Aplikasi Enterprise-4

Enterprise Application Integration & SOA

Wiratmoko Yuwono

Outline

- EAI Overview
- SOA Overview

Enterprise Application Integration

- When systems are very **different** in nature and functionality, using conventional middleware to integrate them:
  - **more difficult**, in some cases simply **infeasible**.

- EAI can be seen as a step **forward** in the **evolution of middleware**, extending its capabilities to combine with application integration…

Conventional Middleware

- Conventional middleware was used to **create applications on top of a heterogeneous set of resources** (e.g. databases)
    - developers focused on **creating application logic**
        - To prepare information in **presentation layer**
    - this process was aided by various **standardization** efforts
        - e.g. JDBC, ODBC, ADO for databases,

EAI

- Enterprise application integration is how to **integrating services** hosted by a **heterogeneous** set of **middleware** platforms, often owned and operated by **different parts** of the **same** companies, or by **different** companies **altogether**

  - It includes:
    - Business Process Integration
    - Enterprise Information Integration

EAI

- EAI is the use of **software** and **computer systems architectural principles** to **integrate** a set of enterprise computer applications

- EAI is the "**sharing** of data and business processes among any connected application or data sources in the enterprise."

  - Using Service Oriented Architectures

EAI

- **Integrates** applications and enterprise data sources so that they can easily share business processes and data

- Integration is **done without significant changes** of applications and data sources

Challenges of EAI

- One large challenge of EAI is that the **various systems** to be linked together
    - different operating systems,
    - use different database solutions and
    - different computer languages, and
    - legacy systems that are no longer supported by the vendor who originally created them.

EAI Level

- **Data Level Integration**
  - Occurs at **database** level within an enterprise.
  - By **migrating** data from on source to another.
  - Move data between applications but it needs application logic **unchanged**.
  - Ex: JDBC, ODBC

- **API Level Integration**
  - Applications are bound together using their **APIs**.
  - Depends on **how feature-rich** an interface is.
  - This type of EAI is mostly applicable for **ERP** applications like SAP, Oracle, PeopleSoft, etc

EAI Level

- **Method Level Integration**
  - Achieved by **sharing the business logic** within the enterprise.
  - Requires **major changes** in source code (in method/function)
  - **Distributed objects** can be used.
    - Ex: CORBA, RMI, .NET Remoting

- **User-Interface Level Integration**
  - Integrate applications using their **user-interface** as a common point.
    - Ex: **mainframe** applications that do not have database or business process-level access may be accessed through the user interface of the application
  - *Not a preferred one*, but in some cases the **only way** of approaching the task.
  - Mostly used for legacy applications / external systems.

EAI Architecture

- EAI platforms are based on two basic principles:
  - **Adapters**: programs that **map** heterogeneous data formats, interfaces, and protocols into a common model and format.
    - The purpose of adapters is to **hide heterogeneity**;
    - **a different adapter** is needed for each type of application that needs to be integrated.
    - Using **adapter pattern**
  - **Message Broker**: the message broker **facilitates** the **interaction** among adapters and a program that contains the **integration logic**
    - Using messages exchange protocol

EAI's purposes

- **Data (information) Integration**: ensuring that information in multiple systems is kept *consistent.*

- **Process Integration**: *linking* business processes across applications.

- **Vendor independence**: extracting *business policies* or rules from applications and implementing them in the EAI system, so that even if one of the business applications is replaced with a different vendor's application, the business rules *do not have to be re-implemented.*

- **Common Front End**: an EAI system could *front-end* a cluster of applications, providing a *single consistent* access interface to these applications and shielding users from having to learn to interact with different software packages.

Examples of **Applicable Standards** to support EAI

- **Earlier middleware:** RPC, Transaction Monitor
- **Object Broker:** CORBA, RMI
- **Communication**: Messaging / JMS
- **Connectivity**: Web Services-> XML, SOAP, WSDL, UDDI
- **Transformation** Standard: XSLT, XQuery
- **Portability**: Java, HTTP, XML, SOAP on Windows, Unix, mainframes
- **Security**: SSL, certificates, signatures, dan WS-* security standard

EAI standards implementation technologies

- **Rosettanet**
  - A non profit organization
  - Group included American Express, Microsoft, Netscape, IBM
  - Implements Standards for **supply chain transactions**
- **UCCNET**
  - Non profit subsidiary of the Uniform Code Council
  - Internet based **Supply Chain Management**
 - Supported by Wal-Mart, Shaw's, Home Depot
- **BPEL4WS**
  - **Web service** orchestration language
  - Standard process integration language
  - Successor to IBM's WSDL and Microsoft's XLANG

EAI Vendors

- **IBM - WebSphere Application Server**
  - Provides support for deploying an application, managing resource requirements for applications

- **SAP NetWeaver™**
  - Unifies integration technologies into a single platform, pre-integrated with business applications

- **BEA System -WebLogic Integration™**
  - Weblogic Integration™ is process integration module of BEA's WebLogic Enterprise Platform™

- **TIBCO - Business Works™**

- **Microsoft BizTalk**
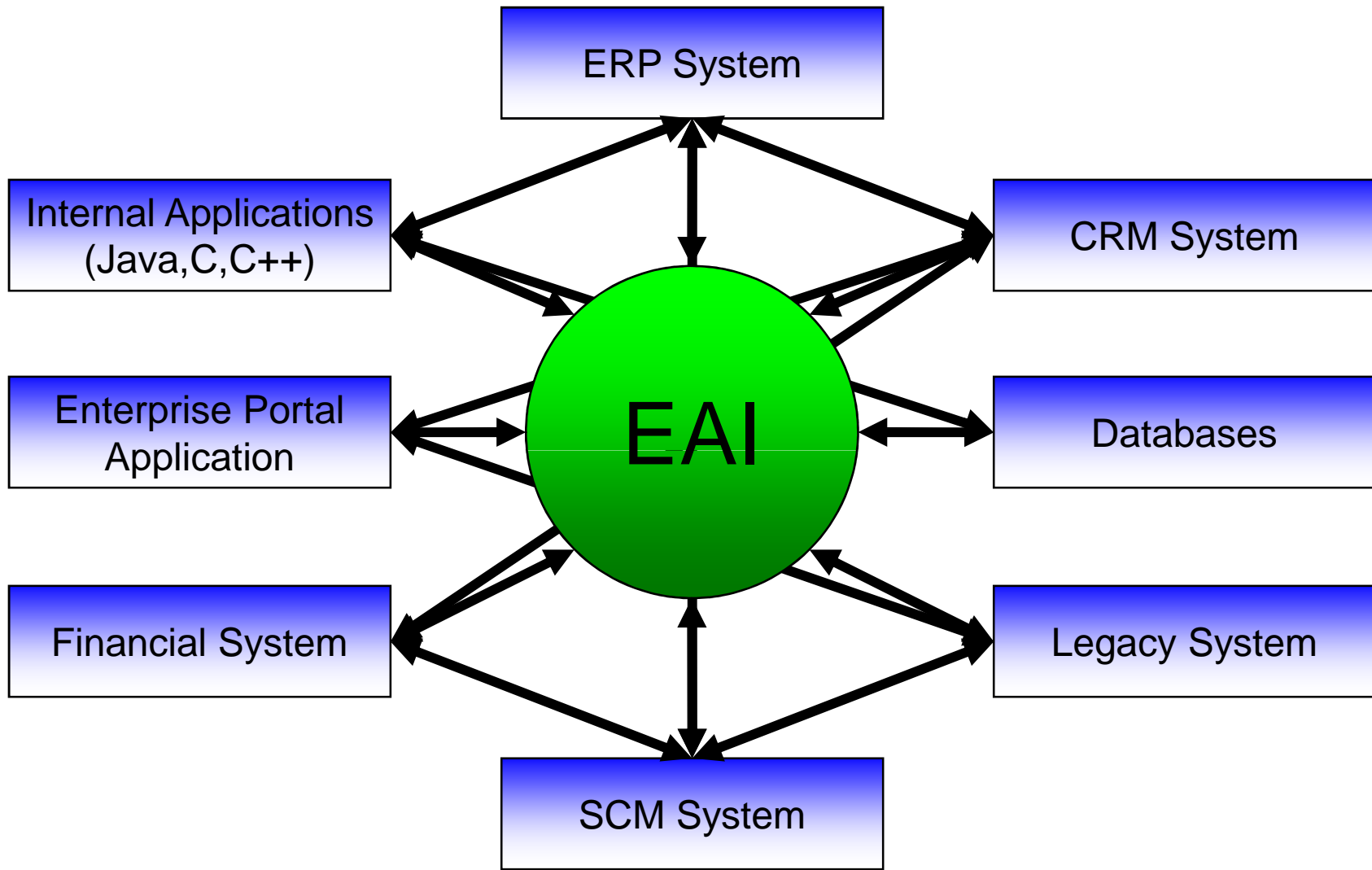
EAI diterapkan misal pada Supply Chain

