Middleware Web Technologies - Overview

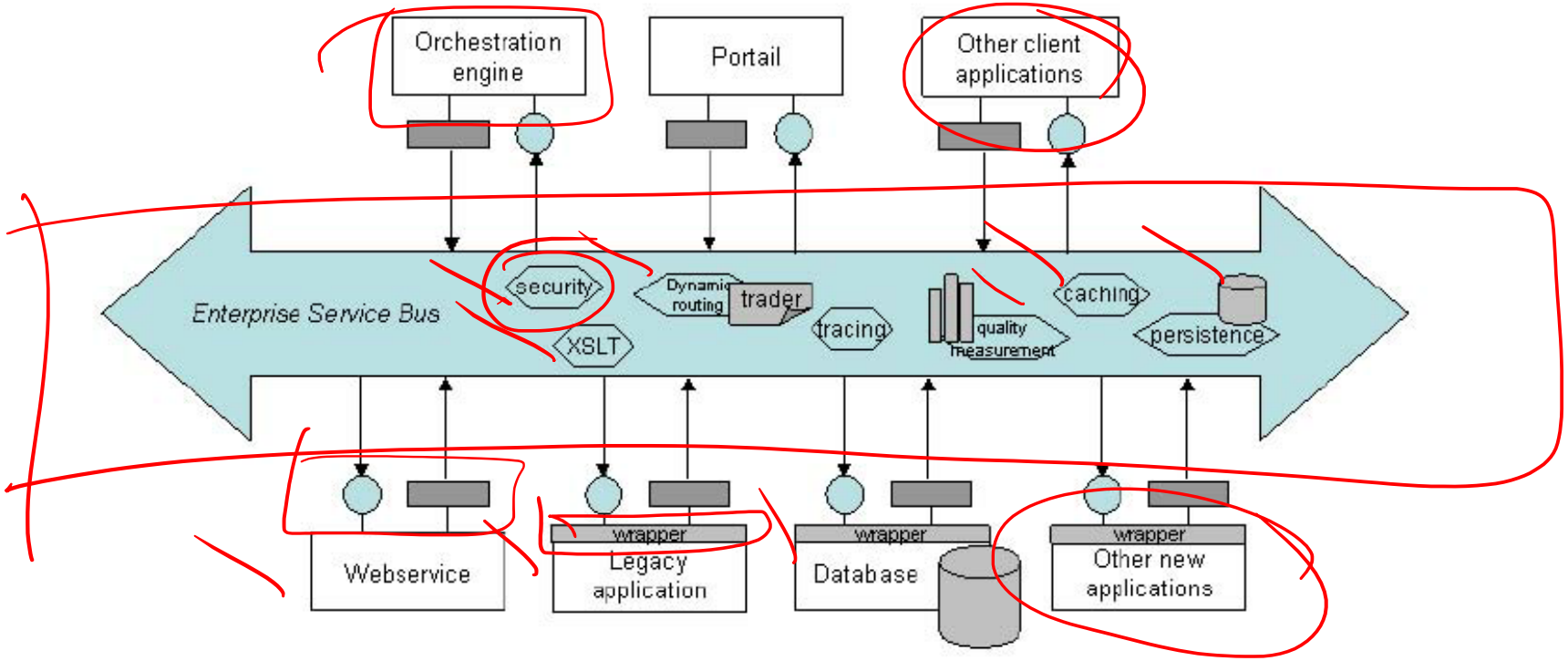
Service Oriented Architecture - Overview

- A Service Oriented Architecture (SOA) is a form of distributed systems architecture that is typically characterized by the following properties
  - Logical view
  - Message Orientation
  - Description Orientation
  - Granularity
  - Network Orientation
  - Platform Neutrality
- Implementation of SOA
  - Arbitrary Web services (SOAP/XML)
  - RESTful Web services
  - Apache Thrift

Middleware Web Technologies - Overview

SOA vs Enterprise Service Bus (ESB)

