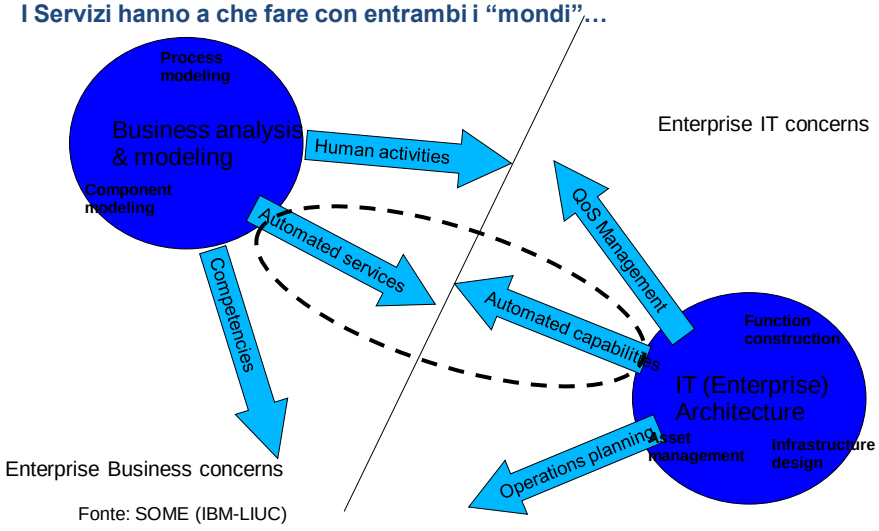
Introduzione a SOA

Information Systems Design

Il Gap tra IT e Business



Fonte: SOME (IBM-LIUC)

Il Gap tra IT e Business

– In numerose aziende l'area IT non svolge il ruolo di centro di raccolta delle istanze di implementazione provenienti dalle specifiche linee/aree di business.

– Le linee/aree di Business non sono integrate (org. a silos). Ciò comporta una crescita senza controllo dell'IT con possibili duplicazioni nelle funzionalità implementate con conseguente aumente dei costi di implementazione e manutenzione.

– I responsabili IT e quelli incaricati di definire i processi di business lavorano a compartimenti stagni, ignorando cosa sta implementando la controparte.

– Gli investimeni in IT delle aziende sono stati molto elevati in passato e spesso sono stati originati da una pianificazione non soddisfacente. Ora, la richiesta è per una strategia di "protezione" degli investimenti effettuati.

– I processi di business non sono più in grado di supportare e rispondere a nuove esigenze e requisiti di business

Fonte: SOME (IBM-LIUC)

Il Gap tra IT e Business

– I processi di Business devono essere rapidamente modificati in risposta ad un cambiamento delle esigenze di mercato. Difficilmente una tale flessibilità è correttamente supportata da processi IT non modificabili
  - Un'azienda è totalmente flessibile se i cambiamenti ai processi non implicano una modifica nelle procedure IT-based (o processi IT).

– Le fusioni aziendali spesso implicano duplicazioni e mancanza di efficienza → sovrapposizioni fra i processi di business delle due realtà (e di conseguenza di quelli IT).

L'area IT deve modificare il proprio ruolo da centro di costo a punto focale per ottenere e/o mantenere un vantaggio competitivo.

Fonte: SOME (IBM-LIUC)

Il concetto di allineamento di business (strategico)

Un approccio collaborativo nelle decisioni fra l'area IT e quella di business assicura:

- *Investimenti IT investments basati sulle priorità di business*
- *che la fornitura di servizi IT-based forniscaun risultato*
- *La valutazione delle priorità di business con le potenzialità ed i limiti dell'IT ben chiari in mente*

"Il processi tramite il quale chi si occupa di business e di IT collaborano per creare ***un ambiente nel quale:***

- ***gli investimenti IT e la fornitura di servizi IT rispecchino le priorità di business***
- ***le priorità di business siano influenzate dalla comprensione delle potenzialità e dei limiti dell'IT"***

"On IT-business Alignment" Macehiter Ward-Dutton, Feb 2005

Vantaggi del modello distribuito

Perchè il modello distribuito ha avuto successo:

- Le risorse che prima lavoravano da sole possono ora interagire tra di loro
- Hardware/Software specifici possono essere condivisi sulla rete tra più sistemi e non devono essere più duplicati su ogni sistema che necessità di funzionalità specifiche
- Avere macchine più piccole che lavorano insieme è in genere più economico e affidabile che un grande sistema centralizzato. E' possibile clonare gli stessi dati e la stessa funzionalità di business su sistemi diversi.

Svantaggi del modello distribuito

Ci sono però anche dei problemi:

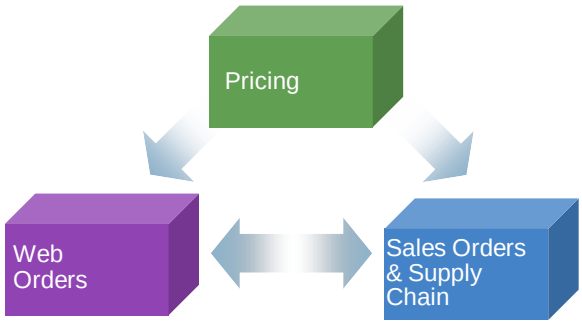
- I dati ridondanti che risiedono su macchine diverse devono essere continuamente sincronizzati (aggiornamenti su database sincronizzati, clock delle macchine sincronizzato)
- La complessità dei sistemi aumenta e si introducono nuovi possibili problemi (es. problemi alla rete, gestione delle singole macchine che possono essere dislocate in edifici o anche città diverse)
- Difficoltà di aggiornamento software/hardware e loro compatibilità.

Le esigenze di oggi per i modelli distribuiti

Oggi, un sistema distribuito deve soddisfare i seguenti requisiti:

- Deve essere basato su standard aperti
- Non deve dipendere dal linguaggio di programmazione, dalla piattaforma, dal Vendor
- Devono essere “Loosely coupled”: il modo in cui il client comunica con il server non dipende da come l’applicazione è stata implementata
- Deve essere semplice poter distribuire le applicazione sw e accedere ai clients e ai servers
- CORBA, DCOM, RMI etc indirizzano parte di questi problemi.