- The process of fulfilling a customer order involving many departments, external suppliers, warehouses, manufacturing sites, etc.
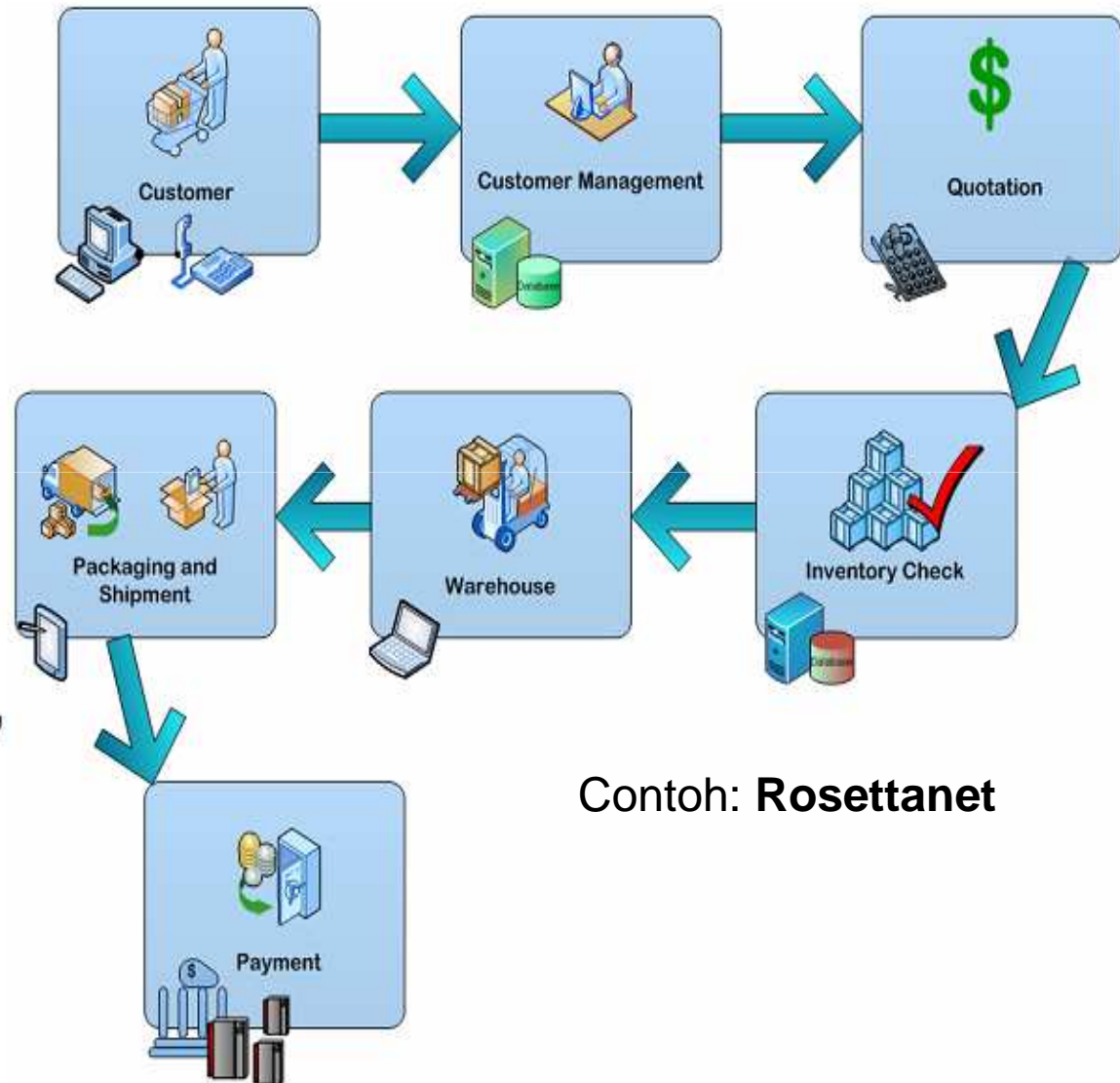

Contoh: **Rosettanet**

Example scenario

Business Scenario

- Capital Equipment Manufacturing Company
- Servicing Hospitals with 9 Warehouses ($30 Million Inventory) owned and operated in North America
- Customer Satisfaction is critical. Need to ship Services parts in time quickly
- Expensive to maintain 9 Warehouses

