# Arsitektur Teknologi Informasi

## Services Oriented Architectures

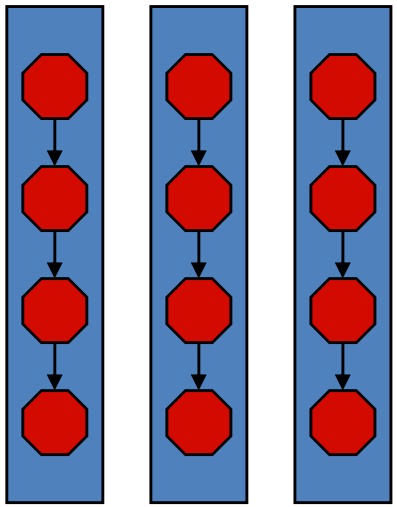
anton@ti.ukdw.ac.id

# Tes Kecil I

- Bagaimana hasilnya?

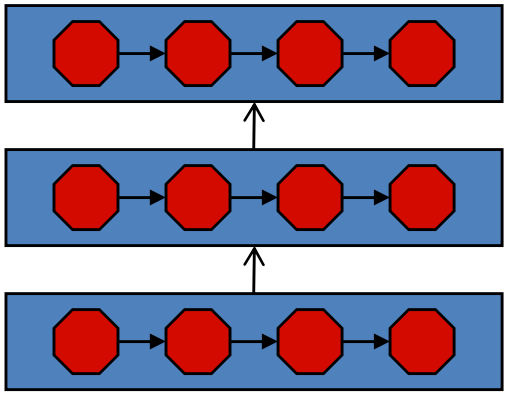
# Directions of System Architecture

1960 - 1980



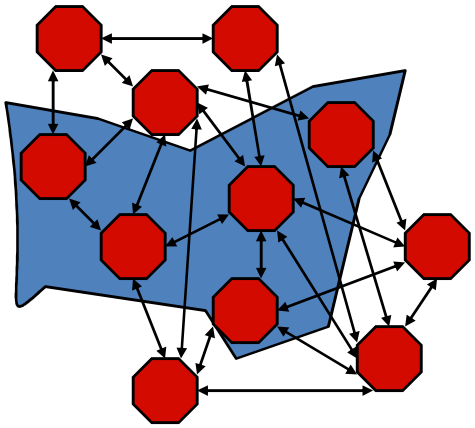
- Organization Focus
- Mainframe Centric
- Internal Use
- Unique Data

1990 - 2000



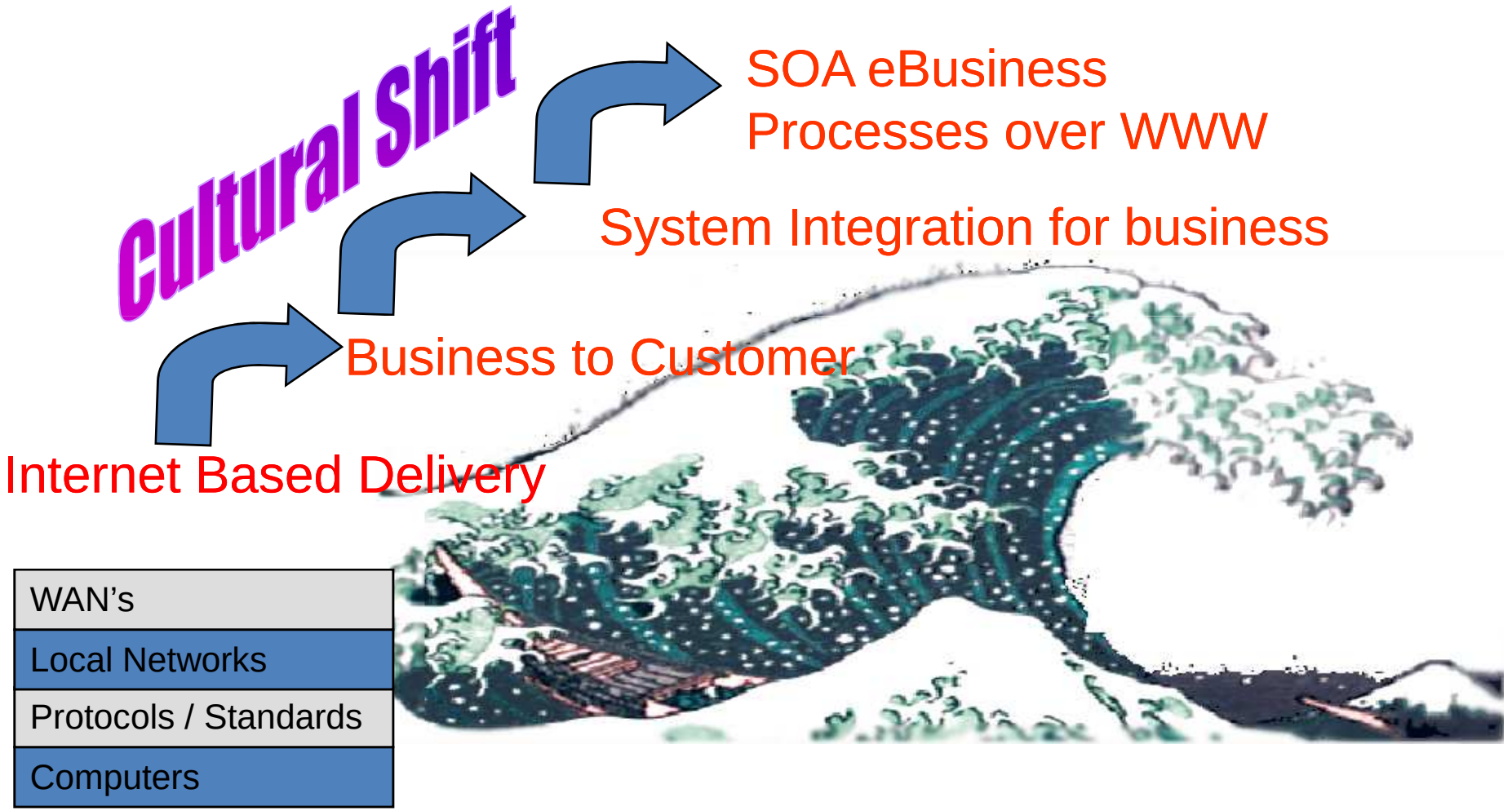
- Process Focus
- Client Server
- Partial Connectivity
- EDI File Transfer

2010 - 2050

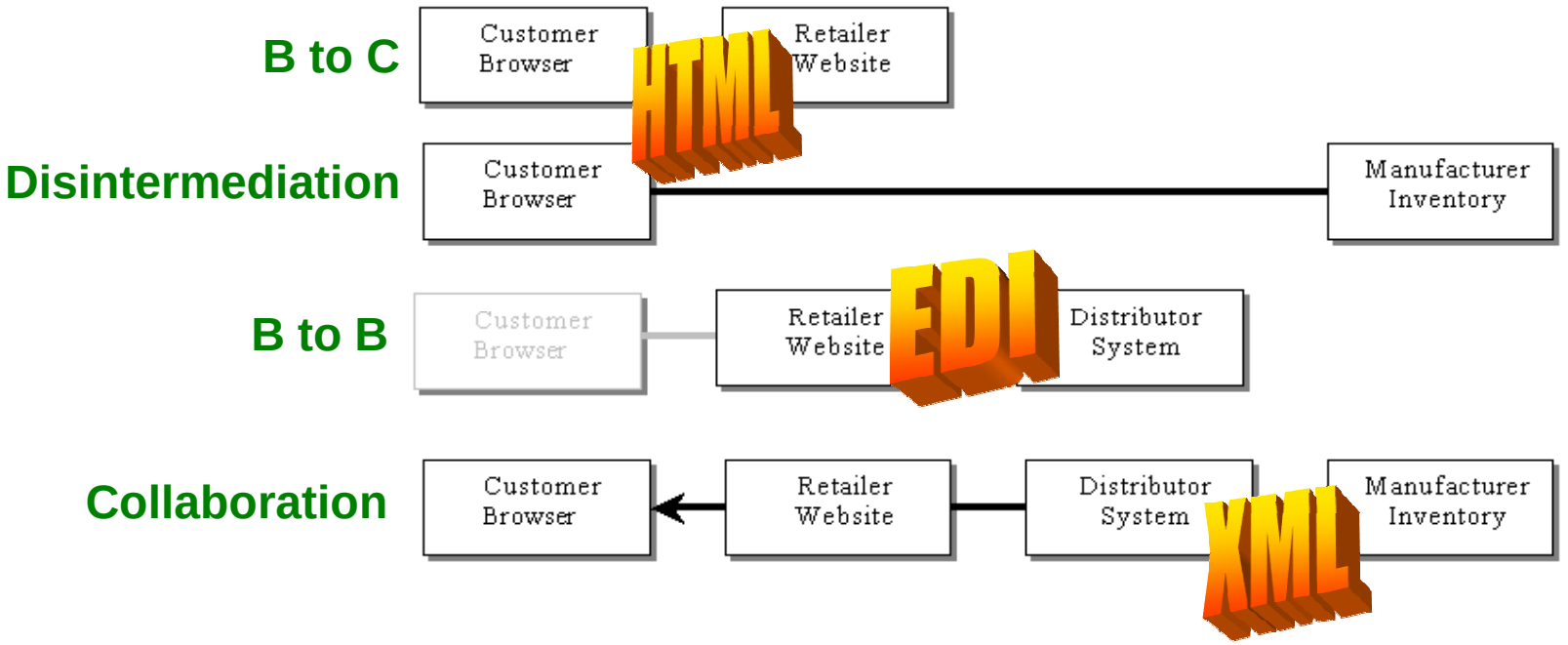


- Distributed Functions
- Data Centric
- Universal Interoperability
- Real-time Connectivity

