Open Cloud Engine - An Open Source Cloud Native Transformer

UEngine

AS-IS: Pain-points in service operation

- Requests for Service upgrade is too frequently, it brings over-time working everyday. Developer's happiness grade is too low.
- Module update of one team effects all the teams' modules, all teams have to test all the systems and standby during every single deployment of teams.

- With Separated operation team and development team,
- Even the development has been done, ops team cannot deploy the new features due to the fear of the errors that brings customer loss.
- With manual operation, it is hard to mange the Service Level Agreements, the claims from customers is increasing.

Remedies:

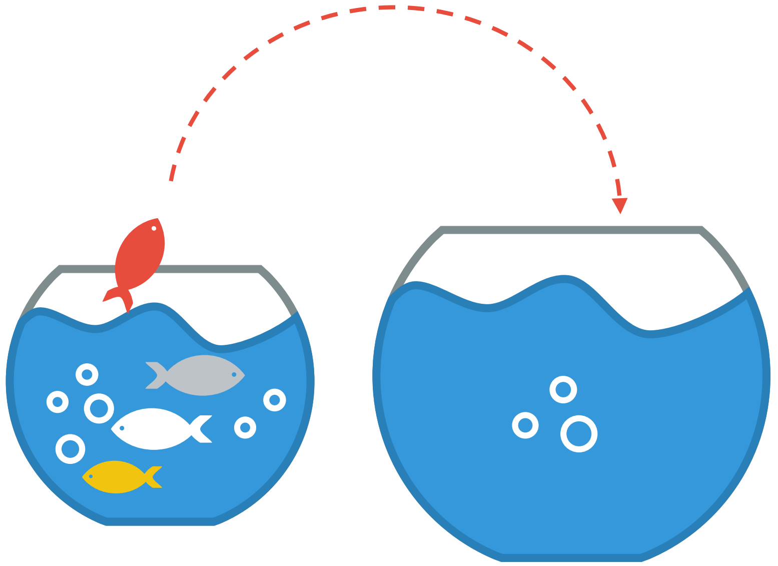
1. Microservices Architecture
2. DevOps

Open Cloud Engine

UEngine

Migrating to Cloud Native Application

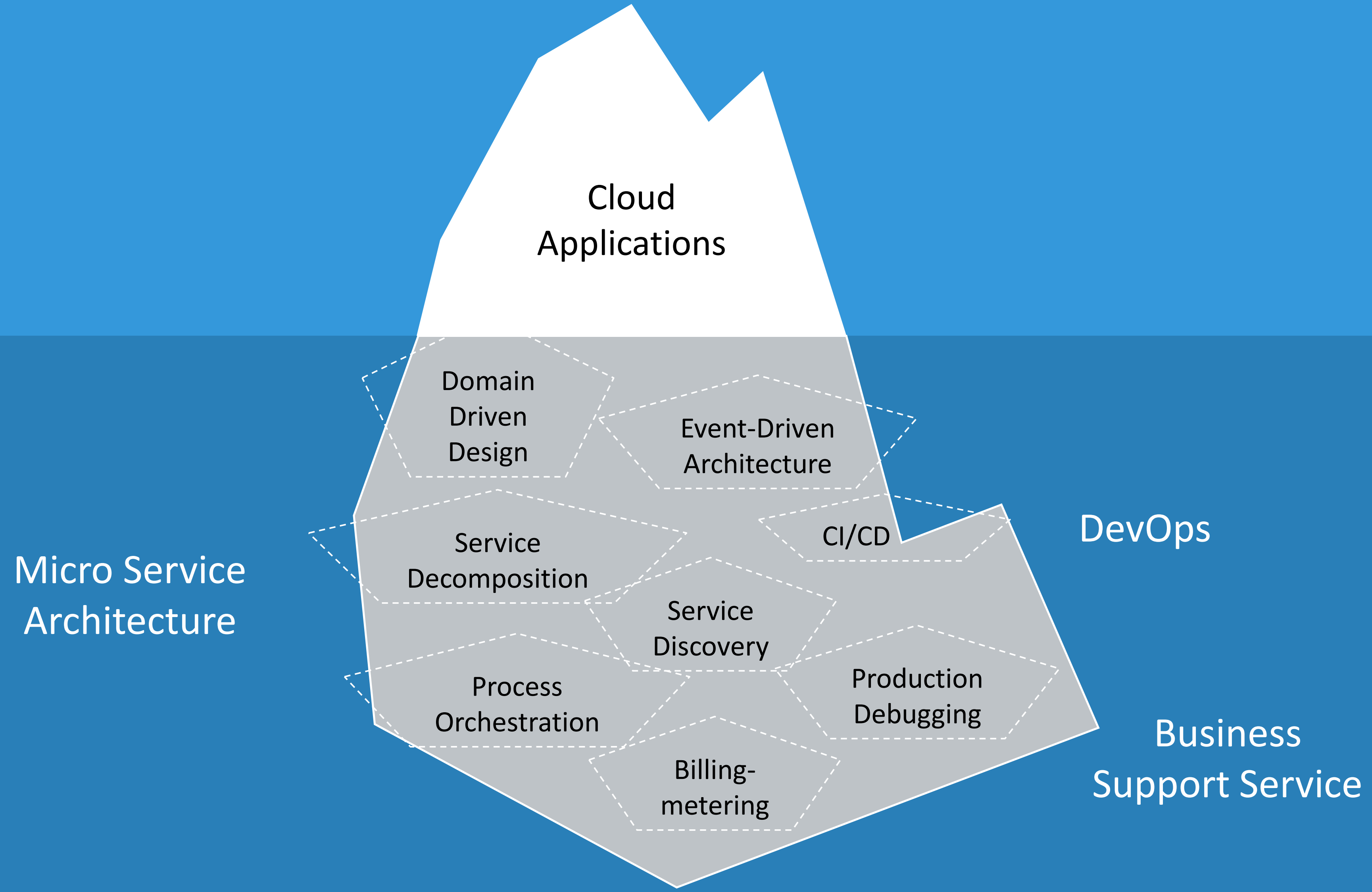
Product and customization
Long-term Delivery (typically 9 mo.)
Monolithic



Subscription and self-serviced
Agile and Continuous Delivery
Micro services

Skills & Expertise on MSA

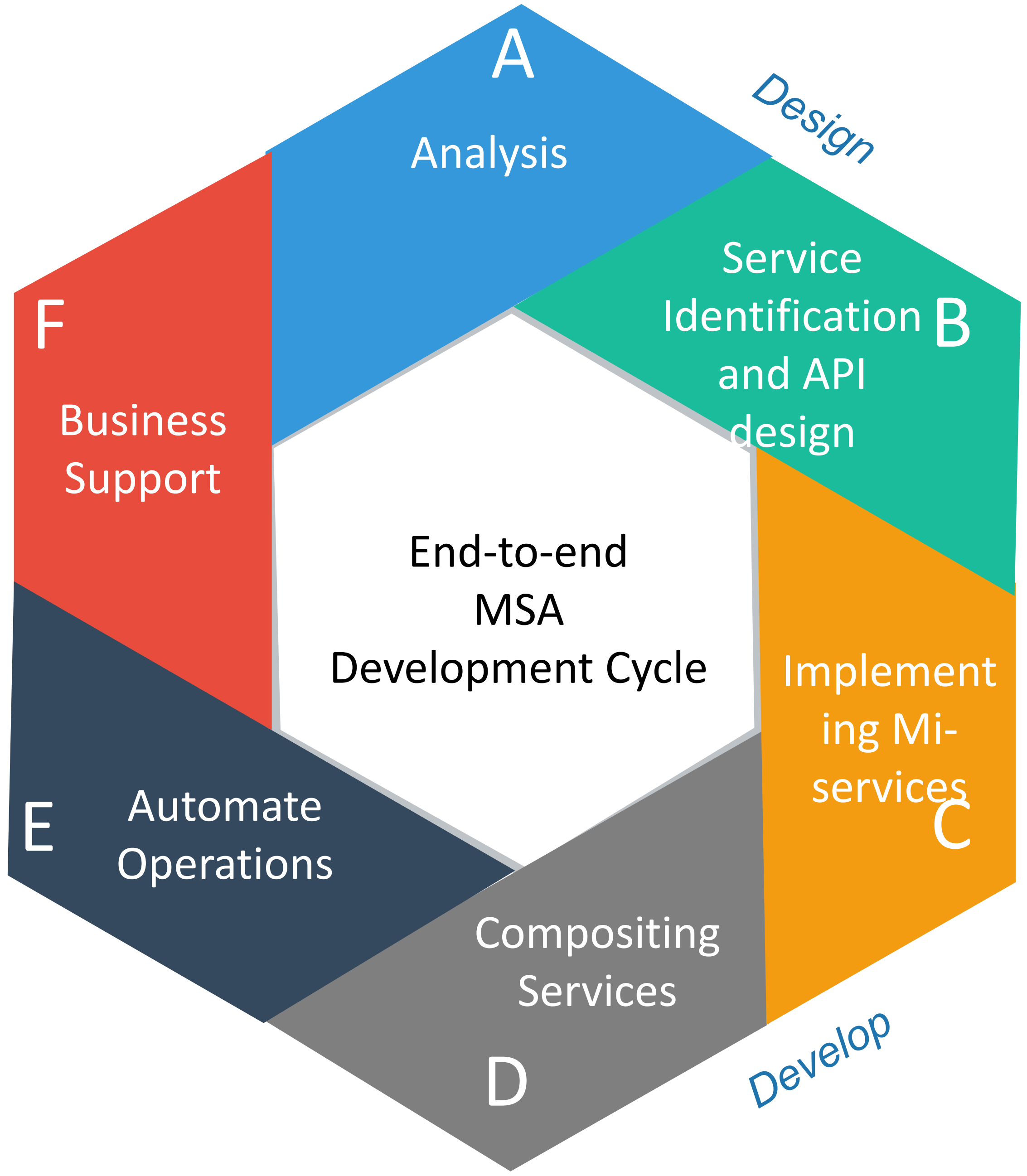
Implementing Successful Native Cloud Application requires 1. **Micro Service Architecture-based Application Design and 2. Tools and environments of DevOps**



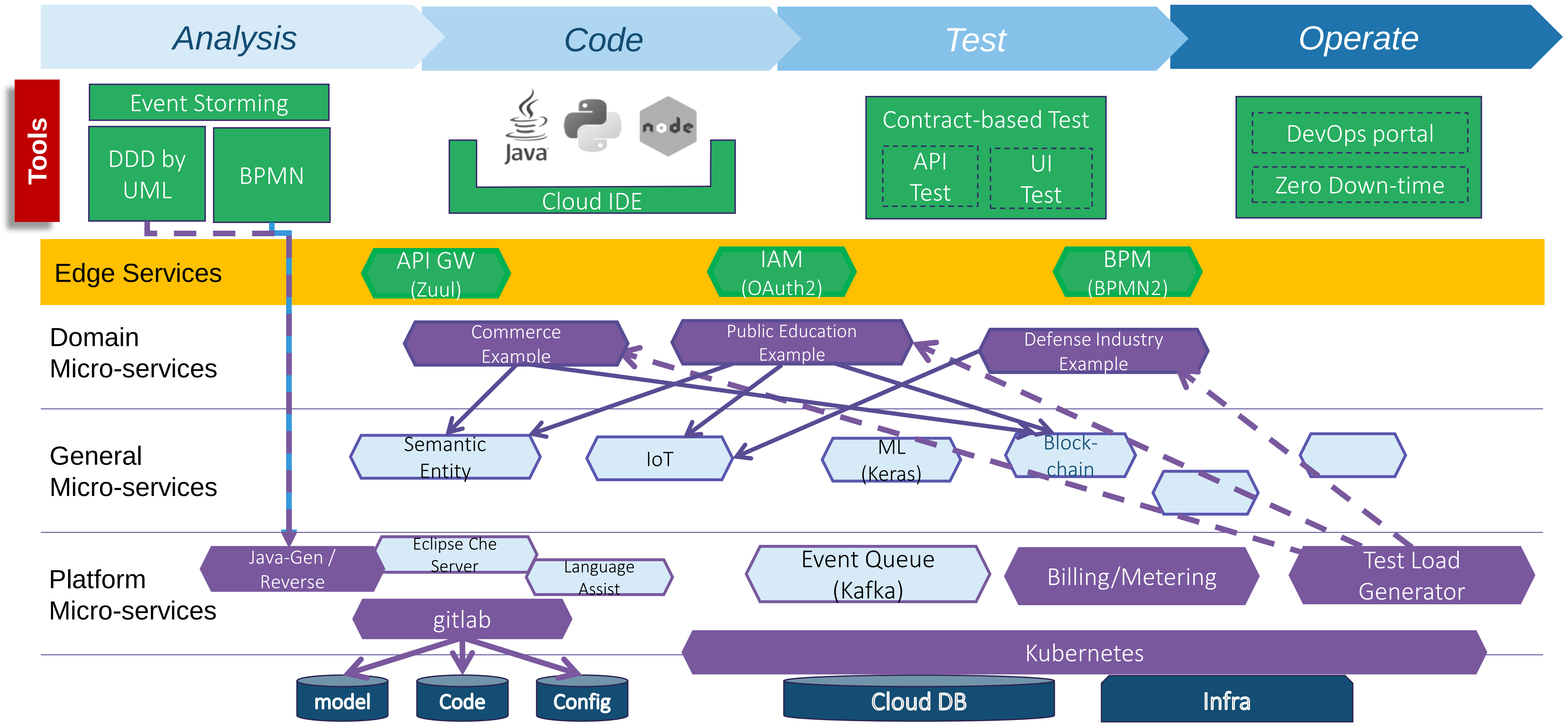
OCE's mission

Open Cloud Engine aims to support full-lifecycle of cloud native application development

http://github.com/TheOpenCloudEngine



OCE components



Analysis / Design Phase

uEngine

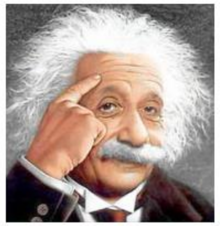
Micro Services Characteristics

Components via Services

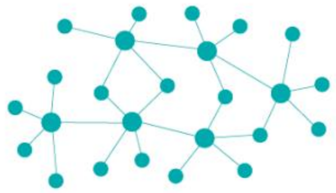
Organized around **Business Capabilities**

Products NOT Projects

Smart Endpoints & Dumb Pipes



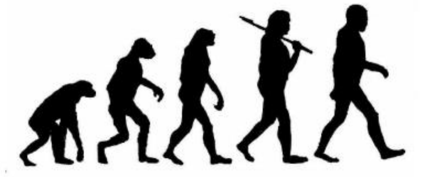
Decentralized Governance & Data Management