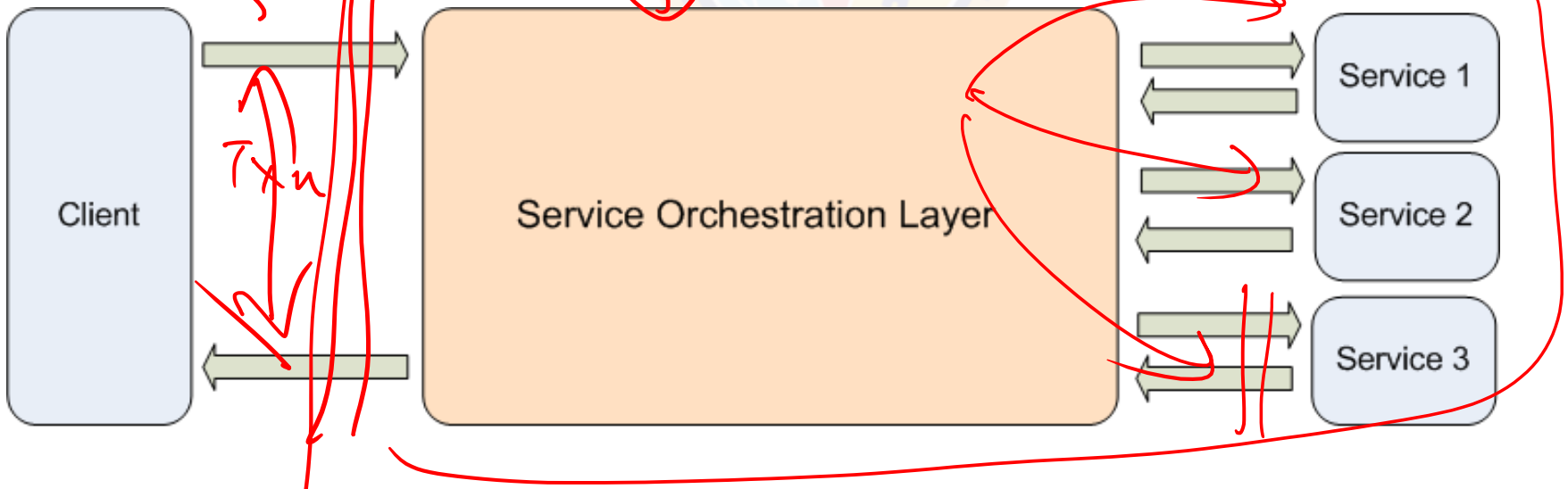
- ESB - Implements a communication system between mutually interacting software applications (including SOA components)
- Can be used to integrate SOA apps with non-SOA apps
- Acts as service orchestrator, API gateway, security manager etc.



Middleware Web Technologies - Overview

Service Orchestration

- Process of integrating two or more applications and/or services together to automate a process, or synchronize data in real-time
- Service orchestration is the combination of service interactions to create higher-level business services
- Service Orchestration involves connecting various applications, web services and legacy components in a business flow
- Examples of Orchestrators: Business Process Modeling engines, ESB etc.



Middleware Web Technologies - Overview

SOA Maturity Levels

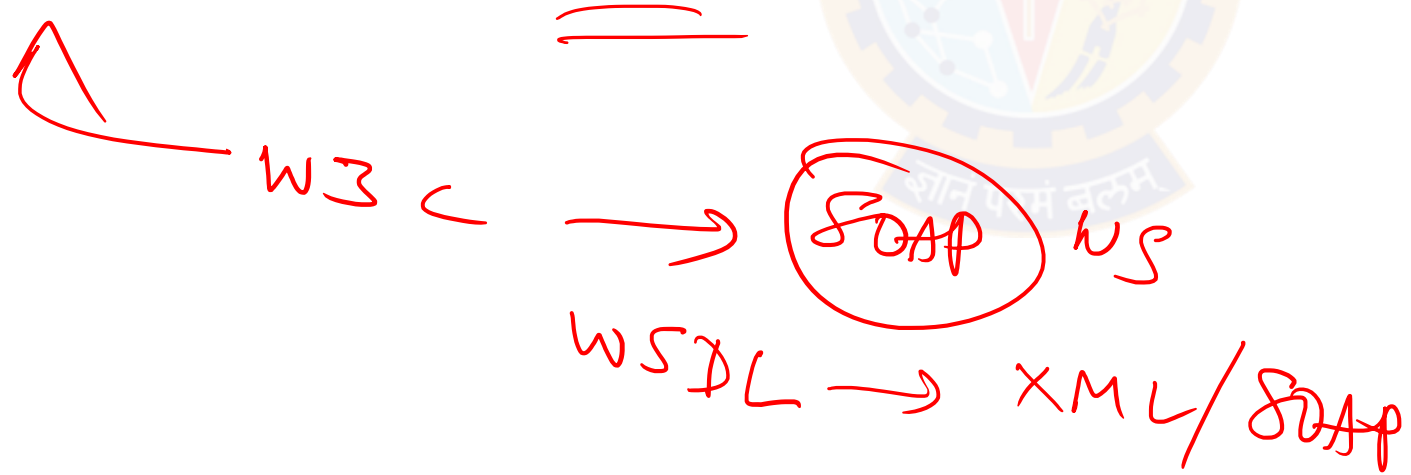
- Analogous to CMM definition for process maturity
- Levels
  - Initial
  - Managed
  - Defined
  - Quantitatively Managed
  - Optimized