Proposed Solution

- All 9 warehouses being outsourced to major Shipping Company – FEDEX-KINKOS
- Need to Integrate with FedEx's System for this outsourcing agreement
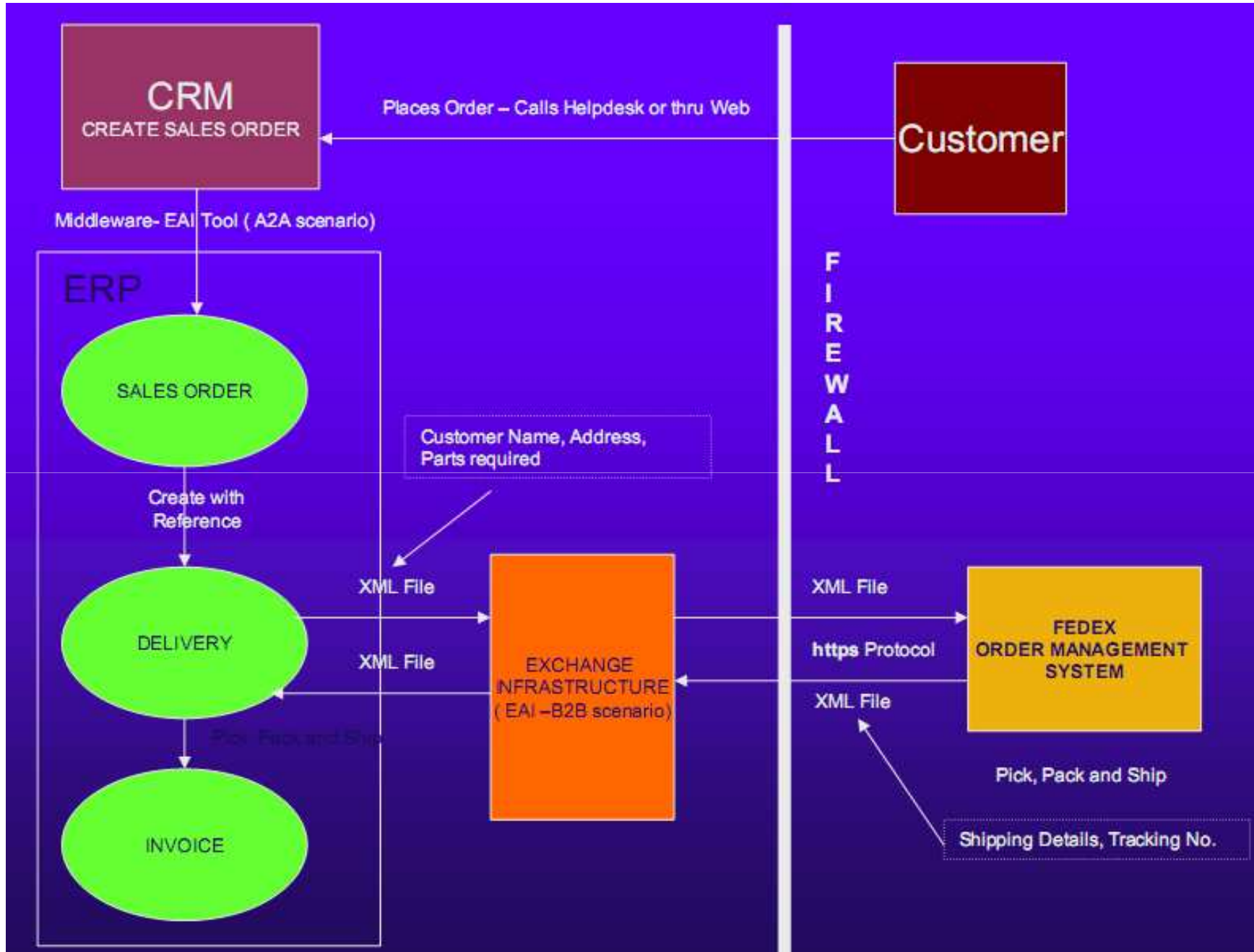
Service Oriented Architecture

- SOA adalah sebuah konsep Software Architecture yang mendefinisikan penggunaan **layanan** untuk mendukung kebutuhan pengguna software.
  - Salah satu bentuk model implementasi EAI
- A service-oriented architecture is a framework for **integrating business processes** and supporting IT infrastructure as secure, standardized components—**services**—that can be **reused** and **combined** to address changing business priorities

# Apakah 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)

Architecture Analogy
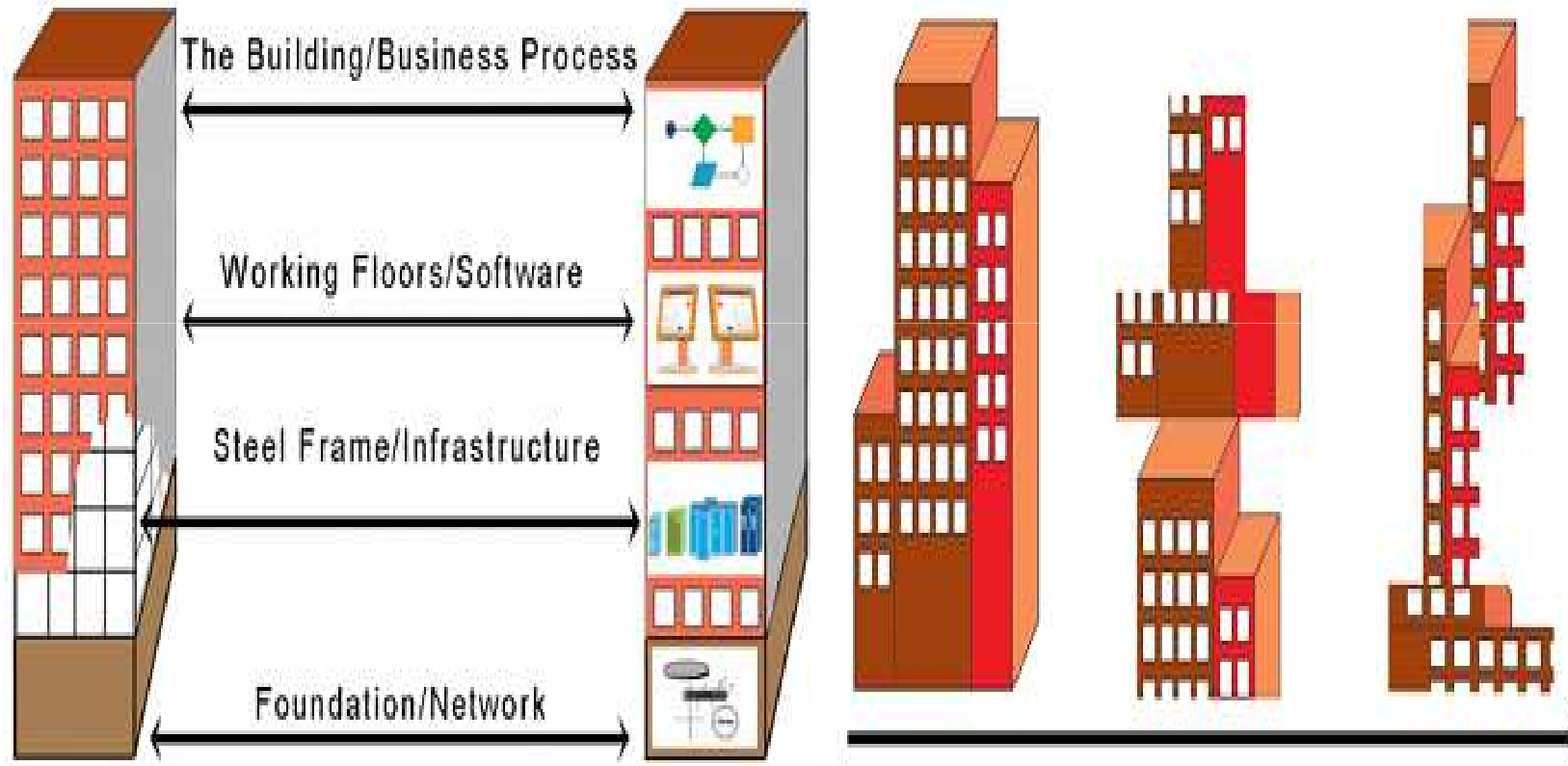
SOA's architecture
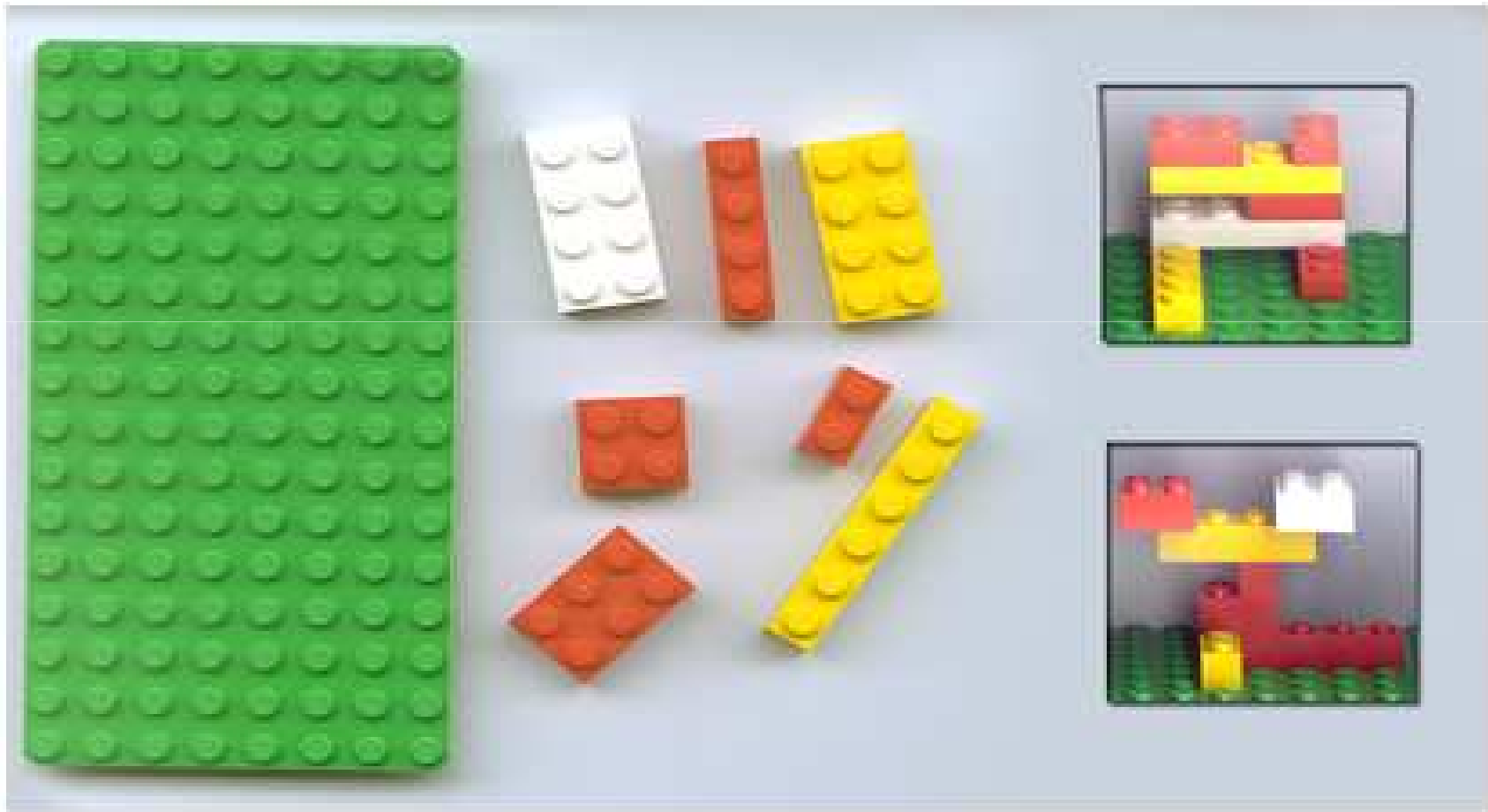
SOA dan Aplikasinya