Infrastructure **Automation**

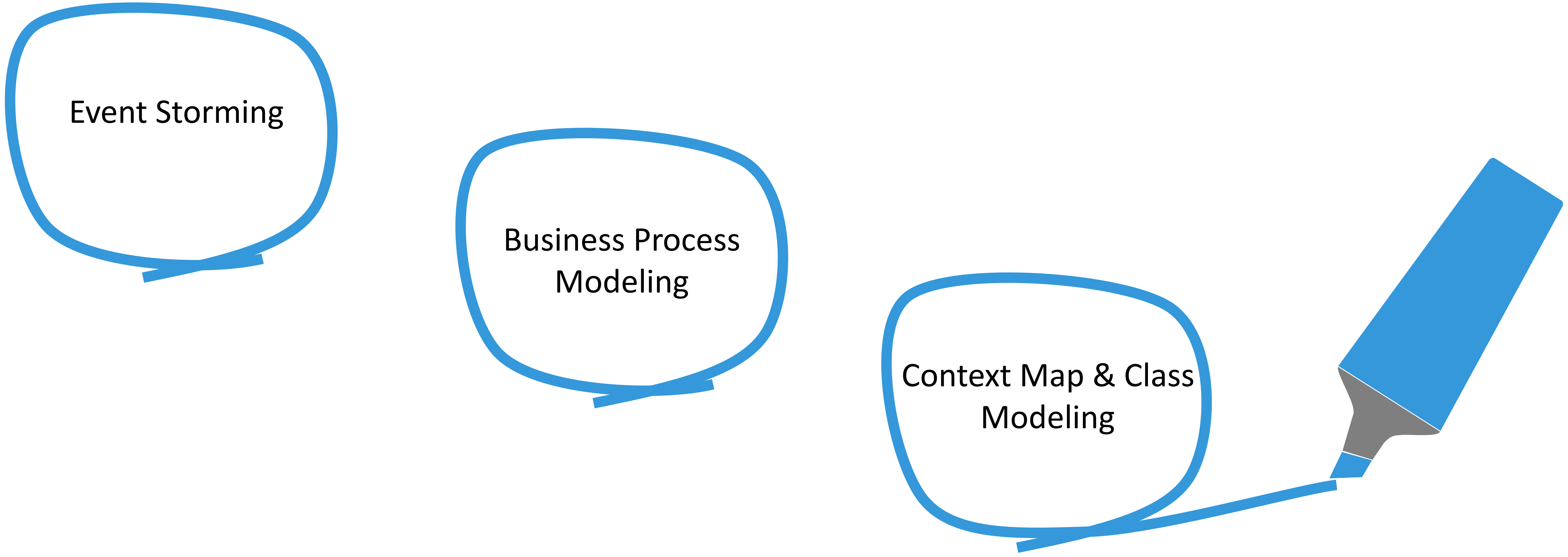
Design for **Failure**

Evolutionary Design

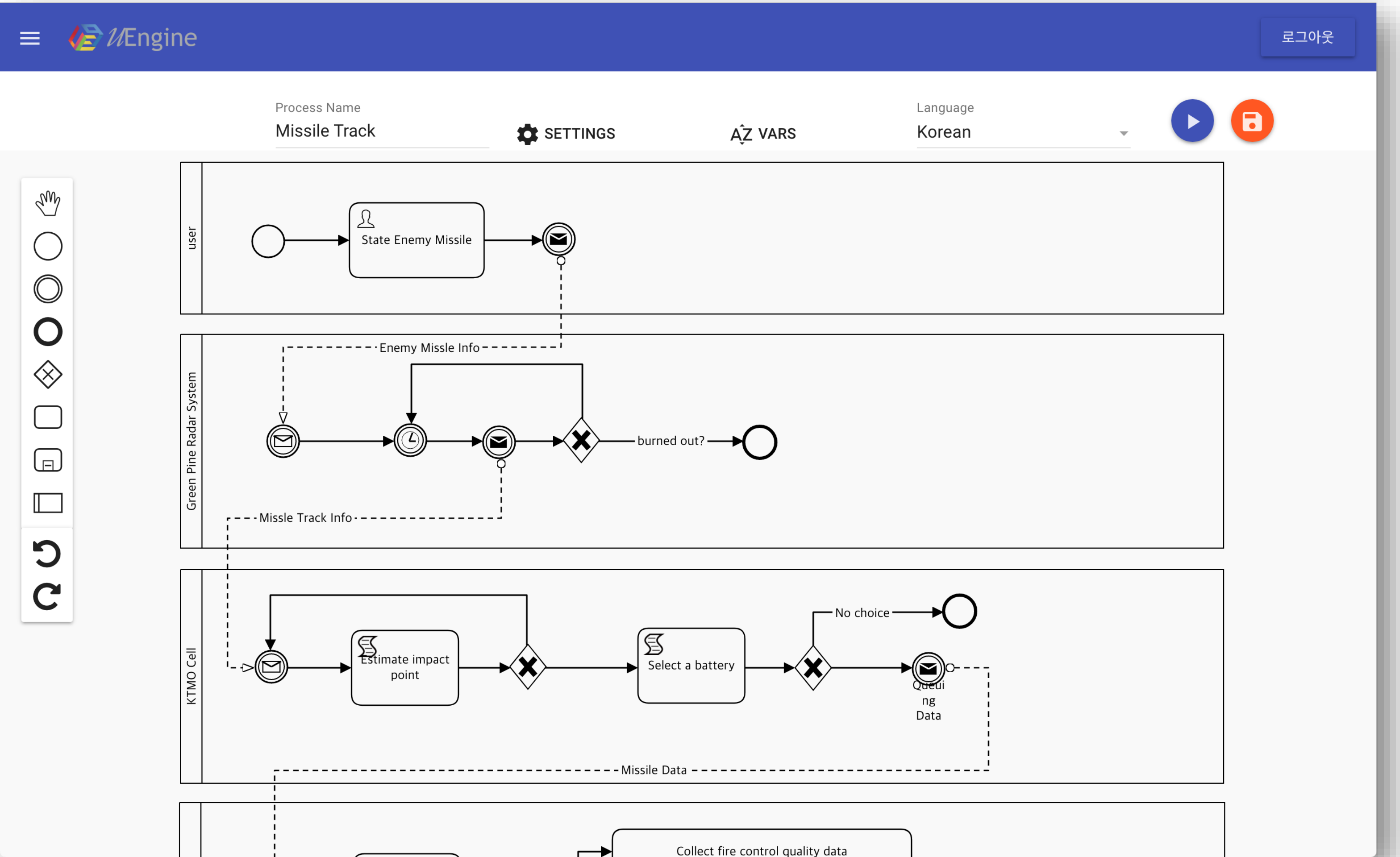


- By James Lewis and Martin Fowl

Doman-Driven Design Process



Process Modeling with BPMN



To analysis event–driven inter–communication of mi–service, BPMN2.0 specification could be used for modeling its choreography with their expressive power such as “Service Pools”, “Web Service Tasks”, and “Signal Events”.

Later, process definitions can be used for generating source code for Java(Spring)–based event–driven applications.

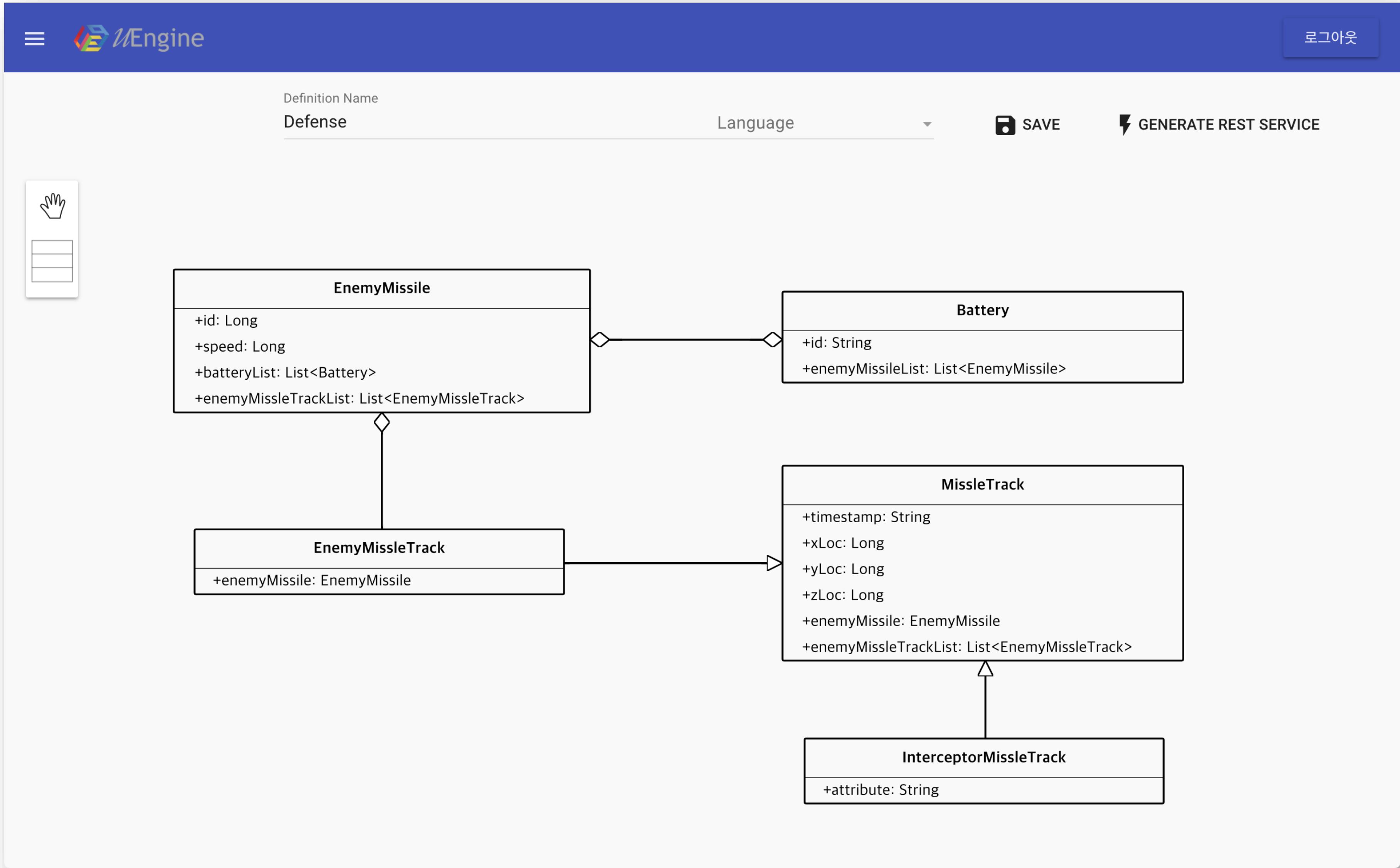
utilized OSS: VueJS(China), OpenGraph(Korea)

Domain Class Modeling with UML

By using the Domain class modeler, domain experts or application architects can draw their domain model for unit Mi-services.

Domain models can be used for generator Java(Spring)-based database applications and the changes in application code will be applied to the model vice-versa. (Round-trip Engineering)

utilized OSS:
Javareverse, VueJS



EnemyMissile
+id: Long
+speed: Long
+batteryList: List<Battery>
+enemyMissleTrackList: List<EnemyMissleTrack>

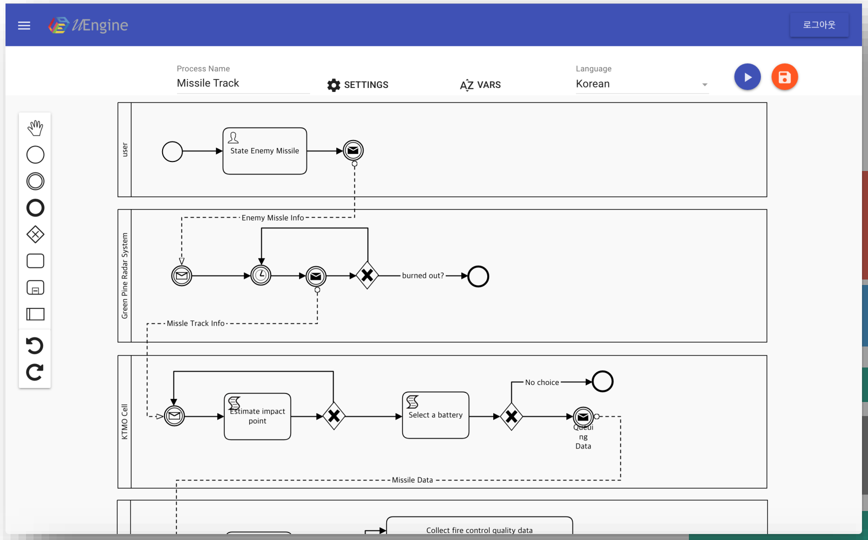
Battery
+id: String
+enemyMissileList: List<EnemyMissile>

EnemyMissleTrack
+enemyMissile: EnemyMissile

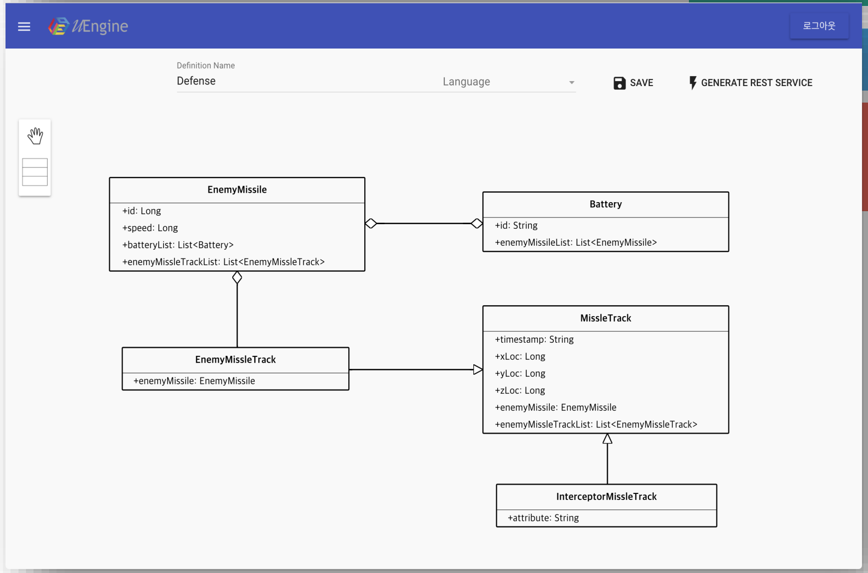
MissleTrack
+timestamp: String
+xLoc: Long
+yLoc: Long
+zLoc: Long
+enemyMissile: EnemyMissile
+enemyMissleTrackList: List<EnemyMissleTrack>

