19. Component and Composite Services

3 November 2008

Bob Glushko

Plan for Today's Class

Abstracting the idea of "Service"

Service Oriented Architecture

Component Business Models

Service Oriented Computing

Web Services

Composite Services

Abstracting the idea of "Service"

Many of the traditional concepts, techniques, and curricula for service design and operations originate in and emphasize person-to-person services

These do not fit so well when person-to-person services are replaced or complemented by self-service or for automated information-intensive services provided by one computational or automated process to another

The concept of SERVICE SYSTEM was proposed to raise the level of abstraction about the service concept to make these different kinds of services more comparable

What Is a Service?

Provider

Client / consumer / requestor

Interface -- the (sometimes implicit) description of what the service does and how to request it

- ... or the information about the service provided to any potential service client / consumer / requestor
- A service can offer multiple interfaces ... you might think of different "quality of service" that way
- The interface can change without affecting the service that is delivered

Service Oriented Architecture - A Conceptual Perspective and Design Philosophy

Business processes are increasingly global and involve widely dispersed parts of an enterprise or multiple enterprises

A business needs to be able to quickly and cost-effectively change how it does business and who it does business with (suppliers, business partners, customers)

A business also needs more flexible relationships with its partners and "assets" to handle variable demands

Service Oriented Architecture - Business Components

So we need to think of "what a business does" in more granular terms

Business functions or services are "components"

A business model is a composition or assembly of components

These business components can be a mix of core, internal ones that a business does itself and outsourced ones provided by other businesses

Definition of Component

A component offers a business service to other components

A component has:

- A business purpose
- One of more activities
- Resources
- Governance
- Services

Cherbakov et al. on "Service Orientation and Business"

"The notion of componentization allows an enterprise to deconstruct, analyze, and then reconstruct into value nets, in which partnerships with customers and suppliers operate in a network supported by real-time information flows and integrated IT systems"

"The process of deconstruction/reconstruction is realized through business components, which correspond to distinct business functions"

"In the on demand environment, the component-based firm links its components efficiently and seamlessly both internally and across the firm’s boundaries with best-of-breed components provided by external partners"

The Component Integration Vision

Proponents of "Service Oriented Architecture," "Service Oriented Computing," and "Web Services" have a vision:

- ... the component-based firm links its components efficiently and *seamlessly* both internally and across the firm's boundaries...
- ... the business components can be *plugged in* or unplugged with relative ease...
- ... permitting applications to be *constructed on the fly*...

Service Oriented Architecture and the "Plug and Play" Economy

"Plug and Play" was a marketing term used by Microsoft Windows 1995

"Plug and Play" as a metaphor for the integration of business components was first used around 1996 from Marty Tenenbaum

For small businesses to participate in the web economy, the solution they used must be technically simple and relatively inexpensive to start with

These requirements for "easy and cheap" integration are especially important in the lower tiers of supply chains

Loose Coupling

Two businesses anywhere in the world can do business with each other using telephones, fax machines, or email

Their interactions are:

- *Ad hoc*
- *Asynchronous*
- *Location and Implementation Transparent*

The Integration Requirement

But firms don't rely on phones, fax machines, or email to do all of their business because they have too many interactions, and these kinds of service requests must often be manually transformed and re-entered as input to the destination application or requested service

This requires INTEGRATION -- the controlled and automated sharing of data and business processes among any services, applications, or data sources, intra- or inter-company

Tight Coupling

"Tight coupling" between two businesses, applications or services means that their interactions and information exchanges are completely automated and *optimized* in performance...