- Ada dua arah pengembangan:
  - 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 (mis: www.pdfonline.com)

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**
  - Implementation **transparency** (common structure, neutral service description)
  - Location **transparency** (no hard binding)

SOA dan Integrasi Aplikasi/Sistem

- **SOA 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

Komponen SOA

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

Sifat SOA

- SOA bersifat **behind the scence**,
  - SOA tidak terlihat secara langsung oleh client, SOA dihadapkan pada client melalui client UI.

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

Characteristics of SOA

- Services have **platform independent**, self describing interfaces (XML)
- Messages are **formally defined**
- Services **can be discovered**
- Services have **quality of service characteristics** defined in policies
- Services can be **provided on any platform**
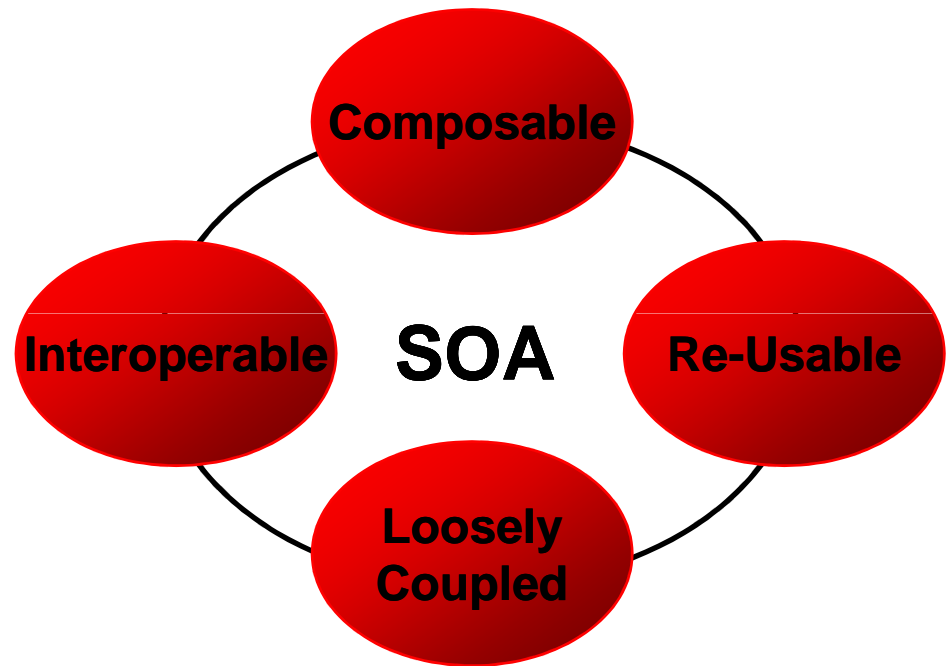
Beberapa Istilah dalam SOA

- **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.
- **Stateless**: tidak menyimpan kondisi apapun.

Beberapa Istilah SOA

- **Provider**: host / application logic yang menyediakan service dan meresponse sebuah request.

- **Consumer**: host yang membutuhkan response dari service yang dihasilkan oleh provider.

- **Binding**: Hubungan antara provider dan consumer bersifat dinamis dan hubungan itu dibuat pada saat runtime berdasarkan mekanisme binding.