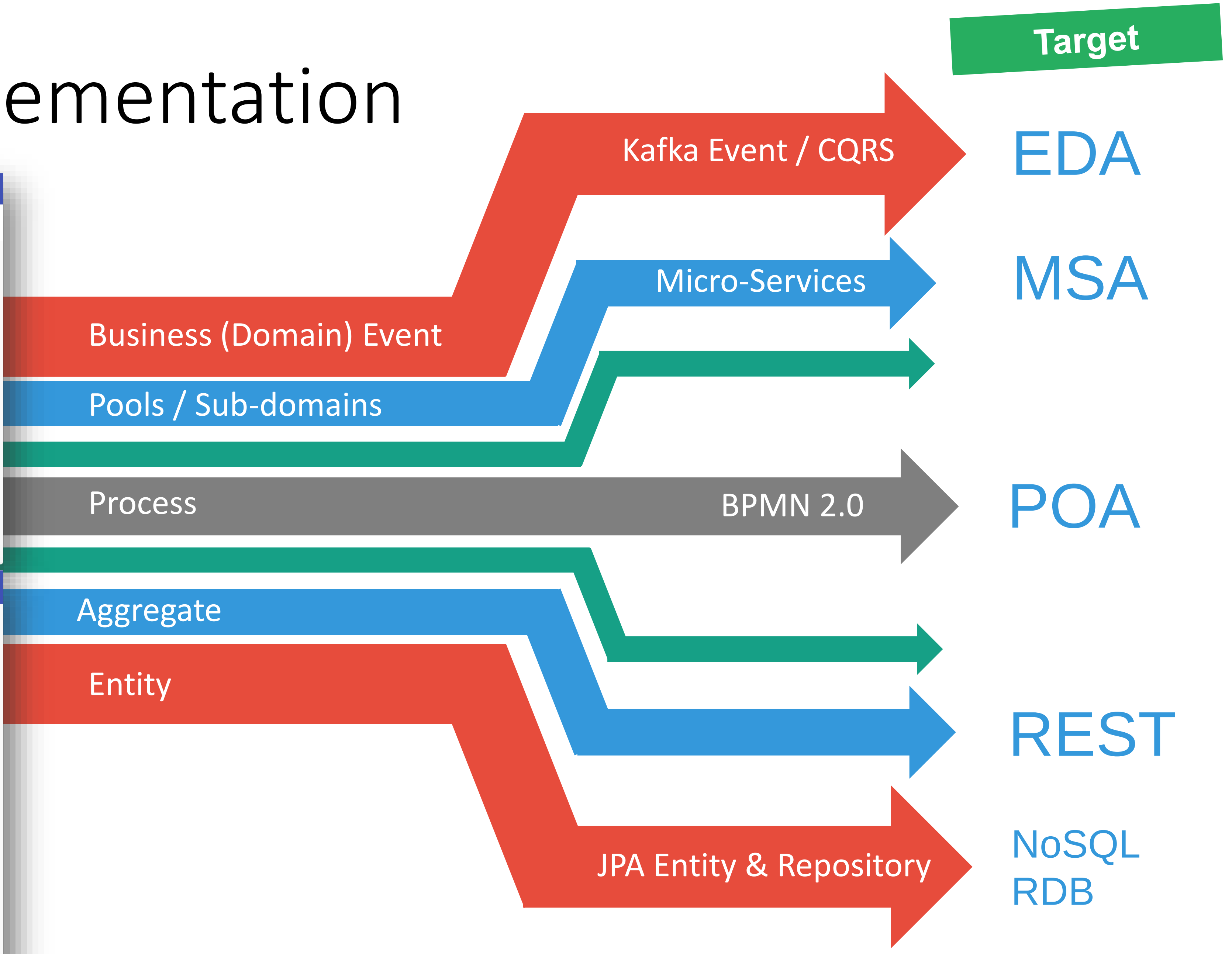
InterceptorMissleTrack
+attribute: String

Model to Implementation


Process Name: Missile Track

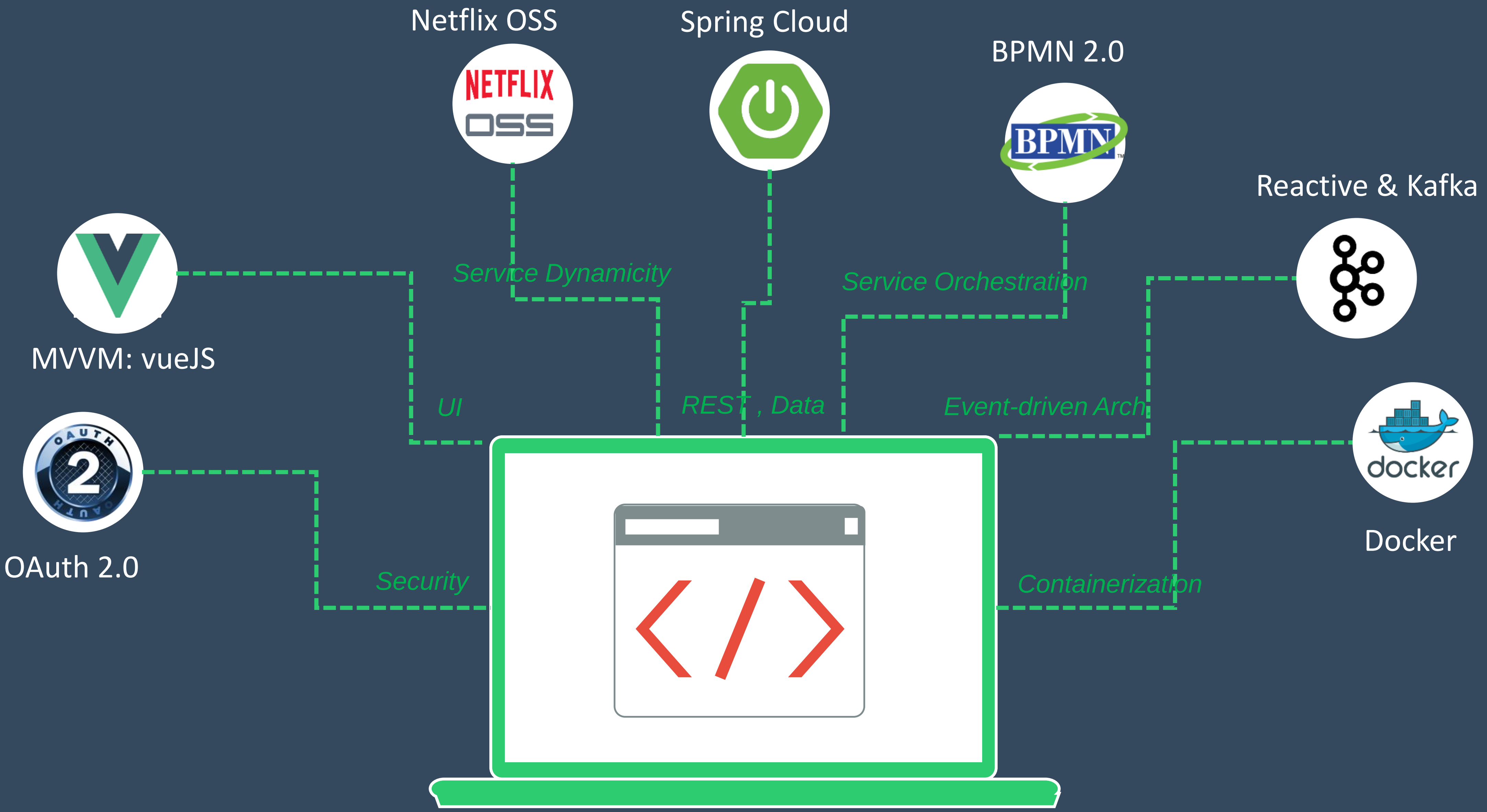
Language: Korean


Definition Name: Defense

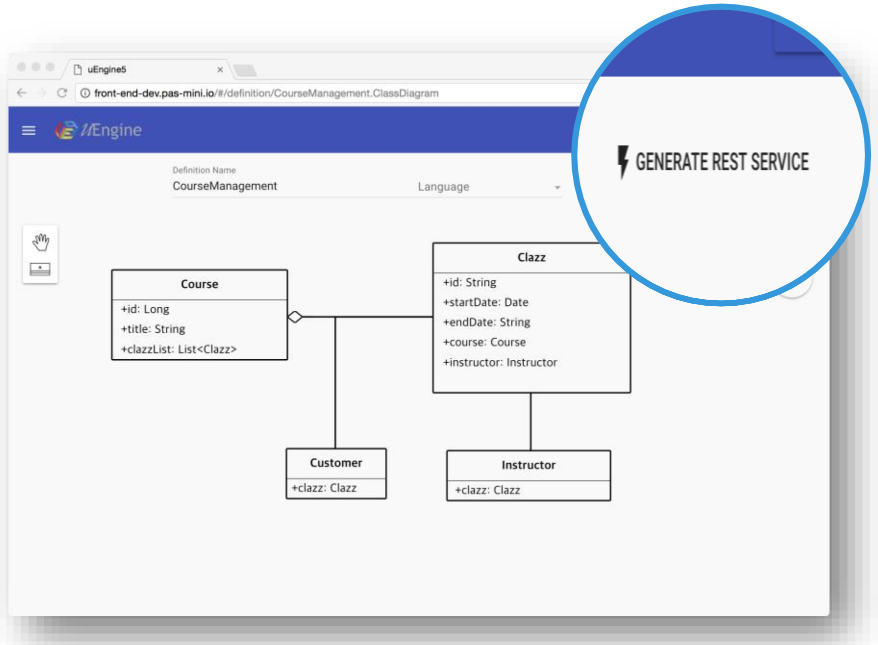


Service Implementation Phase

Output: Best mix of MSA Chassis

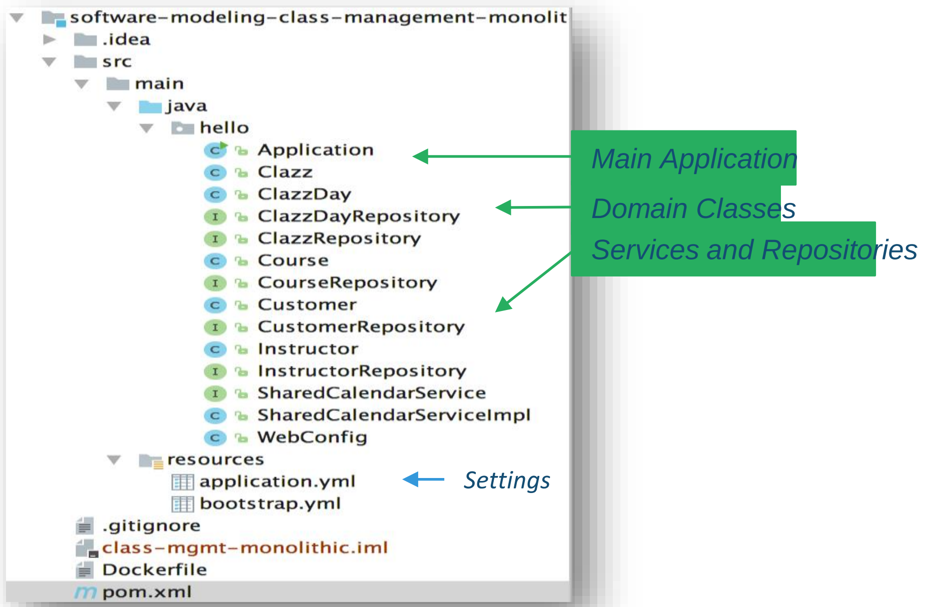


Model to Code & Code to Model



Code

Reverse Engineeri



```
▼ software-modeling-class-management-monolit
  ▶ .idea
  ▼ src
    ▼ main
      ▼ java
        ▼ hello
            Application
            Clazz
            ClazzDay
            ClazzDayRepository
            ClazzRepository
            Course
            CourseRepository
            Customer
            CustomerRepository
            Instructor
            InstructorRepository
            SharedCalendarService
            SharedCalendarServiceImpl
            WebConfig
      ▼ resources
          application.yml
          bootstrap.yml
  .gitignore
  class-mgmt-monolithic.iml
  Dockerfile
  pom.xml
```

Domain Model in UML

Entity, Repository, Service
Decomposed by Business Capabilities

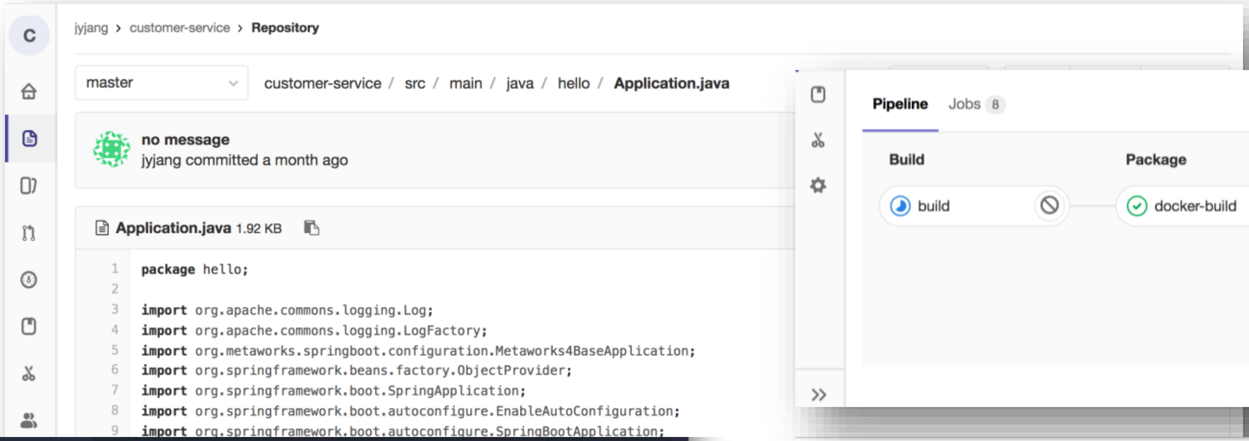
Java(SpringBoot)-based Microservice

JPA Entity, Repository, JAX-RS Service
12-Factors Cloud-Native

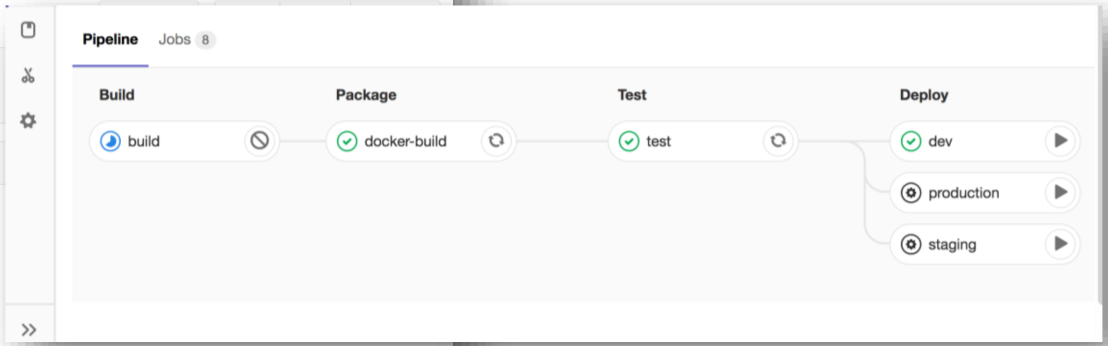
Cloud IDE & Build Pipeline

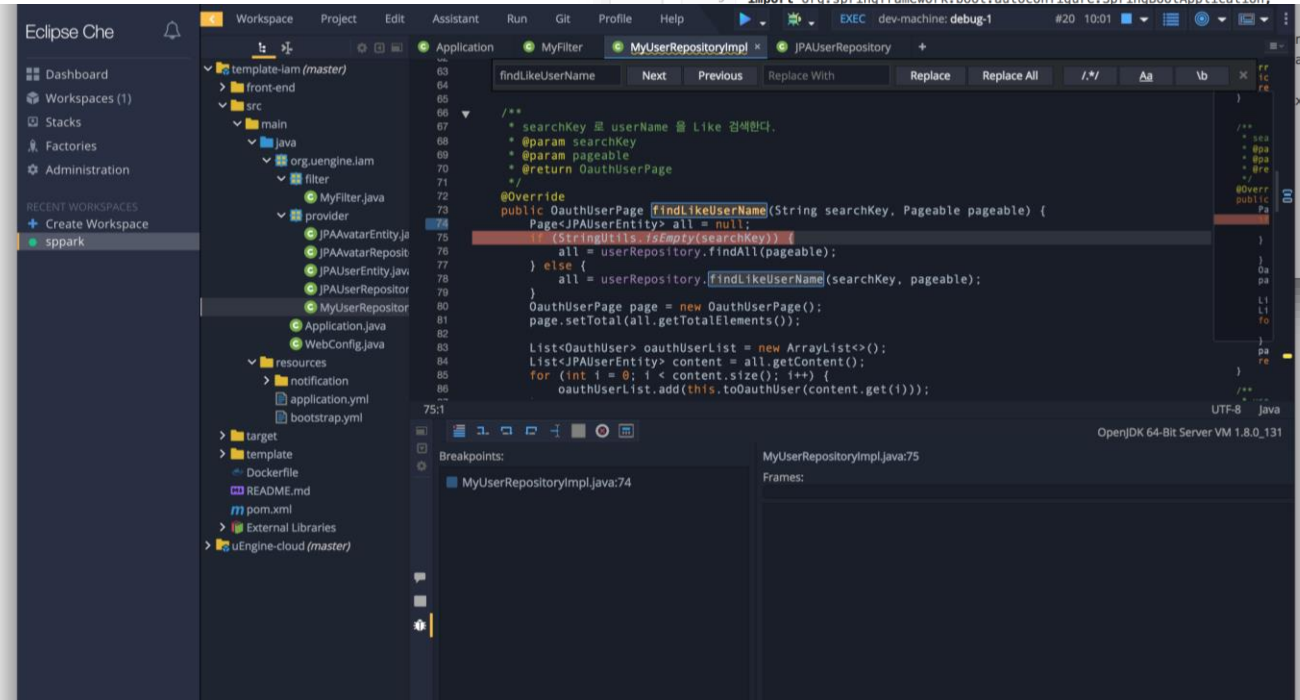
Integrated with DevOps platforms: source version control, agile collaborations, pipelines for CI/CD