... by taking advantage of knowledge of their *internal* processes, information structures, technologies or other *private* characteristics that are not revealed in their *public* interfaces

... and usually implemented with a custom program that fit only between the two of them

Tight coupling is most often used, and usually limited to, situations in which the same party controls both ends of the information exchange

But tight coupling can be a "Faustian bargain" that trades efficiency for flexibility (John Hagel)

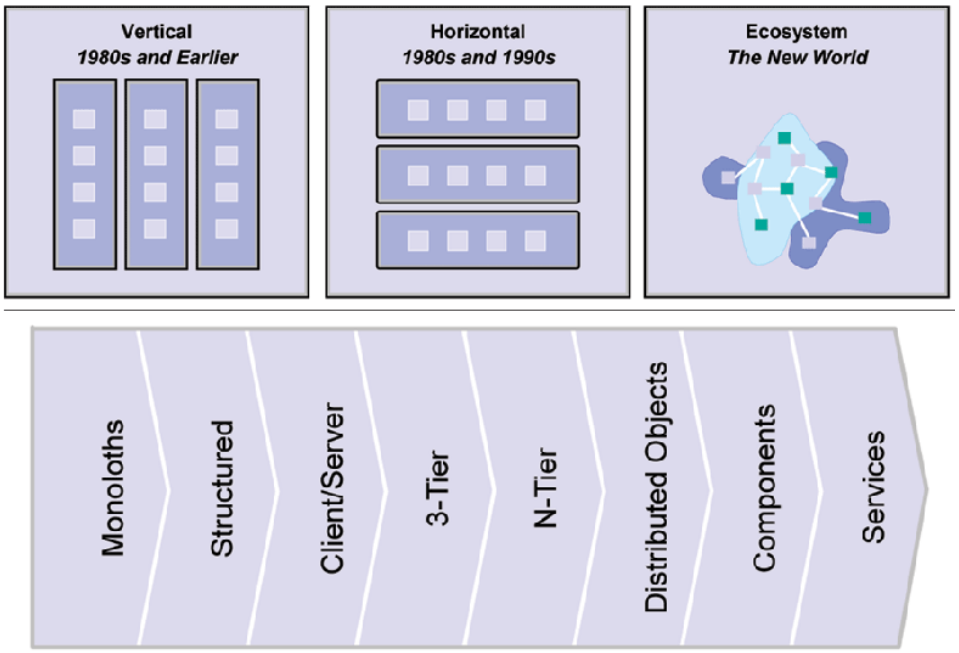
The Integration Challenge

Can we have integration and loose coupling at the same time?

The idea of service-oriented integration says we can

But we can get there from here?

Co-Evolution of Business and Technology Architecture



Service Oriented Architecture -- Discovery and Integration

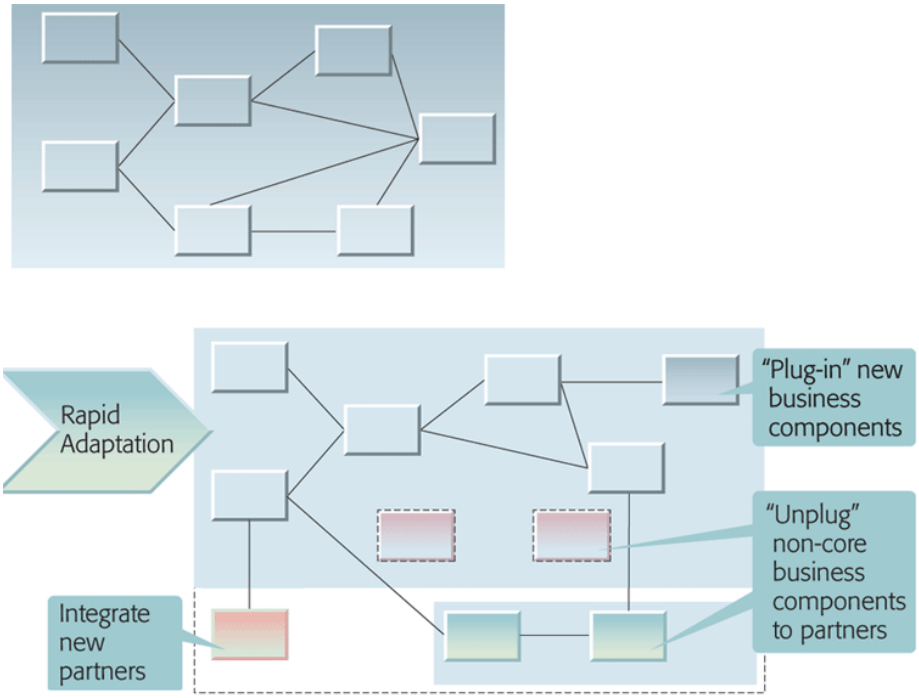
A service-oriented business needs to avoid the "lock-in" by technology or data format created by tightly-coupled integration

