Fundamentals of Software Architecture

aRcH iTe cTuRE Styles & pATtERNs

ThoughtWorks

NEAL FORD

@neal4d
http://nealford.com
Top Level Partitioning

technical partitioning
Top Level Partitioning

technical partitioning

presentation

business rules

persistence

domain partitioning

user interface

module

Catalog

Checkout

ShipTo

Customer
aGEnDA

architecture styles

- Generic
- Domain-specific
  - LMAX
- Technical
  - Layered
  - Microkernel
  - Broker
  - Mediator
  - Event-driven
- ESB-driven SOA
- Serverless
- Modular monolith
- microservices
- Service-based

Quantum = 1
Quanta > 1
Layered Architecture

- presentation layer
- business layer
- persistence layer
- database layer
Layered Architecture

presentation layer

business layer

persistence layer

database layer

request
Layered Architecture

presentation layer

business layer

persistence layer

database layer

separation of concerns
Layered Architecture

hybrids and variants

presentation layer
business layer
services layer
persistence layer
database layer

component component component
component component component
component component component
component component component
component component component

CLOSED
CLOSED
OPEN
CLOSED
CLOSED
Layered Architecture

separation of concerns
We need direct access to the database for performance reasons.
public class LayerDependencyRulesTest {
    private JavaClasses classes;

    @Before
    public void setUp() throws Exception {
        classes = new ClassFileImporter().importPackagesOf(ClassViolatingCodingRules.class);
    }

    @Test
    public void services_should_not_access_controllers() {
        noClasses().that().resideInAPackage("..service..")
            .should().accessClassesThat().resideInAPackage("..controller..").check(classes);
    }

    @Test
    public void persistence_should_not_access_services() {
        noClasses().that().resideInAPackage("..persistence..")
            .should().accessClassesThat().resideInAPackage("..service..").check(classes);
    }

    @Test
    public void services_should_only_be_accessed_by_controllers_or_other_services() {
        classes().that().resideInAPackage("..service..")
            .should().onlyBeAccessed().byAnyPackage("..controller..", "..service..").check(classes);
    }
}

layer dependency
// Controllers should not directly reference repositories
var result = Types.InCurrentDomain()
  .That()
  .ResideInNamespace("NetArchTest.SampleLibrary.Presentation")
  .ShouldNot()
  .HaveDependencyOn("NetArchTest.SampleLibrary.Data")
  .GetResult().IsSuccessful;
LIVE ONLINE TRAINING

Automating Architectural Governance using Fitness Functions

Agile Engineering in Architecture

Next class: 12 March 2020 09:00-12:00 ET (06:00-09:00 PT)
Layered Architecture

Drivers

- Presentation layer
- Business layer
- Services layer
- Persistence layer
- Database layer

✅ simplicity
✅ cost
aGEnDA
Microkernel Architecture

(a.k.a. plug-in architecture pattern)
Microkernel Architecture

architectural components

core system

minimal functionality to run system
general business rules and logic
no custom processing

plug-in module

standalone independent module
specific additional rules or logic
Microkernel Architecture
Microkernel Architecture

source validation tool

- check header standards
- check interceptors
- check contract standards
- check SQL calls
- check audit writes
- check other stuff...

read source files
validation report
Microkernel Architecture

claims processing

NH module

TX module

GA module

CA module

NY module

MA module
Microkernel Architecture

plug-in contracts

core system

plug-in component 1

plug-in component 2

plug-in component 3

plug-in component 4

std

std

std

std
Microkernel Architecture

drivers

- ✔ modularity
- ✔ simplicity
- ✔ cost
- ✔ adaptability
- ✔ evolvability
- ✔ agility
- ✔ testability
- ✔ deployability
Event-driven Architecture

broker topology

mediator topology
broker base architecture
broker base architecture
broker base architecture
broker base architecture
broker base architecture
Event-driven Architecture

broker topology
you move... 

you moved!

Event-driven Architecture

customer process

change address

change address

quote process

recalc quote

notification process

claims process

update claims

update claims

update claims

change address

adjustment process

choreography
Event-driven Architecture

drivers

✓ modularity
✓ agility
✓ fault-tolerance
✓ scalability
✓ performance
✓ elasticity
✓ adaptability
✓ evolvability
mediator base architecture
mediator base architecture

camel.apache.org
mediator base architecture
mediator message flow

you move...

process engine

you moved

change address

recalc quote

update claims

adjust claims

notify insured

customer process

quote process

claims process

adjustment process

notification process
mediator message flow

you move...

process engine

- change address
- recalc quote
- update claims
- adjust claims
- notify insured

- customer process
- quote process
- claims process
- adjustment process
- notification process
mediator message flow

you move...