Perchè SOA ? /1



Applicazioni monolitiche di business

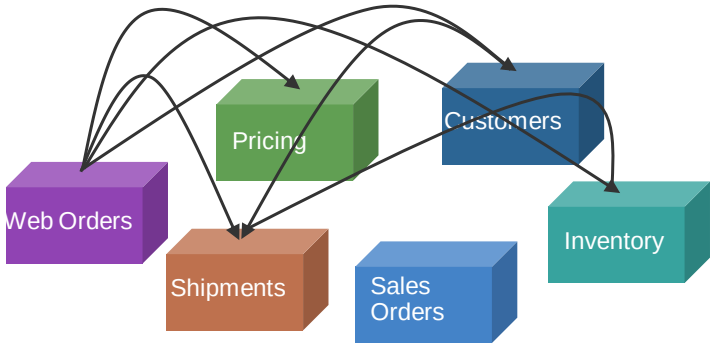
- Sincronizzazione periodica con le informazioni di magazzino
- Informazioni sui pressi inserite spesso inserite in modo dimosomogeneo in dipendenza dalla struttura applicativa
- Mancanza di un database unificato per i clienti ed il magazzino e di flessibilità nei processi di business

Fonte: IBM – SOA Foundation

Applicazioni monolitiche

- Grandi quantità di codice che non utile a descrivere il business
  - Impatto (negativo) sui costi di implementazione e manutenzione
- Scarsa estendibilità
  - Aggiungere o modificare funzionalità richiede la riscrittura di enormi porzioni di codice
- Difficoltà di integrazione con altre soluzioni applicative

Perchè SOA ? /2

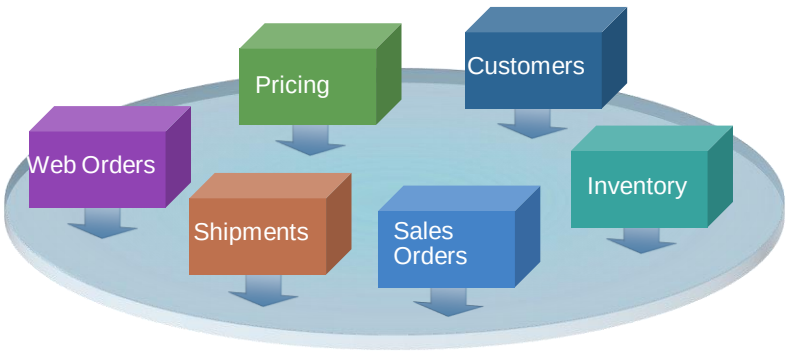


Servizi definiti come unità logiche di business, ma..

- Flusso di controllo – incorporato nella logica del servizio
- La trasformazione/conversione del formato dei dati è incorporato nella logica del servizio
- La stretto accoppiamento dei servizi li rende fragili

Fonte: IBM – SOA Foundation

Perchè SOA ? /3



Servizi definiti come unità di logica di business separate da:

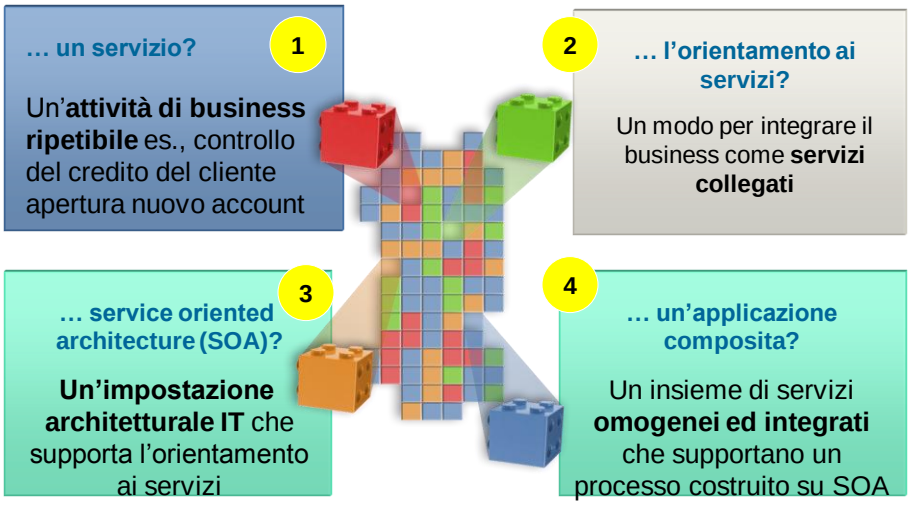
- Il flusso di controllo e indirizzamento
- Conversione del formato dei dati e trasformazione dei protocolli

Fonte: IBM – SOA Foundation

Un assunto fondamentale: "comandano" i processi

- In azienda il supporto dell'IT dev'essere rivolto ai processi aziendali.
- La Service-oriented architecture (SOA) ha a che fare con la creazione di processi:
  - migliori
  - più semplici da modificare
  - più economici da creare

Cos'è.....?



Fonte: SOME (IBM-LIUC)

Service Oriented Architecture /1

- Un approccio per modellare e implementare sistemi distribuiti che consentano una stretta correlazione fra il modello di business e la conseguente implementazione IT
- Un framework applicativo che disgrega le applicazioni di business fino ad ottenere un processo/funzione di business individuale, chiamato ***servizio***.
  Una SOA consente di costruire, sviluppare e integrare questi servizi indipendentemente dalle applicazioni (ERP, CRM) e dalla piattaforma IT (Windows, Linux..) sui quali i servizi vengono utilizzati…
- Focus su **flessibilità e riusabilità**

Service Oriented Architecture /2

- Con l'acronimo "SOA" (Service-Oriented Architecture) si fa riferimento al concetto di architettura (software) per la definizione dei servizi utilizzati a supporto delle richieste operate da parte degli utenti.
- Le singole applicazioni che compongono il processo di business sono dette "servizi".
- Il concetto alla base di SOA consiste nello svincolarsi da una specifica piattaforma considerando l'insieme dei servizi come un componente che può essere riutilizzato o modificato.
- SOA favorisce l'interazione tra le varie realtà aziendali aumentando flessibilità ed adattabilità.

