Business services provide the starting point for services identification and analysis for an organization. Beginning with customers and external touch points into an organization helps focus SOA efforts on areas where transactional friction can endanger customer satisfaction, revenue generation, and process efficiency. However, business services offer another critical benefit: a common business language.

Business Services as a Common Language

Business services extend very naturally into business language and thus can form the bridging terminology between IT and business users/consumers during analysis and design. As the fundamental unit of analysis for SOAs, services can provide common understanding of processes, events, transactions, and IT capabilities that underlie or implement these activities in an organization. This is a critical aspect of services in an SOA. The ability to understand and communicate within an organization using business services concepts will pave the way to a better IT and business relationship as well as provide a common language of business processes and IT functionality. This benefit of SOA and services should be front and center in all organizations.

Technical Services

Technical services are those services that are horizontal in nature or are reusable by all business processes, business units, or process domains. Technical services include security services, logging services, audit services, transformation services, and similar "IT services" that would be leveraged by and across all lines of business. In some organizations, these technical services are described as enterprise common services. They are enterprise-wide and common to all business processes. Another common model is to assign ownership of SOA technical infrastructure to an organization that also manages these enterprise common services. In this approach, the term "services" refers to the infrastructure technology as services as well as the technical services that are running on these infrastructure services. Be careful with your terminology here. The services model is meant to

simplify and clarify, not confuse. Use clear services taxonomies within your organization to eliminate confusion.

Beware of allowing technical services to become the sole focus for your SOA efforts. SOA should be a business-driven initiative, beginning with business services and accommodating technical services as well. This is not to say that an IT-focused SOA initiative cannot deliver tremendous organizational value. It can. However, focusing an SOA initiative solely on the IT organization may overlook opportunities to positively impact business operations with SOA benefits of time to market, process orchestration, and agility. If SOA is reduced to "just another IT initiative" without clear support for and by the business, there may be some organizational value left on the table. If an SOA effort is an IT-driven and IT-focused activity, treat it from an IT business perspective. In other words, make sure SOA addresses the business needs of an IT organization.

Characteristics of Services

Services, in order to meet the needs of the organization, must meet certain criteria to provide the most value to the organization. These attributes are important features of services for this book:

- Coarse-grained services
- Well-defined service contracts
- Loosely coupled
- Discoverable
- Durable
- Composable
- Business aligned
- Reusable
- Interoperable

Coarse-Grained Services Services should be coarse-grained entities. By "coarse-grained," we mean that services should represent business functions, processes, or transactions and encapsulate other fine-grained components or services within them. This term is one of the most used in the rapidly evolving world of SOA. Service granularity depends on how much functionality a service encapsulates and exposes. The internal functionality of a service depends on the scope and functionality of components and transactions encapsulated in the service.

A service that is too big or too coarse-grained will suffer from performance issues and will not be reusable. A service that is too fine-grained will be too narrow in scope to meet requirements of multiple business processes. Services should encapsulate lower-level or fine-grained entities, components, transactions, and other implementation-specific details in order to abstract them from the specific physical and technical implementations they are currently instantiated in.

Fine-grained services provide a narrow scope of business and process utility. Consider software components being exposed as services one for one. These would most likely be fine-grained services. They may encompass little business functionality and hence would be too fine-grained for an SOA. Consuming such fine-grained services could potentially cause excessive messaging traffic to complete the desired business process or transaction. Good services design seeks coarse enough granularity to meet business process requirements while optimizing XML processing and messaging traffic. The art and science of services design is finding the right granularity that solves the business problem, can be reused, and can be technically implemented.

Well-Defined Service Contracts Services must have well-defined contracts that separate the functionality of the service from its specific technical implementation. The service contract informs consumers what the service does as well as how to consume or use the service. Service contracts present the service functionality to the outside world in a standardized, interoperable fashion while hiding the specific internal technical details of the service. In the world of Web services, the service contract is defined by the WSDL document in conjunction with other metadata, such as policies, XML schema, document semantics, and more.