The particular integration technology is less important than the philosophy or business model that requires it – treating different organizations, applications, and devices as loosely-coupled cooperating entities regardless of where they fit within or across enterprise boundaries

Instead of thinking "how do these two applications exchange information," think "how do these two businesses exchange information"

This highlights the principles of discovery and transparent substitutability of service providers

"Plug and Play" (Cherbakov et al.)



Modeling a Business as a Set of Components

Business processes are typically "factored" into components according to the "best practice" patterns in each industry

An emphasis on business model / business process / information exchange patterns facilities component reuse / reassembly into new combinations - virtual enterprise, composite services

"What components do" is defined in abstract, technology-independent terms so we don't have to care about the computer, operating system, or software application that performs each business process

This level of abstraction reduces integration and communication costs between components and is the essence of service orientation

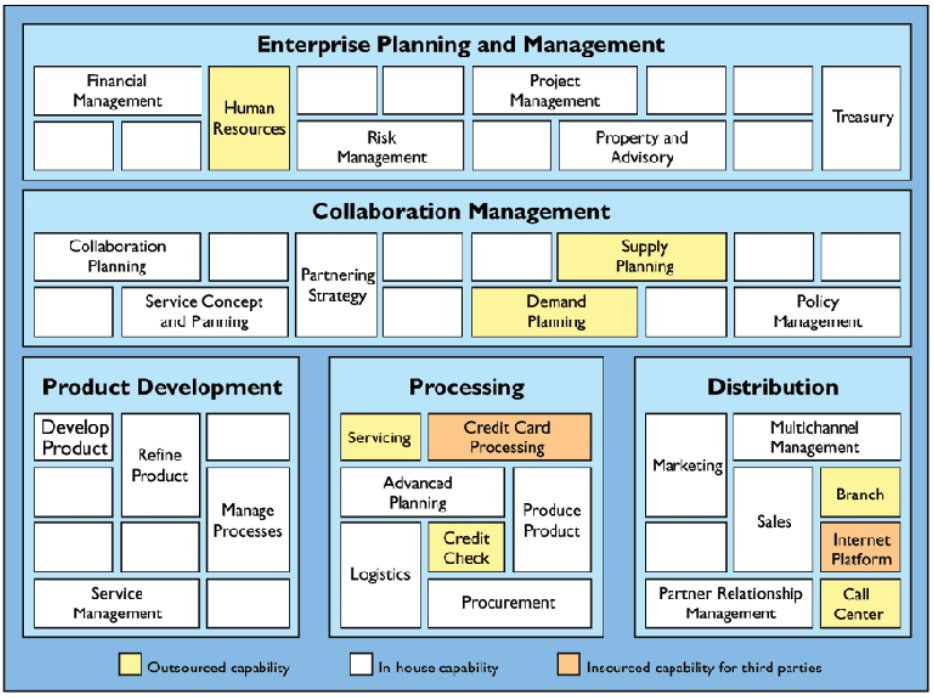
Component Business Map -- Generic

	Business Administration	New Business Development	Relationship Management	Servicing and Sales	Product Fulfillment	Financial Control and Accounting
Directing	Business Planning	Sector Planning	Account Planning	Sales Planning	Fulfillment Planning	Portfolio Planning
Controlling	Business Unit Tracking	Sector Management	Relationship Management	Sales Management	Fulfillment Planning	Compliance
	Staff Appraisals	Product Management	Credit Assessment			Reconciliation
Executing	Staff Administration	Product Directory	Credit Administration	Sales	Product Fulfillment	Customer Accounts
		Marketing Campaigns		Customer Dialog	Document Management	General Ledger
	Production Administration			Contact Routing		

