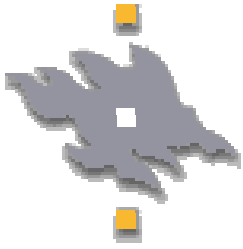


Lecture #9: 1st March 2004

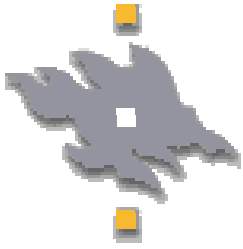
# Service Oriented Architectures: SOA

Suresh Chande

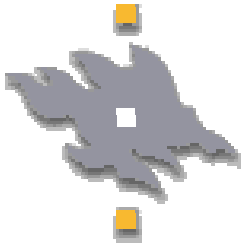


# A Quick Summary

- Web Services architecture
- Web Evolution
- A complete Web Services technology stack
- XML, XML Schema
- Core Web Services stack : SOAP, WSDL, UDDI
- Workflows or process oriented views to services



How do you think Web Services are simplifying/complicating the solutions to the well known integration problems, differently ?



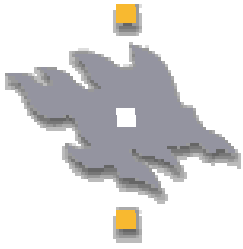
# What do we have ?

- Protocols
- Processing models
- Roles for intermediary processing nodes
- Ubiquitous availability(web)
- Transparency to the Integration environments
- Architecture



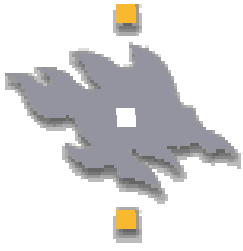
# Service Oriented Architecture

- Web Service technologies as they mature put web technologists to question: How do we put these different technologies together to provide a solution accessible over the web ?
- The increased momentum towards service based computing paradigm has introduced the need to develop a Service-Oriented Architecture



# About SOA

- In SOA, business objectives are achieved by communication with a series of services using the well defined and communication mechanism with the right order to message exchanges.
- It is not a new architectural thinking any way, there have been traces of this since the ~1980's
- The Web Services based SOA is something which is widely accepted and emerged in recent years.



# What's a Service ?

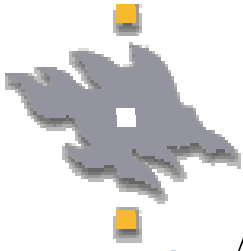
- A Service is an abstract resource that represents a capability of performing tasks that form a coherent functionality from the point of view of providers and requesters[W3C]
- A Service is a logical representation of an organization's functionality it would like to provide/expose to other systems of its consuming partners
- Service provides a means to exchange messages by the requesting client in order to invoke the published functionalities



# What's different with Services ?

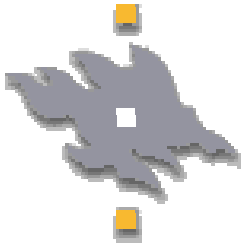
(compared to traditional computing system)

- Does not expose its Target deployment environments to its consumers/invokers
- Does not impose any specific environment except for the need to support interpretation of the XML based messages.
- Provides a loosely coupled means of exposing and integrating service's available functions, maintained at a message level semantics
- Takes the traditional distributed computing(point-2-point) towards End-2-End distributed computing(processing) paradigm
- Represents a higher level interface(service) to a functionally combined set of objects, programs of a specific system.



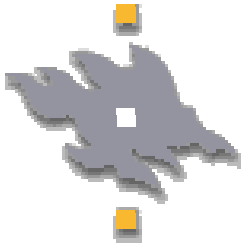
# What is SOA ?

- A set of components which can be invoked, and whose interface descriptions can be published and discovered [W3C]
- A service-oriented architecture is essentially a collection of services. These services communicate with each other. The communication can involve either simple data passing or it could involve two or more services coordinating some activity.
- SOA is an architectural style whose goal is to achieve loose coupling among interacting software agents. A service is a unit of work done by a service provider to achieve desired end results for a service consumer. Both provider and consumer are roles played by software agents on behalf of their owners. [webservices.xml.com]



## What is SOA ?

- SOA is just designing your architecture to best work in a Web service environment, based on the consumer-provider model [ [www.Developer.com](http://www.Developer.com)].
- "An application architecture within which all functions are defined as independent services with well-defined invocable interfaces which can be called in defined sequences to form business processes".