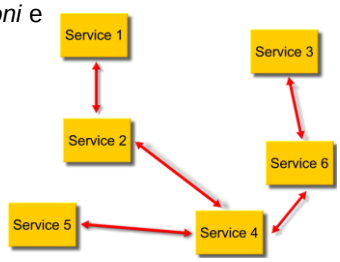
Service Oriented Architecture /3

- Una Service-Oriented Architecture è **un approccio** per organizzare le risorse IT in modo tale che l'accesso a risorse quali
  - i dati
  - la logica di business
  - l'infrastruttura IT

avvenga tramite il routing di messaggi tra interfaccie collegate in rete

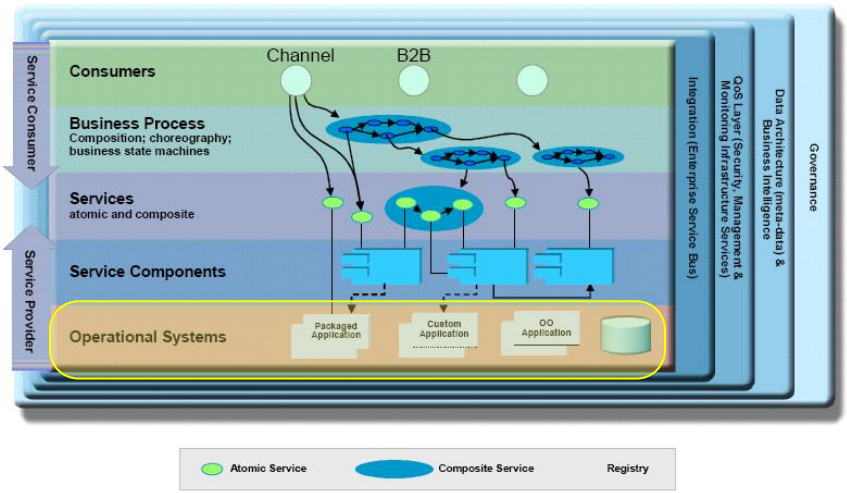
Service Oriented Architecture /4

Service Oriented Architecture è uno approccio architetturale basato su principi di design per lo sviluppo di *applicazioni* e l'integrazione che consente di ottenere:
1.interoperabilità
2.flessibilità
3.riutilizzabilità



- Le applicazioni sono presentate come **servizi**
- ***I servizi*** sono definiti da un'interfaccia standard
- ***I servizi*** sono componenti atomici usati per sviluppare nuove applicazioni (applicationi composite)

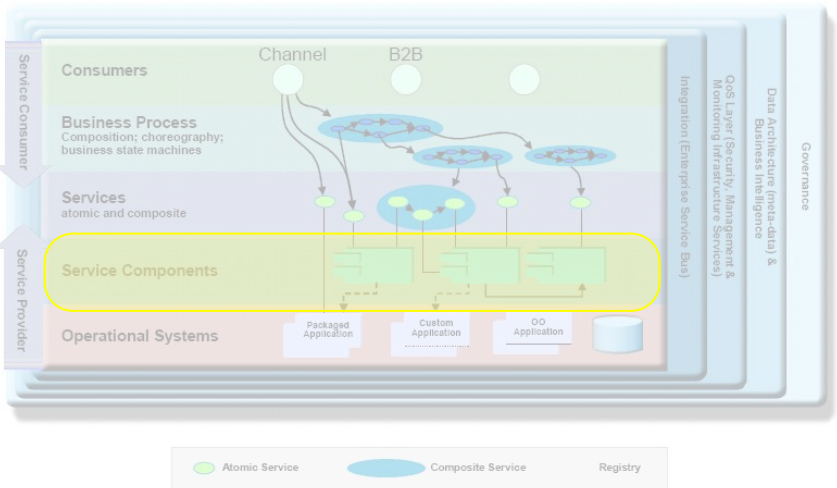
Architettura di riferimento /1



Sistemi operazionali: applicazioni e sistemi esistenti (ERP, CRM, sistemi legacy)

Fonte: SOME (IBM-LIUC)

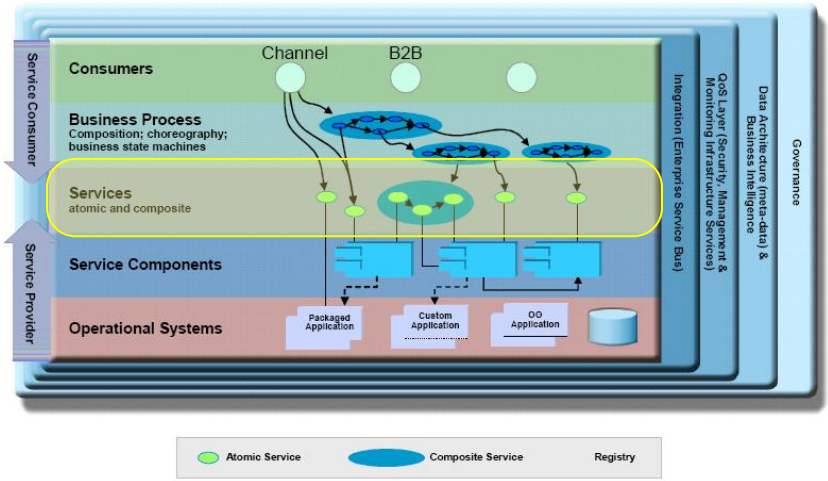
Architettura di riferimento /2



Componenti di servizio: connettori fra i servizi e le applicazioni basati su standard di comunicazione (XML, WSDL)

Fonte: SOME (IBM-LIUC)