# Business shift to Business Services



# Business Interaction and Tool



# What is Service?

- Service is component of distinctive functional meaning that typically encapsulate a high level business concept
- Service contains
  - Contract : message type def, constraint, description (comment)
  - Interface : set of operations
  - Implementation : logic and data

# Service Examples

- Creating a Purchase Order inside an application
- Requesting and reserving a room in a hotel
- Applying for a loan by filling out a loan request form
- Search books/music based on keywords

# Service Oriented Architecture

- SOA adalah sebuah konsep software architecture yang mendefinisikan penggunaan **layanan** untuk mendukung kebutuhan pengguna software.
- A service-oriented architecture is a framework for **integrating business processes** and supporting IT infrastructure, standardized components—**services**—that can be **reused** and **combined** to address **changing** business priorities (Business Perspective)

# SOA

- A set of components which can be invoked, and whose interface description can be published and discovered (W3C).
- Service oriented architecture is a client/server design approach in which an application consists of software services and software service consumers (also known as clients or service requesters). SOA is loose coupling between software components, and in its use of separately standing interfaces (Gartner)

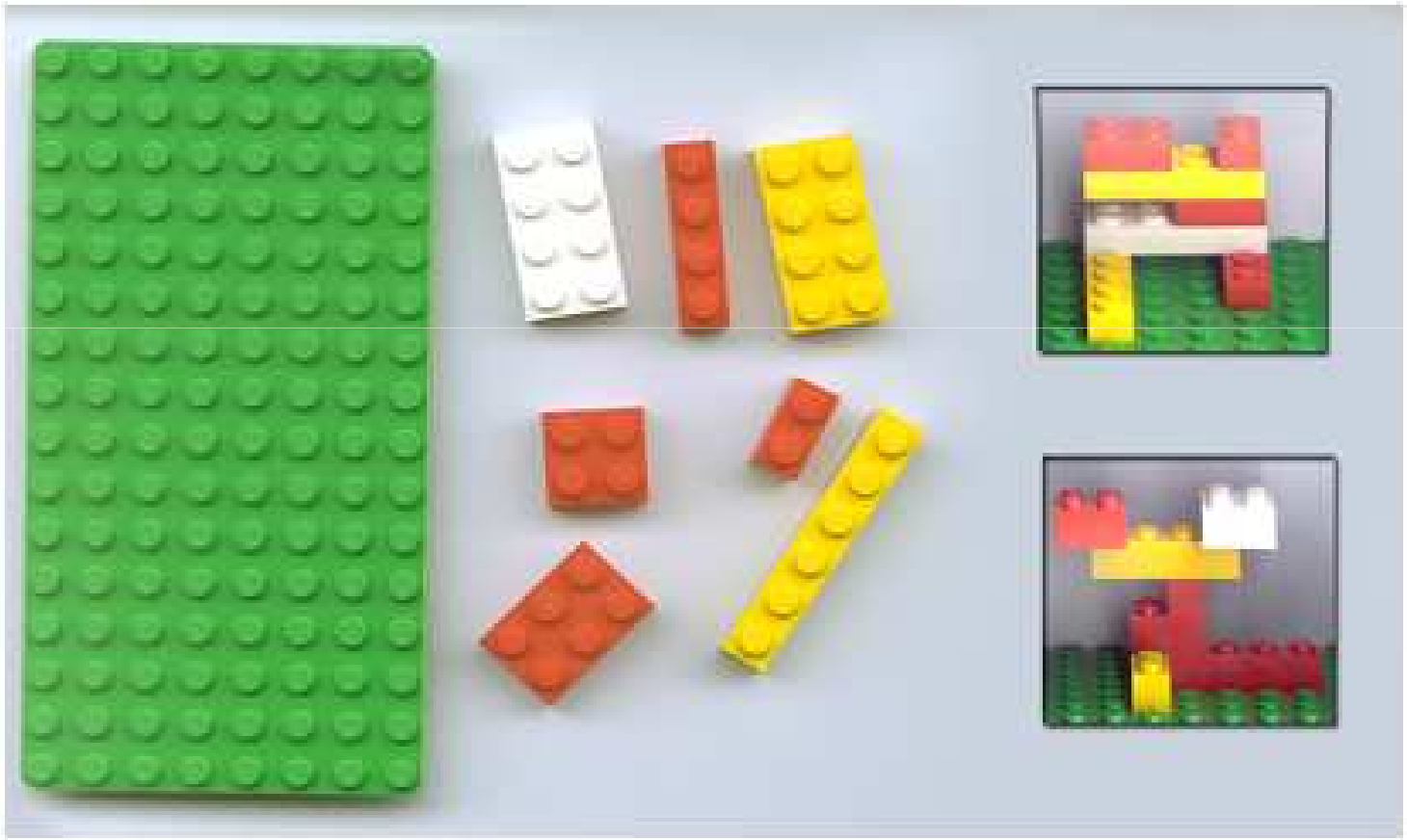
# Agar dapat mengimplementasikan SOA


The core components which make up an SOA implementation

# Bentuk SOA

- SOA adalah sebuah arsitektur yang merepresentasikan fungsi dalam bentuk layanan
  - Mengapa **fungsi**?
    - Karena fungsi menunjukkan **abstraksi aktivitas** – sesuatu yang secara alami dilakukan oleh aplikasi/program, individu, dan organisasi
  - Mengapa **layanan**?
    - Karena layanan mengabstraksikan fungsi dan dapat menunjukkan bentuk **hubungan** yang bermakna antara 2 pihak yang berkomunikasi (**requester** dan **provider**)

# SOA seperti puzzle



# SOA dan implementasinya

- Ada dua arah implementasi SOA:
  - Inward → ke dalam institusi sendiri → integrasi sistem-sistem yang ada untuk membangun fungsionalitas yang lebih luas
    - Misal : untuk Supply Chain Management
  - Outward → memanfaatkannya sebagai perluasan sistem yang ada (external network, peluang bisnis, dsb)
    - Contoh: layanan pembuatan file PDF secara online

# Penyebab SOA dan Tujuan SOA

- Pendorong berkembangnya SOA dari sisi bisnis:
  - Large scale **Enterprise** systems
  - **Internet** scale provisioning of services
  - Want to **reduce** the cost of doing business
- Tujuan
  - **Just-in-time integration** of applications by **discovering** and **orchestrating** network-available services

# SOA dan Integrasi Aplikasi/Sistem

- SOA berfungsi sebagai **platform integrasi**:
  - SOA memisahkan antara **pesan/query/call** dengan **pemrosesan**
  - Pesan/query/call **distandardisasi** dan tidak dikaitkan dengan sebuah produk teknologi tertentu, sehingga bisa dikirimkan/diterima oleh siapapun
  - SOA memisahkan antara bagian **publik** dan bagian **privat**
    - Bagian **publik** dapat diakses oleh siapapun, berupa deskripsi tentang layanan yang ditawarkan
    - Bagian **privat** hanya bisa diakses oleh pemilik/penyedia layanan