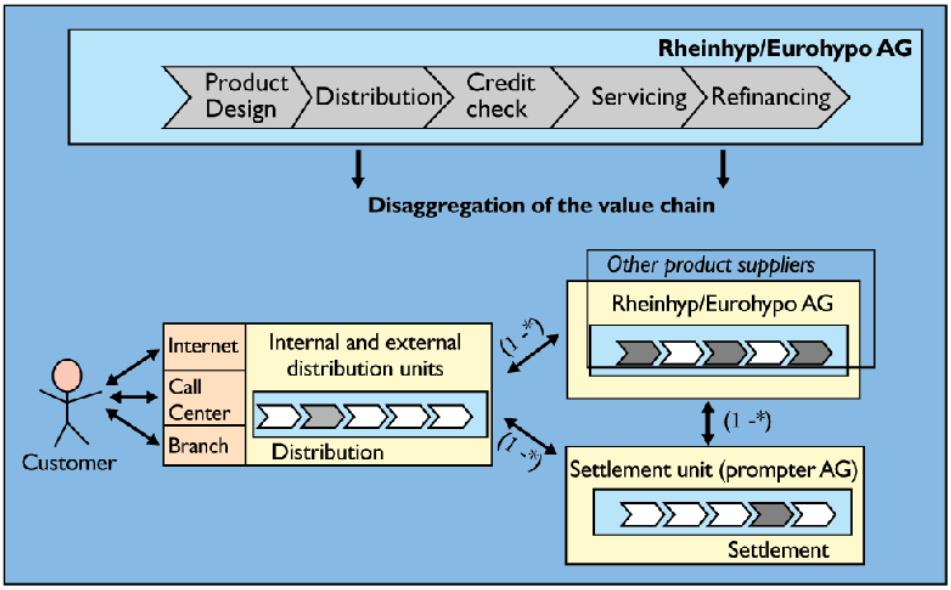
Component Business Map -- Rental Car Operations

	Marketing and Customer Management	Products	Rentals Management	Rental Fleet Logistics	Business Administration
Plan	Customer Segmentation	Rental Product Strategy	Location and Channel Strategy	Fleet Strategy	Corporate / LOB Strategy
	Customer Relationship Strategy	Product Development/ Design	Location Design and Layout	Fleet Planning	Financial Management and Planning
	Marketing Strategy and Planning		Channel Design and Layout	OEM Relationship Planning	Real Estate Planning
Manage	Customer Behavior Modeling	Promotions Management	Channel and Location Profitability	OEM Performance Management	Alliance Management
	Market and Competitor Research	Pricing Management	Location Operations Management	Inbound Logistics	Business Performance Reporting
	Segmentation Management		Reservations Management		Legal and Regulatory Compliance
	Call Center		Workforce Management		Real Estate and Construction Management
					Risk Management
					Stock Ledger
	Campaign Management				HR Management (Career Development, Training, Recruiting)
Execute	Customer Service	Purchasing/ Sourcing	Rentals and Reservations	Location Operations	HR Administration / Payroll
	Preferred Member Management	Demand Forecasting	Time and Attendance	Fleet Servicing	Corporate Audit
	Customer Communications				Corporate Accounting (GL, AP, A/R, Treasury, etc.)
	Mass Marketing and Advertising			Fleet Management	Indirect Procurement
	Target Marketing				PR and Investor Relations
					IT Systems and Operations

The "Componentized" Bank



"Componentized" Bank Value Chains



Service Oriented Computing

Service Oriented Computing extends the abstract principles of Service Oriented Architecture to consider concerns that inherently arise because SOA must be realized in some computing technology