Utilized OSS:
Eclise Che, GitLab, Maven

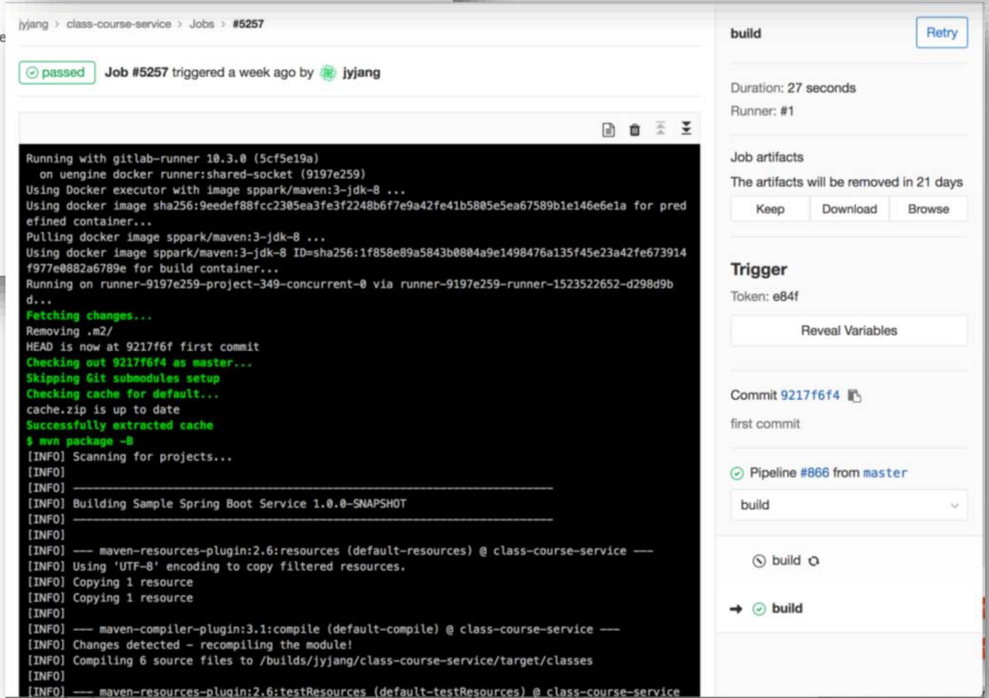

jyjang › customer-service › Repository
master
customer-service / src / main / java / hello / Application.java
no message
jyjang committed a month ago
Application.java 1.92 KB

```
package hello;

import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;
import org.metaworks.springboot.configuration.Metaworks4BaseApplication;
import org.springframework.beans.factory.ObjectProvider;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.EnableAutoConfiguration;
import org.springframework.boot.autoconfigure.SpringBootApplication;
```





```
    /**
     * searchKey 로 userName 을 Like 검색한다.
     * @param searchKey
     * @param pageable
     * @return OauthUserPage
     */
    @Override
    public OauthUserPage findLikeUserName(String searchKey, Pageable pageable) {
        Page<JPAUserEntity> all = null;
        if (StringUtils.isEmpty(searchKey)) {
            all = userRepository.findAll(pageable);
        } else {
            all = userRepository.findLikeUserName(searchKey, pageable);
        }
        OauthUserPage page = new OauthUserPage();
        page.setTotal(all.getTotalElements());

        List<OauthUser> oauthUserList = new ArrayList<>();
        List<JPAUserEntity> content = all.getContent();
        for (int i = 0; i < content.size(); i++) {
            oauthUserList.add(this.toOauthUser(content.get(i)));
```



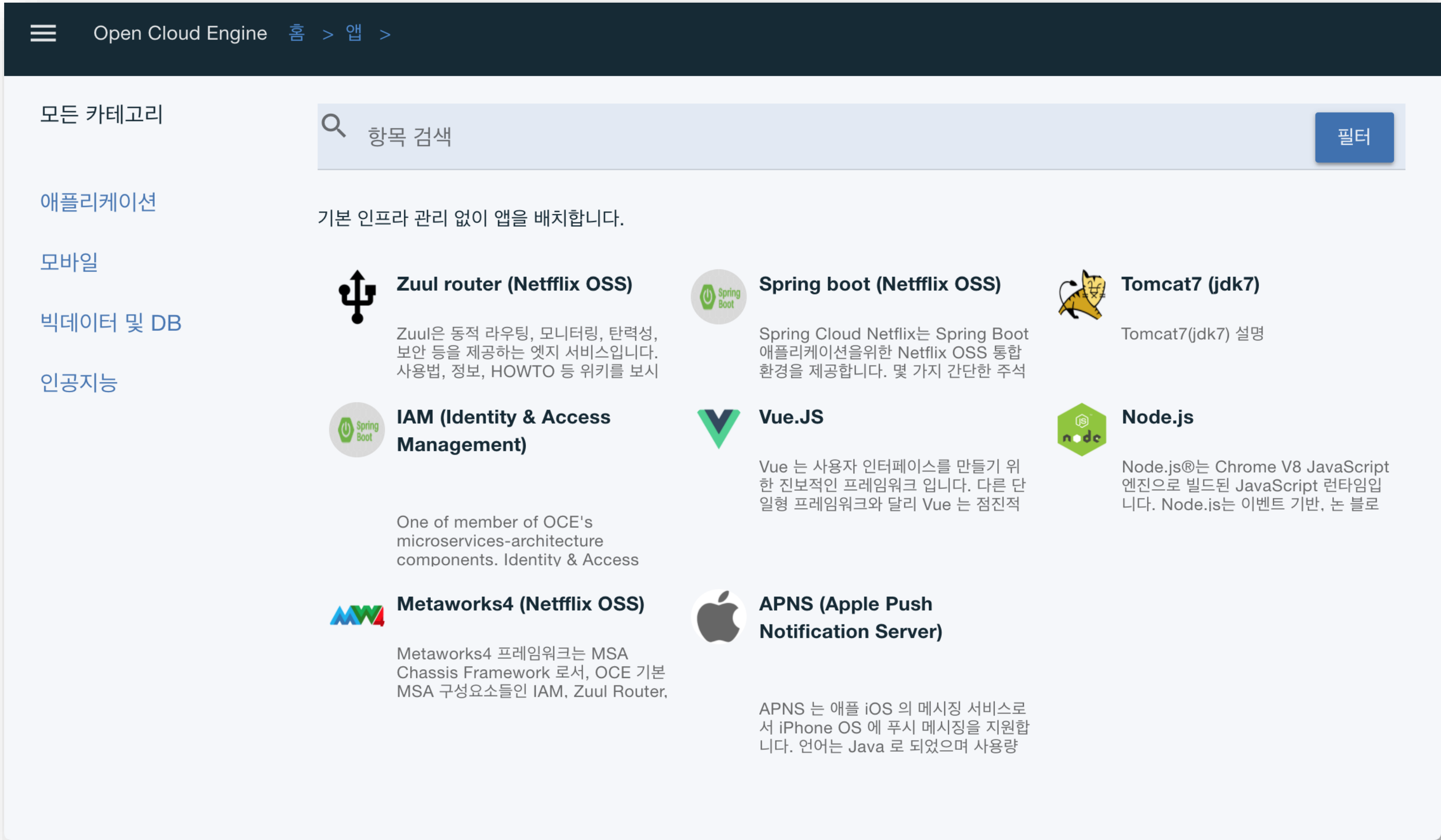
```
Running with gitlab-runner 10.3.0 (5cf5e19a)
  on uengine docker runner:shared-socket (9197e259)
Using Docker executor with image sppark/maven:3-jdk-8 ...
Using docker image sha256:9eedef88fcc2305ea3fe3f2248b6f7e9a42fe41b5805e5ea67589b1e146e6e1a for pred
efined container...
Pulling docker image sppark/maven:3-jdk-8 ...
Using docker image sppark/maven:3-jdk-8 ID=sha256:1f858e89a5843b0804a9e1498476a135f45e23a42fe673914
f977e0882a6789e for build container...
Running on runner-9197e259-project-349-concurrent-0 via runner-9197e259-runner-1523522652-d298d9b
d...
Fetching changes...
Removing .m2/
HEAD is now at 9217f6f first commit
Checking out 9217f6f4 as master...
Skipping Git submodules setup
Checking cache for default...
cache.zip is up to date
Successfully extracted cache
$ mvn package -B
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Building Sample Spring Boot Service 1.0.0-SNAPSHOT
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ class-course-service ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Copying 1 resource
[INFO] Copying 1 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ class-course-service ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 6 source files to /builds/jyjang/class-course-service/target/classes
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ class-course-service
```

Extensible Polyglot Boilerplates

Tailor for users' development environment – various application templates

(Docker File-based)

Utilized OSS: Docker



Open Cloud Engine 홈 > 앱 >

모든 카테고리

항목 검색

필터

애플리케이션

모바일

빅데이터 및 DB

인공지능

기본 인프라 관리 없이 앱을 배치합니다.

Zuul router (Netfflix OSS)

Zuul은 동적 라우팅, 모니터링, 탄력성, 보안 등을 제공하는 엣지 서비스입니다. 사용법, 정보, HOWTO 등 위키를 보시

Spring boot (Netfflix OSS)

Spring Cloud Netflix는 Spring Boot 애플리케이션을위한 Netflix OSS 통합 환경을 제공합니다. 몇 가지 간단한 주석

Tomcat7 (jdk7)

Tomcat7(jdk7) 설명

IAM (Identity & Access Management)

One of member of OCE's microservices-architecture components. Identity & Access

Vue.JS

Vue 는 사용자 인터페이스를 만들기 위한 진보적인 프레임워크 입니다. 다른 단일형 프레임워크와 달리 Vue 는 점진적

Node.js

Node.js®는 Chrome V8 JavaScript 엔진으로 빌드된 JavaScript 런타임입니다. Node.js는 이벤트 기반, 논 블로

Metaworks4 (Netfflix OSS)

Metaworks4 프레임워크는 MSA Chassis Framework 로서, OCE 기본 MSA 구성요소들인 IAM, Zuul Router,

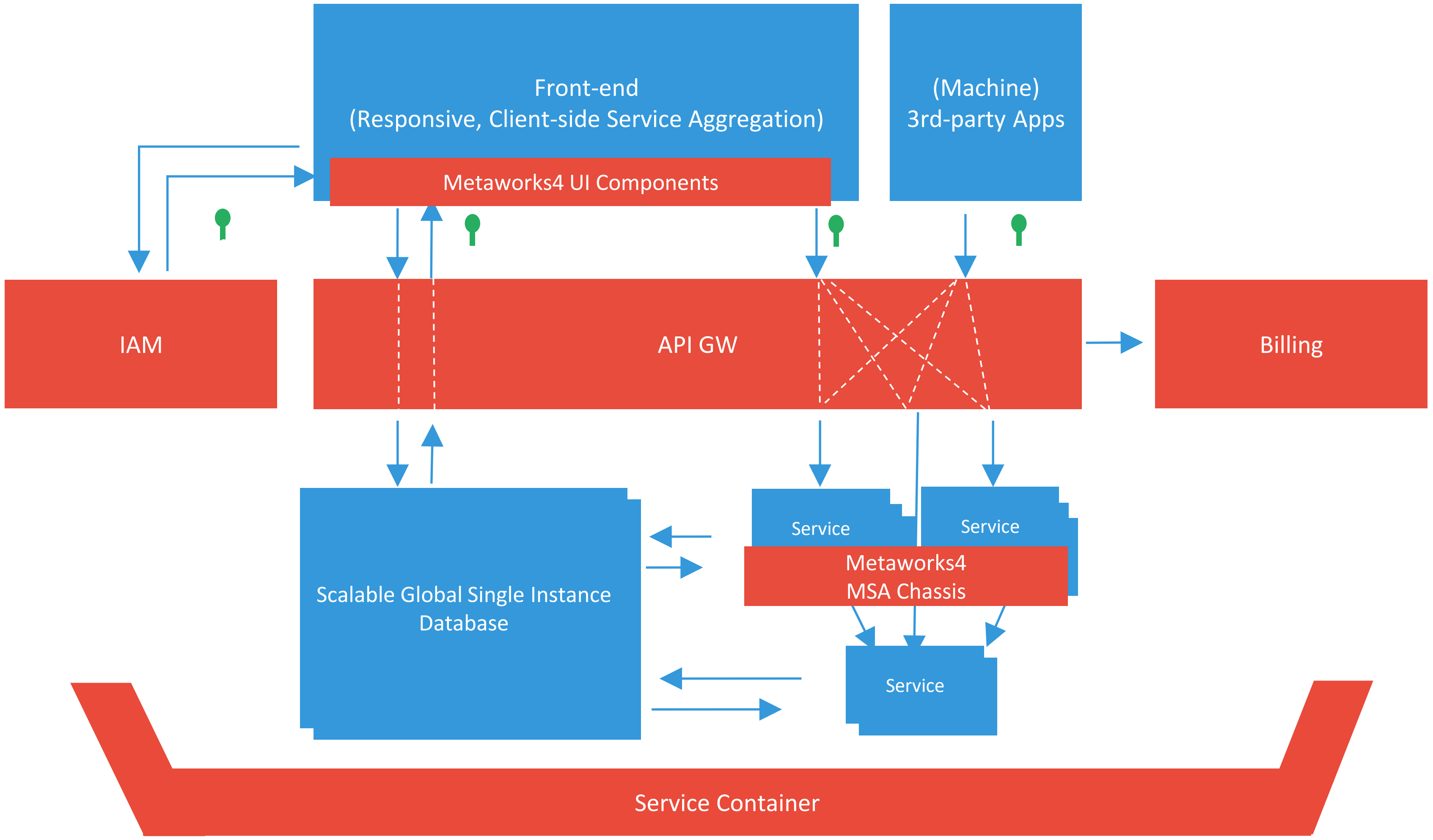
APNS (Apple Push Notification Server)

APNS 는 애플 iOS 의 메시징 서비스로서 iPhone OS 에 푸시 메시징을 지원합니다. 언어는 Java 로 되었으며 사용량

Generated Architecture

Applied MSA Design Patterns:

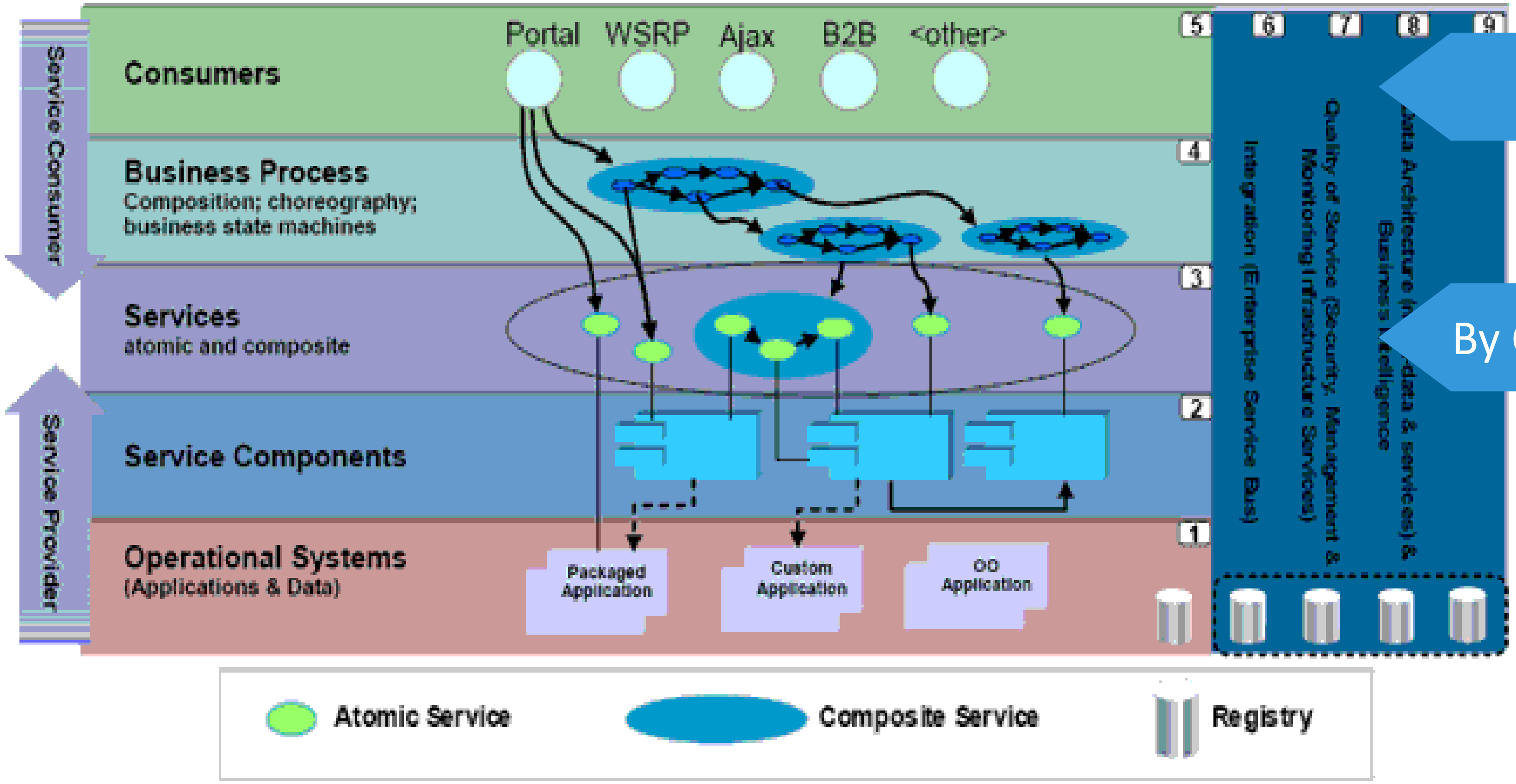
1. Multiple Instances Per Host
2. Externalized Configuration
3. API Gateway
4. Client-side discovery
5. Self-registration
6. Circuit Breaker
7. Database per Service
8. CQRS
9. Event Sourcing
10. Access Token
11. Service Contract Test
12. Log Aggregation
13. Health Checking
14. Distributed Tracing
15. Client-side UI Composition

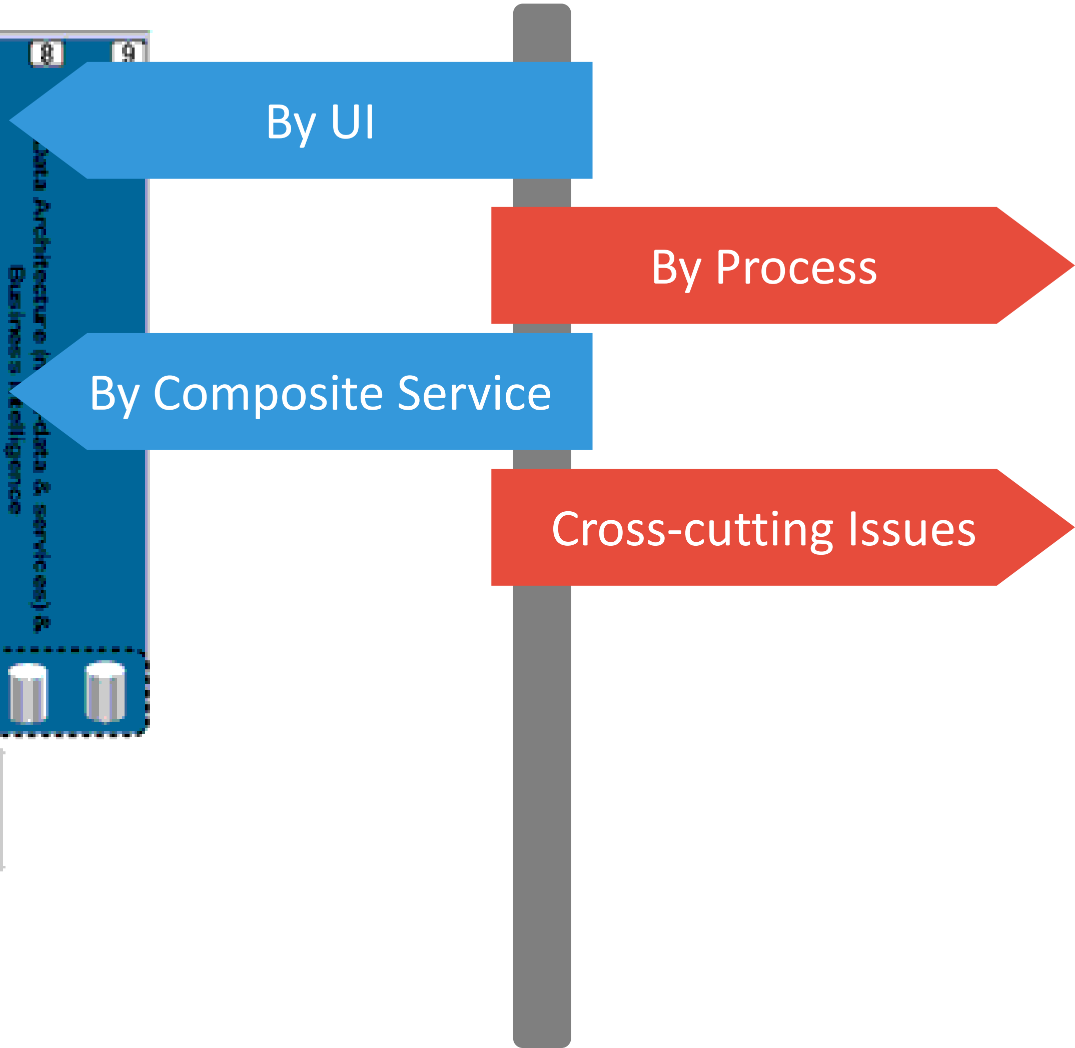


Service Mashup & Monetize Phase

uEngine

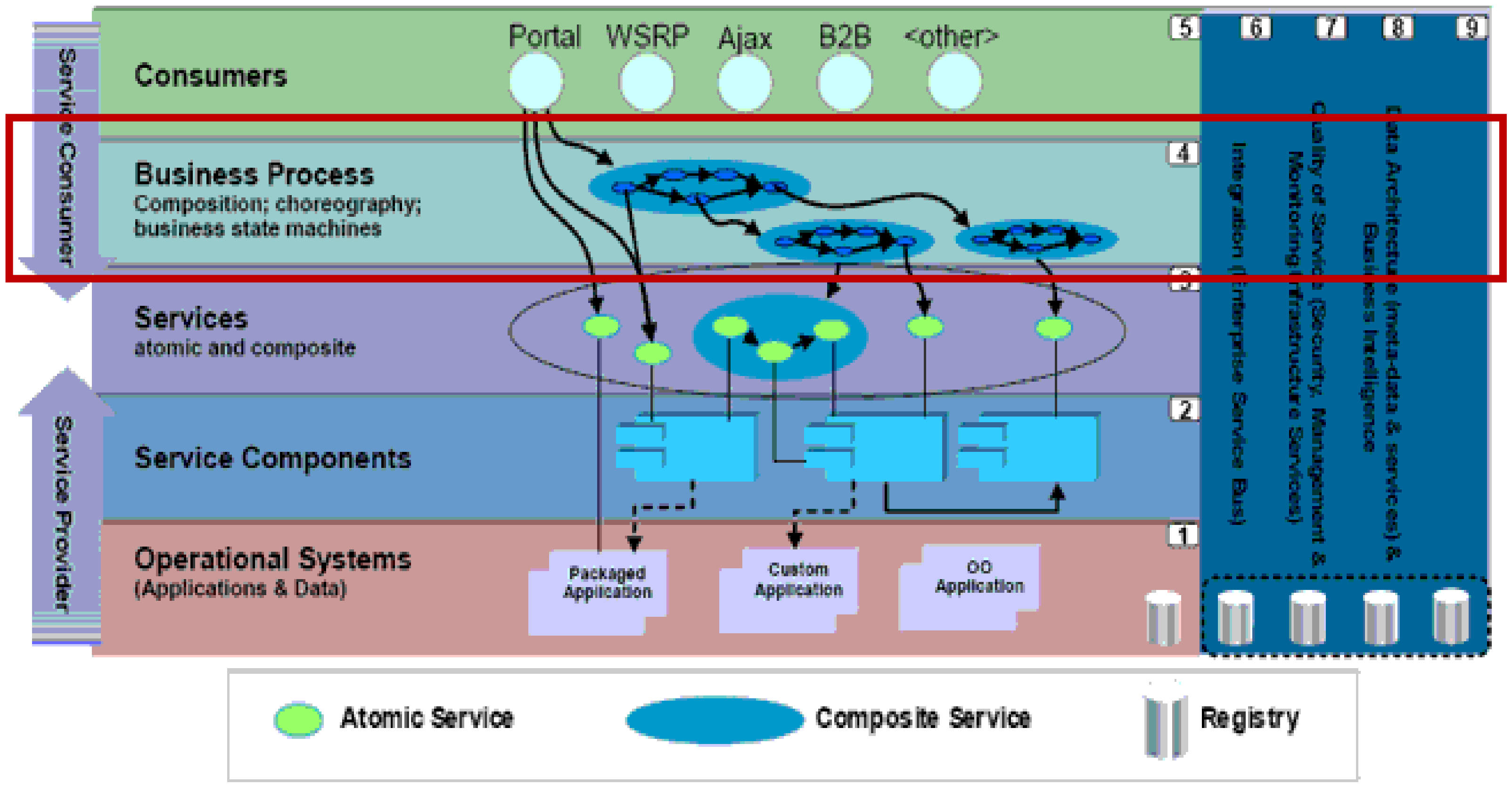
Mashup Strategies





Mashup Strategy 2:
Service Mashups by Process

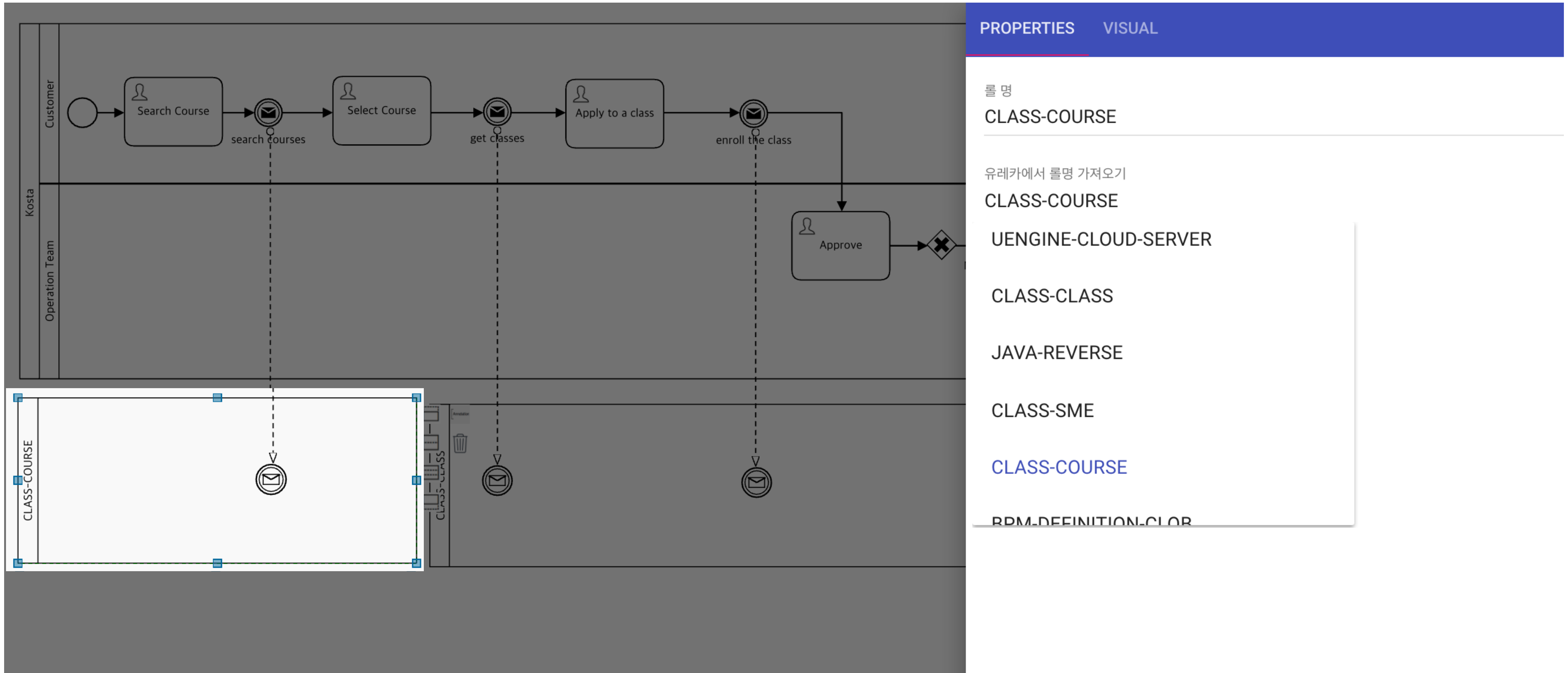
- Process based mashups use modeling tools to create new composite services or UIs to combine pre–existing services without extra development.
- Components like events, tasks by service, task by human(UI creation) can be drawn in the shape of a flow chart and can be executed as drawn.
- Services registered in Eureka registry can be called and set up with GUI, then the created process works as a new service.
- API GateWay,BPN are tools that support these.



System Integration:

Dynamic Service Binding

The instinctive idea of "**service pool**" modeling guides modeling set-ups by dynamically recognizing payload schema, connection methods and resources inside a micro-service just by **point-and-click**ing the connection target registered in Registry.