# Needs for a SOA

- To simplify the complexity of the integration in traditional IT solutions
- Ability to provide quick response to changes
- Integration with new partners and solutions in reduced time.
- Solve the communication among heterogeneous environments
- A Consistent integration model where solutions can be created, integrated and reused
- Increasingly Adaptive business or software environments



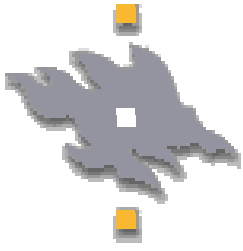
## SOA: Arises from the existing problems

- Legacy utilization and exposure of access to heterogenous platform
- Programmatic access of the legacy systems across widely deployed ubiquitous environments(web)
- Reuse of legacy systems into existing and new infrastructures and vice versa
- Inter-operable distributed computing system



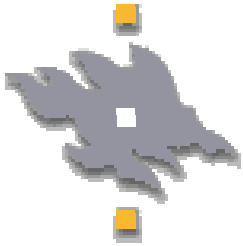
## SOA: Arises from the existing problems

- Discoverable services belonging to specific features or categorization
- Composition of new services making use of existing services and business objects as building blocks
- Transparency from the implementation technology
- Management of the services

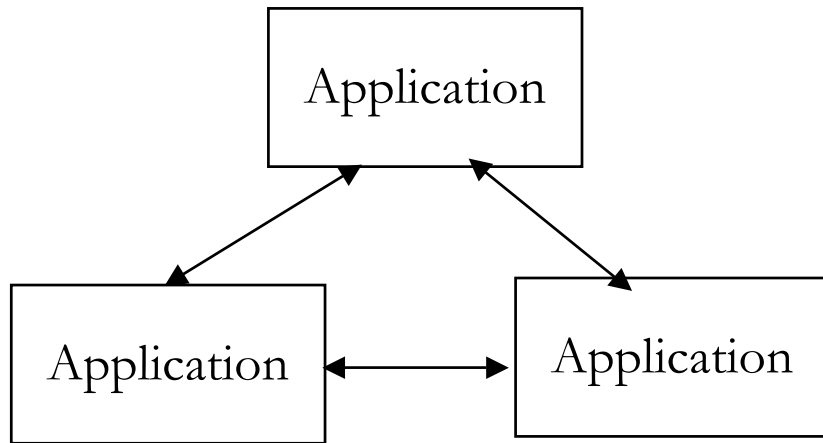


# Service Categorization

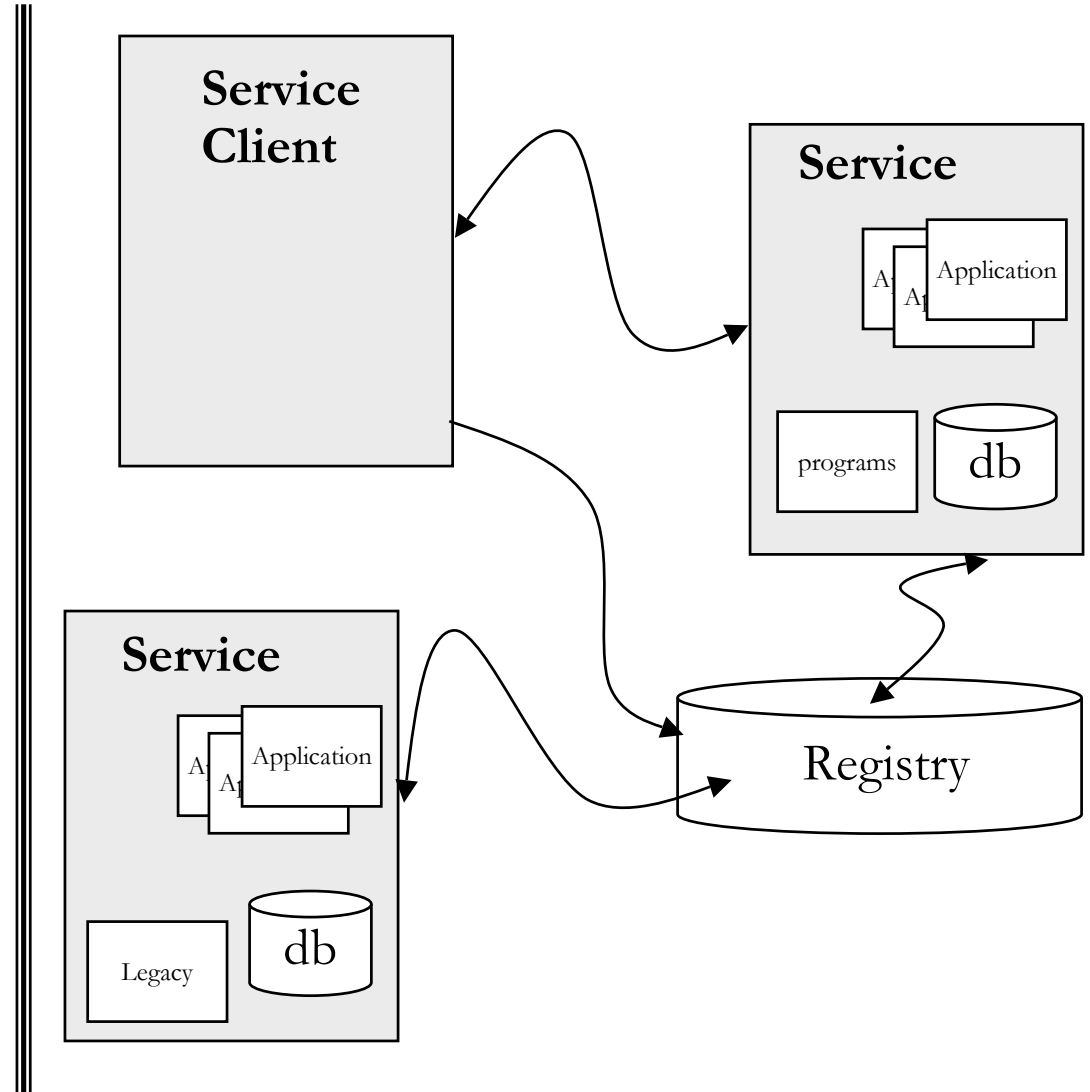
- **Business Services:** get/set business features,
- **System Services:** Trace, Log, Metering, Billing & Charging, Security
- **Transactional Services:** Purchase commits, rollbacks and compensation
- **Management Services:** Start, Stop, Pause, Resume, Cancel.

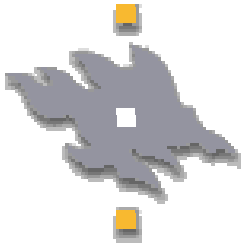


# Service Oriented Architecture: Core Components



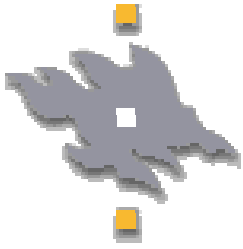
Application Integration





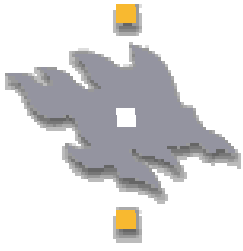
## SOA: Realizations

- Service oriented architectures can be realized by means of different technologies:
  - Web Services
  - REST
  - Web based Message Queue infrastructure  
(**Enterprise Service Bus**)