These include:

- Transaction management and coordination
- Security
- Orchestration
- Resource management to ensure performance and quality of service

Alternatives for Service Realization

Build services internally -- pay entire cost

Outsource the building of services -- pay entire cost

Reuse an external service

- Pay on subscription / lease basis
- Pay by use

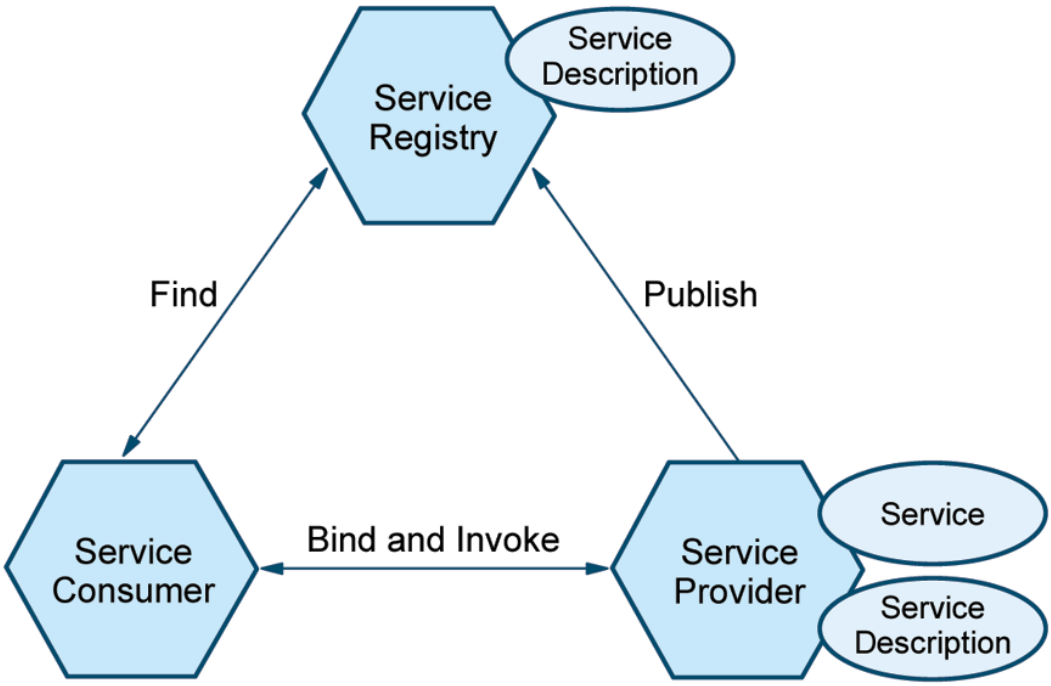
Web Services

Web Services -- with a capital "S" -- generally means a particular set of specifications for doing service-oriented integration with XML documents as the "payload" that conveys the information required by the service interface

(Or put another way -- the interface is specified using an XML schema that defines in a formal way the information the service expects and how it should be structured)

The most important Web Service specifications are those for a service's public interfaces (Web Service Description Language) and for the messaging protocol used to send and receive XML documents through those interfaces (SOAP)

The Web Services Model



Web Services Standards & Mailing a Letter

To whom should we address the letter? ==> need directory so we can "discover" service providers (UDDI)

To what address do we send it, and what should the letter say? ==> need service description (WSDL)

How to address the envelope and put the letter in it? ==> need message packaging protocol (SOAP)