Benefits of SOA

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

Benefits of 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
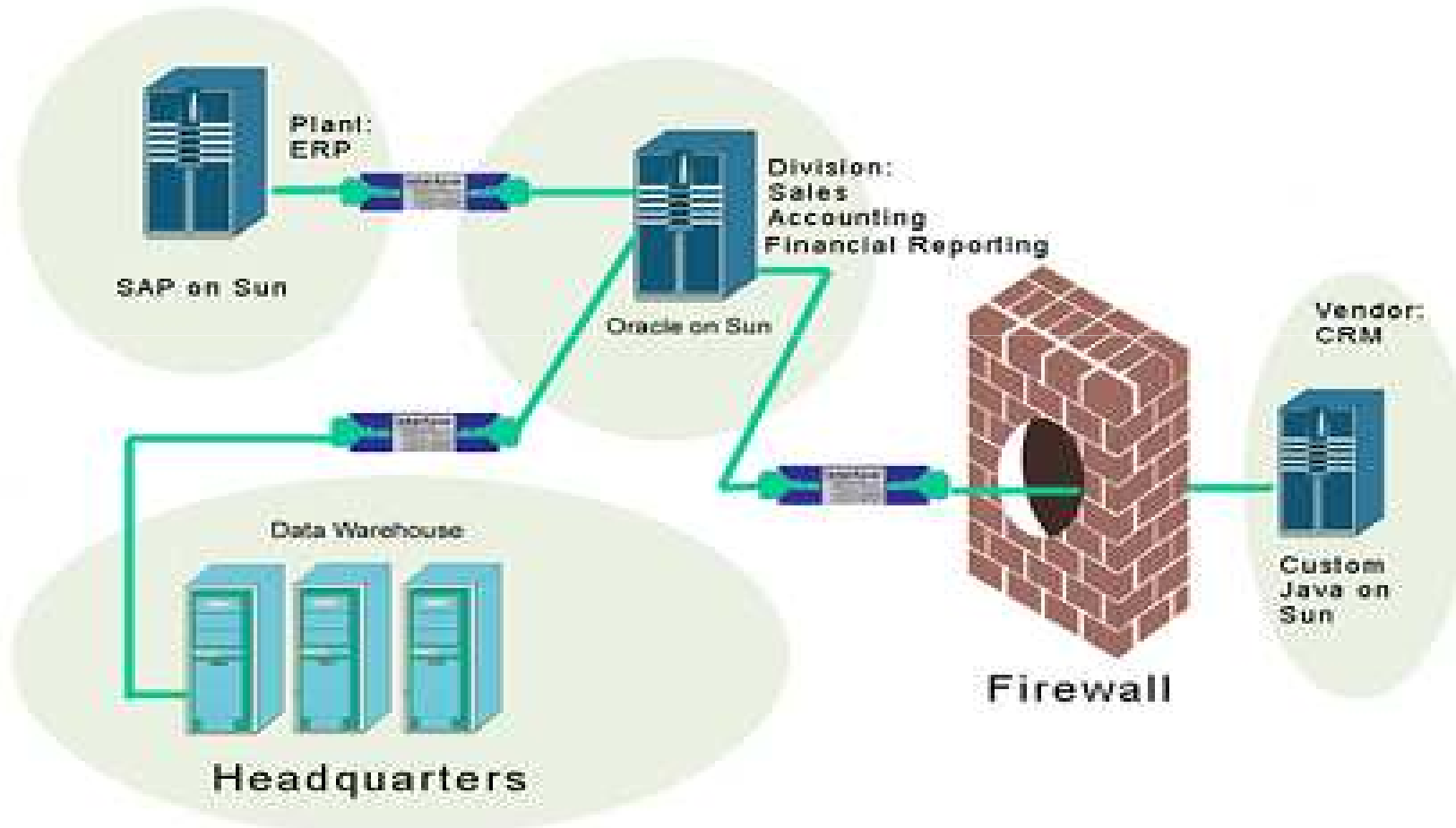- **Better scalability**
  - there can be **multiple** instances of the service running on different servers. This increases scalability
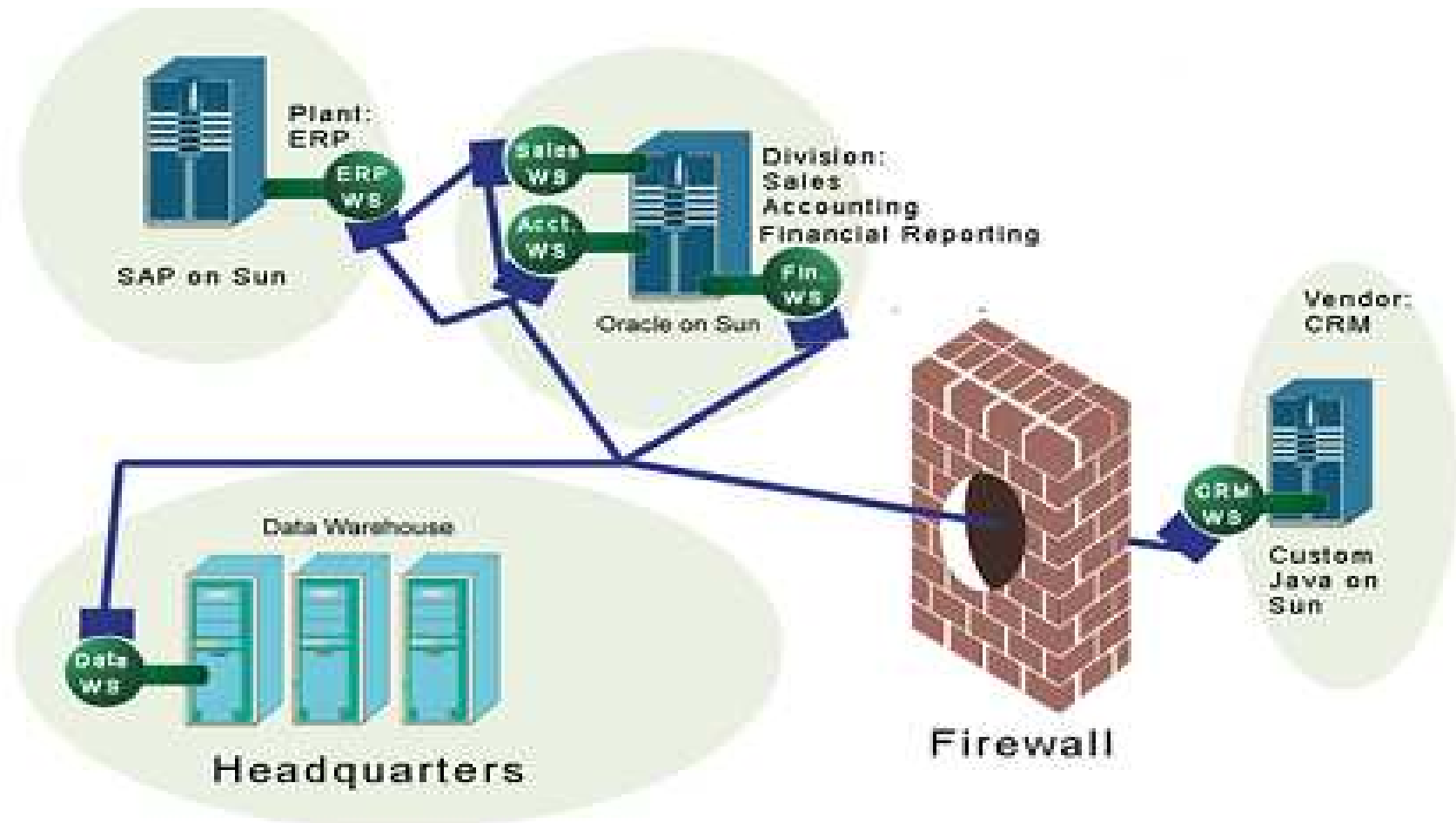- **Higher availability**
  - Since the location of a service does not matter and you can have multiple instances of a service, it is possible to ensure high availability
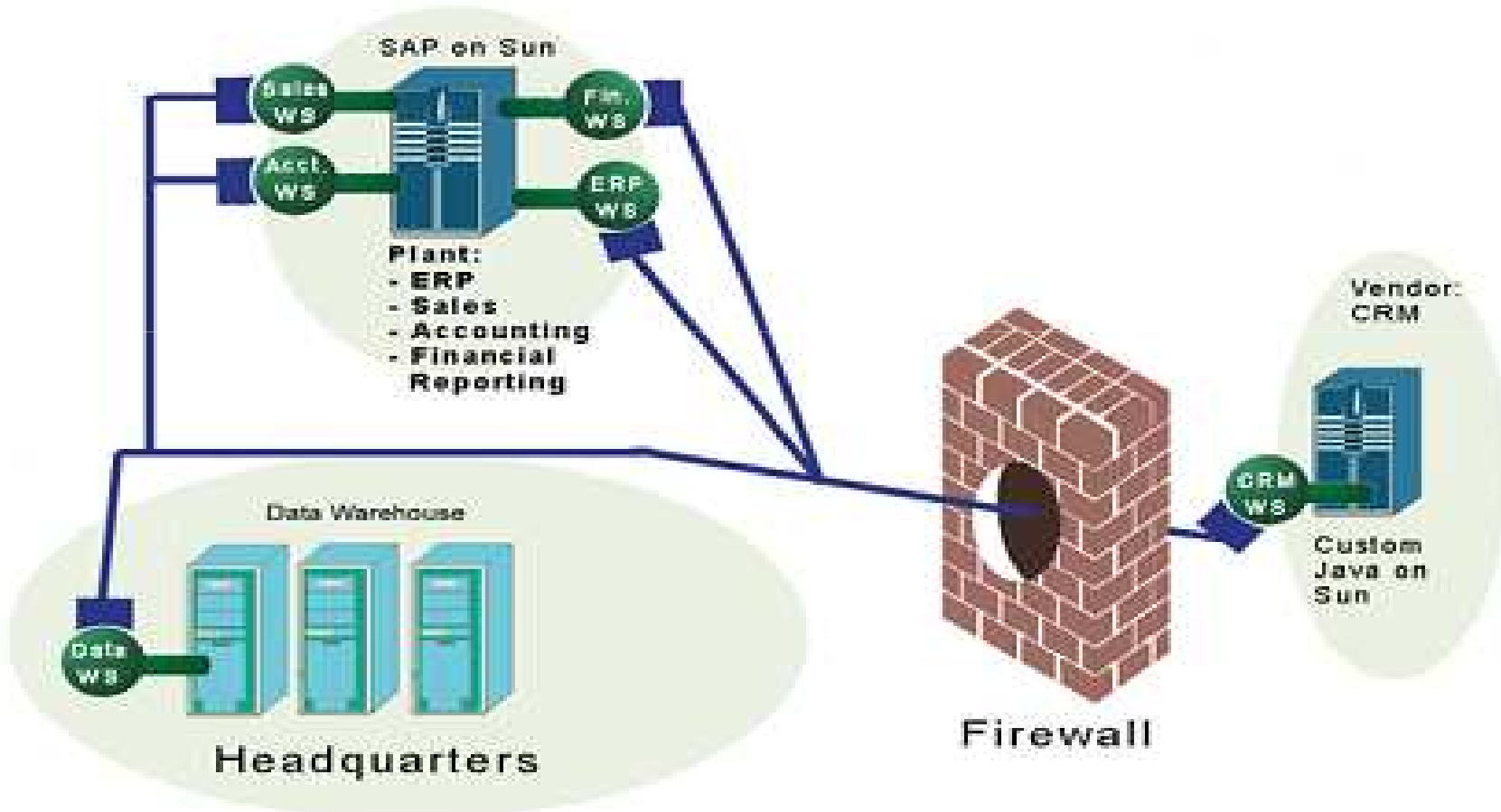
Non-SOA