Maturity level	SOA view	Benefits and metrics	Business involvement	Methodology	Service sourcing	Governance
Initial	Fine-grained software components	Promise of reuse; no metrics	Hidden from business	Minor adaptation of current software methods	Largely derived from existing application programming interfaces	Basic service definition policies at project level
Managed	Emerged software architecture	Standardization of data and resources	Services exposed as part of project cost	Project-level service definition methods	In-house development of new fine-grained software components	Service policies managed by registry monitors
Defined	Business process support	Business process redesign	Services part of requirement capture	Business service definition methods	External sourcing possible by defined function	Full set of service policies and metrics in place
Quantitatively managed	Enterprise service architecture	Agility and flexibility	Organizational "service thinking"	Intra- and inter-organizational service definition	Enterprise service bus to coordinate services	Business and IT governance metrics aligned
Optimized	Adaptive architecture	Autonomic systems	Value-stream "service thinking"	Value-chain service optimization	Full integration up and down the stack	Policy feedback used to adjust delivery

Middleware Web Technologies - Overview

Web Service – Initial Definition

- "A web service is a software system designed to support interoperable machine-to-machine interaction over a network." – W3C glossary
- "It [Web Service] has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other web-related standards." – W3C basic definition, 2002



Middleware Web Technologies - Overview

Web Service Redefined – REST + Non-REST

"An even more constrained architectural style for reliable Web applications known as Representation State Transfer (REST) has been proposed by Roy Fielding and has inspired both the W3C Technical Architecture Group's architecture document [Web Arch] and many who see it as a model for how to build Web services [Fielding]. The REST Web is the subset of the WWW (based on HTTP) in which agents provide uniform interface semantics -- essentially create, retrieve, update and delete -- rather than arbitrary or application-specific interfaces, and manipulate resources only by the exchange of representations. Furthermore, the REST interactions are "stateless" in the sense that the meaning of a message does not depend on the state of the conversation.

We can identify two major classes of Web services:

- REST-compliant Web services, in which the primary purpose of the service is to manipulate XML representations of Web resources using a uniform set of "stateless" operations; and
- arbitrary Web services, in which the service may expose an arbitrary set of operations."

-W3C extended definition, 2004

Middleware Web Technologies - Overview

Non-REST Vs RESTful Web services

Non-REST Web services (SOAP WS)	RESTful Web services
Exposes services as individual operations, each to achieve a specific business objective.	Representational State Transfer is an architectural style for defining interoperability in terms of resources or concepts
Introduced in W3C draft for web services in 2002	Introduced in W3C extended revision in 2004
Uses Simple Object Oriented Protocol (SOAP), a variant of XML for data transfer	REST supports SOAP/XML/JSON or any other data protocol supported by HTTP
Service contract definition is specified via Web service description language (WSDL)	Service contract is defined via resource or concept endpoints and HTTP methods (POST/ GET/ PUT/ DELETE) defined on them
JAX-WS is the binding protocol used	**JAX-RS** is binding protocol used
Imposes a rigid structure for service definition and request/response	Does not impose any specific structure – any format supported by HTTP is accessible
Requires more footprint and bandwidth, resulting in performance issues at large scale	Light-weight formats (like JSON) require less footprint there by improving performance
Defines its own security, as defined by SOAP-WS security	I**nherits security measures** from the underlying transport protocol (TCP, UDP etc.)

Thank You!

In our next session:

Non-REST (SOAP) Web Services