The Service Discovery Myth

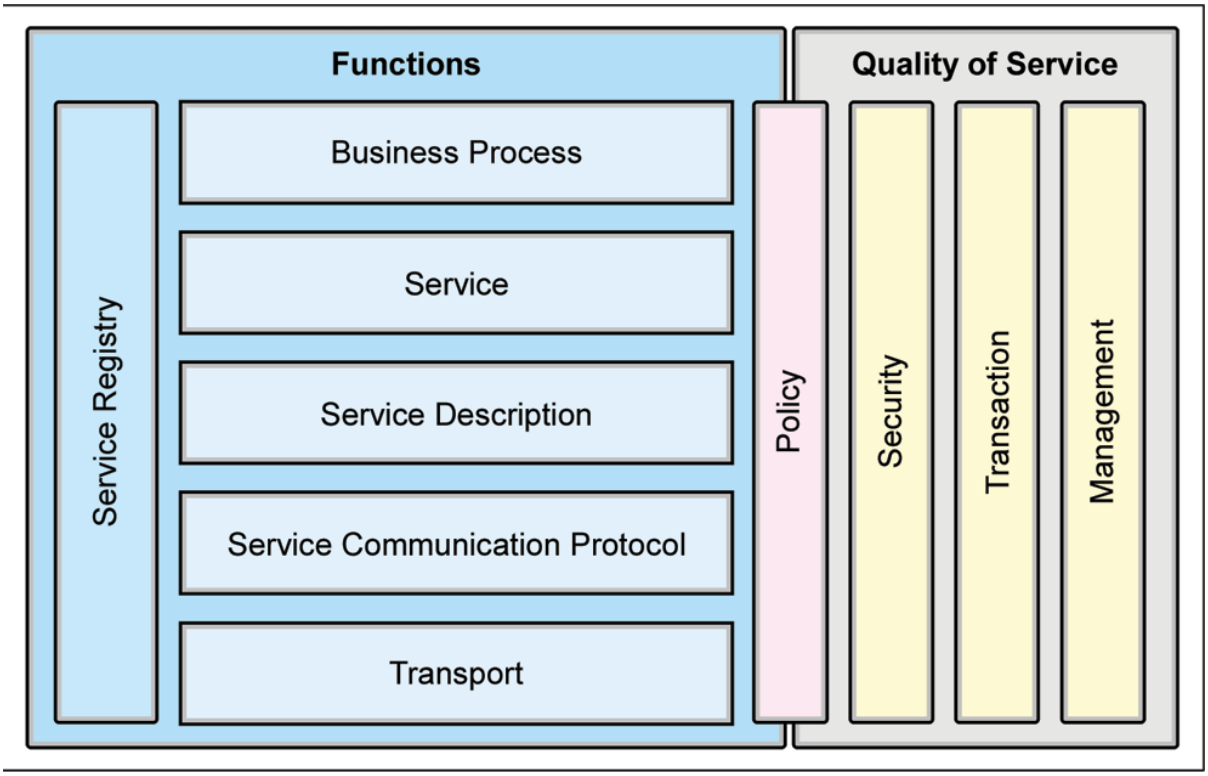
Many discussions about services highlight the concept of service discovery and a specification called UDDI (Universal Description, Discovery and Integration)

UDDI was proposed as a kind of services "white" and "yellow" pages directory that would enable services to be registered by their providers and discovered by potential users, all by automated means

But UDDI is mostly used for "internal" service directories and rarely for "public" ones

- When you use an ATM, would the machine search for the bank's service interface in a directory?
- Most service relationships are established "offline" and then the information about how to access the service is built into the service requestor's implementation
- Would you trust the information published in a UDDI directory?

Web Services Abstract View



WS-* ("star" or "splat")

The major platform and enterprise software vendors (Microsoft, IBM, Sun, Oracle, BEA, HP, SAP...) have developed and "standardized" a few dozen specifications for extending the basic Web Services specifications to handle issues that emerge in complex distributed applications and service systems

These specifications cover things like security, multi-hop addressing, process choreography, policy assertion, performance management, ...