SOA – Web Services

Changing SOA

From Components to (Web) Services

- Requires a client library

- Extendable
- Stateless

- Fast

- Loose coupling via
  - Message exchanges

- Composable
- Stateless & Context independent

- Some overhead

Shift From Application To A Service-Oriented Architecture

From

To

- Function oriented
- Build to last
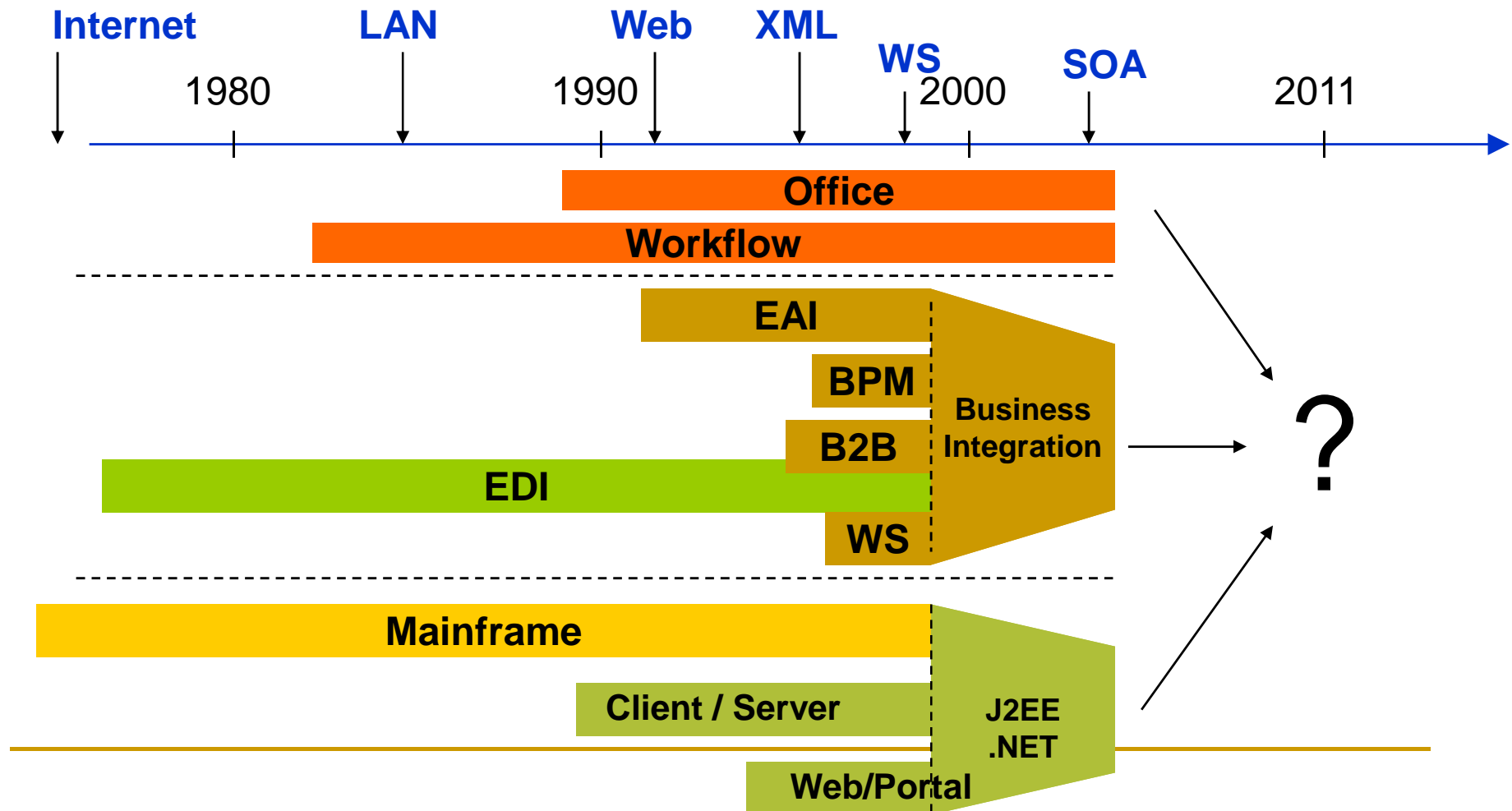- Prolonged development cycles