# Sifat SOA

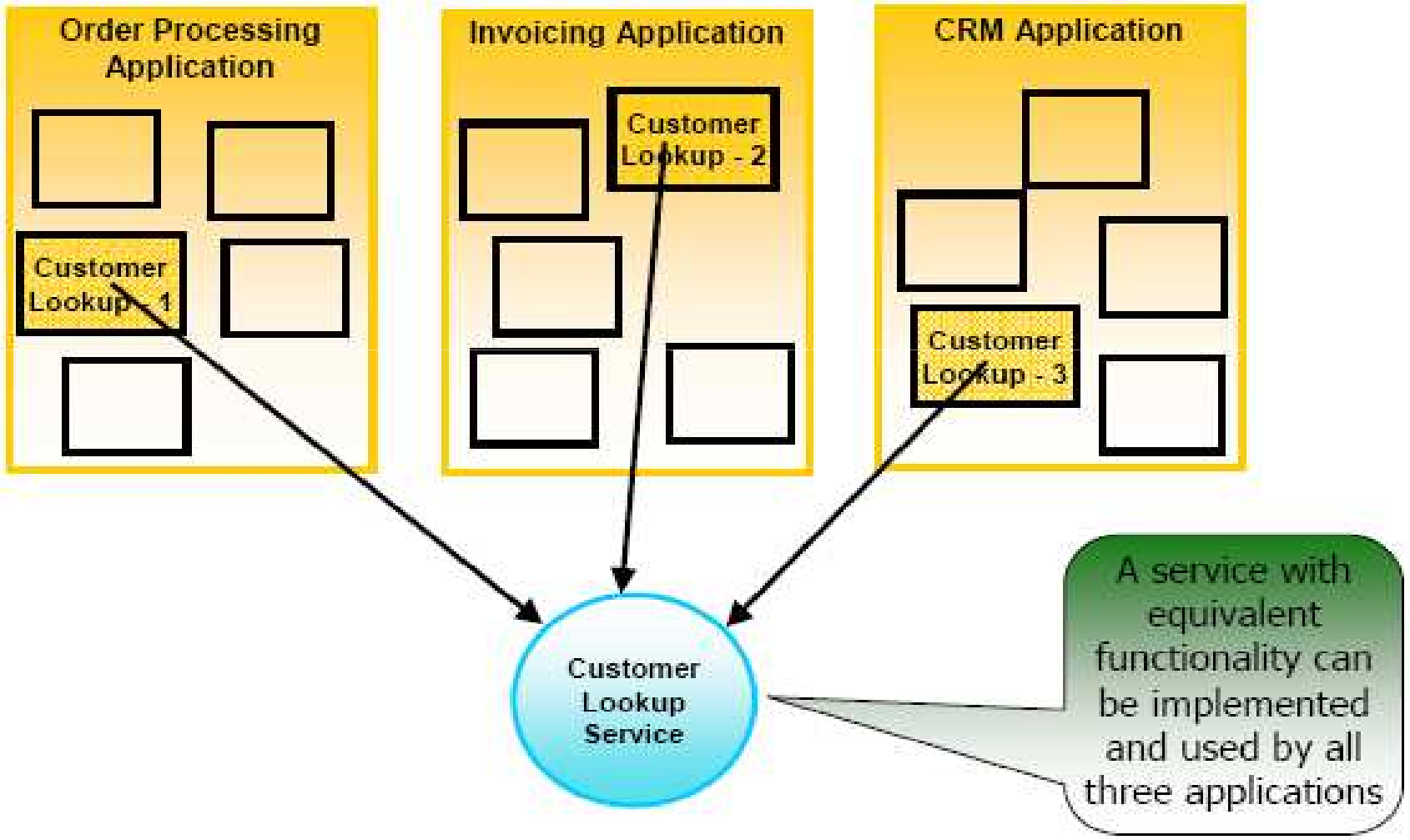
- SOA bersifat **behind the scence**,
  - SOA tidak terlihat secara langsung oleh client, SOA dihadapkan pada client melalui client UI
  - Digunakan untuk berkomunikasi antar aplikasi

- SOA merupakan suatu **service** yang “hanya menunggu” (**listen**) secara terus-menerus untuk digunakan.

# Keuntungan SOA

- **Better reuse of services**
  - Build new client functionality on top of existing Business Services
- **Loosely coupling**
  - Make changes without affecting another
- **Easier to maintain**
  - Changes/Versions are ok!
- **Platform Independence**
  - An enterprise can leverage its existing legacy applications that reside on different types of servers

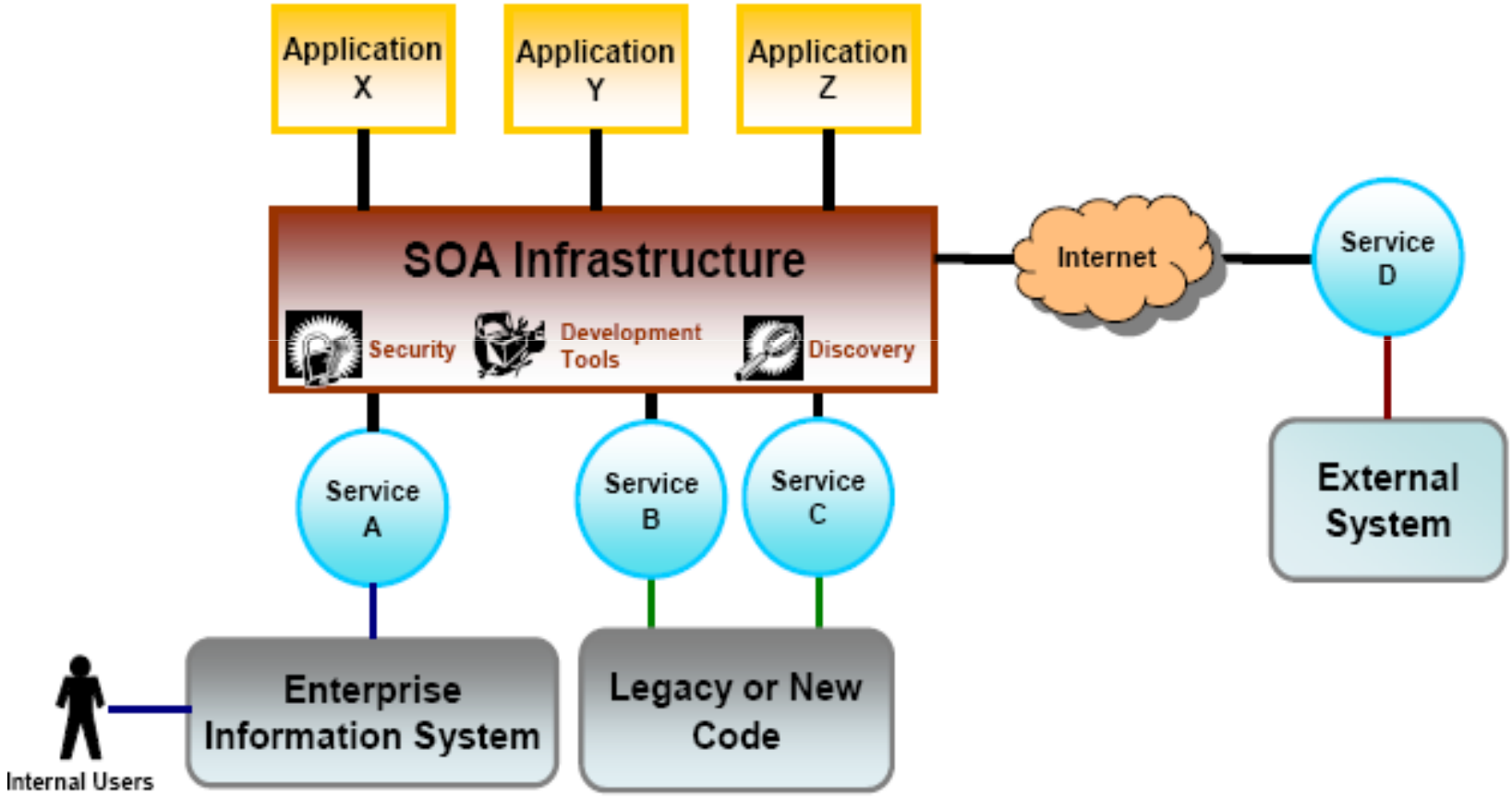
# Reusability



# Keuntungan SOA (2)

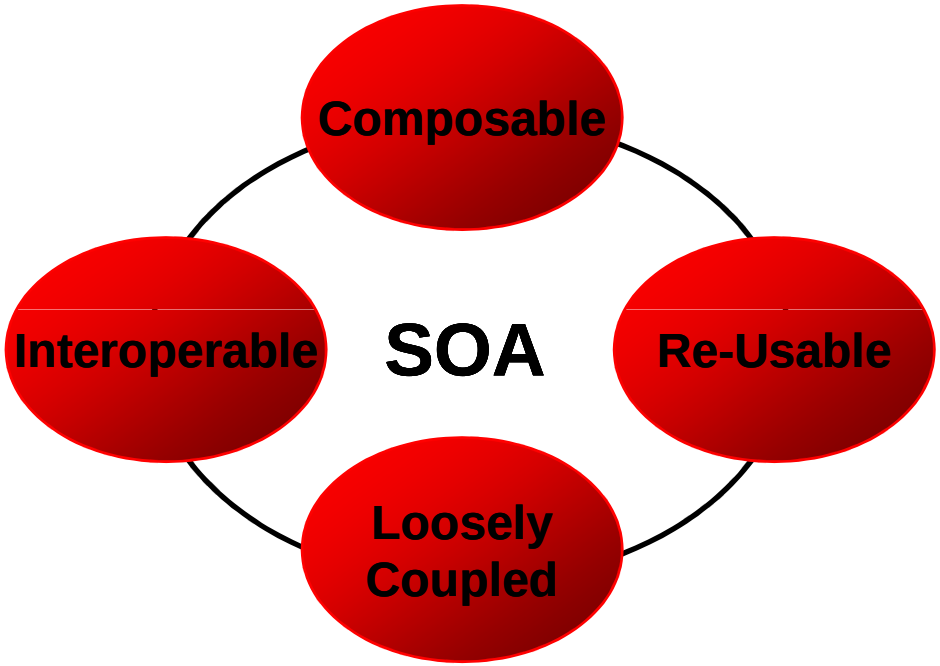
- **Code Reuse**
  - the services can be **reused** in multiple applications
- **Location Transparency**
  - Web services are often published to a directory where consumers can look them up
- **Better scalability**
  - there can be **multiple** instances of the service running on different servers. This increases scalability
- **Higher availability**
  - Since the location of a service does not matter and you can have multiple instances of a service, it is possible to ensure high availability