Their proponents argue that these additional specifications are essential for service oriented computing to be viable for enterprise-level applications and services

But they've made Web Services (with a capital "S") seem needlessly complex for a great many applications where they might have been useful

Many services are being implemented today with simpler protocols

Web-based Services

This is a category coined by Erik Wilde for his courses at the I-school to mean "Web Services and any services that use any Internet protocol"

This includes services implemented using the basic HTTP protocol and its mechanisms for providing "better service" using content negotiation (provide different information to the client based on the type of browser, etc.)

This broader category makes it easier to understand and make tradeoffs in the design and implementation of services

Composite Services

A composite service provides a single user interface to a set of services linked by overlapping information requirements, business rules, and processes

The application logic is reused from the component services, which can still function as services in their original contexts

A common type combines services from internal legacy applications, where the lack of integration had made it difficult to provide good customer service

Motivations for Internal Composite Services

BUSINESS STRATEGY perspective - can provide improved and new services to customers by combining customer information, order management, product information, etc.

BUSINESS OPERATIONS perspective - Reduce errors and time for information to flow from one "stovepipe" system or process to another by replacing manual methods of information sharing like re-typing or phone with automated interchange

TECHNOLOGY perspective - reuse of legacy applications extends their useful lives and can be technically easier than reimplementing their functionality in a new application

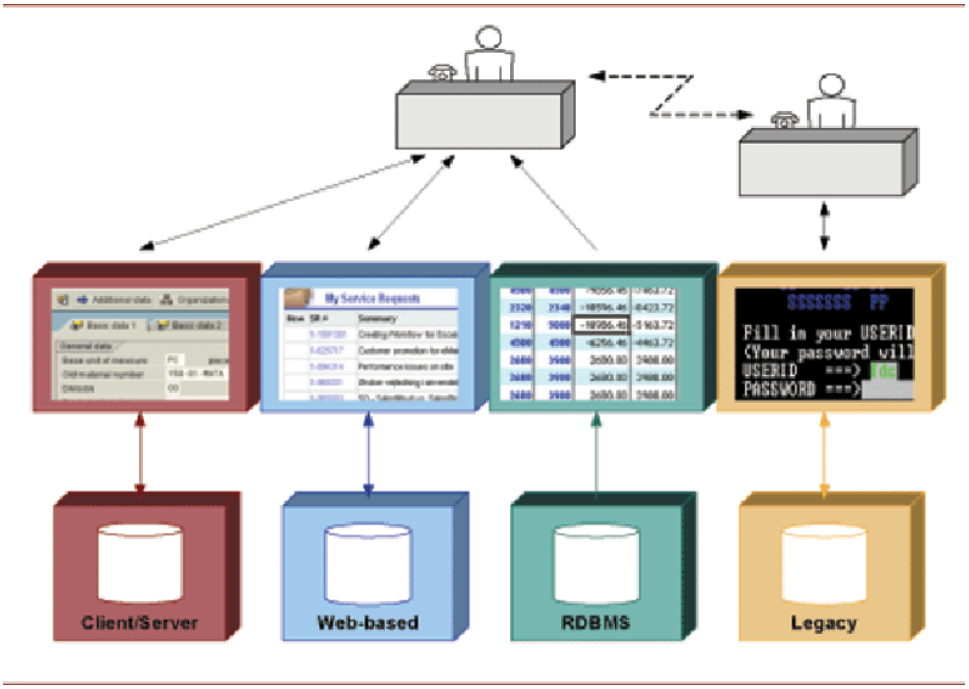
Example: Why Enterprises Want Composite Services

An existing customer calls a service representation to increase an order

The service representative must:

- locate information about the customer (Customer Database)
- locate the existing order (Order Management Application)
- determine if the order can be changed or whether a new order must be created
- determine whether to accept the order based on the customer's payment history and credit (Accounting Application; Credit Check Application)

Before Service Composition



After Service Composition

Inter-Enterprise Composite Services

Rather than being provided by internal legacy systems, the separate services being composed might be provided by other firms

This makes the composite service the initiator, and possibly the controller of a long-running multistep collaboration