Loosely Coupled The term "loosely coupled services" is another one that is clear yet ambiguous. The reason is that "loosely coupled" has implications both for services and for the enabling technology required to operate services. In the services design aspect of the term, "loosely coupled" means designing services such that specific implementations of services can be replaced, modified, and evolved over

time without disrupting the current service consumers and the overall activities of an SOA. In this sense, then, loosely coupled services reduce lifecycle costs, such as development and maintenance costs, by isolating the impact of changes to the internal implementation of services and encouraging reuse of services.

Loosely coupled services are more typically associated with document-style services than remote procedure call (RPC)-style services, which sometimes have a tendency to be tightly coupled to specific technology platforms from which the services were exposed.

The fact that services are loosely coupled also implies certain aspects of the SOA enabling technology along with services design. Consider the synchronous-asynchronous dichotomy of message exchange patterns. Services designed to be asynchronous take advantage of messaging platforms supporting message queue technology, publish-subscribe (pub-sub) functionality, and related message exchange patterns (MEPs). Services that implement asynchronous messaging have a tendency to be more loosely coupled than services that are synchronous. This is a generalization, as synchronous services can be implemented using loose-coupling concepts. However, implementing loosely coupled services and taking advantage of asynchronous messaging when appropriate will allow a transition to an event-based model of SOA, where an asynchronous event-driven services paradigm replaces a synchronous tightly coupled and brittle approach.

Discoverable Services should be discoverable. This means not only that the services are designed well, but that their contracts are published and visible to an intended audience—the consumers. Discoverable services implies that the service contracts—WSDL documents in the case of Web services—are published to a location where they can be discovered, whether that location is a service registry, metadata repository, subdirectory, or some known location. Once the service contracts are published, they must be advertised to potential consumers. An important point must be made here. Services should have known consumers and reuse patterns before they are created or exposed. These intended consumers will use the services as "advertised" in the service contract. However, one of the benefits of an SOA is the unintended or *emergent* consumption of services as more of them are available for consumption. These emergent patterns of service consumption and reuse may lead to new innovations, new business processes, and other organizational value derived by creating reusable interoperable services in your SOA.

Durable Services should be durable yet elastic. Durable services are those that map to lasting business or process themes. For example, insurance companies will always have a claims process. The claims process is thus an enduring process theme. The business services that comprise the claims processes may change, and the claims process itself may change, but there will always be a claims business process in an insurance organization. The services may change but the business or process themes will remain. We like to say that services are the lasting and enduring assets of your SOA. Design them well, pick the proper services, and make them the central asset of your SOA process.

Composable Services should be composable. The word "composability" has as many definitions as the term "loosely coupled." Composable services are designed to be incorporated into other services as composite services as necessary. In addition, composable services can be assembled into orchestrated process flows. In this sense, composable services are stateless and atomic in nature. They stand on their own yet rely on other services or infrastructure for state and context.

The World Wide Web Consortium (W3C) defines composability in this way: "Composability of web services refers to the building, from a set of web services, of something at a higher level, typically itself exposed as a larger web service. Web services are required to be composable—you should be able to make a web service implementation by building it out of component web services."[3]

However, IBM and Microsoft have advocated another aspect of composability during the standards process for services and SOA: "Composability enables *incremental consumption* or *progressive discovery* of new concepts, tools and services. Developers only need to learn and implement what is necessary, and no more. The complexity of the solution increases only because the problem's requirements increase, and is not due to technology 'bloat.'"[4]

In many respects, composability of services is a result of Web services standards and the multipart message structure. This modular structure enables the *composition* of new functionality. New message

elements supporting new services may be added to messages in a manner that does not alter the processing of existing functionality. Composability also means that incremental functionality may be added to existing services without breaking the contract with existing services consumers, and new standards may be implemented similarly without compromising the existing interoperability and functionality of the same services. In both of these contexts, composability of services is an essential requirement.