# Scalability



# Teknologi yg digunakan SOA

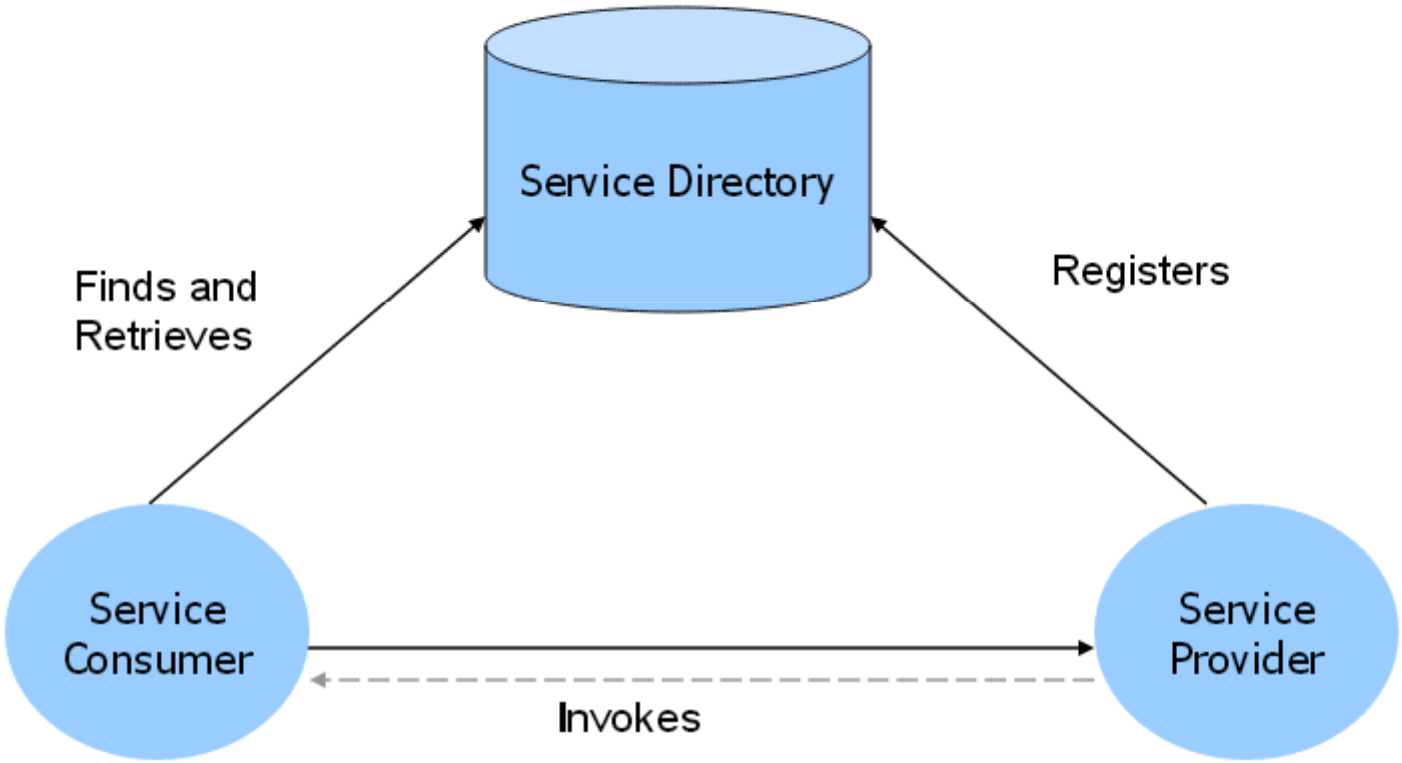
- Services have **platform independent**, self describing interfaces (XML)
- Messages are **formally defined** (WSDL)
- Services **can be discovered** (UDDI)
- Services have **quality of service characteristics** defined in policies (SOAP)
- Services can be **provided on any platform** (HTTP)
- Services can be **secured** (WS-Security)



# Komponen/Bagian SOA

- Layanan / **Service**
- Penyedia layanan / **Provider**
- Pemakai layanan / Consumer / **Requester**
- Tempat penyimpanan / **Registry**
- Pesan / **query** / call

# SOA model relation



# Beberapa Istilah dalam SOA

- **Service**: suatu fungsi yang menerima satu atau lebih request dan mengembalikan satu atau lebih response yang terdefinisi dengan baik dengan menggunakan interface yang standar.
  - Service is **self-contained**. That is, the service maintains its own state
  - Interface contract to the service is **platform-independent**
  - Service can be **dynamically located and invoked**
  - Pengguna service **dapat menentukan** service yang diperoleh untuk digunakan dalam application logic mereka.

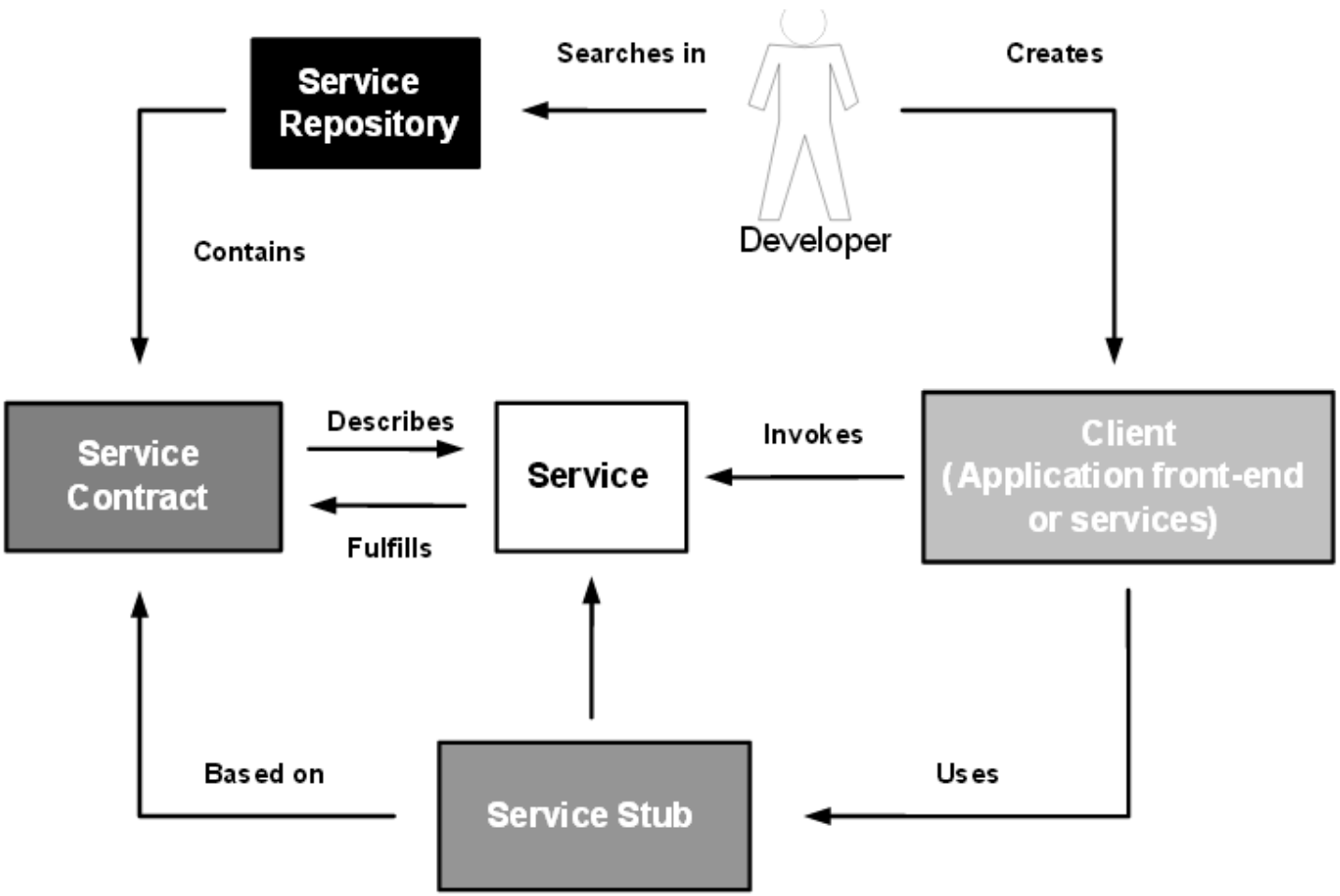
# Beberapa Istilah dalam SOA

- **Provider**: bagian dalam SOA yang menyediakan services
  - Terdiri dari >= 1 service
  - Harus dapat ditemukan oleh requester
  - Mendaftarkan dulu ke registry
- **Requester**: bagian dalam SOA yang mencari dan menggunakan services
  - Dapat menggunakan lebih dari 1 service
  - Harus dapat mencari provider
  - Mungkin mencari di registry