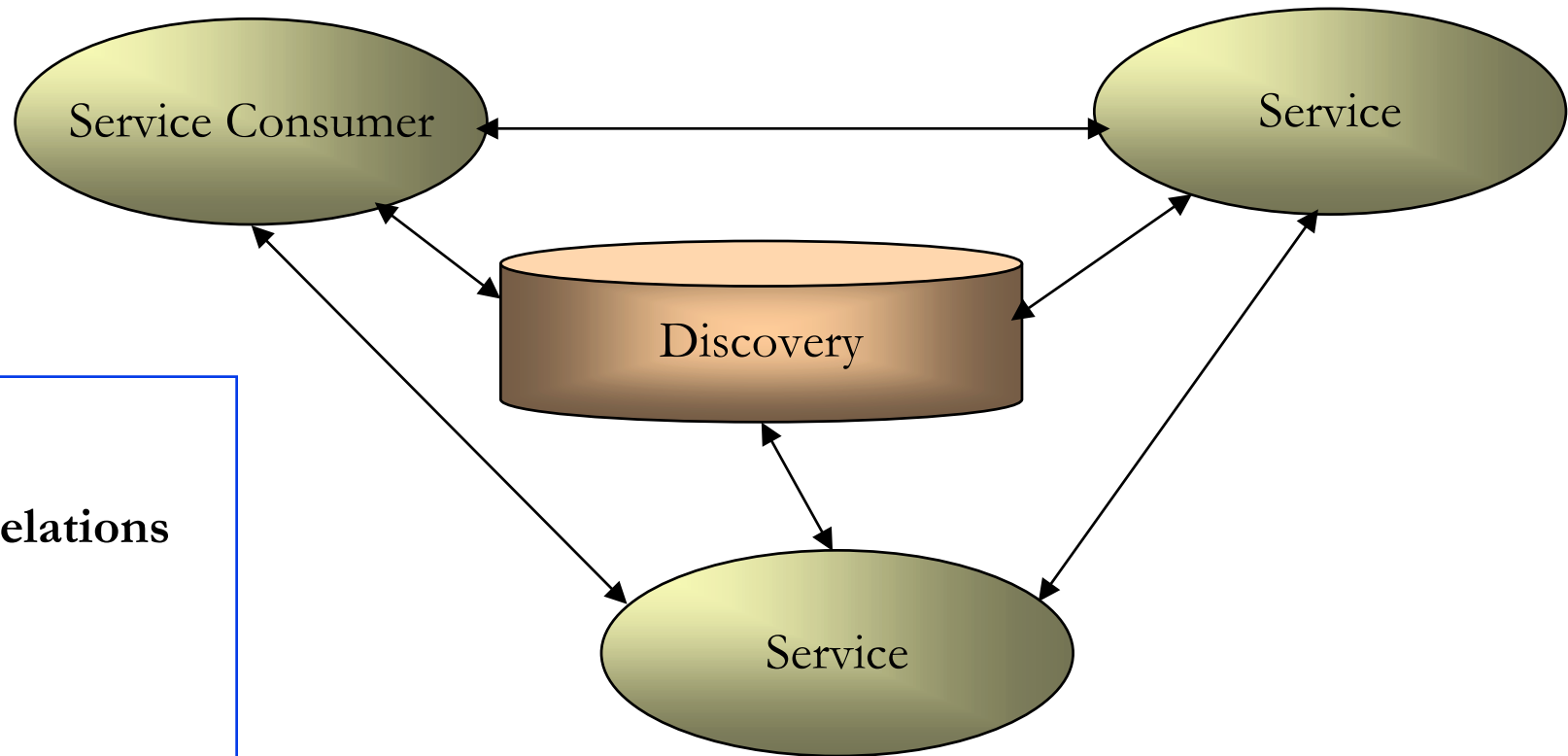
# SOA: Models of Deployment

- Basic Point to Point
- Composed/process oriented SOA
- Shared Context SOA



# Basic SOA: Point 2 Point

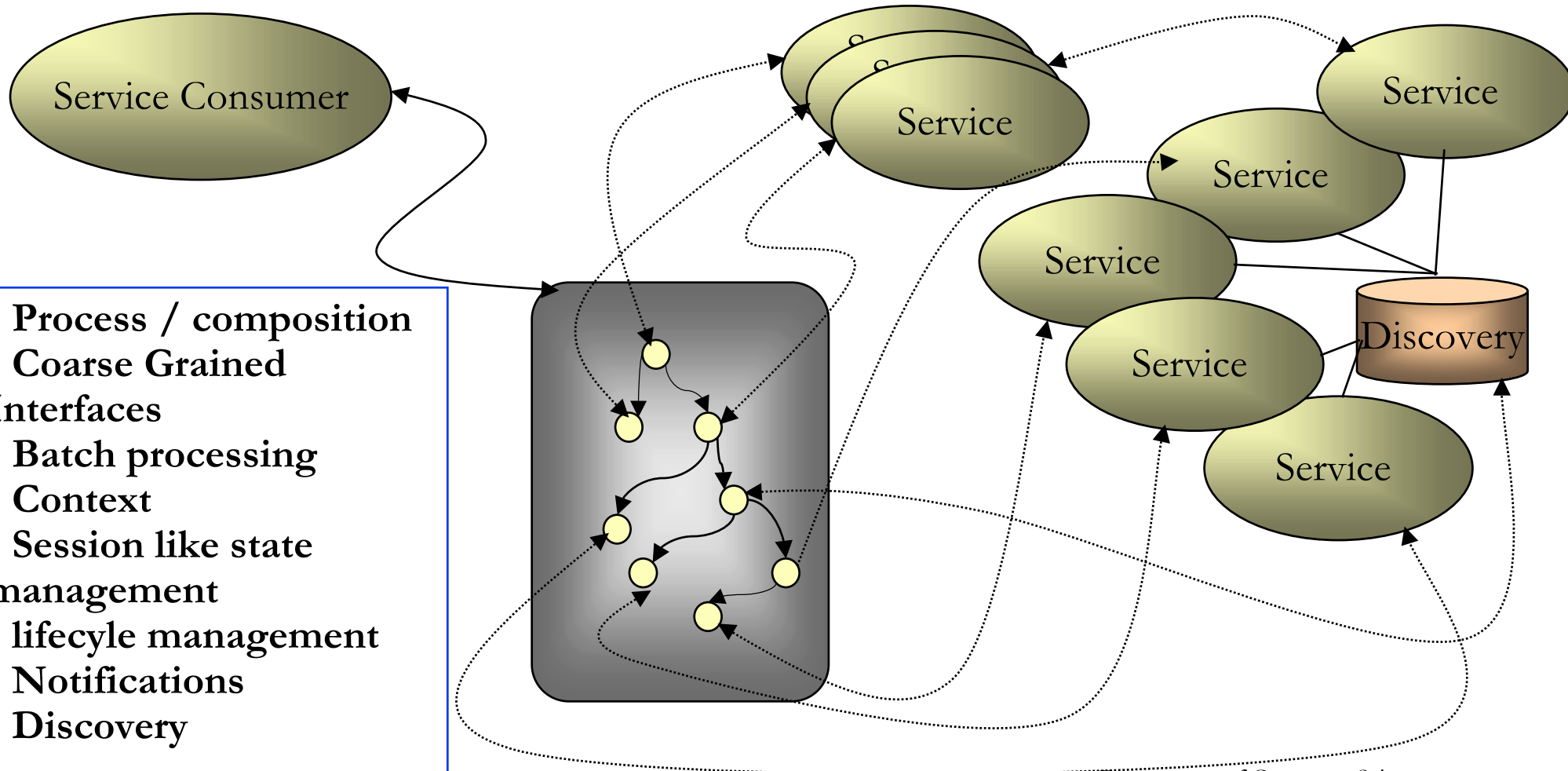
Communication and integration established by point to point  
Message Semantics and message exchange patterns



- **MEPs**
- **Reliability**
- **Message Co-relations**
- **Security**
- **Policies**
- **Notifications**
- **Subscription**