The interactions between the composed services is called a CHOREOGRAPHY where there is distributed coordination with equivalent responsibility, where the sequence is determined by the service outputs, or where we don't need to take a position on how the overall process is controlled

The coordination is said to be ORCHESTRATED when one of the services controls the invocations of the others, serves as the CONDUCTOR, or when all the services are controlled by an intermediary service (the "traffic cop" analogy)

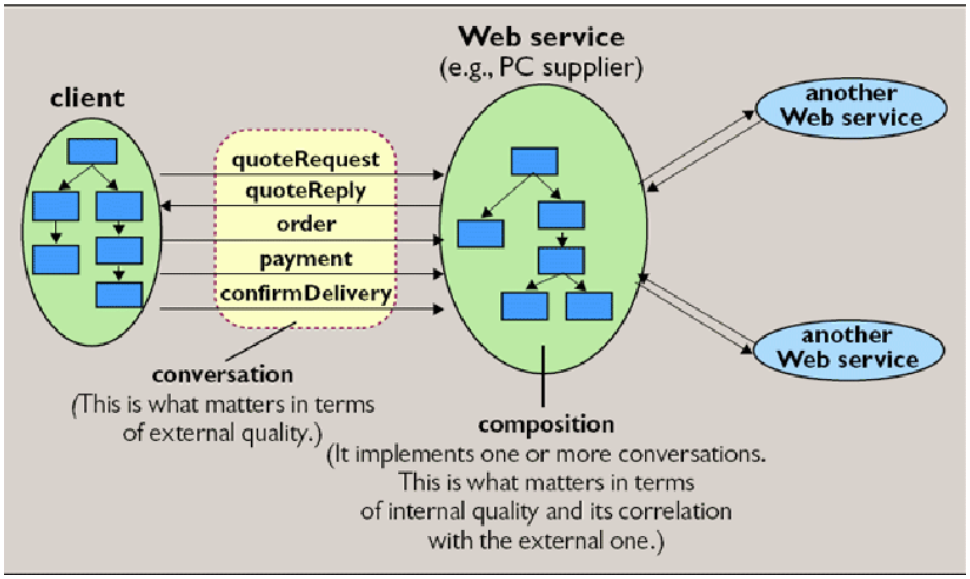
Motivations for External or "Inter-Enterprise" Composite Services

BUSINESS STRATEGY perspective - can focus on core competencies by using services provided by other firms in component business models

BUSINESS OPERATIONS perspective - can improve collaboration with "real-time" information exchange with suppliers and other partners

TECHNOLOGY perspective - composite application platform can exploit "on demand" services and reuse the models and techniques needed to integrate (and interoperate) applications and services

Web Services Composition



A Simple Service?

Simple vs Composite Services -- Not That Simple

If a service interface is the *published or public description of what the service does and how to request it*
then all services are simple from the service consumer's perspective

That's because any of the services that have been composed "behind the scenes" are invisible to the service consumer... THAT'S THE POINT OF SERVICE-ORIENTED COMPUTING

This COMPONENT ANONYMITY gives service providers flexibility in implementing and providing services because of TRANSPARENT SUBSTITUTABILITY

But the GRANULARITY or ABSTRACTION of the service presented to the service consumer matters a lot

Composite Services vs. Mash-ups

The key idea that a composite service reuses information from another service or information resource without changing it is shared by "mash-ups"

The boundary between them isn't sharp, but "mash-up" more often applies to service combinations created by people in an "individual" rather than a "business" or "organizational" context

Nevertheless, "enterprise mash-up" is an emerging term, which blurs the boundary even more

Readings for November 5

L. Cherbakov, A. Bravery, B. D. Goodman, A. Pandya, & J. Baggett, “Changing the corporate IT development model: Tapping the power of grassroots computing,” IBM Systems Journal, 2007.

L.Cherbakov, A. Bravery, & A. Pandya, “SOA meets situational applications, part 1: Changing computing in the enterprise,” IBM Developer Works, 23 August 2007.