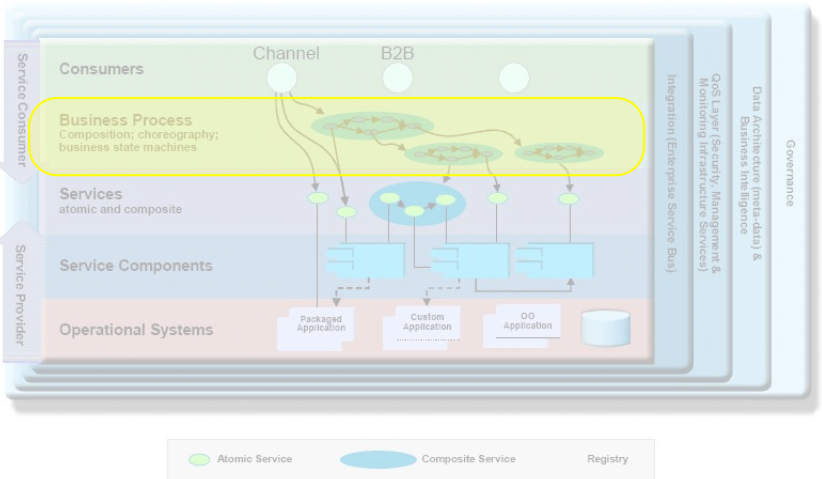
Architettura di riferimento /3



Servizi: forniscono appunti i servizi allo stadio superiore, quello dei processi aziendali

Fonte: SOME (IBM-LIUC)

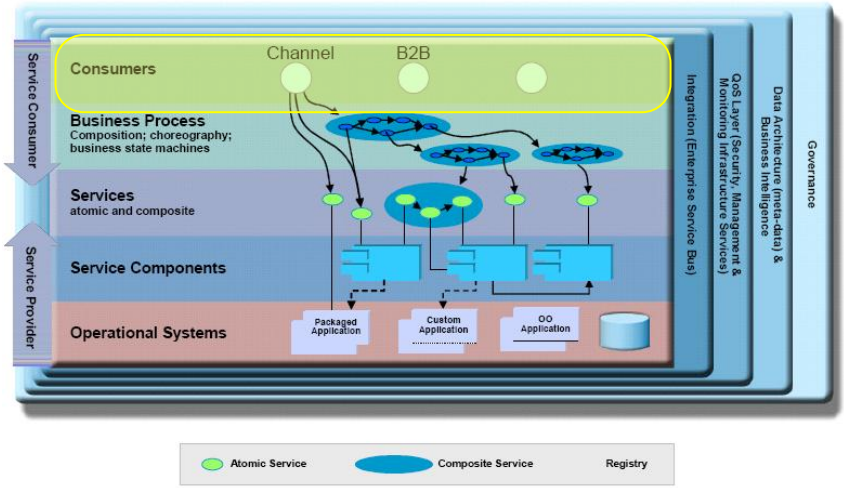
Architettura di riferimento /4



Processi: lo stadio che "coreografa" i servizi al fine di implementare i casi d'uso ed i processi stessi

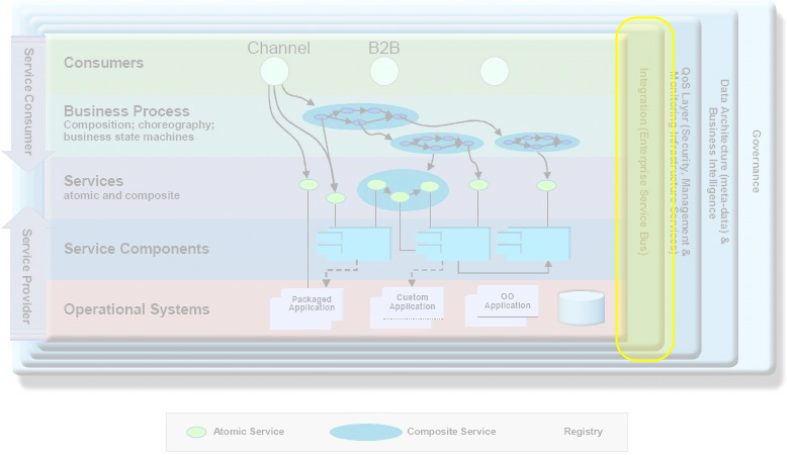
Fonte: SOME (IBM-LIUC)

Architettura di riferimento /5



Consumatori: lo strato di presentazione che fornisce l'esposizione dei processi

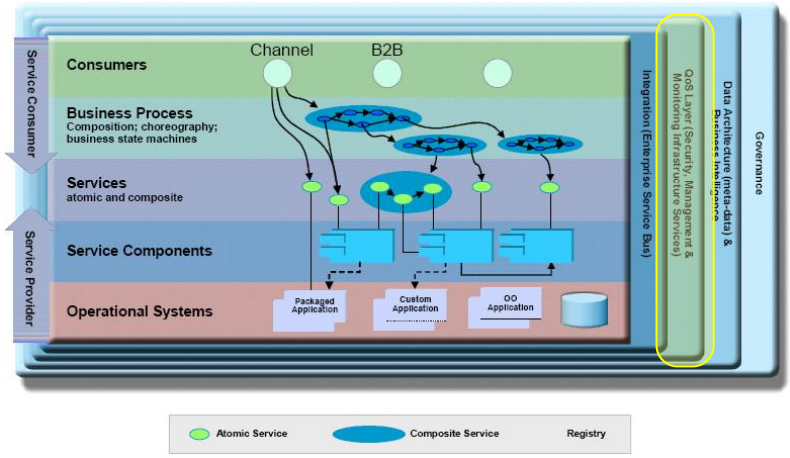
Architettura di riferimento /6



Integrazione: l'infrastruttura tecnologica che fornisce l'accesso e l'integrazione dei servizi (ESB - Enterprise Service Bus).

Fonte: SOME (IBM-LIUC)

Architettura di riferimento /7



QoS: strumenti di monitoraggio e gestione

Fonte: SOME (IBM-LIUC)

SOA: principi architetturali

- **Loose coupling** – I servizi mantengono relazioni che minimizzano le dipendenze (è richiesta sola conoscenza reciproca da parte dei servizi stessi)
- **Service contract** – I servizi aderiscono ad un accordo di comunicazione, come definito da uno o più documenti di descrizione del servizio
- **Service abstraction** – Oltre a quanto descritto nel service contract, i servizi nascondono la propria logica all'esterno
- **Service reusability** – La logica di business è divisa tra i servizi per promuovere la riusabilità
- **Service composability** – Collezioni di servizi possono essere coordinati e assemblati per creare un servizio composito
- **Service autonomy** – I servizi controllano la logica che incapsulano
- **Service discoverability** – I servizi sono progettati per essere esteriormente descrittivi, in modo da poter essere trovati e valutati tramite apposititi meccanismi (repository e ricerca)

Descrizione di una SOA /elementi

1. FRONTEND DELLE APPLICAZIONI (MASCHERA INSERIMENTO ORDINI....)
   - Instaurano e controllano tutta l'attività del sistema aziendale
   - Attiva un business process e riceve i risultati
2. SERVICE (NUOVO ORDINE...)