# Composed/Process oriented SOA: Basic extension

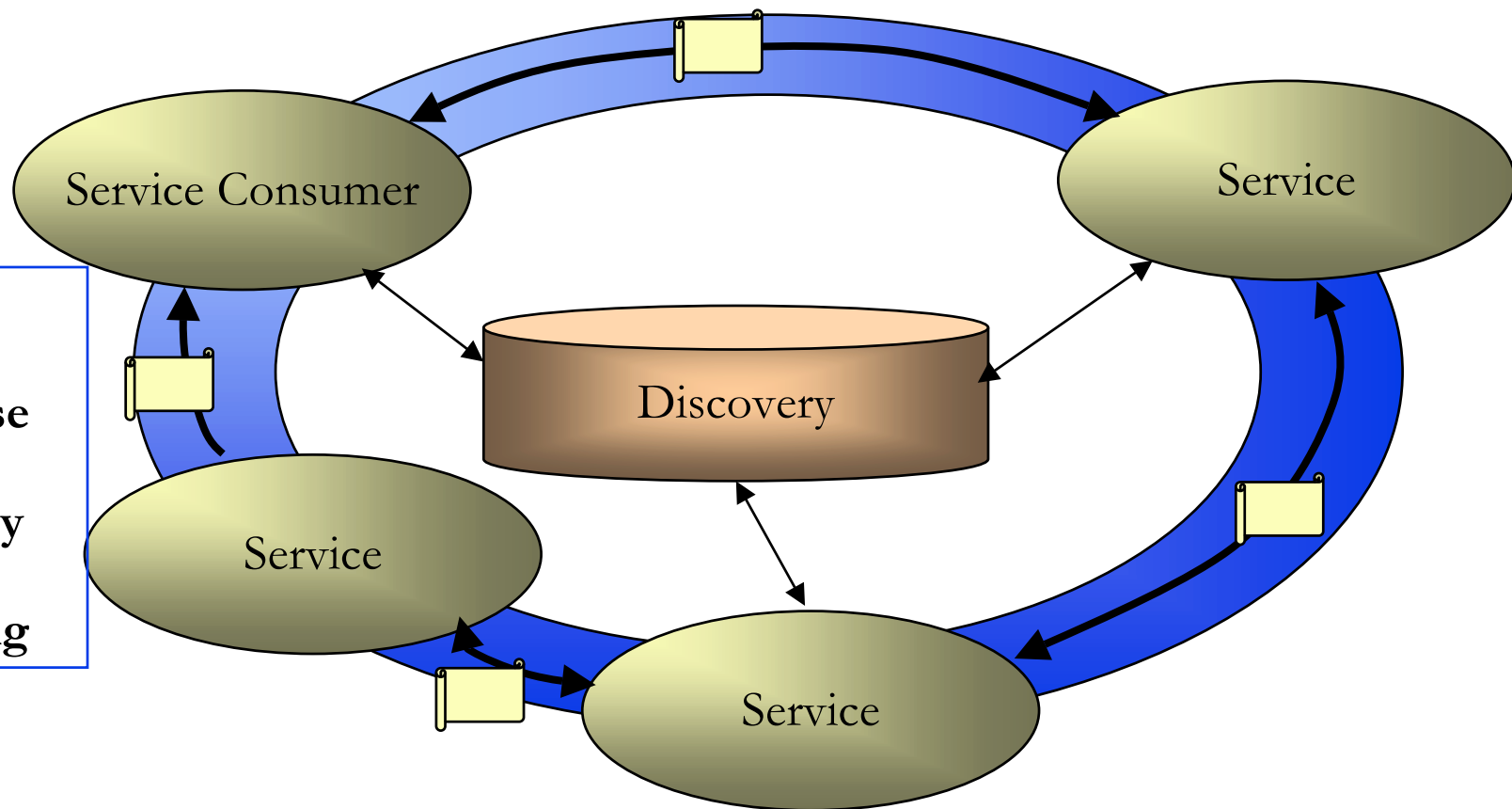
Communication and integration established by a single point / process oriented view to services



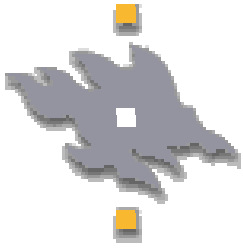
- **Process / composition**
- **Coarse Grained Interfaces**
- **Batch processing**
- **Context**
- **Session like state management**
- **lifecycle management**
- **Notifications**
- **Discovery**

# Shared Context SOA: End 2 End

Communication and service integration established by End-2-End shared Message Semantics and message exchange patterns

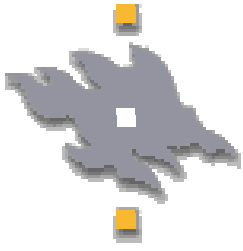


- Shared context
- Routing
- Varying Response / Request path
- Federated security
- Distributed Message processing



# Challenges

- What is the level of granularity in which a service should be exposed ?
- How can you represent the real world documents (messages such as: forms, purchase orders, expense claims) and their processing in SOA which would include both Services and non-service based legacy systems.
- How are the multiple set of services bound together in SOA ?



# Benefits

- Evolve on top of existing infrastructures (leverage reuse of existing infrastructure)
- Extract system/service complexity and rely on the infrastructure of the SOA
- Integration of the services can also benefit from the SOA features
- Time to Market is speedened
- Process oriented view and management on consumption of the service components.