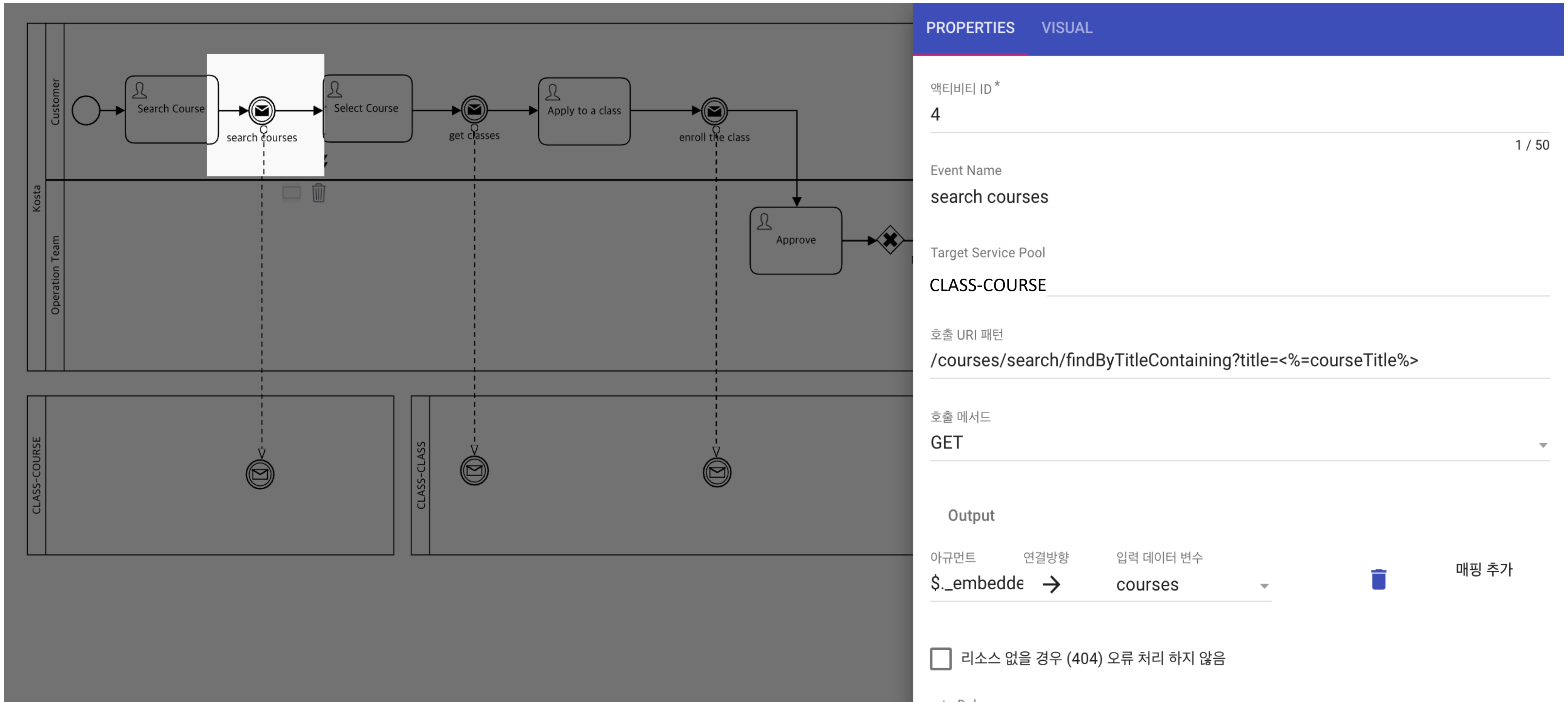


롤 명
CLASS-COURSE

유레카에서 롤명 가져오기
CLASS-COURSE

System Integration:
Dynamic REST/SOAP invocation

An event can be published through modeling. Published event can choose between **Synchronous calls like REST/WEB service and Blocking event calls like Kafka Event.** This ables analysis stage modeling to be used as an implementation.



액티비티 ID *
4
1 / 50

Event Name
search courses

Target Service Pool
CLASS-COURSE

호출 URI 패턴
/courses/search/findByTitleContaining?title=<%=courseTitle%>

호출 메서드
GET

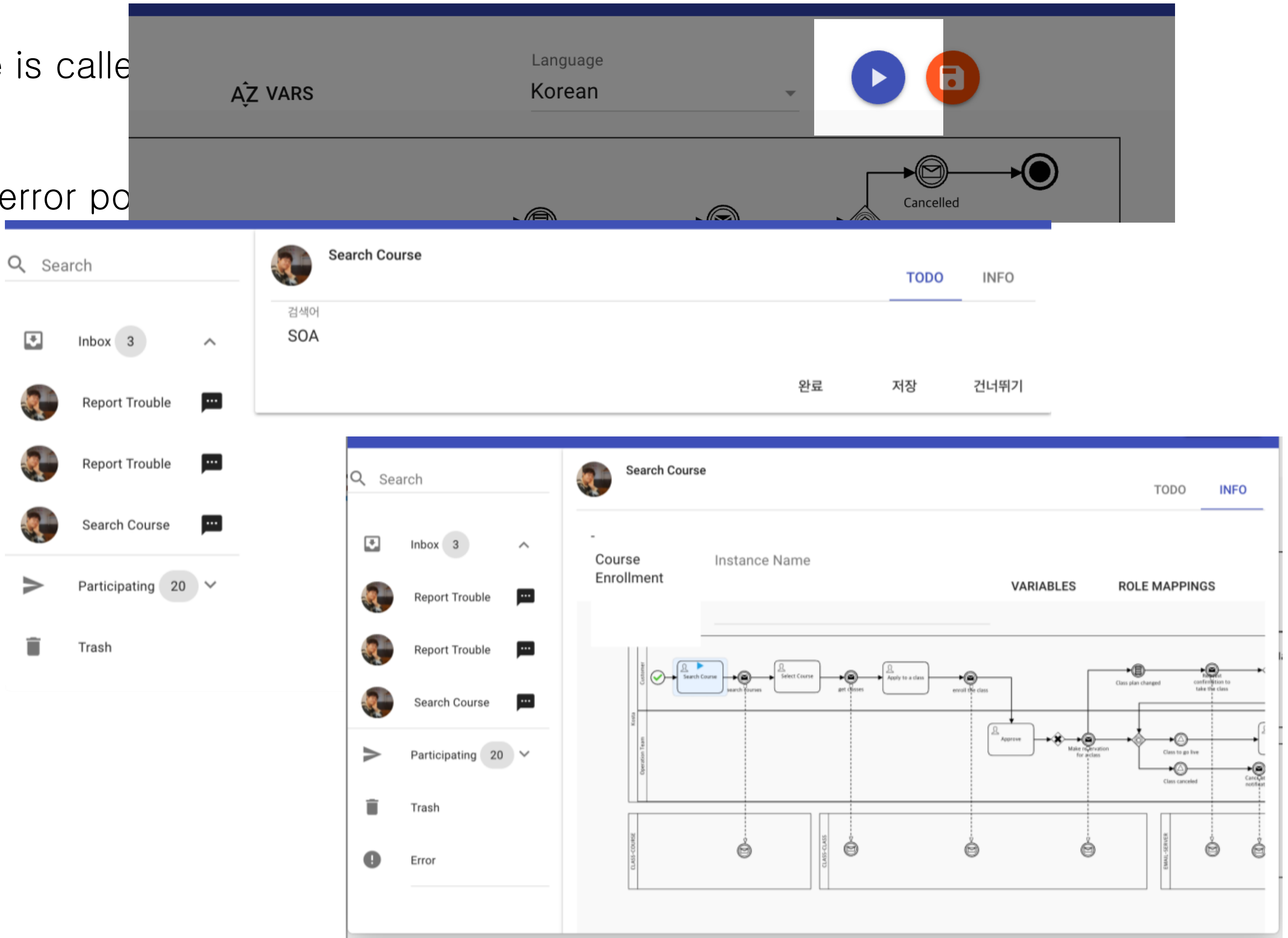
Output

아규먼트	연결방향	입력 데이터 변수
$._embedde	→	courses

☐ 리소스 없을 경우 (404) 오류 처리 하지 않음

Process Execution & Monitoring

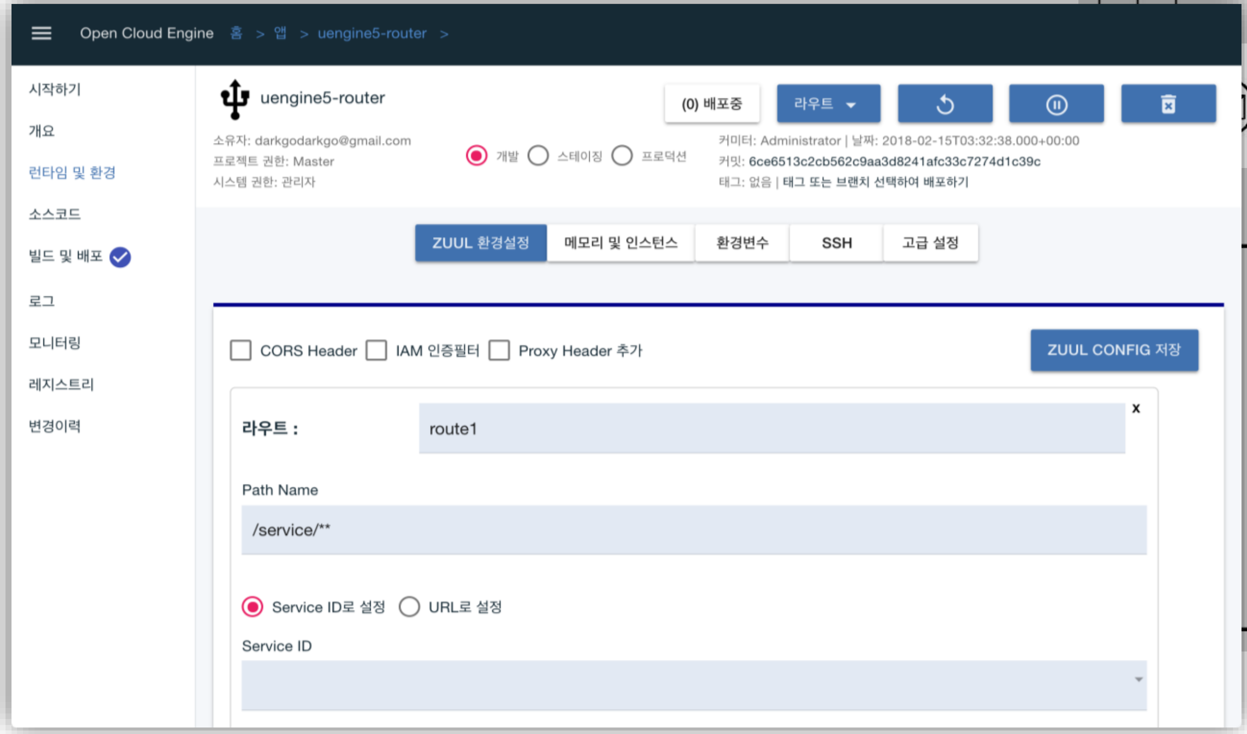
- **Simulatin g& Debugging** modeled process
- Auto-creates a page to handle human tasks
- Shows payload and results when a micro-service is calle
- Finds Error logs in the process
- **Restarts and restores to previous phase** from the error po
- **Applies it to production** after sucessful simulation



Exposing Process as REST Services

- **Security, integration, performances** about external paths to access micro services can controlled.
  APIs need for new business requirements can be created through **mash-up of existing micro service assets**.
- The created process can be exposed right away in the shape of **REST API or Kafka Event Consumer**.
- **Endpoint creation** through service endpoint designation
- **Correlation** between invoker and process instance

Used OSS:
Zuul, Kafka


Open Cloud Engine 홈 > 앱 > uengine5-router >

시작하기
개요
런타임 및 환경
소스코드
빌드 및 배포
로그
모니터링
레지스트리
변경이력

uengine5-router

(0) 배포중 | 라우트

소유자: darkgodarkgo@gmail.com
프로젝트 권한: Master
시스템 권한: 관리자

[✓ 개발] 스테이징 프로덕션

커미터: Administrator | 날짜: 2018-02-15T03:32:38.000+00:00
커밋: 6ce6513c2cb562c9aa3d8241afc33c7274d1c39c
태그: 없음 | 태그 또는 브랜치 선택하여 배포하기

ZUUL 환경설정 | 메모리 및 인스턴스 | 환경변수 | SSH | 고급 설정

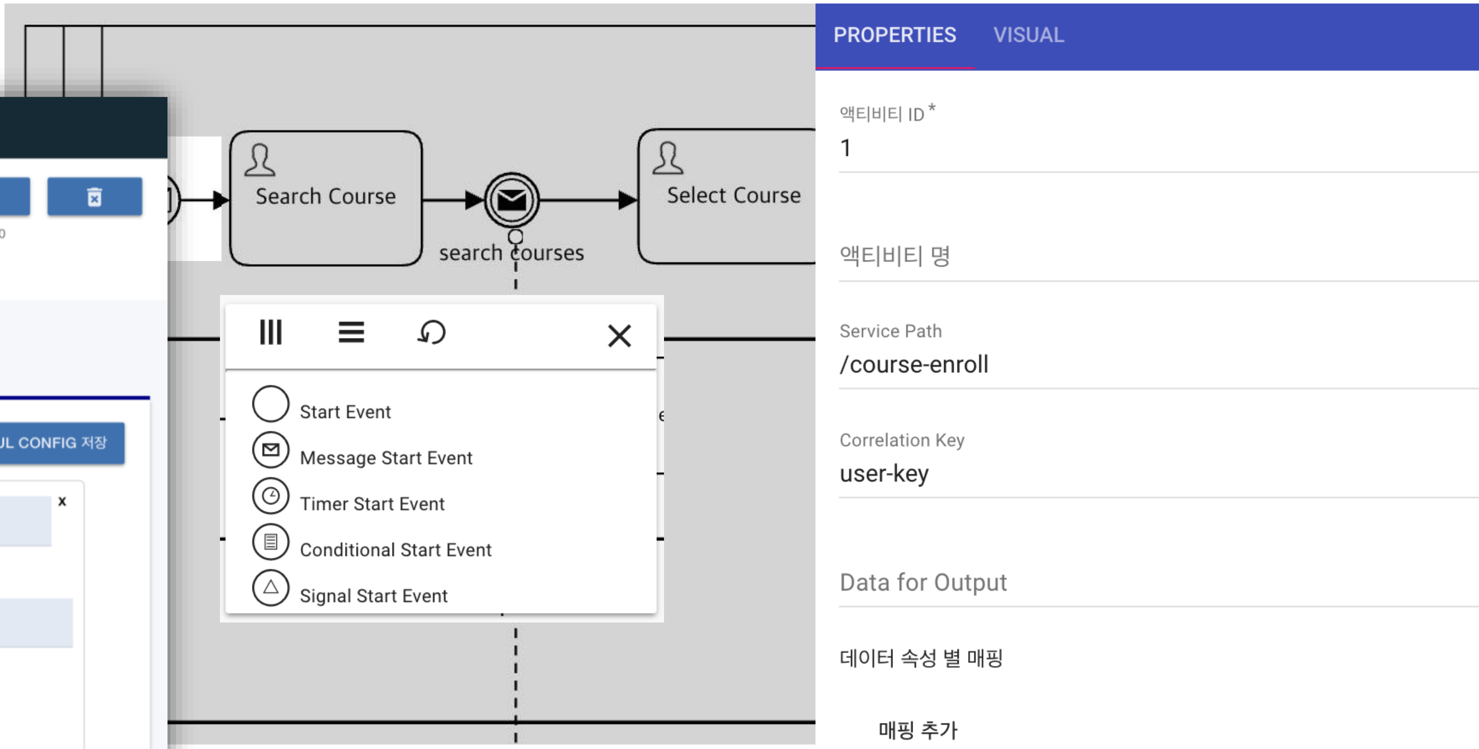
☐ CORS Header ☐ IAM 인증필터 ☐ Proxy Header 추가 — ZUUL CONFIG 저장