Componente software che incapsula un concetto di business ad alto livello:

   - Contract
   - Interface
   - Implementation
   - Business logic
   - Data
3. SERVICE REPOSITORY
   - Contiene le informazioni per accedere ai servizi disponibili
   - Mette a disposizione i mezzi necessari per scoprire i servizi disponibili
   - Un repository non è necessario, ma lo diviene a lungo termine
4. SERVICE BUS

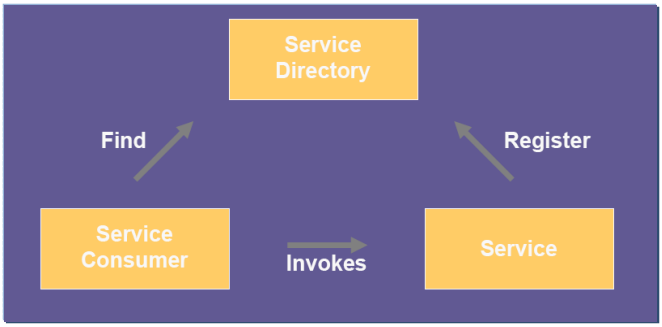
Connette tra loro tutti i componenti di una SOA, spesso basati su linguaggi di programmazione, sistemi operativi e ambienti di run time eterogenei

   - Deve supportare diversi tipi di comunicazione (almeno sincrona e asincrona)
   - Deve fornire servizi "tecnici": es. logging, auditing, security …

Caratteristiche di un servizio

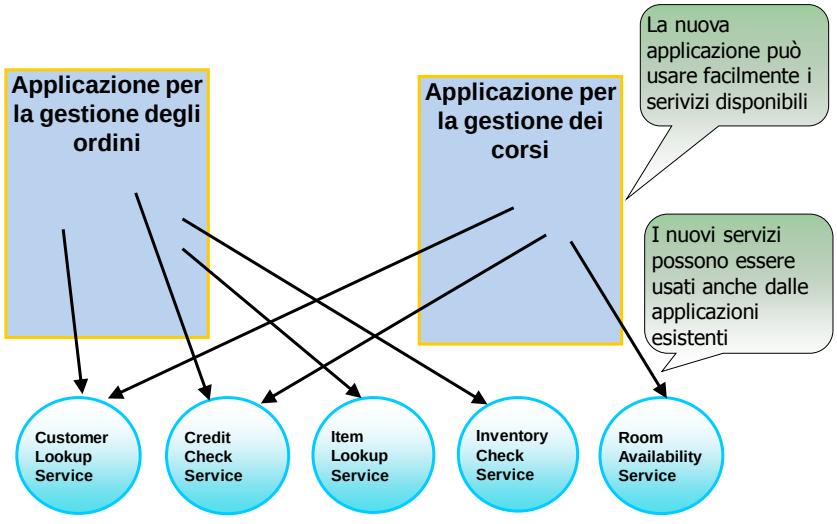
- Moduli che eseguono un compito predefinito
- Componenti software che hanno contratti/interfacce pubblicate (pubbliche)
- Sono Platform-Independent
- Sono Language-Independent
- Sono Operating-System-Independent
- I "consumatori/clienti" possono dinamicamente scoprire i servizi
- Sono interoperabili
- Sono riusabili

La logica di utilizzo di un servizio



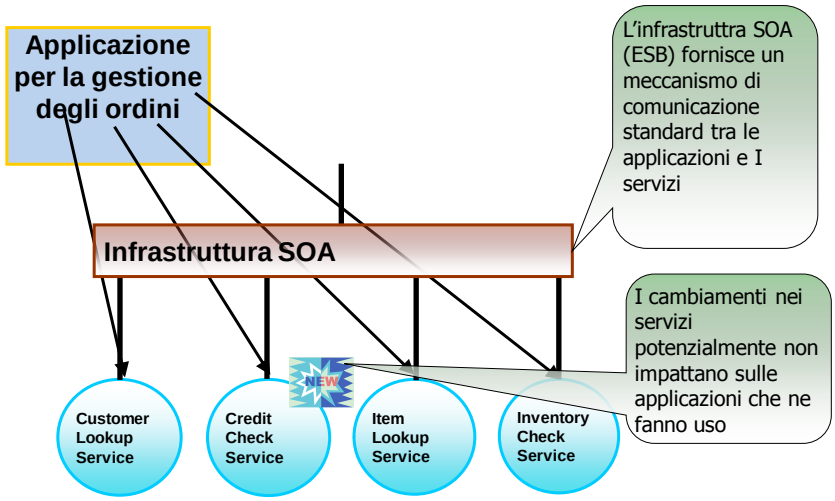
Fonte: David Pilling, Service Oriented Architecture for Law Firms: SOA is inevitable, are you ready?

Servizi e agilità (riusabilità)



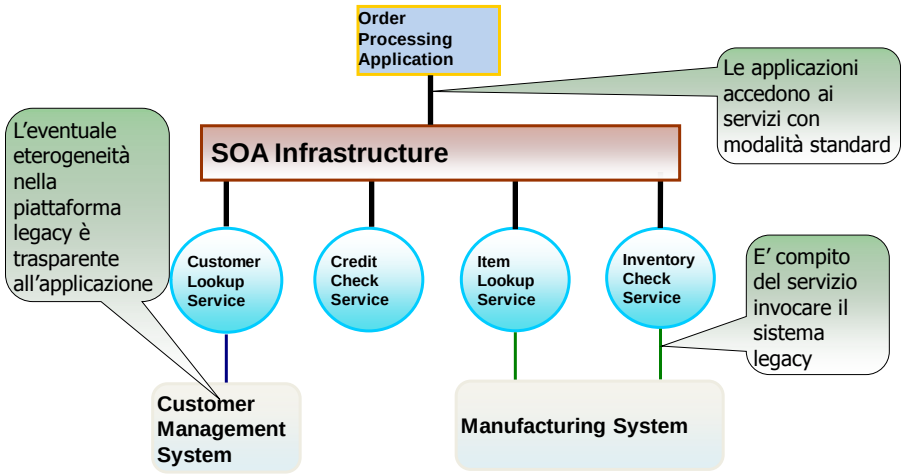
Fonte: Dennis Smith, Software Engineering Institute, 2006 Carnegie Mellon Univ.

