Service-Oriented Architecture MCN 7206

Benjamin Kanagwa,PhD

Monolith

- A Software Monolit
  - One build and deployment unit
  - One code base
  - One technology stack (Linux, JVM, Tomcat, Libraries)
- Benefits
  - Simple mental model for developers
  - one unit of access for coding, building, and deploying
  - Simple scaling model for operations
  - just run multiple copies behind a load balancer

Problems with Monolith

- Huge and intimidating code base for developers
- Development tools get overburdened
  - refactorings take minutes
  - builds take hours
  - testing in continuous integration takes days
- Scaling is limited
  - Running a copy of the whole system is resource-intense
  - It doesn’t scale with the data volume out-of-the-box
- Deployment frequency is limited
  - Re-deploying means halting the whole system
  - Re-deployments will fail and increase the perceived risk of deployment

What is a Service (2)

- In economics and marketing, a service is the non-material equivalent of a good. Service provision has been defined as an economic activity that does not result in ownership, and this is what differentiates it from providing physical goods.

- a process that creates benefits by facilitating either a change in customers, a change in their physical possessions, or a change in their intangible assets.

What is a service (3)

- A Windows Service?
  - RPC Locator, EventLog, DHCP Client,
- Software Service?
  - Distribution Service, Alert Service
  - Security Service, Log Service
- Business Service?
  - Common Operational Picture, Navigation
  - Accounts Receivable, Customers

Context, Composition and State

- Services are most often designed to ignore the context in which they are used
  - It does not mean that services are stateless they are rather context independent !
  - This may be seen as a form of “loose coupling”
    - Services can be reused in contexts not known at design time
- Value can be created by combining, i.e. “composing” services
  - Pay for water a versus bank the money,check bill and then , pay, …

Intents and Offers

- Service consumer expresses “intent”
- Service providers define “offers”

- Sometimes a mediator will:
  - Find the best offer matching an intent
  - Advertise an offer in different ways such that it matches different intent
- Request / Response is just a very particular case of an Intent / Offer protocol

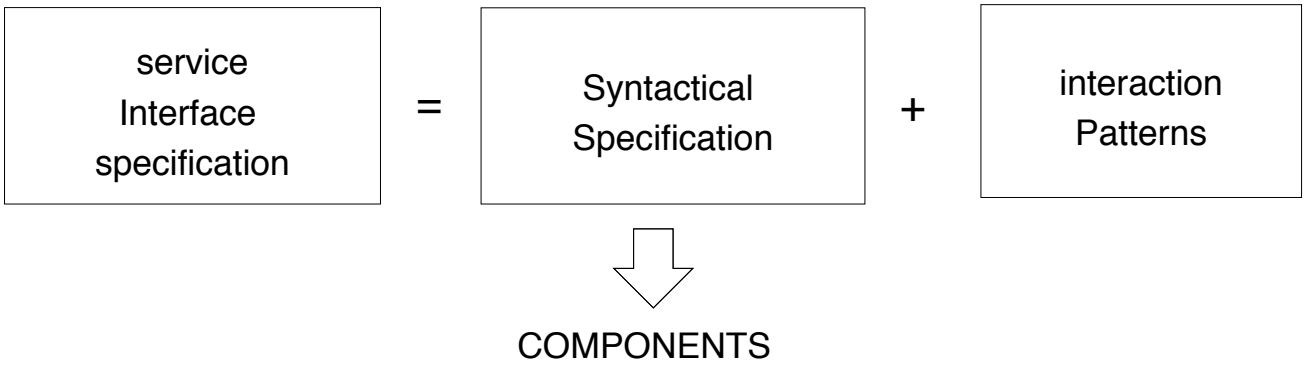
Service Orientation and Directories

Learn & Compose

Service Interfaces

- Non proprietary
  - All service providers offer somewhat the same interface
- Highly Polymorphic
  - Intent is enough
- Implementation can be changed in ways that do not break all the service consumers
  - Real world services interact with thousands of consumers
  - Service providers cannot afford to “break” the context of their consumers