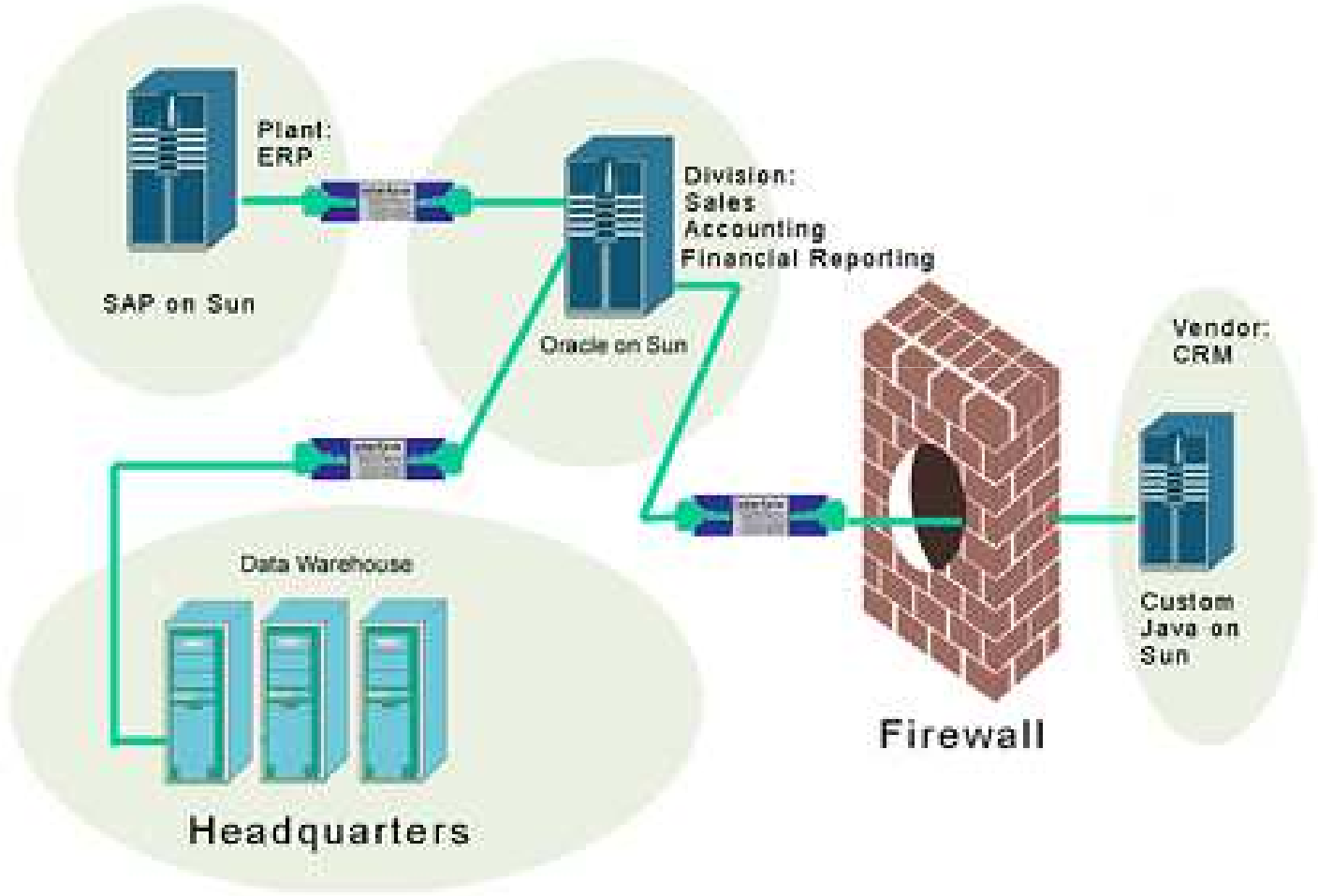
# Beberapa Istilah dalam SOA

- **Registry**: tempat penyimpan informasi provider-provider yang menyediakan berbagai services
  - Berupa layanan yang listen terus menerus
  - Bisa berbayar atau gratis
- **Query**: mekanisme invocation service
  - Berupa permintaan service yang bersifat standar
  - Menggunakan format khusus agar dapat dibaca oleh service

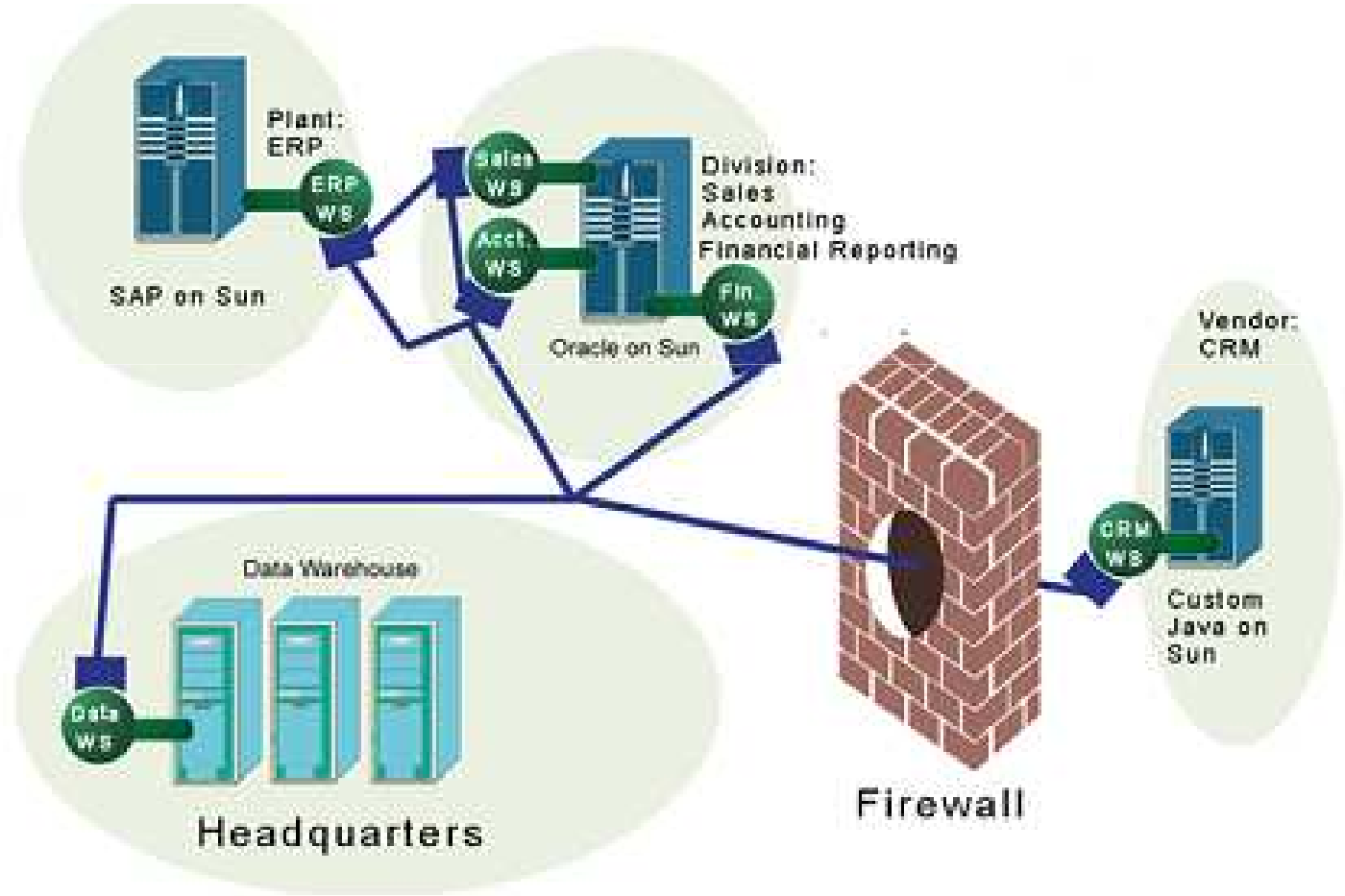
# SOA at Works


Service Repository
Searches in
Creates
Developer
Contains
Service Contract
Describes
Fulfills
Service
Invokes
Client (Application front-end or services)
Based on
Service Stub
Uses

