional services firms that offer XML and Web Services-based products and services in order to help them understand their competitive landscape, plan their product roadmaps, and communicate their value proposition to their customers within the context of Service Orientation.

ZapThink provides guidance and expertise to professional services firms to help them grow and innovate their services as well as promote their capabilities to end-users and vendors looking to grow their businesses.

ZapThink also provides implementation intelligence to IT users who are seeking guidance and clarity into the best practices for planning and implementing SOA, including how to assemble the available products and services into a coherent plan.

ZapThink's senior analysts are widely regarded as the "go to analysts" for XML, Web Services, and SOA by vendors, end-users, and the press. Respected for their candid, insightful opinions, they are in great demand as speakers, and have presented at conferences and industry events around the world. They are among the most quoted industry analysts in the IT industry. ZapThink was founded in November 2000 and is headquartered in Baltimore, Maryland.

ZAPTHINK CONTACT:
ZapThink, LLC
108 Woodlawn Road
Baltimore, MD 21210
Phone: +1 (781) 207 0203
Fax: +1 (815) 301 3171
info@zapthink.com

ZapThink, LLC • 108 Woodlawn Road • Baltimore, MD 21210 • info@zapthink.com • www.zapthink.com