Model-Driven Continuous Deployment for Quality DevOps

D. A. Tamburri, E. Di Nitto, M. Guerriero, M. Artac, T. Borovšak

DICE
Horizon 2020 Research & Innovation Action
Grant Agreement no. 644869
http://www.dice-h2020.eu
Roadmap

- Intro: The Rise of Big Data
- Research Solution: Project DICE
- DICE Meta-models
- M2M and TOSCA in DICE
Software market rapidly shifting to Big data

- Expected 32% compound annual growth rate in EU through 2016
- Just 35% of Big data projects are successful
  [CapGemini’15]
Intro: Building blocks for DIAs today

Lambda architecture

Data Source → Coordinator (Kafka) → Serving Layer → Data Store

Data Source → Coordinator (Kafka) → Speed Layer

Data Source → Coordinator (Kafka) → Batch Layer

Distributed storage

Cloud infrastructure

Data streaming

Distributed computation
Intro: example challenges!

Legacy Systems and Processes

Learning curves
Initial prototype
Risk of failure

Big Data Analytics Adoption

Spark
Cloudera
(+ others...)

Products
Services
DICE Objectives

- Tackling skill shortage and steep learning curves
  - High-degree of MDE automation via DICE tools

- Shorten time to market for DIAs
  - Push out new products, without sacrificing quality

- Decrease development and testing costs
  - Fast iterative definition of application prototypes

- Reduce number and severity of quality incidents
  - DevOps-fashioned Iterative refinement of application design
DICE incremental modeling and analysis

- **DICE Platform Independent Model (DPIM)**
  - M2M transformation
  - is implemented by

- **DICE Technology Specific Model (DTSM)**
  - M2M transformation
  - is deployed onto

- **DICE Deployment Specific Model (DDSM)**
  - M2T transformation

- **Analysis**
  - To provide sound abstractions and UML Profile for DIA

- **Analysis & Optimization**
  - To provide a sound method for:
    - DICE-based Continuous DIA Architecting
    - DICE-based Continuous DIA Deployment
DICE incremental modeling and analysis

- DICE Platform Independent Model (DPIM)
- DICE Technology Specific Model (DTSM)
- DICE Deployment Specific Model (DDSM)

Analysis

- M2M transformation is implemented by
- M2M transformation is deployed onto
- M2T transformation

Analysis & Optimization

To provide sound abstractions and UML Profile for DIA

To provide a sound method for:
- DICE-based Continuous DIA Architecting
- DICE-based Continuous DIA Deployment

In a DevOps Fashion!
Where does this come from?

Dev Area Views (Development structure, business logic, quality verification)

Ops Area Views (Deployment, process, monitoring)
Where does this come from? DICE!

Dev Area Views
(Development structure, business logic,...)

Ops Area Views
(Deployment,...)

Featuring the DICE H2020 EU Project
Where does this come from? DICE!

Deployment and Operations area of our architecture framework

Featureing the DICE H2020 EU Project
Where does this come from? DICE!

Featuring the DICE H2020 EU Project
Where does this come from? DICE!

Featuring the DICE H2020 EU Project
DICE Meta-models - DTSM

*Metamodels here are illustrative mockups - You can think of using ADOxx to reproduce them!
DICE Meta-models* - DTSM Core

1. Common to all DICE tech-packs;
2. Distinguishing analysable elements;
3. Distinguishing properties;
4. Simplified property definition;

*Metamodels here are illustrative mockups - You can think of using ADOxx to reproduce them!
DICE Meta-models – DDSM\textsuperscript{[5]}

MODACloudsML4DICE

MODA4DICE:: HadoopMR

MODA4DICE:: Oryx2

DDSIMetamodels

MODACloudsML Specification

DICER M2T Tool

Automated transformation

TOSCA4DICE Specification

TOSCA Blueprint *.yaml

DDSIMetamodel
Where does this come from? DICE!
"Topology and Orchestration Specification for Cloud Applications"

1. OASIS Standard;
2. Cloud Infrastructure-as-code language;

1. Type-strength Intent modeling;
2. In DICE? Big-Data TOSCA “profile”;

Service template
“TOSCA is used to describe the **deployment view** for cloud applications”

- **Nodes** - are the resources or components that will be materialized or consumed in the deployment topology
- **Relationships** express the dependencies between the nodes (not the traffic flow)

- **Node templates** to describe components in the topology structure
- **Relationship templates** to describe connections, dependencies, deployment ordering

**Groups**
Create Logical, Management or Policy groups (1 or more nodes)

**Requirement - Capability**
Relationships can be customized to match specific source requirements to target capabilities
With M2M a TOSCA blueprint happens!

```
"storm_nimbus_host": {  
"relationships": [
  {
"type": "cloudify.openstack.server_connected_to_floating_ip",
"target": "storm_floating_ip"
  },
  {
"type": "cloudify.openstack.server_connected_to_security_group",
"target": "storm_security_group"
  }
],
"type": "dice.medium_host"
},

"zookeeper": {
  "relationships": [
  {
"type": "cloudify.relationships.contained_in",
"target": "zookeeper_host",
"source_interfaces": {
  "cloudify.interfaces.relationship_lifecycle": {
"preconfigure": "scripts/connect_zookeeper_servers.sh"
  }
  }
  }
],
"type": "dice.medium_host"
}
```

And after that... DevOps!

In a DevOps Fashion!
4. Deploy improved-by-verification architecture
5. Monitor and continuously enhance/architect/deploy DIA
Conclusions, take-home messages

- MDE+TOSCA = DevOps Synergy!

- Big-Data and TOSCA are fit for each other...
  - but technologies are not mature, yet!
Bibliography