- Coordination oriented
- Build to change
- Incrementally built and deployed

- Application based solution
- Tightly coupled
- Function / Object oriented
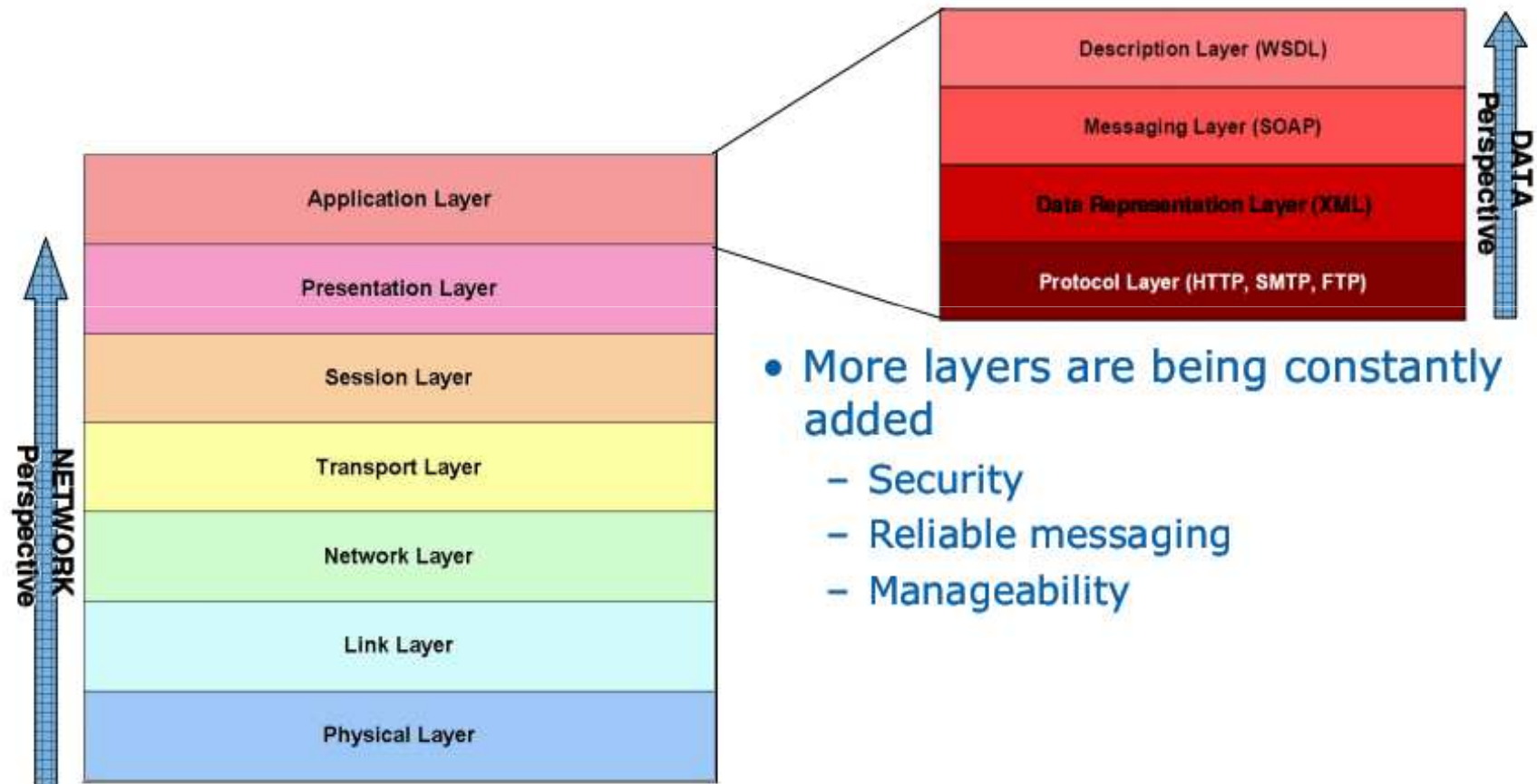- Known implementation

- Enterprise solutions
- Loosely coupled
- Message oriented
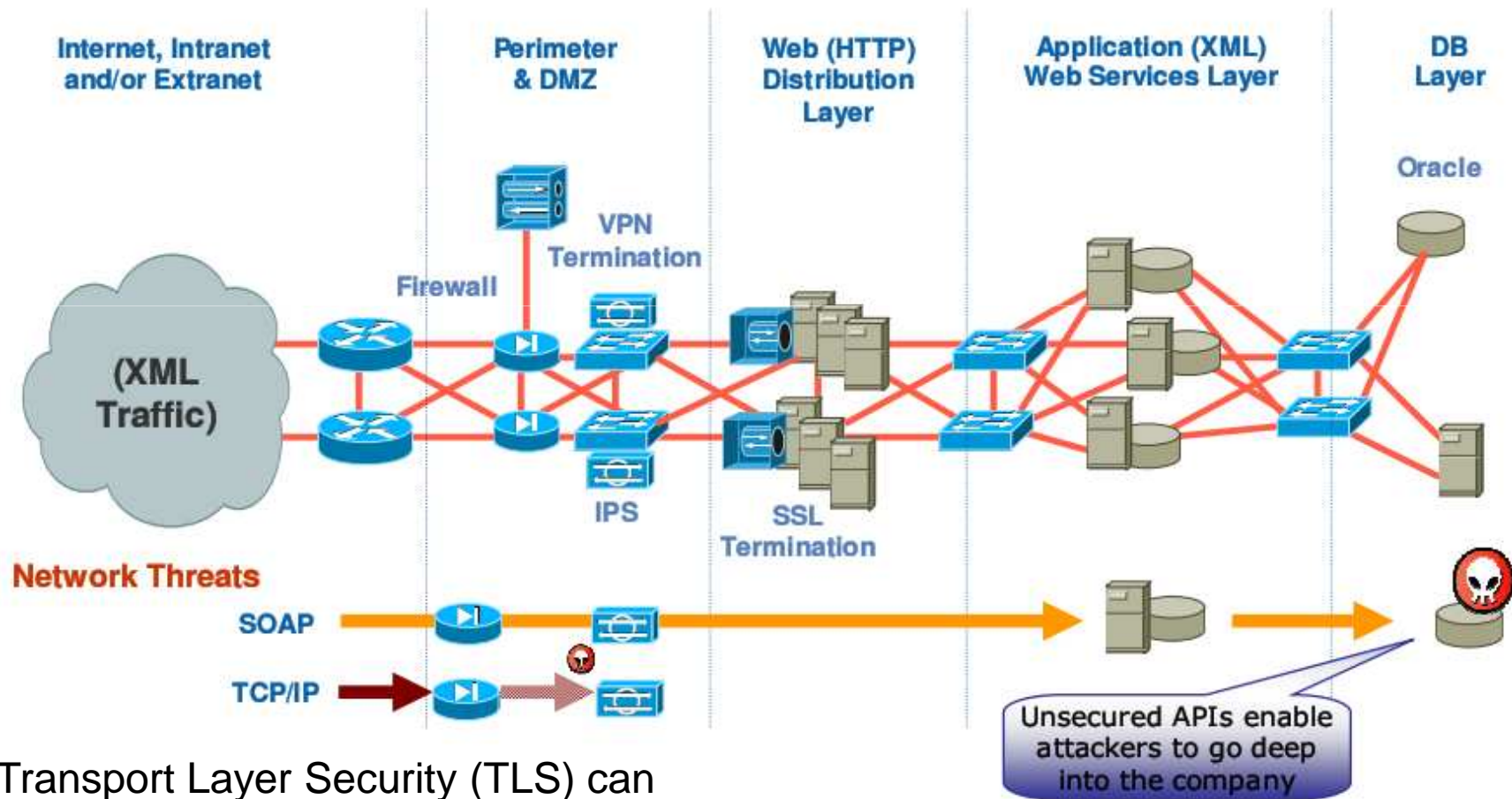- Abstraction