process engine

- change address
- recalq quote
- update claims
- adjust claims
- notify insured

customer process
quote process
claims process
adjustment process
notification process
mediator message flow

You move...

you moved

process engine

change address
recalc quote
update claims
adjust claims
notify insured

customer process
quote process
claims process
adjustment process
notification process
Question:
Which is more performant and more scalable—broker or mediator?
aGEnDA
ESB-driven SOA

- Message bus
  - Process choreographer
  - Service orchestrator
- Business services (BS)
- Enterprise services (ES)
- Application services (AS)
- Infrastructure services (IS)
ESB-driven SOA
ESB-driven SOA
ESB-driven SOA

- Business services (BS)
- Message bus
  - Process choreographer
  - Service orchestrator
- Enterprise services (ES)
- Application services (AS)
- Infrastructure services (IS)
ESB-driven SOA

business services

message bus

process choreographer

service orchestrator

enterprise services

application services

infrastructure services

business services

enterprise services

application services

infrastructure services

business services
ESB-driven SOA

enterprise services

business services

message bus

process choreographer

service orchestrator

enterprise services

application services

infrastructure services
ESB-driven SOA

application services

business services

message bus

process choreographer

service orchestrator

enterprise services

application services

infrastructure services
ESB-driven SOA

infrastructure services

- business services
- message bus
- process choreographer
- service orchestrator
- enterprise services
- application services

infrastructure services
ESB-driven SOA

enterprise scope

service taxonomy

shared resources
ESB-driven SOA

shared resources

- auto and homeowners insurance division
- commercial insurance division
- casualty insurance division
- life insurance division
- disability insurance division
- travel insurance division
ESB-driven SOA

shared resources

- Auto and homeowners insurance division
- Commercial insurance division
- Casualty insurance division
- Life insurance division
- Disability insurance division
- Travel insurance division

Customer service
ESB-driven SOA

shared resources

- auto and homeowners insurance division
- commercial insurance division
- casualty insurance division
- customer service
- life insurance division
- disability insurance division
- travel insurance division
ESB-driven SOA
ESB-driven SOA

protocol-agnostic heterogeneous interoperability

- service consumer (C#/.NET)
  - REST
- service consumer (Java)
  - AMQP
- service consumer (Java)
  - REST
- messaging
- middleware
- service (EJB3)
  - RMI
- service (C++/Tux)
  - ATMI
- service (Java)
  - SOAP
ESB-driven SOA

contract decoupling

SKU
mm/dd/yy

business service

transformation and enhancement

enterprise service

product id
yyyy.mm.dd

quantity

xml

184684538
04/12/17

xml

184684538
04/12/17

java

184684538
04/12/17

A432-98
2017.04.12
100

A432-98
2017.04.12
100

xml

java

java
service-oriented architecture

- enterprise scope
- abstraction (api layers)
- contract decoupling
- heterogeneous integration
- shared resources
- service taxonomy
ESB-driven SOA

Drivers

- Integration
- Integration
- Orchestration
Serverless Architectures

BaaS

FaaS
Serverless
BaaS
(Backend as a Service)

traditional client/server architecture
BaaS

Client (browser) → API Gateway → Authentication Service → Purchase Function → Purchase Database → Product Database → Purchase Function → Search Function → API Gateway → Pet Store Server → Database

- API gateway
- Authentication service
- Purchase function
- Search function
- Product database
- Database

80
Serverless Architectures

BaaS

FaaS
FaaS

(Functions as a Service)
Serverless

drivers

- agility
- lack of (apparent) infrastructure
- discrete functionality
Last 10% Trap

“Users always want 100% of what they want (& are never satisfied with less).”
What happened to the 4GLs?
aGEnDA
Modular Monolith
Modular Monolith

drivers

- agility
- inverse Conway Maneuver
- modularity
- deployability
- restructurability
Microservices Architecture
Microservices Architecture
Microservices Architecture

protocol-aware heterogeneous interoperability
Microservices Architecture

distributed

separately deployed
Microservices Architecture
Microservices Architecture

distributed

separately deployed

service component
Microservices Architecture

Client requests → API layer → Service components → Databases

Diagram showing the flow of client requests through the API layer to various service components, each leading to a database.
Microservices Architecture
Microservices Architecture

transactions

no acid transaction
Microservices Architecture

distributed

separately deployed

service component

bounded context
Microservices Architecture
microservices architecture

distributed

separately deployed

service component

bounded context

data domains
Microservices Architecture
Microservices Architecture

Client requests → API layer → Service components ➔ Databases ➔ Client requests

Hybrid
Microservices Architecture

distributed

separately deployed

service component

bounded context

data domains

api layer
API Layer

hides the actual endpoint of the service, exposing only those services available for public consumption
API Layer

internal and external client requests
API Layer

internal and external client requests

feature toggle
API Layer

internal and external client requests
microservices architecture

distributed
separately deployed
service component
bounded context
data domains
api layer

event driven
Microservices Architecture
Microservices Architecture

drivers

- modularity
- agility
- fault-tolerance
- scalability
- testability
- deployability
- evolvability
architecture styles