라우트 : route1

Path Name: /service/**

[✓ Service ID로 설정] URL로 설정

Service ID:



PROPERTIES | VISUAL

액티비티 ID*: 1

액티비티 명

Service Path: /course-enroll

Correlation Key: user-key

Data for Output

데이터 속성 별 매핑

매핑 추가

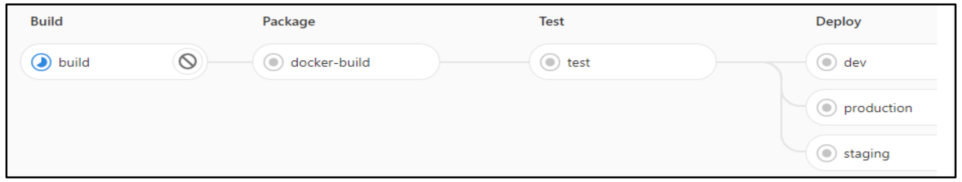
Operation Phase

UEngine

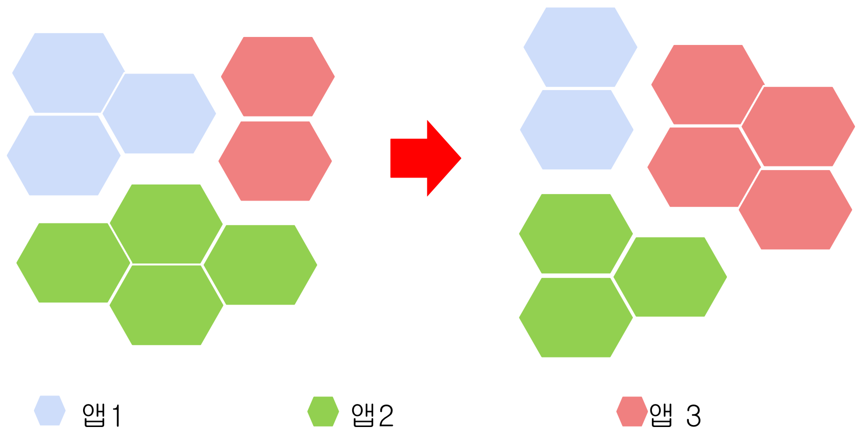
Zero-downtime Deployment & Scale

자동화된 시스템 배포 및 확장

지속적인 서비스의 출시를 위하여 배포할 서비스만을 격리하여 배포하면서도 연계된 서비스들과 동적으로 연계를 유지함(Contract-based). 이러한 배포 과정은 자동으로 기계에 의하여 수행됨

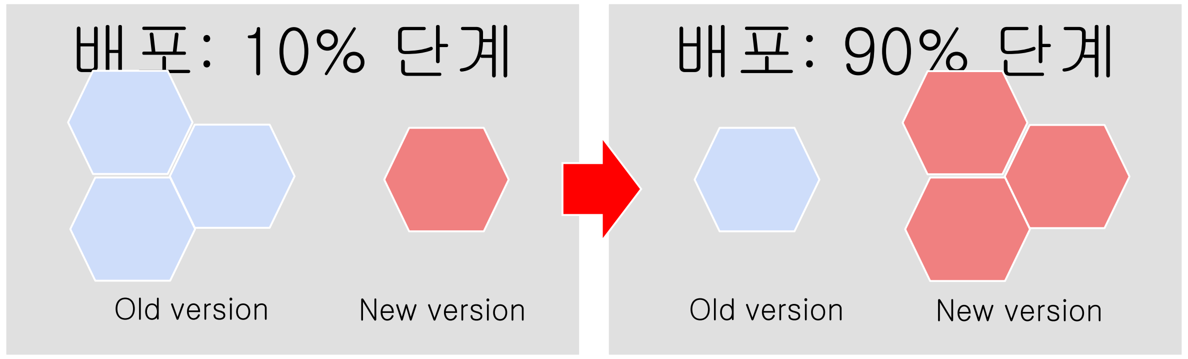


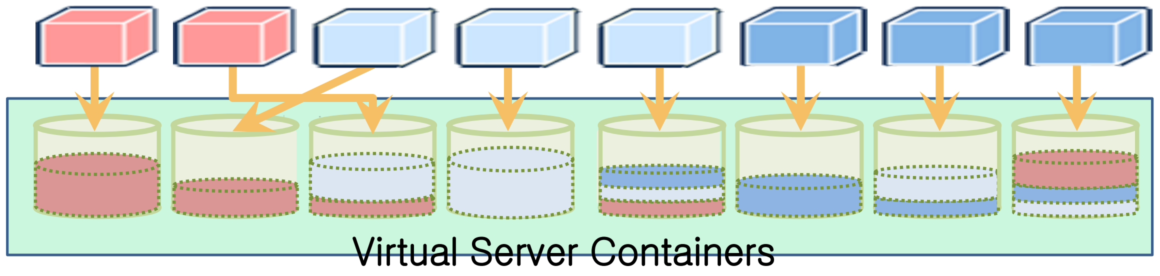
마이크로 서비스로 개발된 서비스들은 요청량에 따라 동적으로 워크로드가 분산되고 HA 구성이 이루어져 자원 가용률을 최대화하며 요청에 따른 운영노동력을 최소화 합니다.



무정지 재배포 (Zero-downtime Deployment)

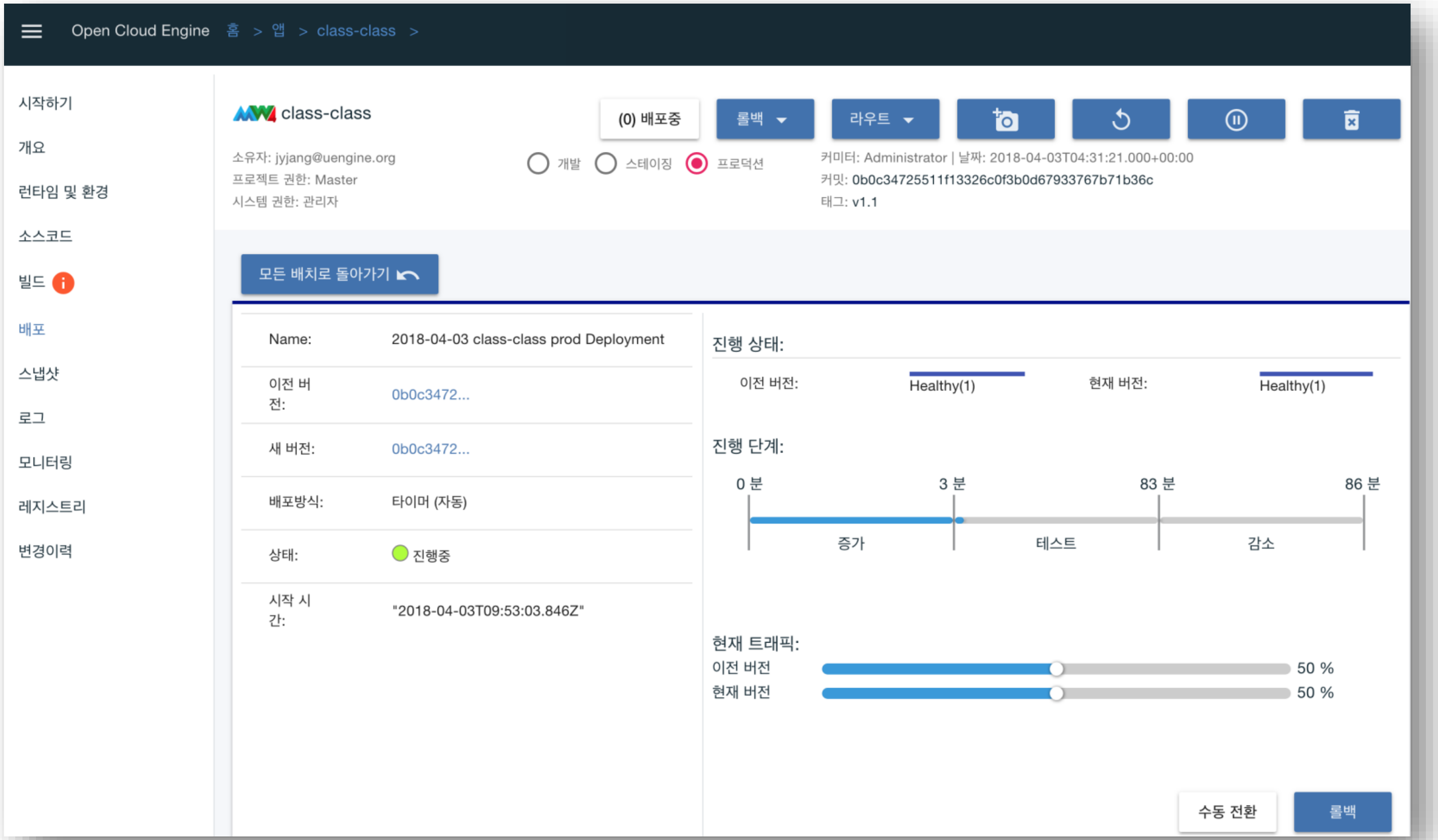
Canary Deployment 를 통하여 무정지 상태로 각 마이크로 서비스 별로 지속적인 개선과 수정이 가능하도록 함. (참고: 아마존의 경우 하루 23000회 배포를 하여 SaaS 서비스로서의 경쟁력을 내고 있음)





Self-healing and Canary Deployment

OCE provides production-grade DevOps dashboards and GUIs for controlling and managing various application deployment strategies



Open Cloud Engine 홈 > 앱 > class-class >

시작하기
개요
런타임 및 환경
소스코드
빌드
배포
스냅샷
로그
모니터링
레지스트리
변경이력

class-class

(0) 배포중 | 롤백 | 라우트

소유자: jyjang@uengine.org
프로젝트 권한: Master
시스템 권한: 관리자

개발 [ ] 스테이징 [ ] [✓ 프로덕션]

커미터: Administrator | 날짜: 2018-04-03T04:31:21.000+00:00
커밋: 0b0c34725511f13326c0f3b0d67933767b71b36c
태그: v1.1

모든 배치로 돌아가기

Name:	2018-04-03 class-class prod Deployment
이전 버전:	0b0c3472...
새 버전:	0b0c3472...
배포방식:	타이머 (자동)
상태:	진행중
시작 시간:	"2018-04-03T09:53:03.846Z"

진행 상태:
이전 버전: Healthy(1) 현재 버전: Healthy(1)

진행 단계:
0 분 3 분 83 분 86 분
증가 테스트 감소

현재 트래픽:
이전 버전 50 %
현재 버전 50 %

수동 전환 | 롤백

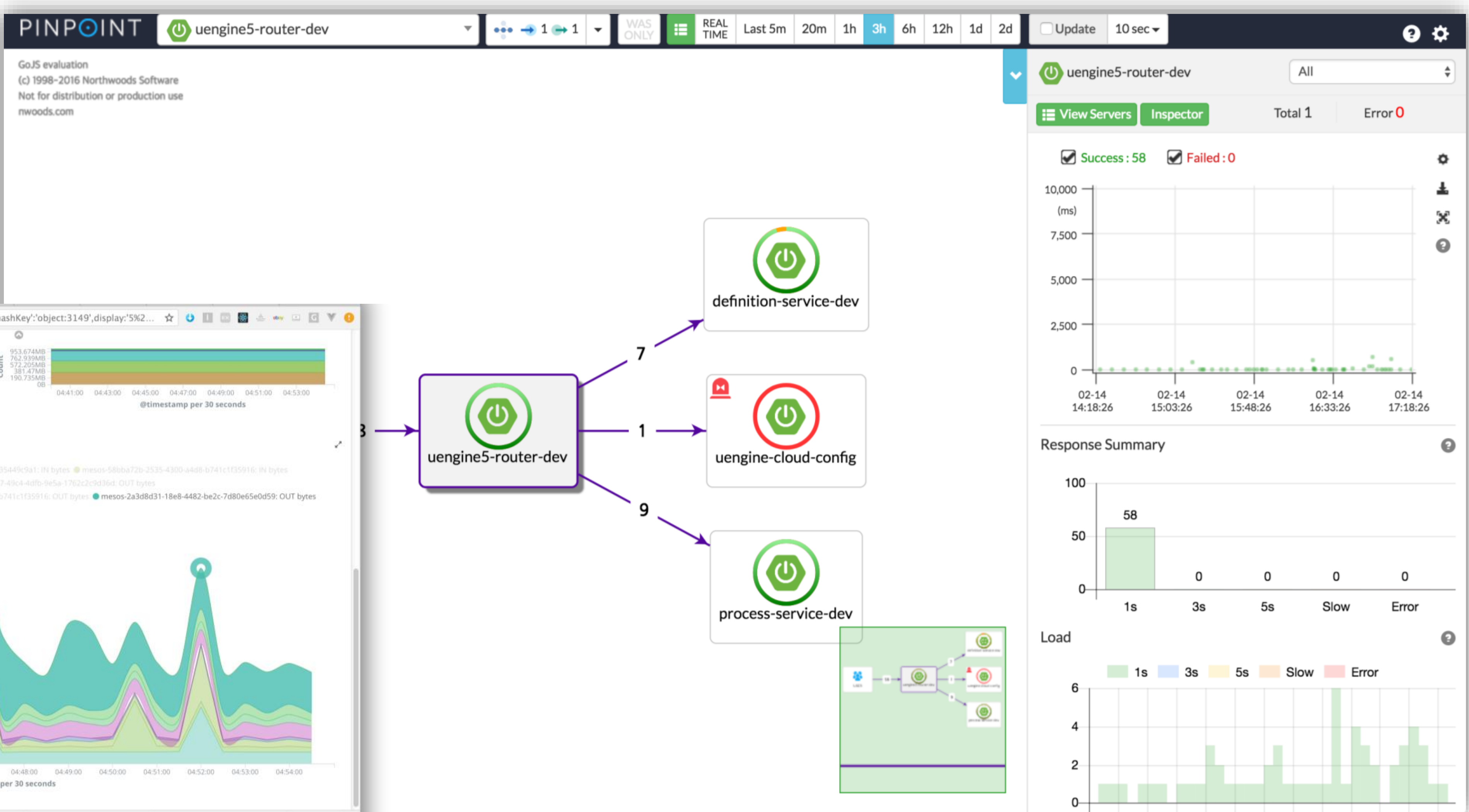
- <u>Auto-healing and scaling:</u> Using Kubernetes engine, service auto-healing and auto scaling can be done. In OCE, provides the GUI for that desired states and monitor for the actions done by kubernetes engine.
- <u>Canary deployment:</u> Using L7-layer software routers, it is ever easier for smart deployment such as Canary deploy, AB Testing, and Dark Launch (Shadow deploy). OCE provides comprehensive GUI and dashboard for controlling these deployment strategies.

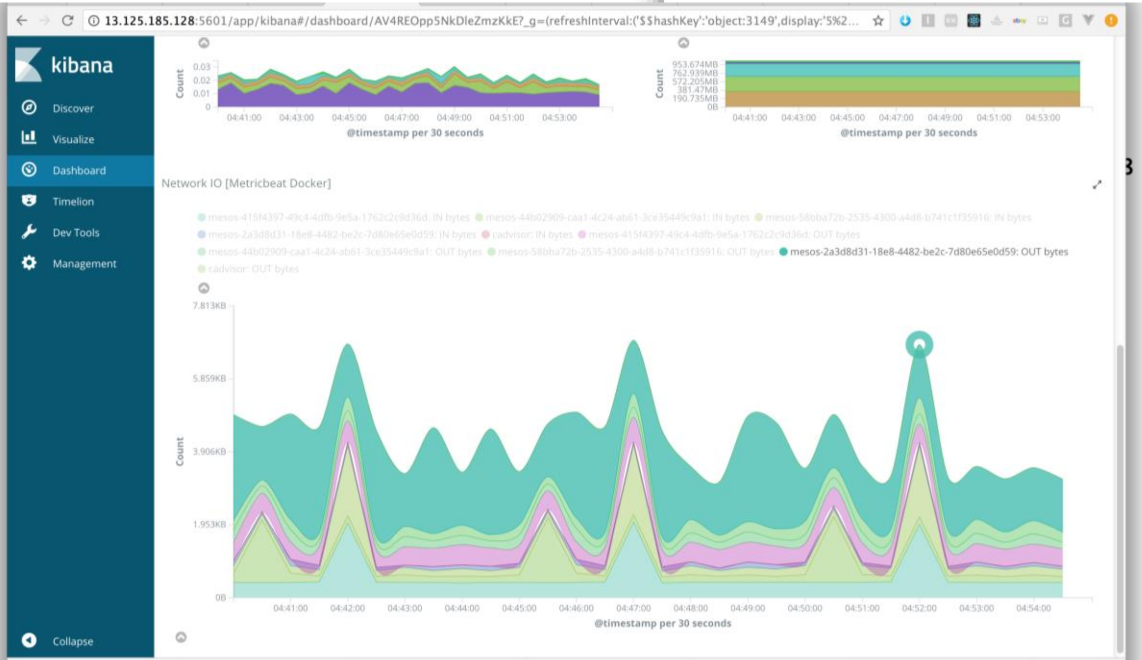
Utilized OSS:
Kubernetes, Istio

Production Debugging

Distributed tracing and monitoring measures for production debugging and reliable service operations.

Utilized OSS:
Naver Pinpoint, ELK





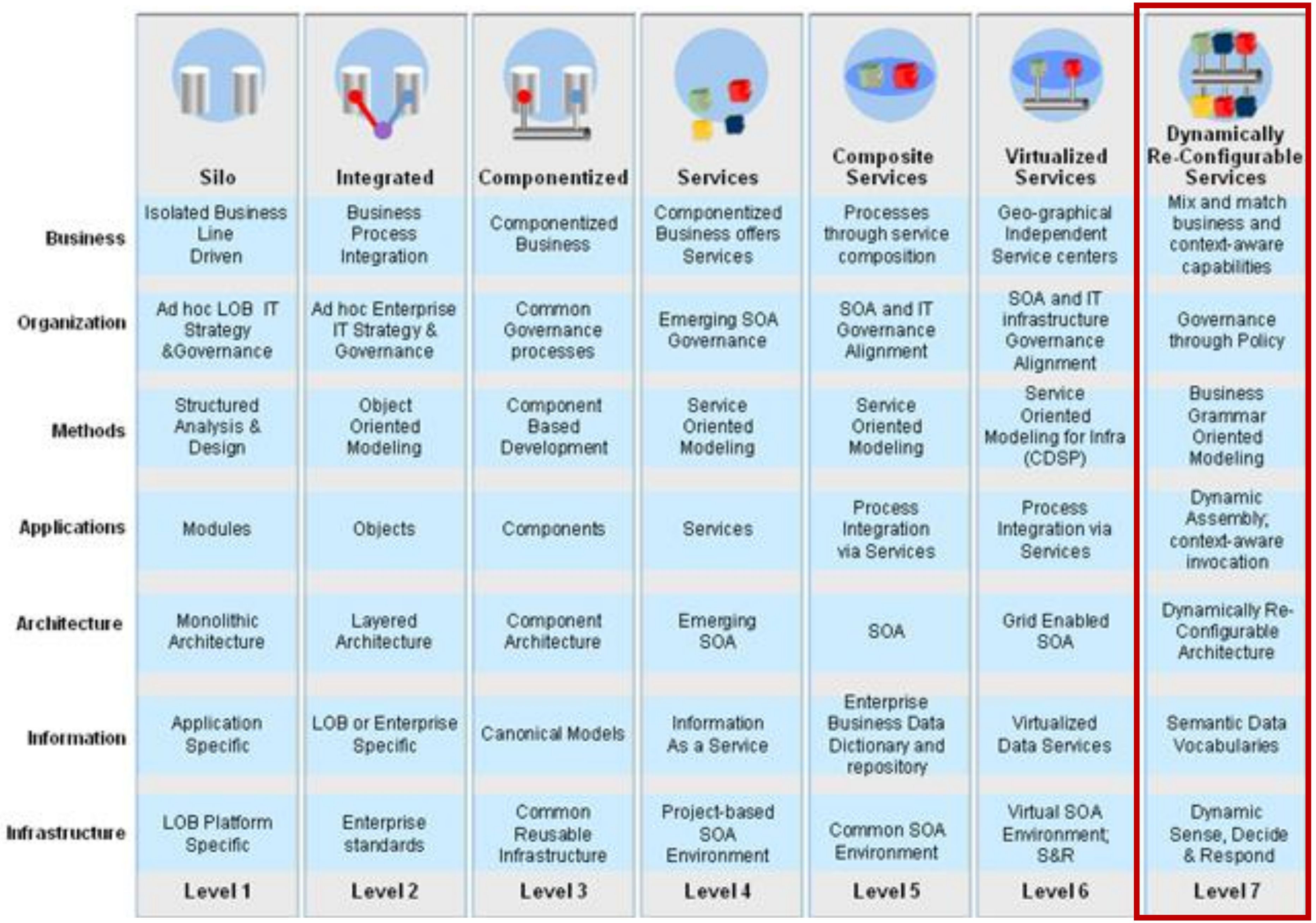
Values

uEngine

Our target:

SOA MM7:

Dynamically Re-configurable

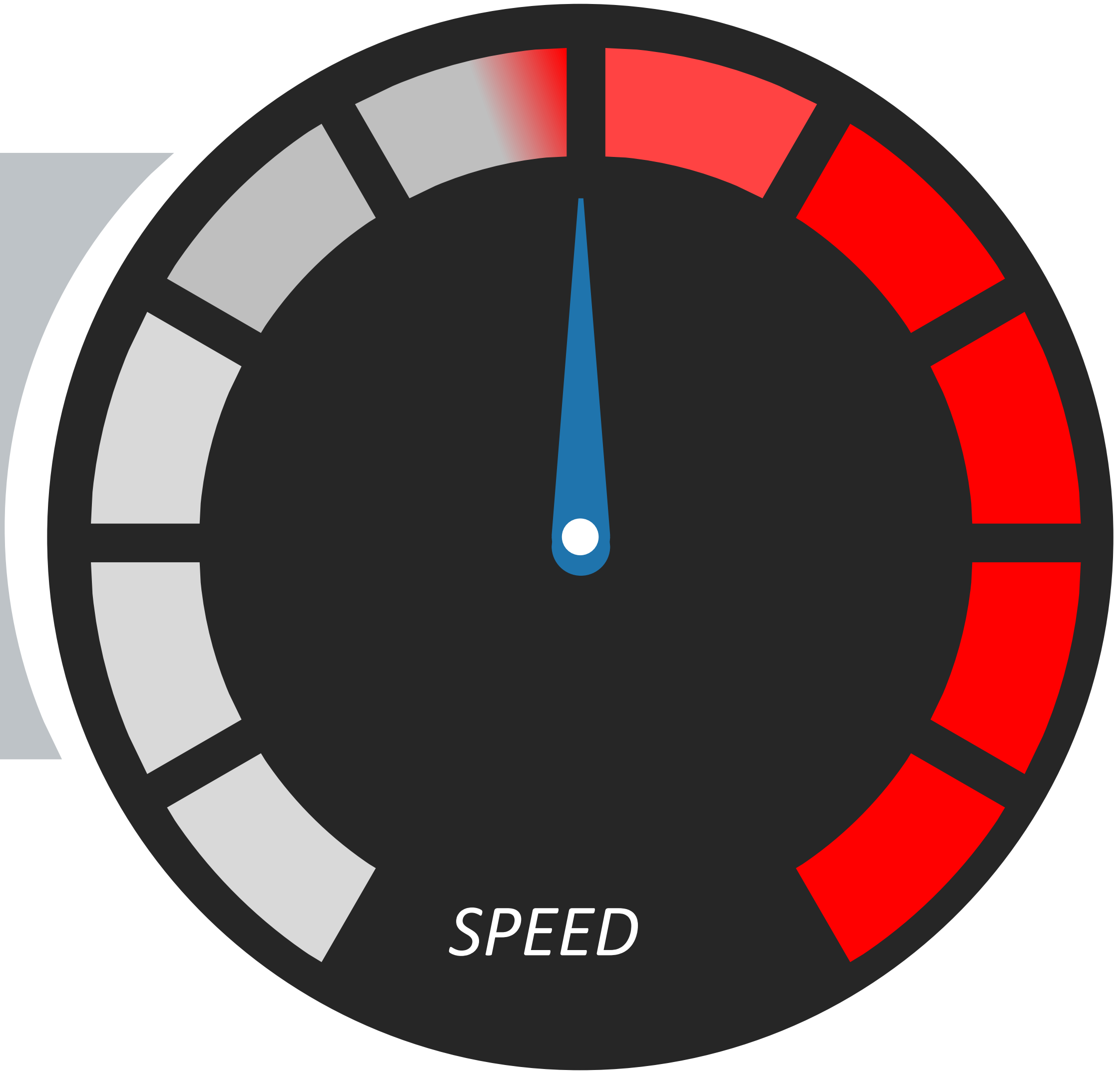


	Silo	Integrated	Componentized	Services	Composite Services	Virtualized Services	Dynamically Re-Configurable Services
Business	Isolated Business Line Driven	Business Process Integration	Componentized Business	Componentized Business offers Services	Processes through service composition	Geo-graphical Independent Service centers	Mix and match business and context-aware capabilities
Organization	Ad hoc LOB IT Strategy &Governance	Ad hoc Enterprise IT Strategy & Governance	Common Governance processes	Emerging SOA Governance	SOA and IT Governance Alignment	SOA and IT infrastructure Governance Alignment	Governance through Policy
Methods	Structured Analysis & Design	Object Oriented Modeling	Component Based Development	Service Oriented Modeling	Service Oriented Modeling	Service Oriented Modeling for Infra (CDSP)	Business Grammar Oriented Modeling
Applications	Modules	Objects	Components	Services	Process Integration via Services	Process Integration via Services	Dynamic Assembly; context-aware invocation
Architecture	Monolithic Architecture	Layered Architecture	Component Architecture	Emerging SOA	SOA	Grid Enabled SOA	Dynamically Re-Configurable Architecture
Information	Application Specific	LOB or Enterprise Specific	Canonical Models	Information As a Service	Enterprise Business Data Dictionary and repository	Virtualized Data Services	Semantic Data Vocabularies
Infrastructure	LOB Platform Specific	Enterprise standards	Common Reusable Infrastructure	Project-based SOA Environment	Common SOA Environment	Virtual SOA Environment; S&R	Dynamic Sense, Decide & Respond
	Level 1	Level 2	Level 3	Level 4	Level 5	Level 6	Level 7

Speed to digital transformation

Software development is a learning process,
working code is a side effect. -
Martin Fowler

1. Microservice analysis and design with software modeling tools (DDD modeling) and code generation
2. DevOps environment of integrated GUI support for utilizing K8S and Istio.



Company Intro

OPEN CLOUD ENGINE

uEngine BPM

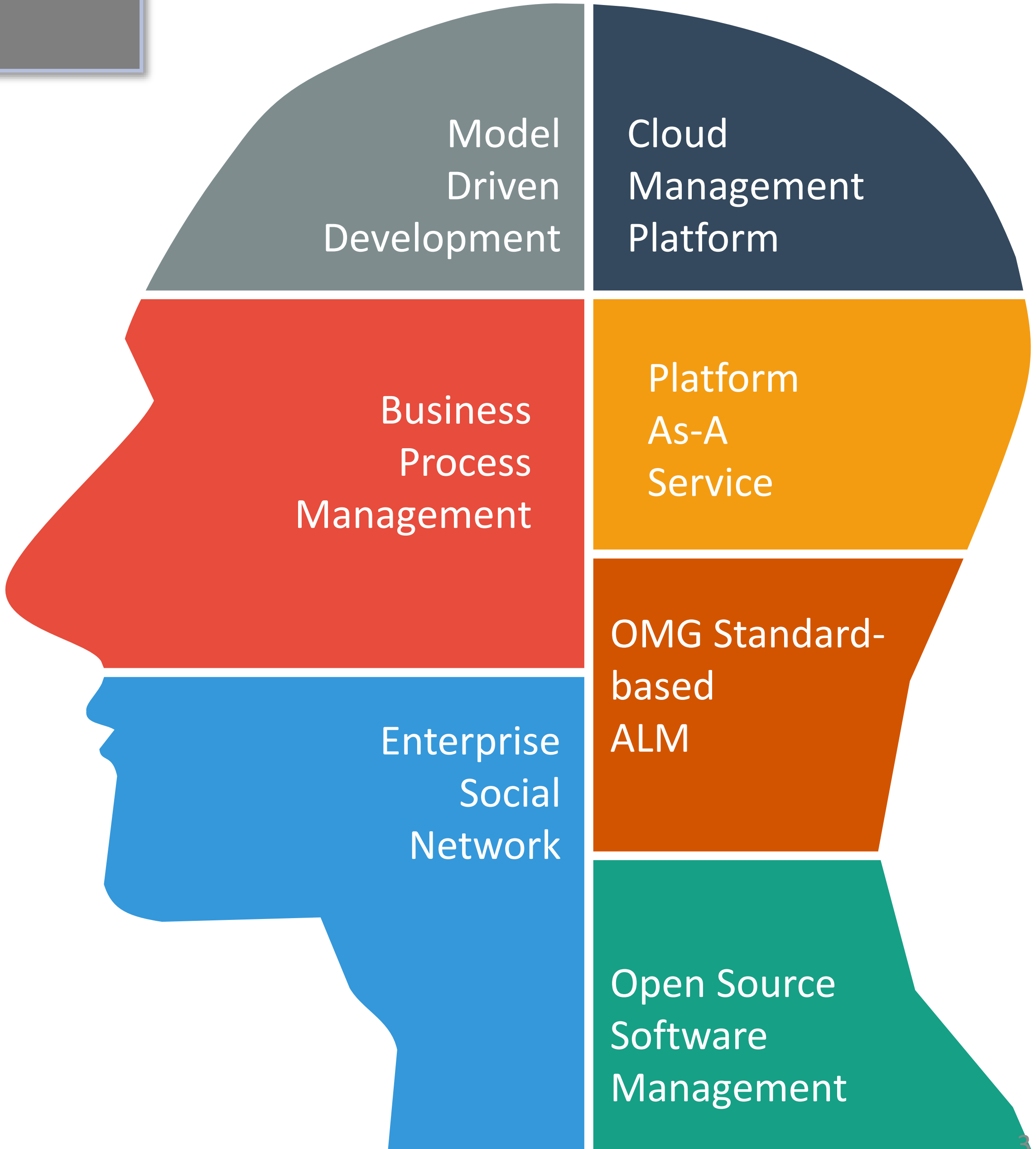
Hanwha

PROCESS CODI

IBK

SAMSUNG 삼성전기

LG Display


Model Driven Development
Cloud Management Platform
Business Process Management
Platform As-A Service
OMG Standard-based ALM
Enterprise Social Network
Open Source Software Management