Perkembangan SOA

SOA at application layer of OSI

Anatomy of the SOA Security challenge

WS-Security is to SOA, SSL is to HTTP

Internet, Intranet and/or Extranet — (XML Traffic)

Perimeter & DMZ — Firewall, VPN Termination, IPS

Web (HTTP) Distribution Layer — SSL Termination

Application (XML) Web Services Layer

DB Layer — Oracle

Network Threats

SOAP

TCP/IP

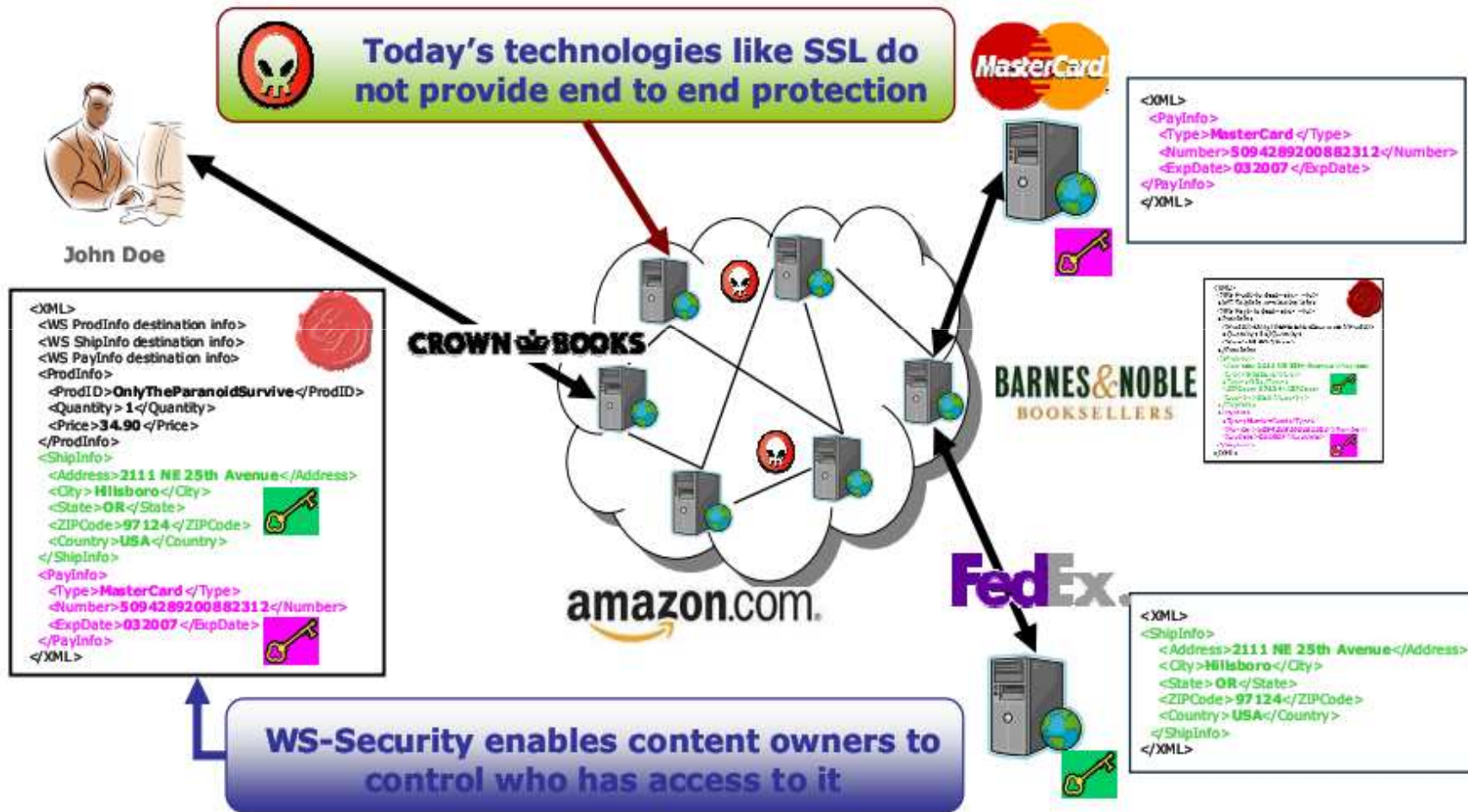Unsecured APIs enable attackers to go deep into the company

Transport Layer Security (TLS) can provide point-to-point security, but not end-to-end, problem with proxies

WS-Security

- **Enhancement to SOAP**
- **How to sign SOAP messages to assure integrity.**
  - Signed messages provide also non-repudiation.
- **How to encrypt SOAP messages to assure confidentiality.**
- **How to attach security tokens.**
- **Uses**
  - XML Encryption
  - XML Digital Signatures
  - SSL/TLS

SOAP Envelope
SOAP Body

SOAP Envelope
Security Feeder
Security Token
Signature
SOAP Body
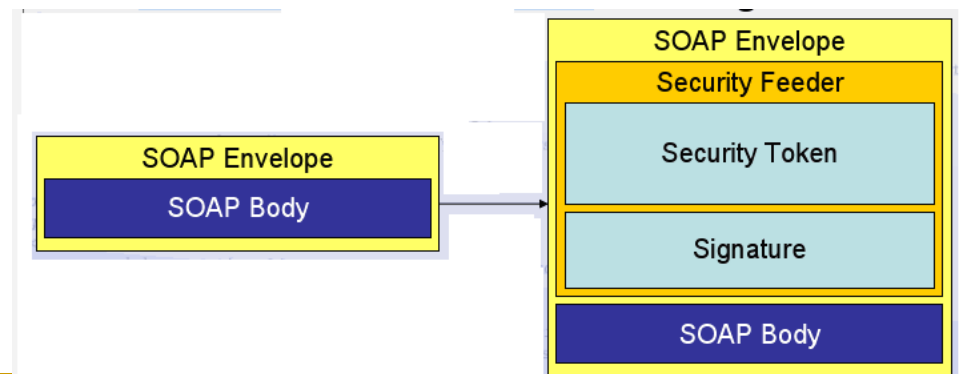
Implementasi: **WS-Security**

SOA in practice