Business Aligned Services should be business aligned. By this we mean that services identification and analysis should begin with business imperatives and business requirements, and then cascade into the other services we expect to find in most organizations, such as technical services, data services, infrastructure services, and more. Services, or business services, represent business concepts and match business needs as determined through business strategy and planning and the associated business discovery processes that should precede an SOA initiative.

Reusable Services must be reusable. This is a function of proper services identification, analysis, and design. Spending time and effort on services that are not reusable is dangerous and wasteful. You must implement services that have clear and defined reuse across and within business processes and that have multiple consumption patterns in your current and planned business processes. Again, bear in mind that the intended reuse of services may lead to unintended or emergent reuse by other consumers who find value in a particular service. This is a good thing. However, you must have a management infrastructure in place to track actual consumption patterns in order to ensure proper performance of the services and appropriate sizing of hardware and bandwidth of your networks.

Interoperable Services must be interoperable. This sounds obvious, but many services do not interoperate. This lack of interoperability can result from the differential application of policies, standards, and other design criteria during the services design and development process. The way to achieve interoperable services is to enforce a body of SOA policies across the services lifecycle: identification, design, and implementation.

Services First, Not Web Services First

A common misconception with SOA and services is to focus only on Web services, which are really a special case of interoperable reusable services that make use of the core SOA standards such as XML, SOAP and WSDL. We suggest taking the conceptual view of all possible reusable services first, and then implementing Web services where they make sense. Intradomain services may not require WSDL documents and SOAP messaging due to their domain-specific behavior and context. However, cross-domain services are great candidates for Web services because they will be exposed outside of a particular domain for shared usage by a diverse community of interest. Consider these four simple rules for services:

1. Focus on business services first.
2. Determine services that map to business processes.
3. Decide what services are best exposed as Web services.
4. Implement as priorities dictate.

SERVICES ADDRESS PERSISTENT CHALLENGES AND PRESENT NEW OPPORTUNITIES

The services paradigm has the potential to introduce many new opportunities to IT organizations. The organizations that embrace SOA and can capitalize on service orientation will realize a wealth of new concepts, technologies, methodologies, and development approaches that have never been practiced in their environments. Major service opportunities for enterprises include:

- Service strategies and operating models
- Abstraction and conceptualization
- Breaking silos of asset ownership
- Service reuse and asset leverage
- Business and consumer growth

Services Strategies and Operating Models

SOA and services provide new business opportunities through the ability to impact both IT and business processes in a variety of beneficial ways. For example, SOA can shorten time to market for products and services through the introduction of service identification, analysis, and design best practices that can influence product development lifecycles. These new methodologies encourage the formation of business and technology strategies that may influence management structures, business models, business processes, and software development lifecycle activities. Many organizations already had begun the migration to a services delivery model before the advent of SOA and services. However, with the industry adoption of standards for SOA and Web services, the opportunity to drive IT and business change through service delivery models is clear. Shared reusable and interoperable services offer many benefits to the organizations that embrace the opportunity.

Abstraction and Conceptualization

Service-oriented development helps elevate software abstractions and generalizations beyond the traditional software development process.[5] This higher level of abstraction based on services facilitates the establishment of discovery, analysis, and design practices that can meet business requirements much earlier in the organization's business process and software development lifecycle. These opportunities encourage a top-down conceptual analysis, services design, and architecture process that are aligned with business requirements and strategies. SOA discourages ad hoc approaches to IT and business solutions, yet it will provide a more agile approach to rapid business solutions based on reusable and interoperable services.

Breaking Silos of Asset Ownership

Services provide an opportunity to break the organizational silos of asset ownership. This problem is partly responsible for the silos of application functionality that exist today, which traditionally required the point-to-point integration solutions of the past. An SOA model of shared reusable services allows an organization to change the asset utilization model to one of shared business services instead of domain ownership, business unit ownership, or even departmental ownership. The movements toward grid computing, utility computing, and other similar trends are all based on improving return on assets (ROA) and