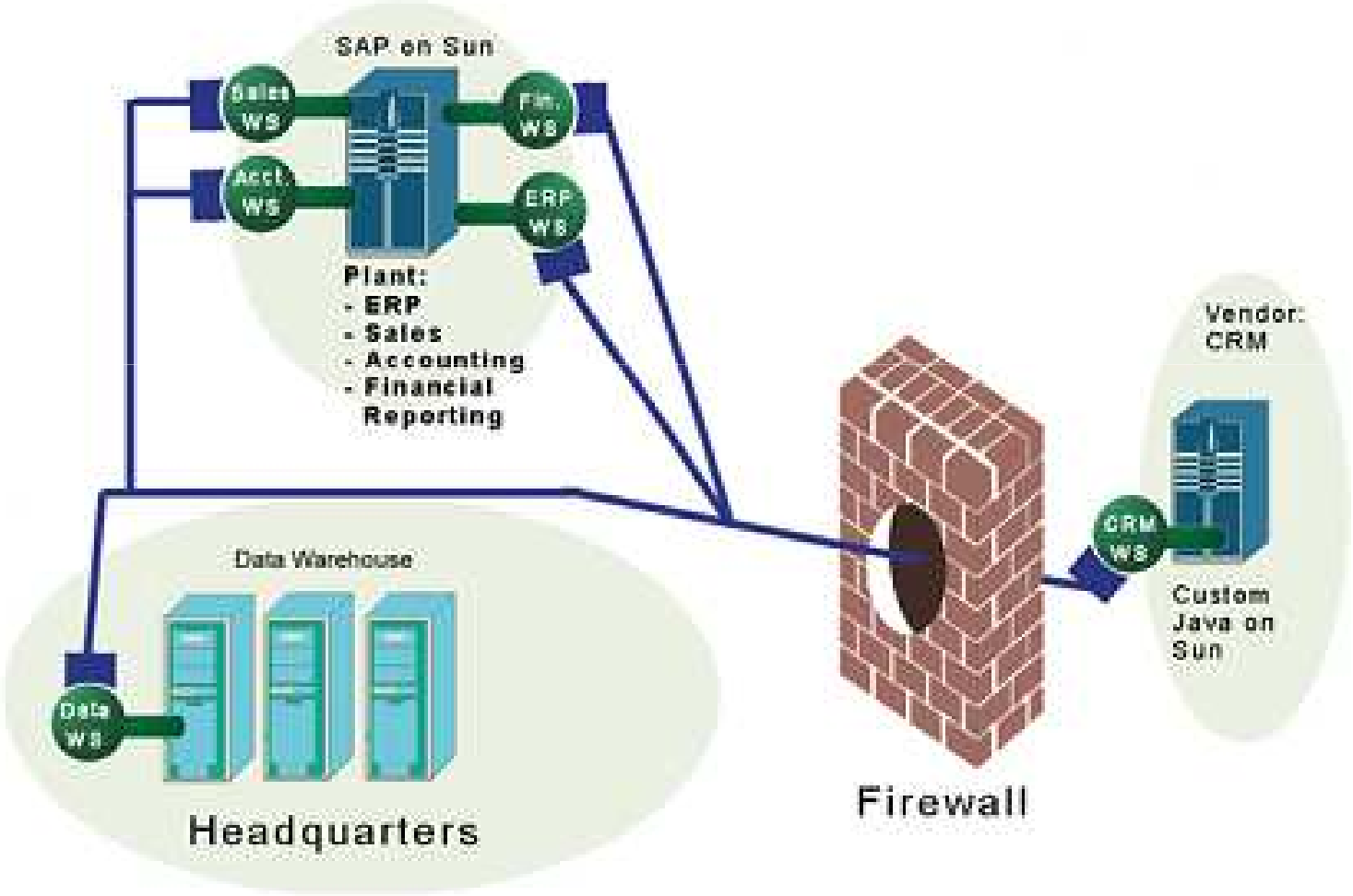
# Non-SOA (Integration)