Servizi e adattabilità



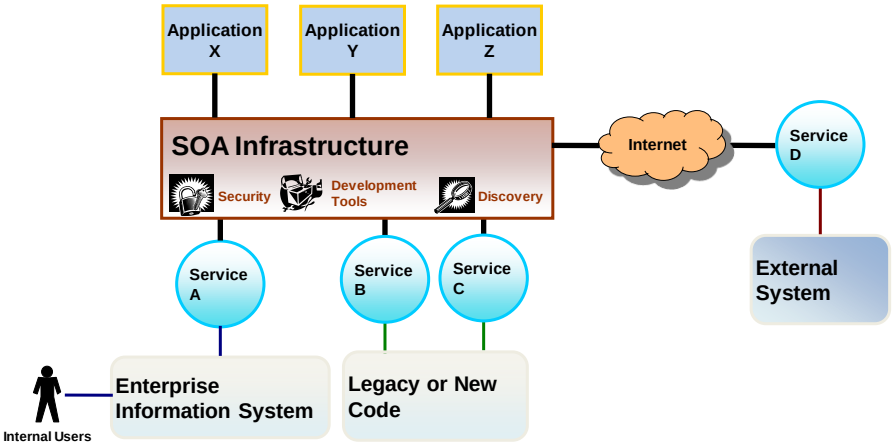
Fonte: Dennis Smith, Software Engineering Institute, 2006 Carnegie Mellon Univ.

Servizi e Legacy System



Fonte: Dennis Smith, Software Engineering Institute, 2006 Carnegie Mellon Univ.

Componenti di un sistema basato su SOA



Fonte: Dennis Smith, Software Engineering Institute, 2006 Carnegie Mellon Univ.

Benefici tecnologici

- Costruire i processi è più semplice e rapido:
  - I servizi esistenti possono essere riutilizzati
  - Le applicazioni possono invocare i servizi con modalità standard
- Le applicazioni possono essere interfacciate più facilmente con diversi client:
  - Windows clients, ASP.NET/JSP, etc. Platform Indipendenza: i servizi “Loosely coupled” non richiedono più la stessa implementazione tecnologica ai 2 estremi (applicazione e sistema ERP/CRM/legacy)

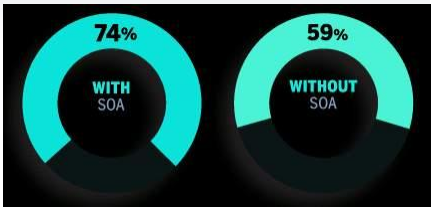
Benefici di business

- I manager capiscono i servizi
  - Migliore interazione fra manager e responsabili IT
- I processi diventano più "espliciti"
  - Quindi più comprensibili e migliorabili
- Le applicazioni e/o i processi diventano più facilmente esternalizzabili
  - Perchè ben definiti e separati

CIO's e SOA

SOA Leads to the Executive Committee's Door

CIO is part of the Executive Committee



SOA Spells Money

Average Compensation of CIOs Working WITH an SOA	Average Compensation of CIOs Working WITHOUT an SOA
$250,000	*$159,000*

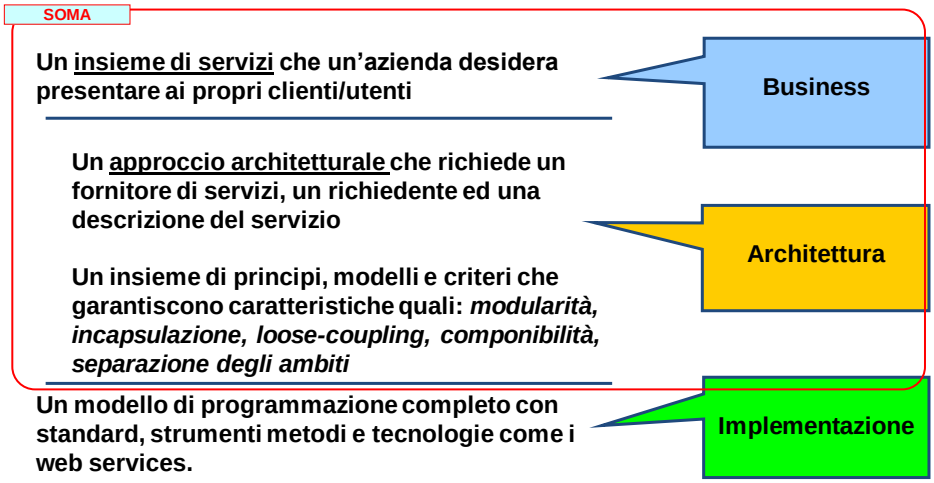
And their budgets are Bigger too

WITH SOA	WITHOUT SOA
8.9%	*5.8%*

Source: "State of the CIO", January 1, 2007, CIO Magazine
http://www.cio.com/state/survey_slideshow/index.html

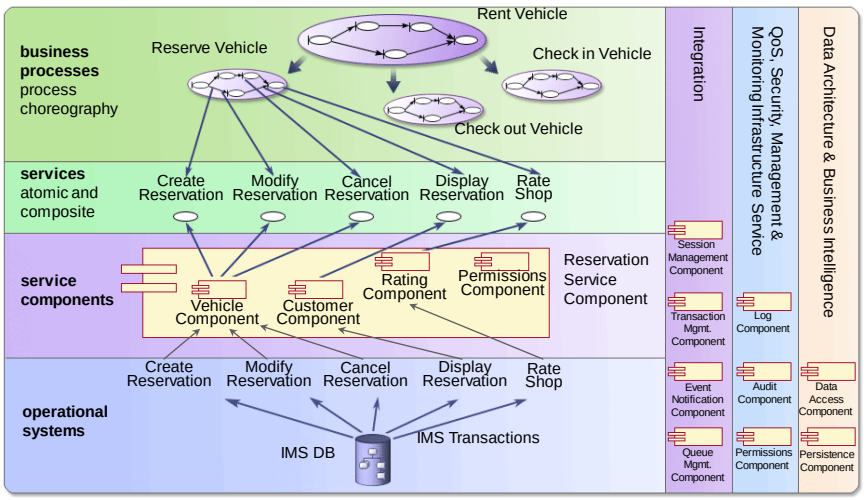
Note: IT budget as percent of overall revenue

SOA: non solo tecnologia anzi….