Layered

Microkernel

Broker

Mediator

Technical

Event-driven

ESB-driven SOA

Serverless

Modular monolith

Domain

Generic

Domain-specific

Space-based

LMAX

Quantum = 1

Quanta > 1

microservices

Service-based

aGEnDA
Service-based Architecture
Service-based Architecture

- service granularity
- database scope
- deployment pipeline
Service-based Architecture

service granularity

client requests

client requests

client requests

api layer

service component

module

module

service component

module

module

service component

module

module

service component

module

module

service component

module

module

service component

module

module
Service-based Architecture

Service granularity

client requests

client requests

client requests

user interface layer

service component

module
module
module
module

module
module
module
module

module
module
module
module

module
module
module
module

module
module
module
module

module
module
module
module

service component

module
module
module
module

module
module
module
module

module
module
module
module

module
module
module
module

module
module
module
module

Service-based Architecture

Service granularity

client requests

client requests

client requests

user interface layer

service component

module
module
module
module

module
module
module
module

module
module
module
module

module
module
module
module

module
module
module
module

module
module
module
module

module
module
module
module

module
module
module
module

module
module
module
module

module
module
module
module

Service-based Architecture

service granularity

tradeoffs

👍 performance
👍 robustness
👍 domain scope

👎 service development
👎 service testing
👎 deployment pipeline
Service-based Architecture

database scope

client requests

client requests

client requests

api layer

service component

module

module

module

module

module

module

module

service component

module

module

module

module

module

module

module

database

client requests
Service-based Architecture

database scope

client requests

user interface layer

client requests

client requests

Service-based Architecture

database scope

client requests

user interface layer

client requests

client requests

database scope

Service-based Architecture
Service-based Architecture

database scope

client requests → user interface layer → service component → database

client requests → user interface layer → service component → database

client requests → user interface layer → service component → database
Service-based Architecture

database scope

tradeoffs

👍 performance
👍 feasibility

👎 service coupling
👎 schema changes
aGEnDA

architecture styles

Layered
- Microkernel
  - Broker
    - Mediator
  - Event-driven
    - ESB-driven SOA
      - Serverless
        - Modular monolith
          - Domain
            - microservices
              - Service-based

Quantum = 1
Quanta > 1

Domain-specific
  - Space-based
    - LMAX
Space-based Architecture

let's talk about scalability for a moment...
Space-based Architecture
Space-based Architecture

processing unit

processing unit

module module module

in memory data grid

data replication engine
Space-based Architecture

middleware

- messaging grid
- data grid
- processing grid
- deployment manager
Space-based Architecture

middleware

manages input request and session

- messaging grid
- data grid
- processing grid
- deployment manager

processing unit
processing unit
processing unit
processing unit

virtualized middleware

messaging grid
data grid
processing grid
deployment manager

http://
Space-based Architecture

middleware
manages data replication between processing units

- messaging grid
- data grid
- processing grid
- deployment manager
Space-based Architecture

middleware
manages distributed request processing
Space-based Architecture

classified

middleware
manages dynamic processing unit deployment

- messaging grid
- data grid
- processing grid
- deployment manager
Space-based Architecture

product implementations

javaspaces

gigaspaces

ibm object grid

gemfire

ncache

oracle coherence
it's all about variable scalability...

good for applications that have variable load or inconsistent peak times

not a good fit for traditional large-scale relational database systems

relatively complex and expensive pattern to implement
Space-based Architecture

drivers

- scalability
- elasticity
- performance
aGENDA
Overall Structure

6,000,000 transactions /second!
Business Logic Processor

in-memory snapshots

multiple instances running

(full restart—JVM + snapshots— < 1 min)

each event processed by multiple processors

but only one result used
Overall Structure

- Input Disruptor
- Business Logic Processor
- Output Disruptor
Input Disruptor

custom concurrency component
20x10^6 slots for input buffer
4x10^6 slots for output buffer
Input Disruptor

custom concurrency component
Overall Structure
aGEnDA

architecture styles

Layered
Microkernel
Broker
Mediator
Event-driven
ESB-driven SOA
Serverless
Modular monolith
Domain
microservices
Service-based

Quantum = 1
Quanta > 1

Domain-specific
LMAX

aGEnDA