SOA and Previous Advances



- Abstraction
- Compose-ability
- Extended Interfaces
- Separation of Concerns (internals, interfaces, interoperability enterprise level concerns

- Autonomy
- Louse Coupling

Benjamin Kanagwa, Ezra Kaahwa Mugisa (2007):Architecture Analysis of Service Oriented Architecture. Software Engineering Research and Practice 2007: 658-663

Seamless Connectivity enables “On Demand” Computing

- Use software as you need
- Much lower setup time, forget about
  - Installation
  - Implementation
  - Training
  - Maintenance
- Scalable and effective usage of resources
  - Provision
  - Billed on true usage
  - Parallelism (CPU, Storage, Bandwidth…)

Opportunities with Seamless Connectivity

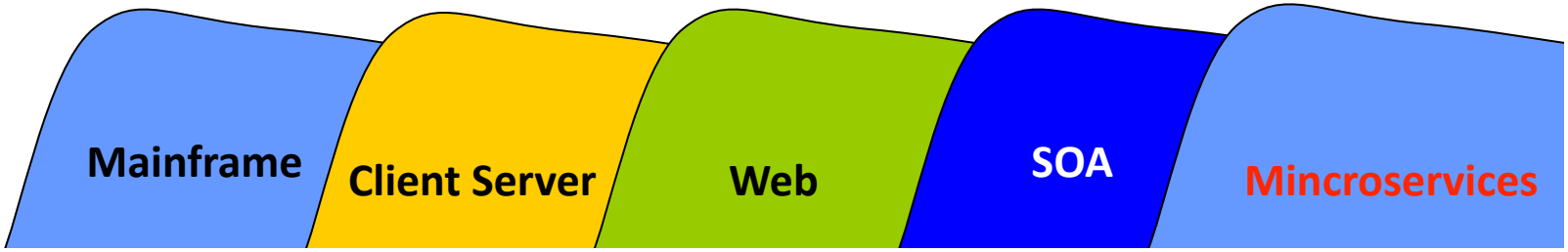
- An application is NOT a single system running on a single device and bounded by a single organization
- Continuum Object … Document
- Messages and Services
  - As opposed « distributed objects »
  - Exchanges becomes peer-to-peer
- Asynchronous communications
- Concurrency becomes the norm while our languages and infrastructures did not plan for it

SOA-New thinking?

- Federation and Collaboration
  - As opposed to « Integration »
- Language(s)
  - Semantic (not syntactic)
  - Declarative and Model driven (not procedural)
- Licensing and Deployment models

So…

- Today, the value is not defined as much by functionality anymore but by connectivity
  - However, we need a new programming model
- Why SOA today?
  - We are reaching a new threshold of connectivity and computing power



What is Service-Oriented Architecture?

- Service-Oriented Architecture (SOA) is an architectural style.

- Applications built using an SOA style deliver functionality as services that can be used or reused when building applications or integrating within the enterprise or trading partners.

SOA

- Uses open standards to integrate software assets as services
- Standardizes interactions of services
- Services become building blocks that form business flows
- Services can be reused by other applications

What is a Service (SOA)?

- A service is a reusable component that can be used as a building block to form larger, more complex business-application functionality.
- A service may be as simple as “get me some person data,” or as complex as “process a disbursement.”

What is a Service(SOA)?

- A service provides a discrete business function that operates on data. Its job is to ensure that the business functionality is applied consistently, returns predictable results, and operates within the quality of service required.

What is a Service(SOA)?

- How the service is implemented, and how a user of the service accesses it, are limited only by the SOA infrastructure choices of the enterprise.
- From a theory point of view, it really doesn’t matter how a service is implemented.

Conclusion

- Service is an architectural style. A way of building systems that exhibit certain architectural qualities