Fonte: SOME (IBM-LIUC)

SOMA:
Service Oriented Modeling and Architecture



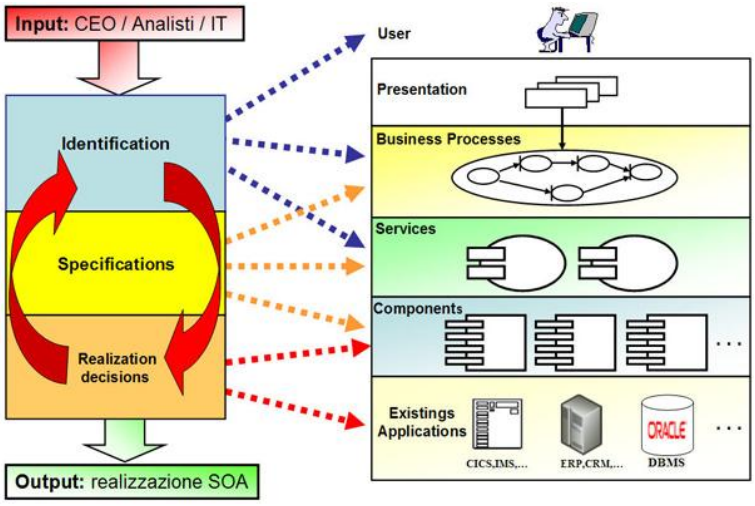
SOMA: diversi approcci (Top-Down)

- Se il punto di partenza è il ridesign o dei processi → approccio Top-Down
- Si parte dall'analisi del processo per poi identificare i servizi con un processo suddiviso in due fasi
  - la prima di analisi, che deve definire un elenco di servizi "candidati" scelti sulla base della ridondanza (riusabilità)
  - la fase di design che deve delineare in maniera netta e precisa i servizi e le relazioni che hanno con i processi e i componenti, attraverso raffinamenti successivi.

SOMA, Top-down../1

- SOMA (Service-Oriented Modeling and Architecture) è una metodologia che aiuta ad applicare l'approccio Meet-in-the-Middle atraverso le fasi di
  - **Identification** (si individuano i possibili candidati a diventare servizi → ridondanza)
  - **Specification** (si decide quali dei servizi candidati precedentemente individuati esportare come servizi veri e propri e per ognuno di essi si specifica quali sono i componenti enterprise che lo compongono)
  - **Realization** (si prendono le decisioni operative per la realizzazione dei servizi)

SOMA, Top-down../2

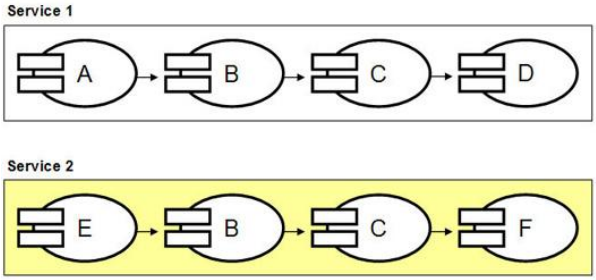


SOMA: diversi approcci (Bottom-Up)

- Se il punto di partenza è il software esistente (preservare l'investimento, applicazioni consolidate, budget)→ approccio Top-Down
  - è importante capire quanto dell'esistente può essere promosso a servizio e con che granularità.
  - Un punto di partenza per definire i candidati ai servizi, partendo da una situazione esistente, può essere l'identificazione degli scenari d'uso più frequenti.
  - Si identifica quindi il codice che in effetti viene maggiormente utilizzato (cioè per la maggior parte del tempo).

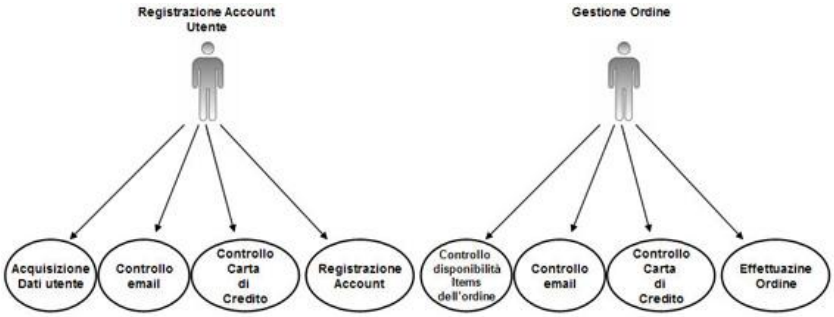
SOMA
servizi “coarse-grained” e riusabilità

- Supponiamo di avere 2 servizi composti rispettivamente, dai servizi ABCD e EBCF



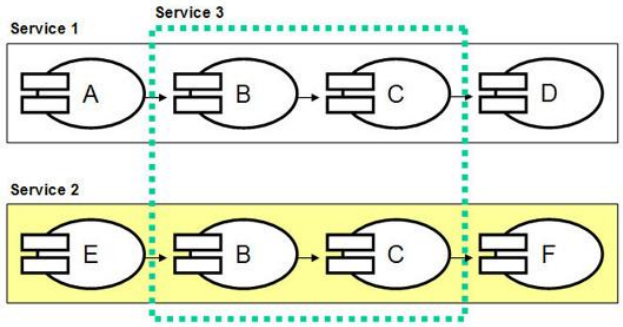
SOMA
servizi “coarse-grained” e riusabilità