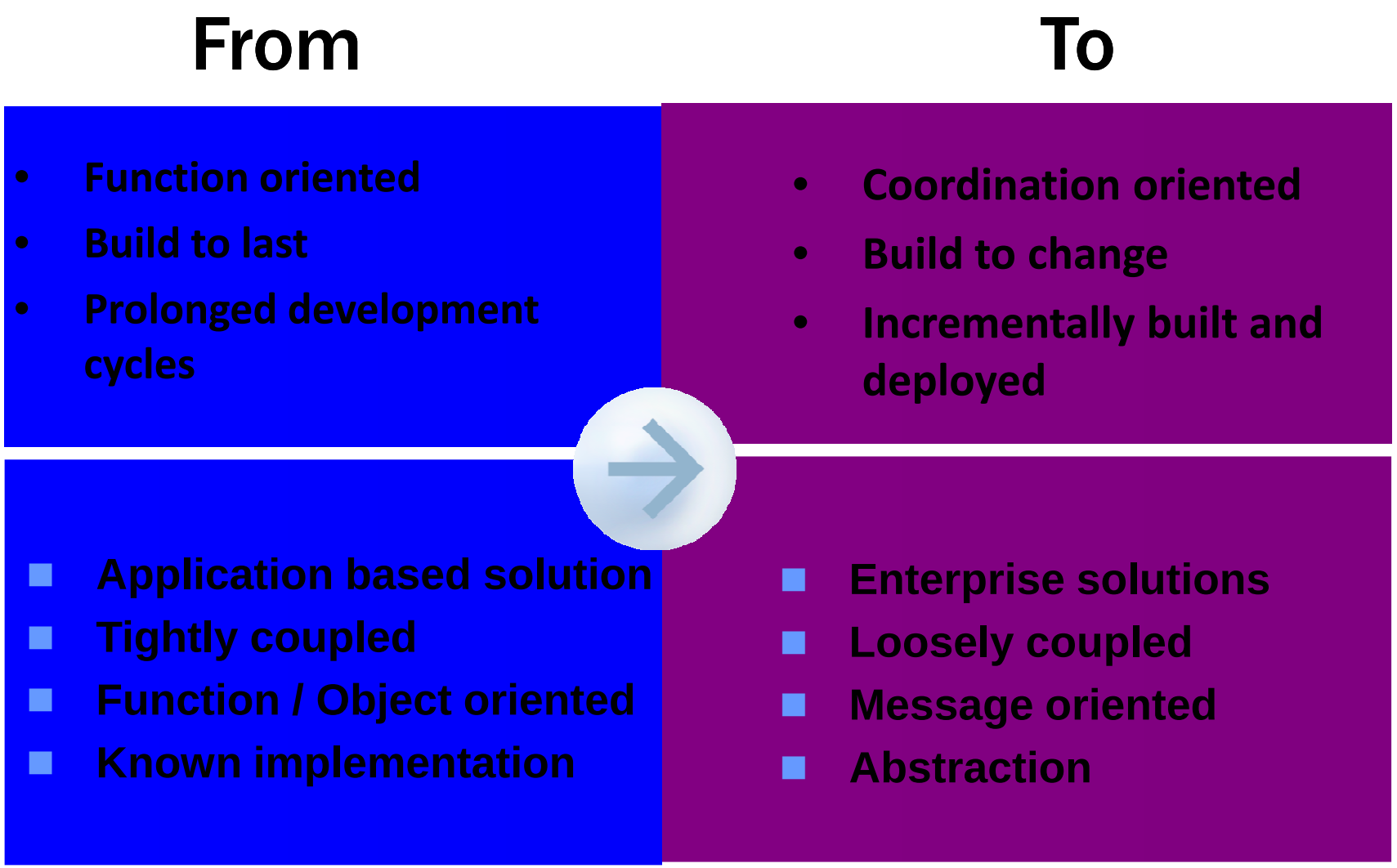
# SOA – Integration



# Changing SOA (Integration)

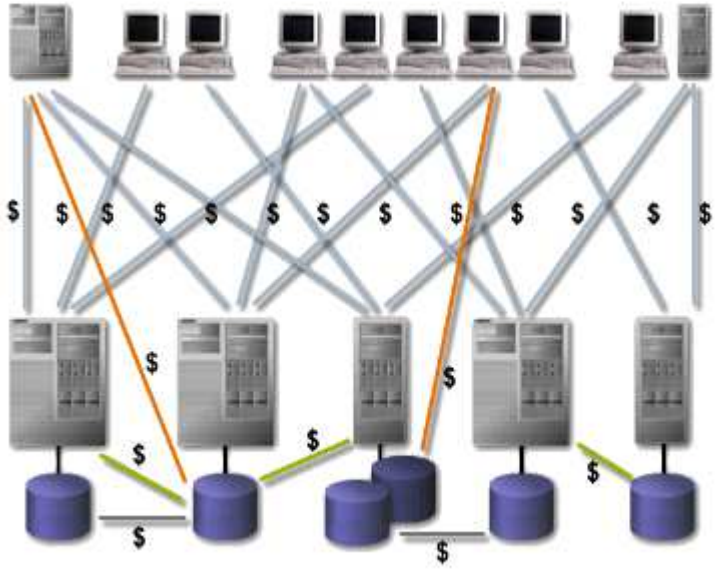


## Shift From Application To A Service-Oriented Architecture

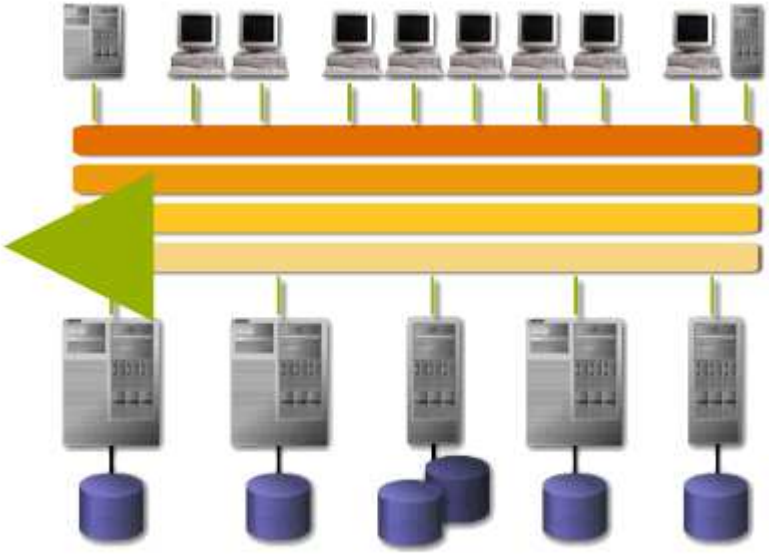


# Shift to SOA

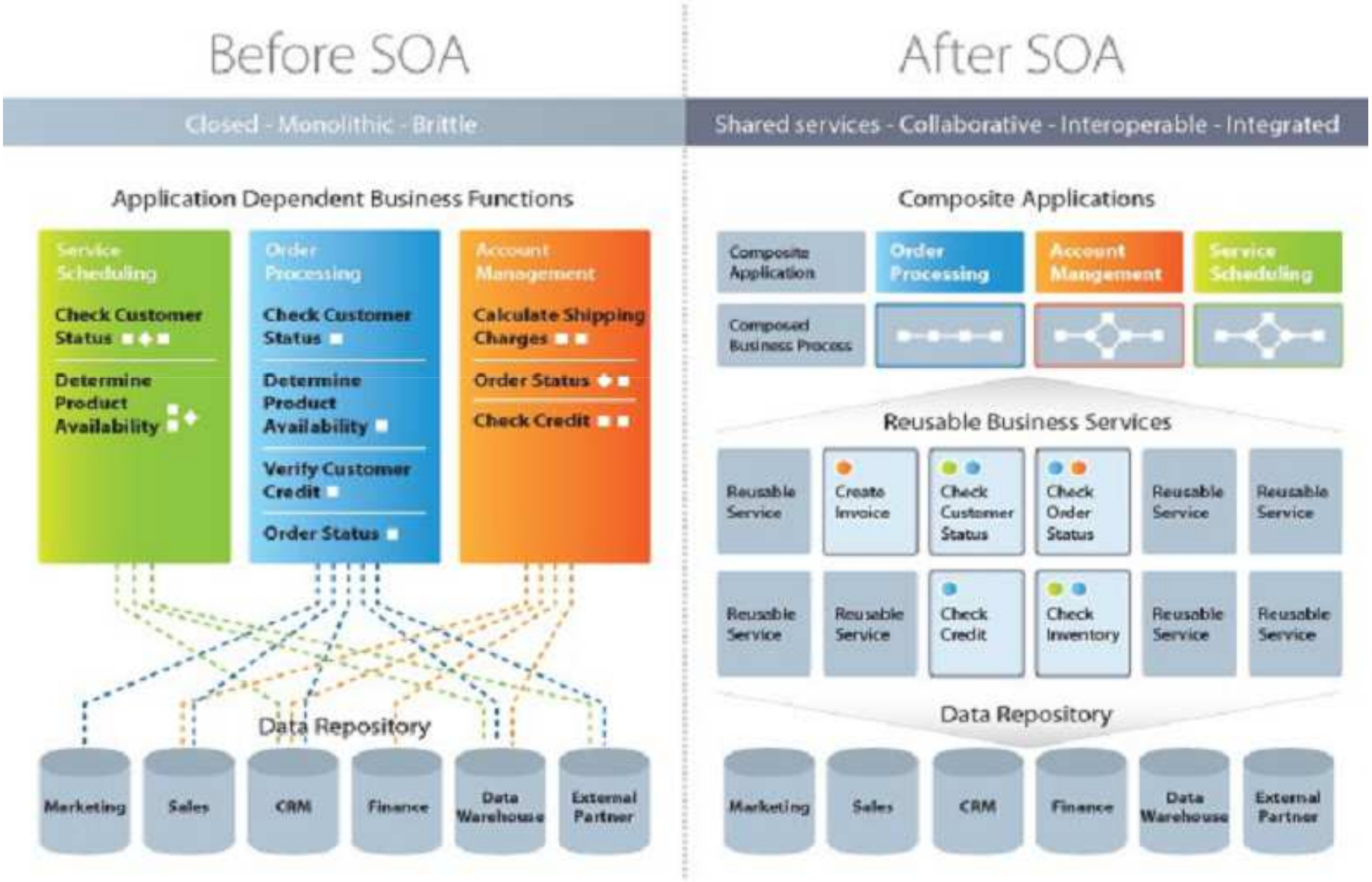
**Accidental**
**Rigid**
***Silo*-Oriented**



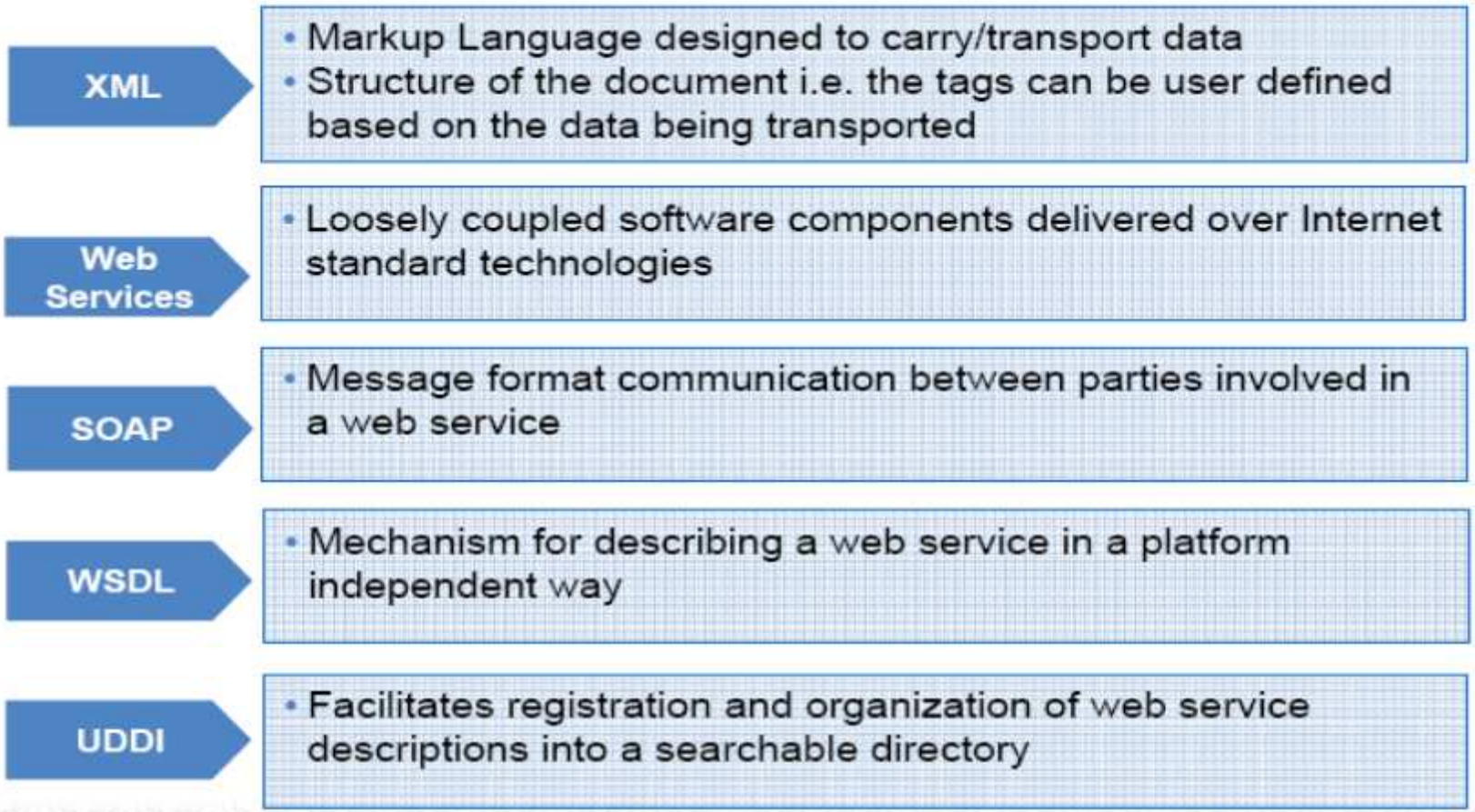
**Layered**
**Extensible**
***Service*-Oriented**



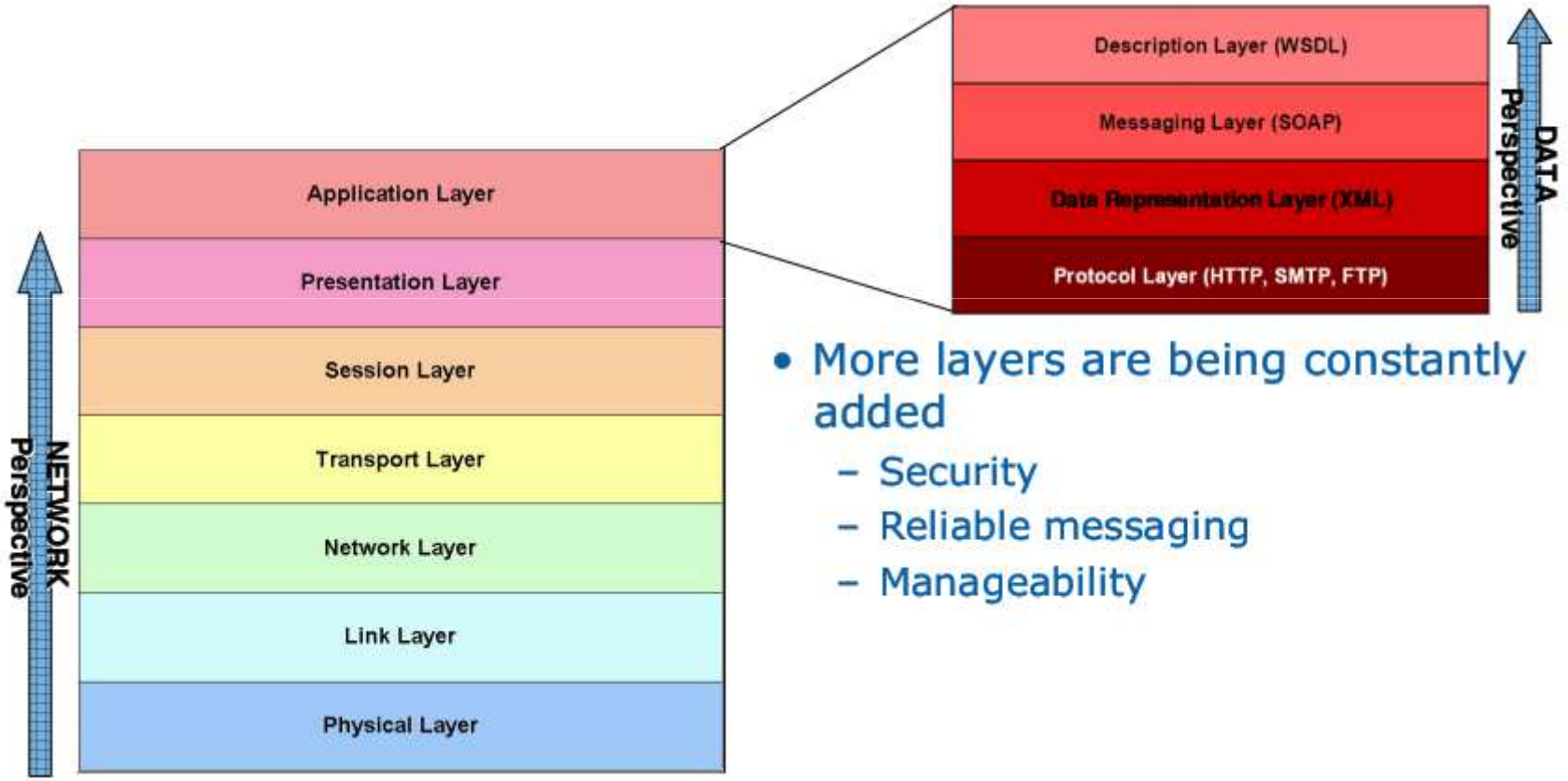
# Before and After SOA



# SOA implementation architecture



# SOA at application layer of OSI

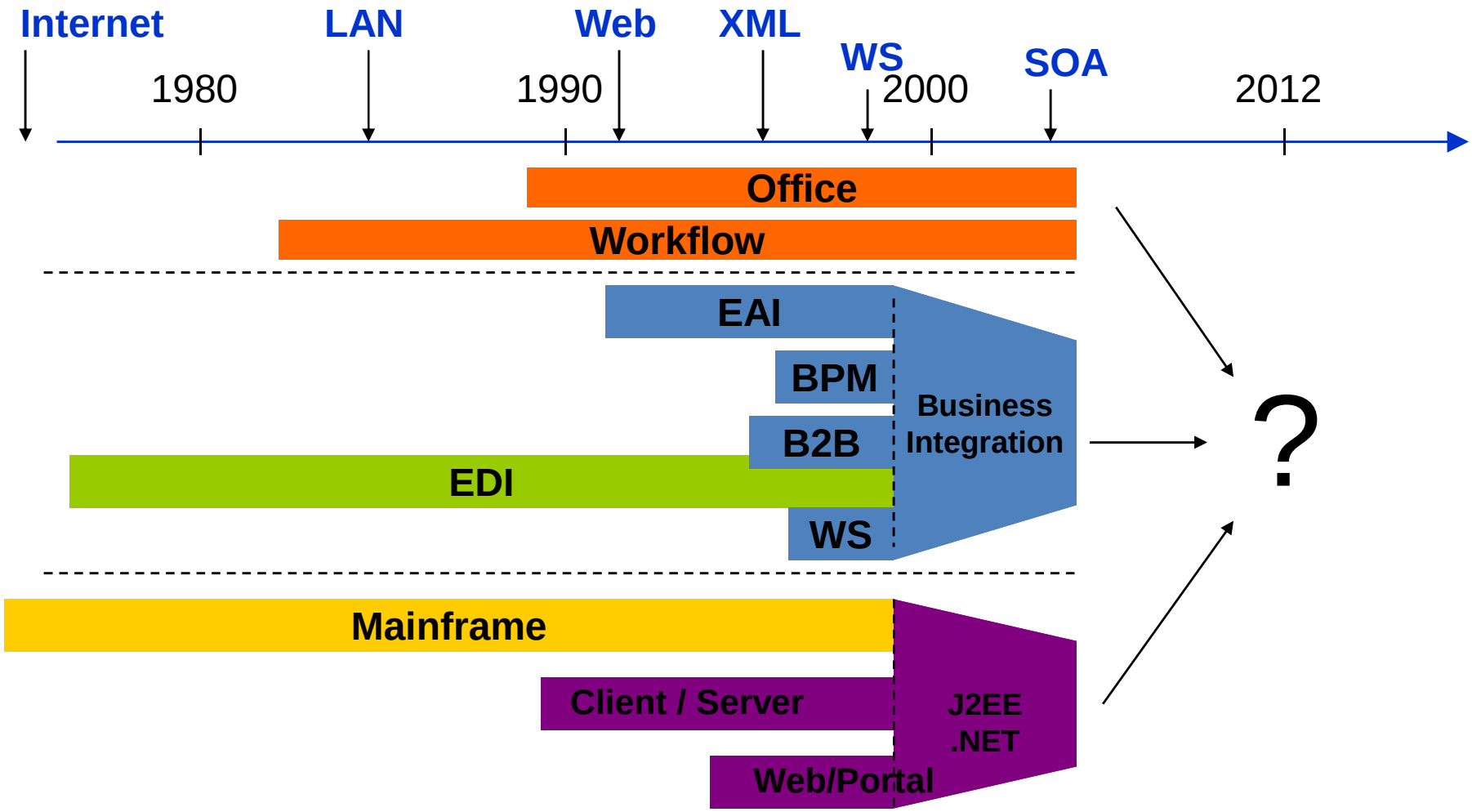



- More layers are being constantly added
  - Security
  - Reliable messaging
  - Manageability

# SOA challenges

- Trust
- Security challenges
- Performance
- Optimization
- Organizing the services
- Finding the right services and right interface

# SOA is a Trend



# NEXT

- Web Services (SOAP)