Microservices and Choreographies | The SMAAll Project

Saverio Giallorenzo
EIT Digital Project

- **Smart Mobility for All**

- **Main Objective**: creation of a global market of services for transportation;

- **Project Partners**: University of Bologna, FBK@Trento, Aalto University / Forum Virium@Helsinki

- **Business Partners**: Reply S.p.A., Emilia-Romagna Region, Trento Municipality, MaaS.fi/MaaS.global, …
Microservices

Architectures
A SMAll Pilot | BusCheck
Microservices:
- cohesiveness & minimality
- fine-grained
- access policies;
- scalability;
- deployment (multistage continuous integration!).
The First Language for Microservices
The Jolie Way

Services communicate through **ports**.
*Ports* give access to an **interface**.
An interface is a set of **operations**.
An **output port** is used to invoke interfaces exposed by other services.
An **input port** is used to expose an interface.
The Jolie Way

- Services communicate through ports.
- Ports give access to an interface.
- An interface is a set of operations.
- An output port is used to invoke interfaces exposed by other services.
- An input port is used to expose an interface.

Deployment:
```java
OutputPort B {
  Location: socket://myaddress.org:80
  Protocol: json/rpc (xml/rpc, …)
  Interface: sendNumber( int )
}
```

Behaviour:
```java
main {
  sendNumber@B( 5 )
}
```
The Jolie Way

interface PositionAPIInterface {
    OneWay: passPosition( PositionType )
}

include "PositionInterface.iol"
outputPort Tracker {
    Location: "socket://Tracker.com:80"
    Protocol: json/rpc
    Interfaces: PositionAPIInterface
}

main {
    passPosition @ Tracker ( gps )
}

include "PositionInterface.iol"
inputPort Tracker {
    Location: "socket://Tracker.com:80"
    Protocol: json/rpc
    Interfaces: PositionAPIInterface
}

main {
    passPosition ( gps )
}

Bus Agency

Tracker
The Jolie Way

interface PositionAPIInterface {
    OneWay: passPosition( PositionType )
}

include "PositionInterface.iol"
outputPort Tracker {
    Location: "socket://Tracker.com:80"
    Protocol: json/rpc
    Interfaces: PositionAPIInterface
}
main {
    passPosition @ Tracker( gps )
}

include "PositionInterface.iol"
inputPort Tracker {
    Location: "socket://Tracker.com:80"
    Protocol: json/rpc
    Interfaces: PositionAPIInterface
}
main {
    passPosition( gps )
}
Choreographies

Protocols,
Correct implementations
include `getInput` from Region.AdministrationConsole
include `insertDelay` from Region.TrackingDatabase
include `hasNextStop` from UniBo.TrackingScheduler
include `calculateDelay` from UniBo.DelayCalculator
include `getBusSchedule` from BusAgency.BusScheduling
include `getPosition` from BusAgency.GpsAPI

preamble{ starter: Admin }

aioc {
  line@Admin = getInput( "Insert line to track" );
  {
    setLine: Admin( line ) -> DB( line )
    |
    setLine: Admin( line ) -> BusAgency( line )
  };
  schdl@BusAgency = getBusSchedule( line );
  passSchdl: BusAgency( schdl ) -> Tracker( schdl );
  hasNext@Tracker = hasNextStop( schdl );
  while( hasNext )@Tracker {
    gps@BusAgency = getPosition( line );
    passPosition: BusAgency( gps ) -> Tracker( gps );
    delay@Tracker = calculateDelay( schdl, gps );
    storeDelay: Tracker( delay ) -> DB( delay );
    {
      _@DB = insertDelay( line, delay )
      |
      hasNext@Tracker = hasNextStop( sched )
    }
  }
}
A SMAll Pilot | BusCheck Architecture

Legend
- M: Microservice
- S: External Service
- Interaction

Regional Government
- Tracking Database
- Review Panel
- Administration Console

Università di Bologna
- Delay Calculator
- Tracking Scheduler

Bus Agency
- GPS API
- Bus Scheduling

Database
Administrator
Tracker
Bus Agency
Correctness by design and by construction

Choreography \[ \xrightarrow{EPP} \] Endpoint Projection

(Correct by design) \[ \xrightarrow{EPP} \] (Correct by construction)

```
line@Admin = getInput( "Insert line to track" );
{ setLine: Admin( line ) -> DB( line ) } | setLine: Admin( line ) -> BusAgency( line ) }
```

```
Admin
getOption@UI("Insert line to track")( line );
{ setLine@Database( line )
  | setLine@BusAgency( line ) };...
```

```
Database
setLine( line );
...
```

```
BusAgency
setLine( line );
...
```
Time for discussion!
MOBILITY AS A SERVICE

Bundling
Routing
Service Promise
Payments
Customer Experience

TRANSPORTATION PROVIDERS

INFRASTRUCTURE

 saverio.giallorenzo@gmail.com | DISI at Unibo | Sophia Antipolis | Evaluation des Projects
Microservices

- Cohesiveness & Minimality
- API design is paramount;
- Partition of work and parallel development;
- Breakdown of complexity into “simple” and specialised services;
- Integrate ESB-like functionalities;
- Lightweight and human-oriented protocols (REST, JSON, etc.).
Triggers compilation to Jolie