- Supponiamo che i 2 processi siano la registrazione di un utente e la gestione di un ordine



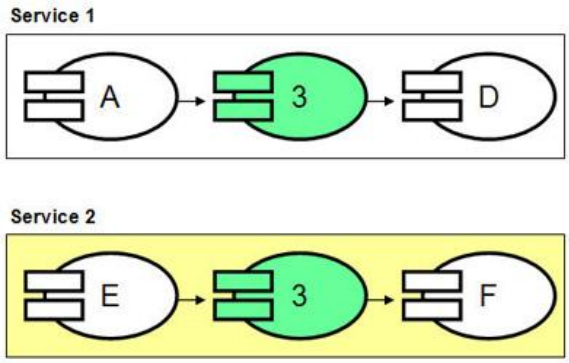
SOMA
servizi “coarse-grained” e riusabilità

- Questo suggerisce di portare a fattore comune i servizi B, C in un nuovo servizio di grana più grossa: il servizio 3.



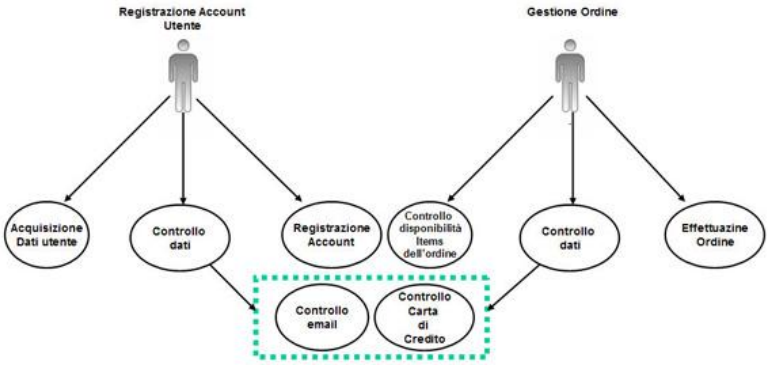
SOMA
servizi “coarse-grained” e riusabilità

- La nuova configurazione è quindi la seguente:



SOMA
servizi “coarse-grained” e riusabilità

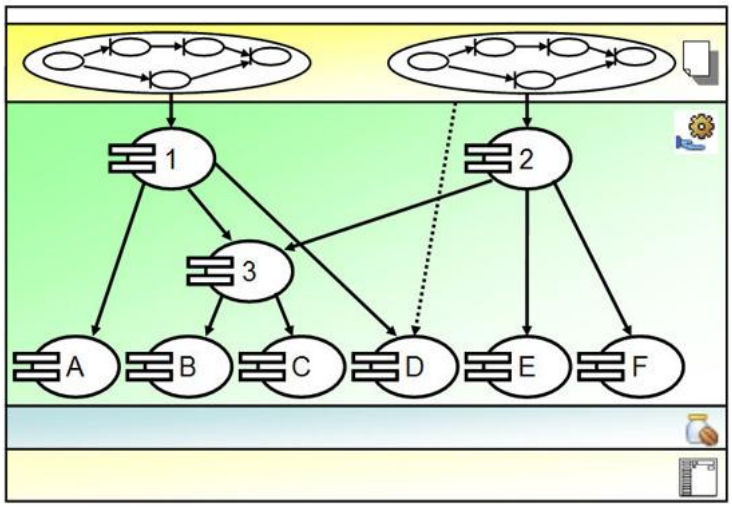
- Applicando questo concetto all'esempio pratico vuole dire che al posto di due singoli servizi di controllo email e di controllo carta di credito ne avremo uno solo.



SOMA
servizi “coarse-grained” e riusabilità

- Dopo questa fase di refactoring proviamo a chiederci: il Servizio 3 è più o meno riusabile dei singoli servizi B, C e D?
- È evidente che il servizio 3 sarà in generale meno riusabile dei singoli servizi che ha aggregato.
- È altrettanto vero però, che il servizio 3 è a grana più grossa. Sicuramente esprime funzionalità di business più chiaramente identificabili e quindi ha un maggiore valore. Inoltre, poichè il servizio 3 è composto da (B + C), di fatto i servizi elementari B e C sono ancora disponibili e usabili nel sistema.
- Nel caso specifico si è quindi definito un servizio a grana più grossa e con un valore di business maggiore: questo può essere più importante dell'effettiva possibilità di riuso.

SOMA
servizi “coarse-grained” e riusabilità



Cosa è un Web Service?

Un Web Service non è altro che un’applicazione distribuita che espone una interfaccia XML.

In particolare espone delle funzioni che si possono chiamare e che sono basate su uno standard. Non dipendono dal linguaggio di programmazione in cui sono scritte e dalla piattaforma sulla quale Girano. In questo modo:

- **L’applicazione non dipende dalla piattaforma o dall’implementazione**
- **L’applicazione si basa su uno standard aperto per la sua descrizione e la sua invocazione**
- **Connette diversi processi tra di loro tramite un’interfaccia programmatica**

Perchè i Web Services sono utili in Internet

Internet è cambiato: dal dominio delle applicazioni business to-consumer (B2C) che si basano su transazioni iniziate manualmente dall'utente su un browser (browsing di documenti, Download di files..), si sta passando ad interazioni business-to-business (B2B) dove le transazioni sono iniziate automaticamente da un programma.
In questo nuovo contesto i Web Services possono integrare le diverse risorse disponibili sul Web da un punto di vista programmatico e farle interagire insieme in maniera neutra

Web 2.0: Una nuova visione per Internet

- Non un software specifico, ma piuttosto un insieme di patterns
- Non è legato ad un particolare vendor, non ha un marchio
- L'obiettivo è quello di sviluppare su tool e tecnologie che permettano ai dati di poter essere consumati senza sapere chi li ha prodotti.
- Per ottenere ciò occorre usare nuove tecnologie: Ajax, javaScript, e XML ad esempio
- Applicazioni non più ad "unico blocco" ma composte da piccole sub-applicazioni: riuso, flessibilità, semplicità

Web 2.0, SOA e Web Services

- Web 2.0 è composto da 3 core patterns: