A Jolie based platform for speeding-up the digitalization of system integration processes

Claudio Guidi, Balint Maschio

italianaSoftware s.r.l

Microservices Community 2019

19-22/02/2019, Dortmund

Digitalisation as a competitive challenge

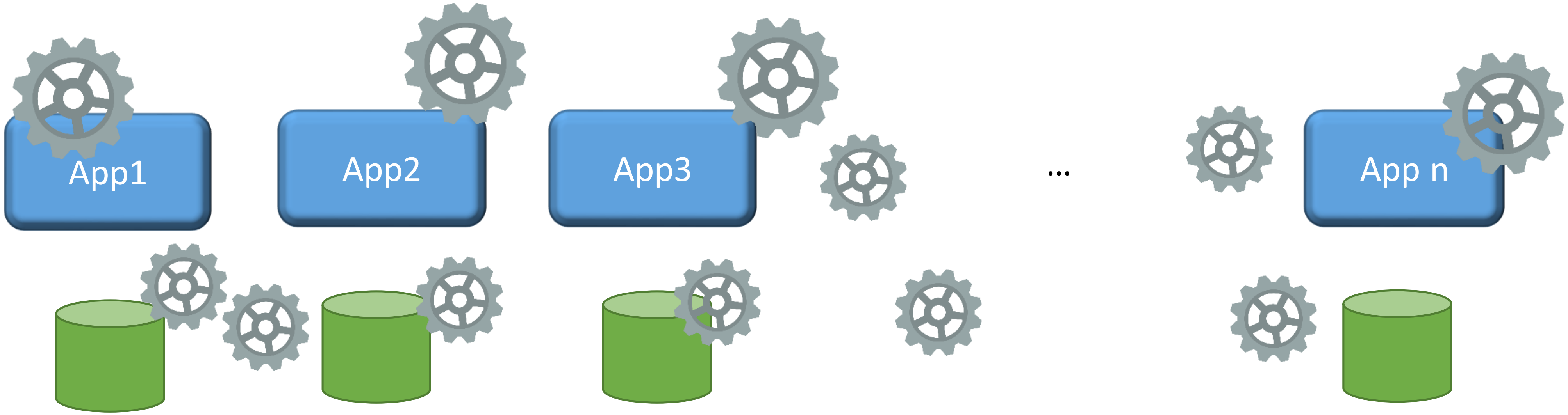
The competitive challenge

- The digitalisation demand is growing in order to increase efficiency
  - any size of companies need to digitalize their own business processes
  - interoperability among companies and among organization units within the same company can be achieved by means of digital process integration
  - Quite all the aspects of the business can be digitalized, from human resources management to production lines

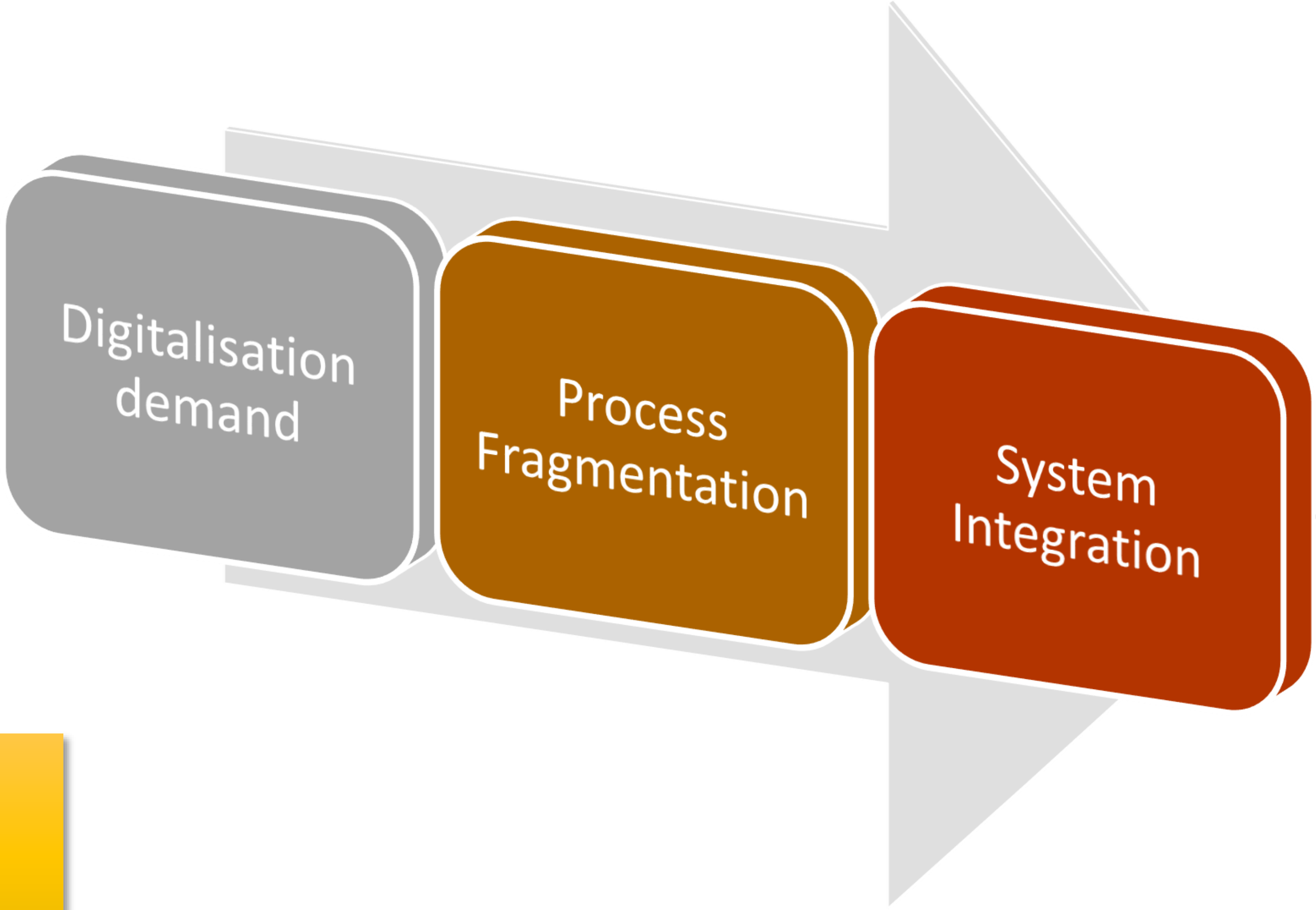
Industry 4.0, Smart Factories, …

The risk is digital process fragmentation

- **Increasing of the number of IT applications**: digital processes are spread over several applications
- **Increasing of the volume of managed data**: new digital processes are required for managing new data
- **Heterogeneous infrastructures:** cloud computing, big data, IoT, etc. : they require specific digital processes to be adapted on them



System integration as a key factor for economic growth

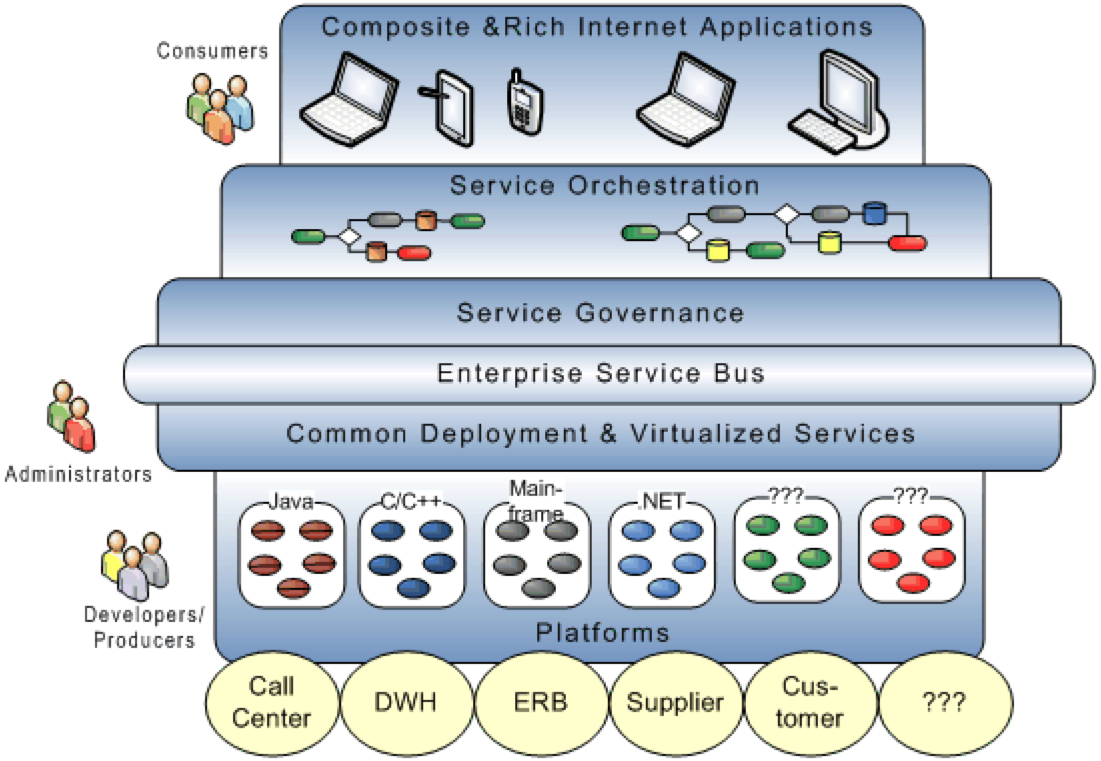


System integration plays a fundamental role for dealing with digitalisation strategies and economic growth

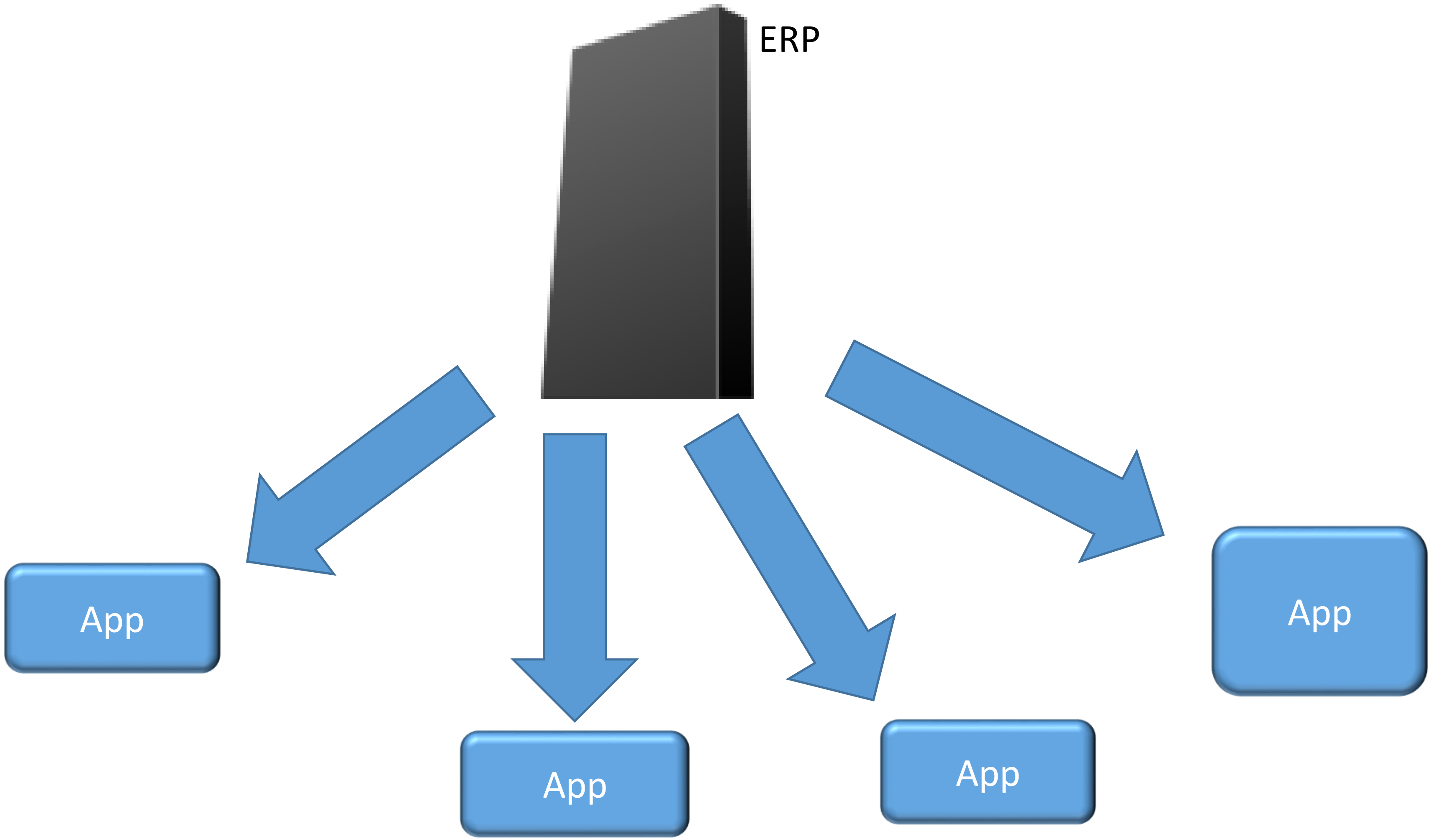
Service Oriented Architectures

- Service Oriented Architectures represents the structured choice for dealing with system integration

- **Good:** service oriented modelling, designing and developing approach
- **Bad:** Expensive, in terms of tools, governance and required skills. **Usually out of the radar for SMEs**.



Usually system integration in SME…



Microservices

- microservices are independent service-based components naturally modular and reusable,
- They offer a high level of **flexibility** making the design and development of integrated processes more **gradual and easy** to handle;
- ideally, they can be adapted to run in different kind of infrastructures.

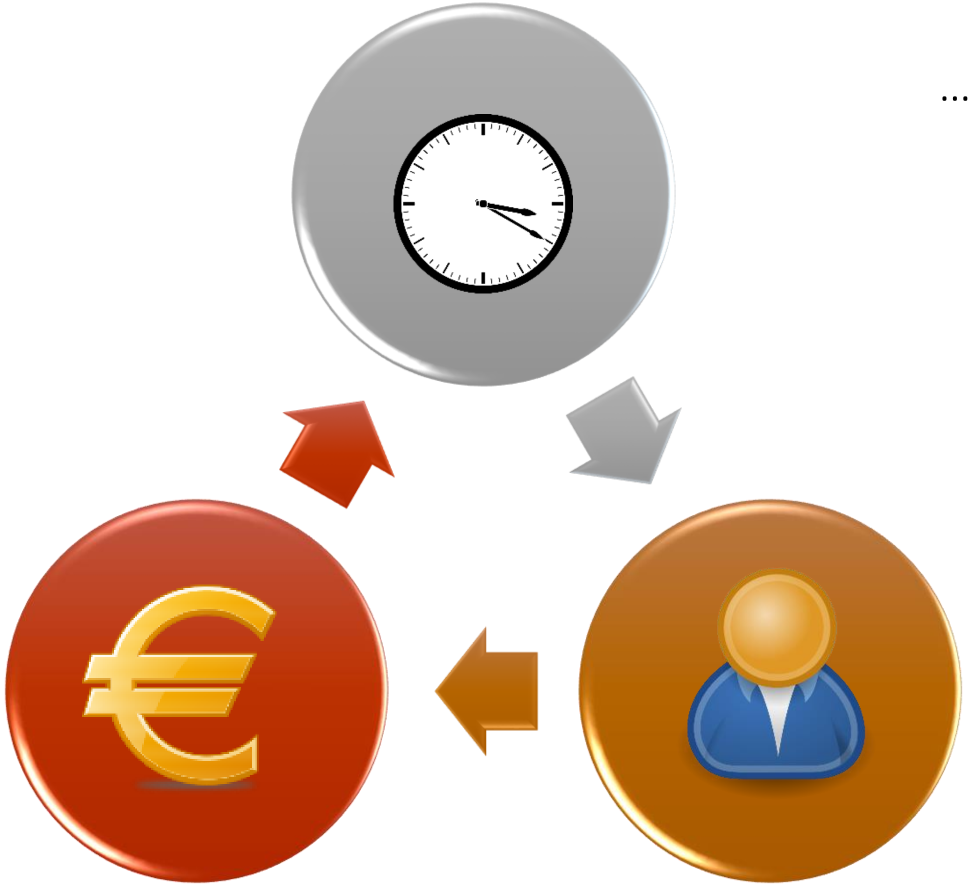
➡ **But**, they introduce **a high level of complexity** which could lead them to be as expensive as a SOA solution

Our challenge

Dealing with system integration by using a fully microservice architecture approach by guaranteeing the following objectives:

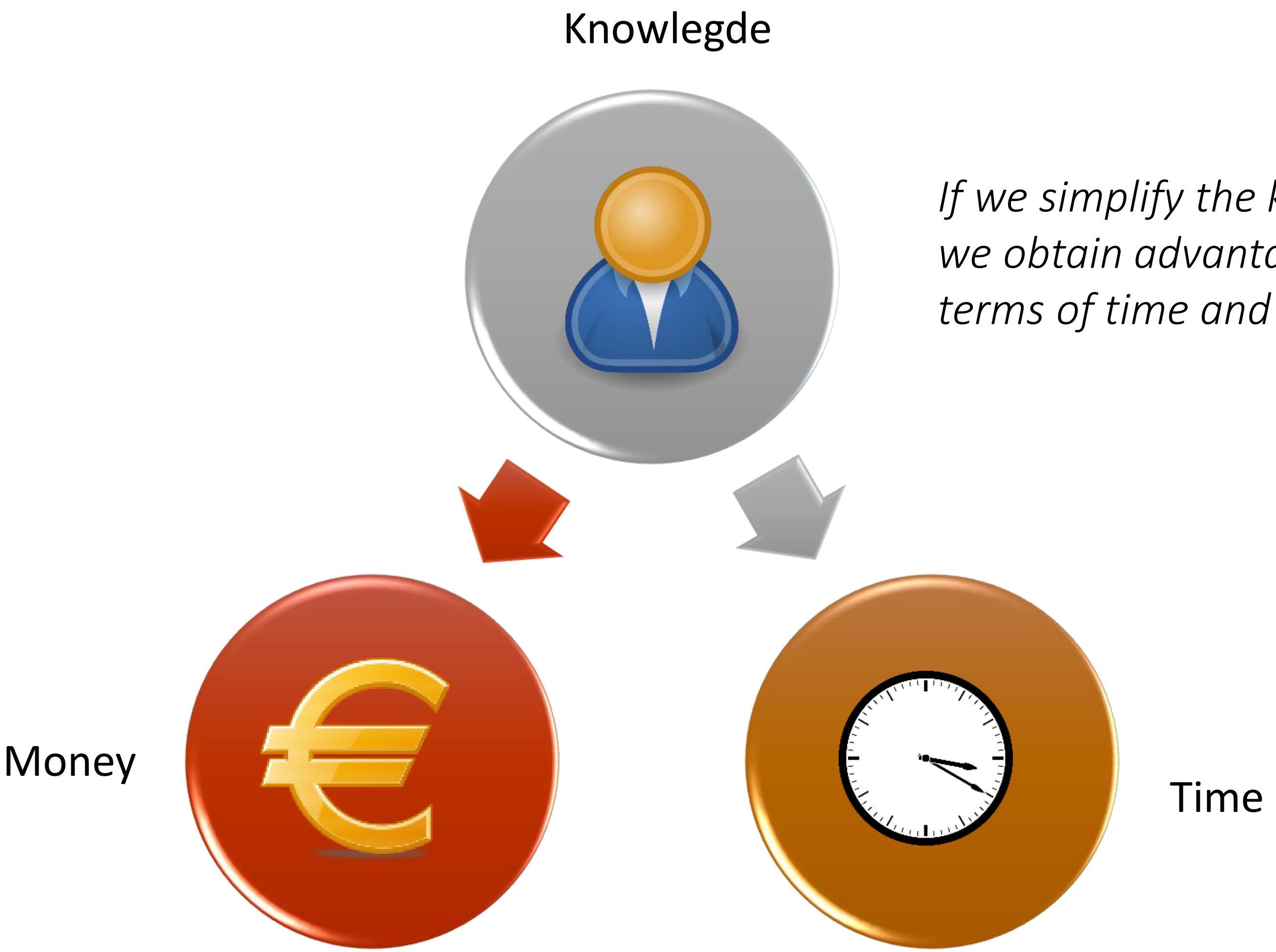
- Reducing time to market
- Reducing costs
- Reducing the knowledge required for managing system integration and enabling small teams to do the job.

Time, money and knowledge…



…they are the same

We focus on knowledge



Knowlegde

Money

Time

If we simplify the knowledge we obtain advantages in terms of time and money

What is Jolie

- Jolie is a programming language which crystalizes the basic concepts of microservices (à la microSOA)
  - Synchronous and asynchronouns communication
  - Workflow-like behaviour
  - Protocol agnostic
  - Integrated syntax for interfaces
  - Engine developed in Java
  - Interpreted language
  - Open source

http://www.jolie-lang.org

In Jolie the single unit of programmable software is a service.
No objects, no functions but services.

A hello world service in Jolie

```
type RequestMsg: void {
    .msg: string
}
type ResponseMsg:void {
     .responseMsg:string
}
```

Types ≅ Data model

```
interface HelloWorldInterface {
RequestResponse:
               hello( RequestMsg )( ResponseMsg )
}
```

Interface

```
inputPort HelloService {
  Location: ’’socket://localhost:8000’’
  Protocol: http { .format=‘’json’’ }
  Interfaces: HelloWorldInterface
}
```

Communication Endpoint

```
main {
    hello( request )( response ) {
               response = ’’Hello World’’ + request
    }
}
```

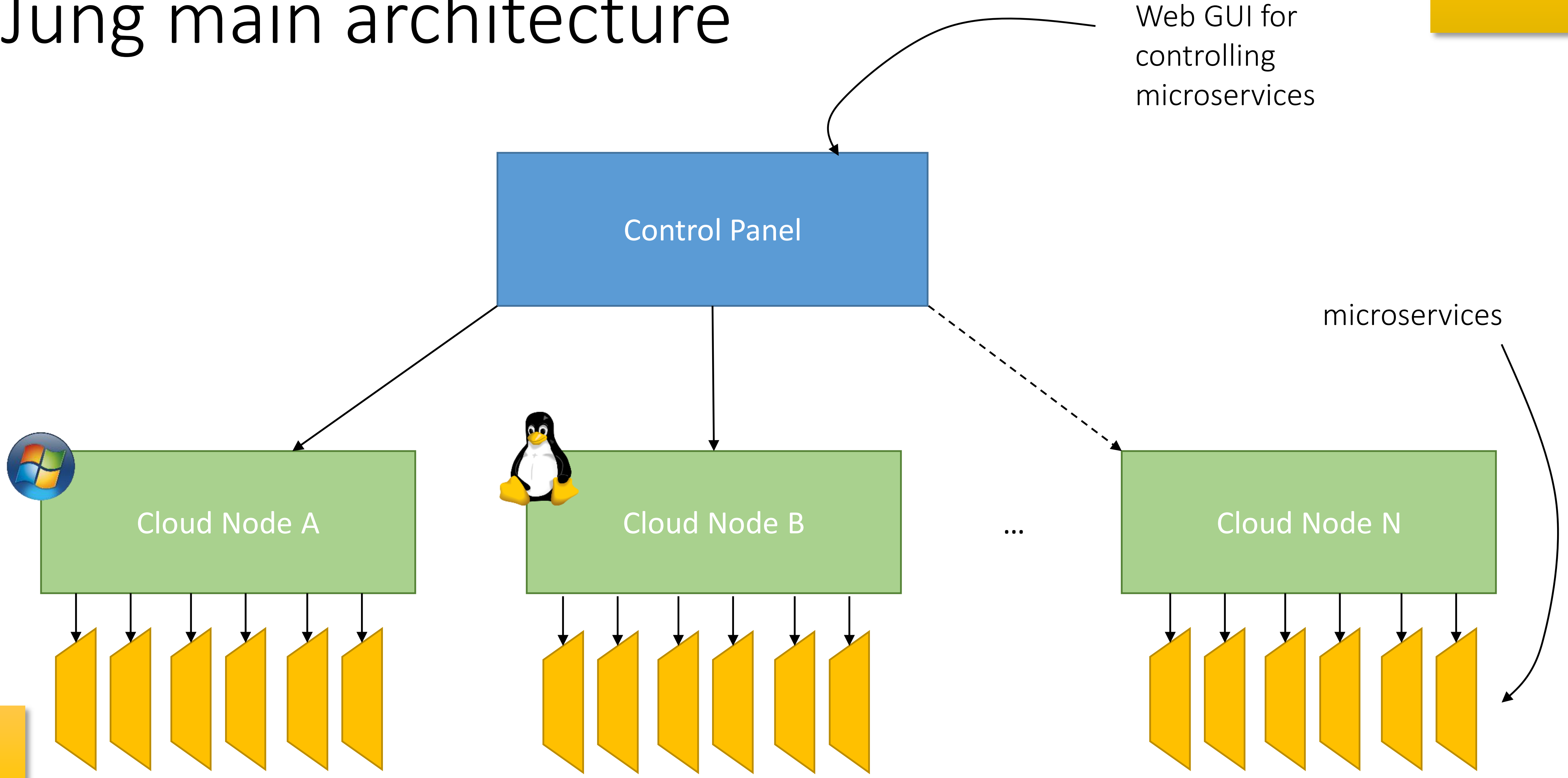
Behaviour

```
jolie helloworld.ol
```

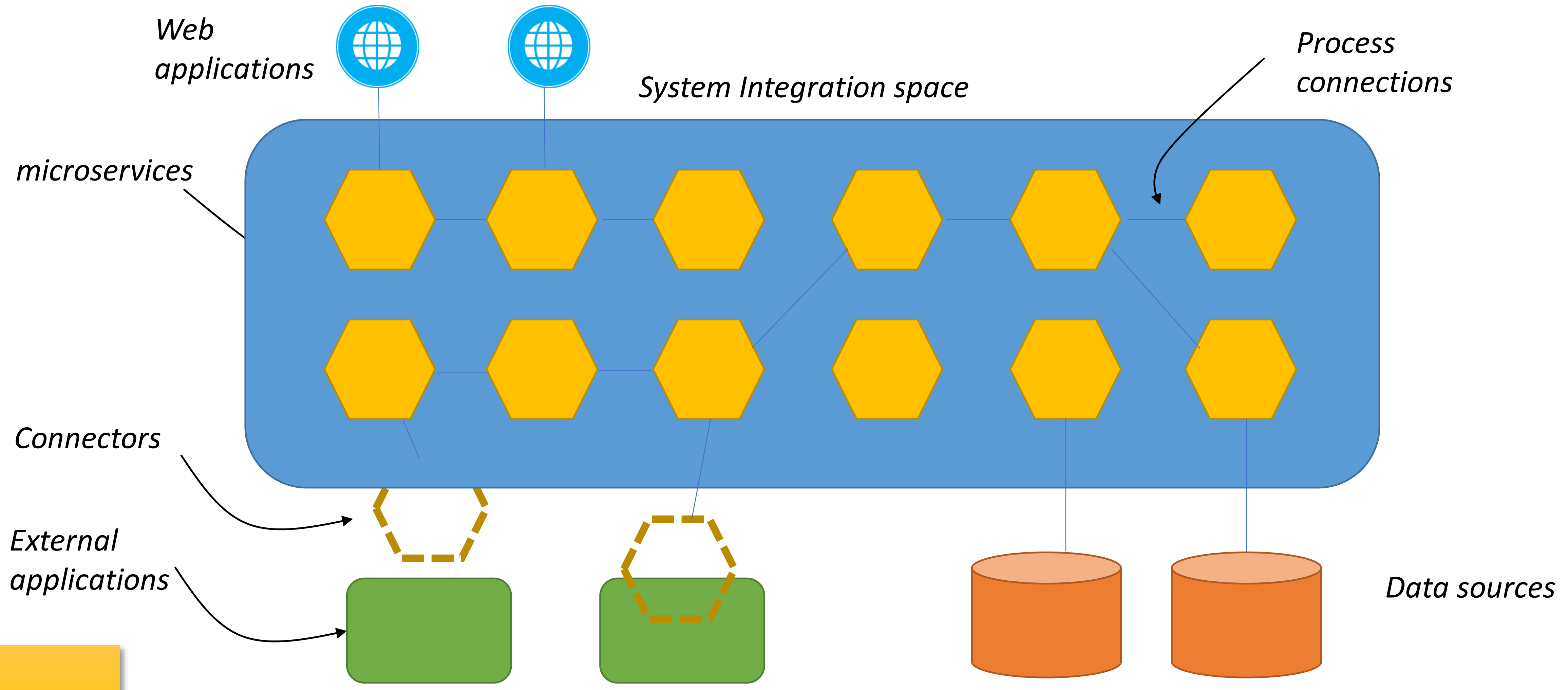
What is Jung

- Jung is an administrative platform developed by italianaSoftware for managing microservices developed with Jolie. It is a commercial product.
- It is fully developed in Jolie
- It does not lock-in the customer: all the microservices deployed by means fo Jung can be independetly deployed without Jung
- It provides some facilities for microservices govenance
  - Deployment management
  - Start/stop microservices
  - Catalog of available microservices and interfaces
  - Automatic monitoring of all microservices
  - Account manager for microservices equipped with GUI.

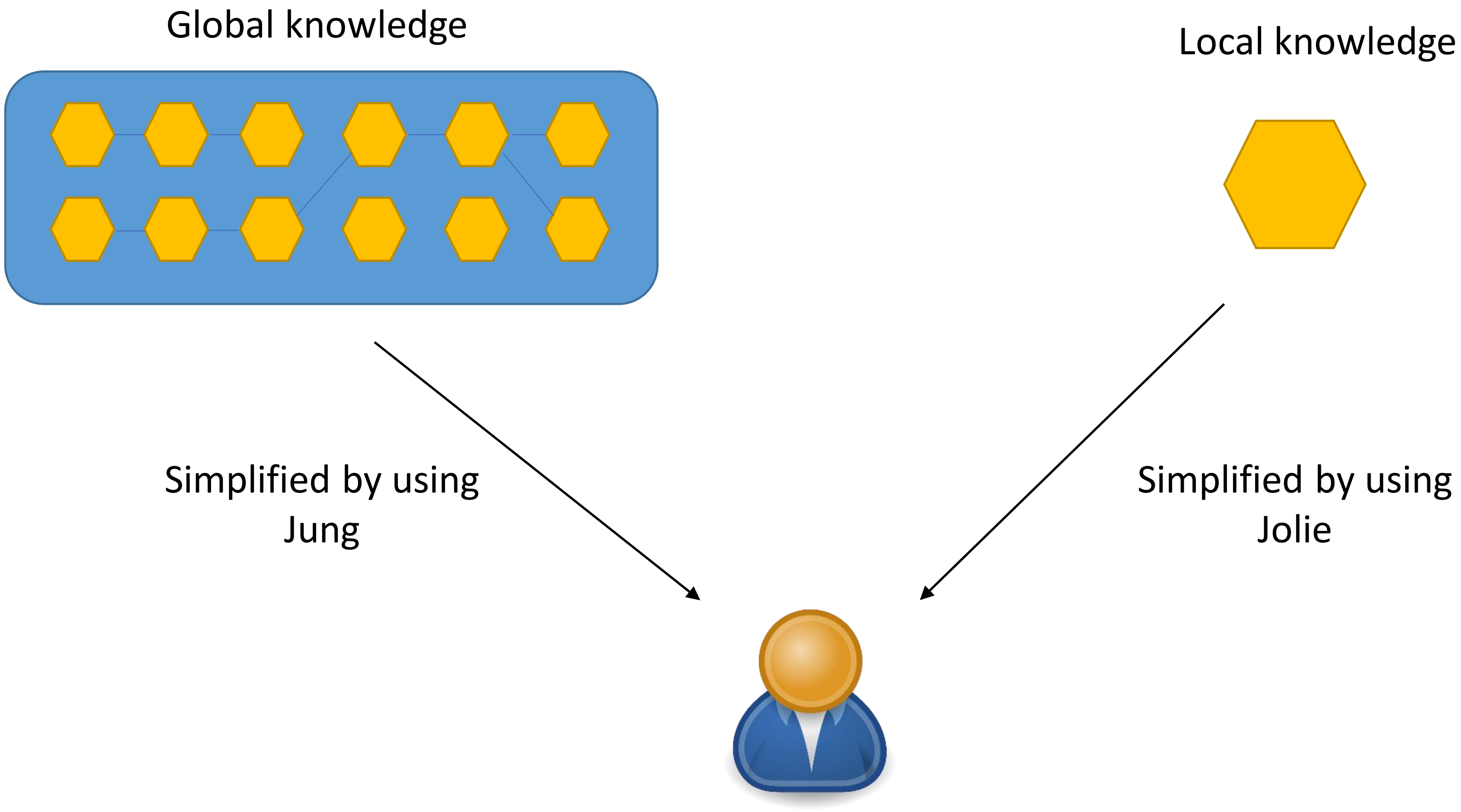
Jung main architecture



Reference Architecture



Simplification of Global knowledge and Local knowledge



Global knowledge

Local knowledge

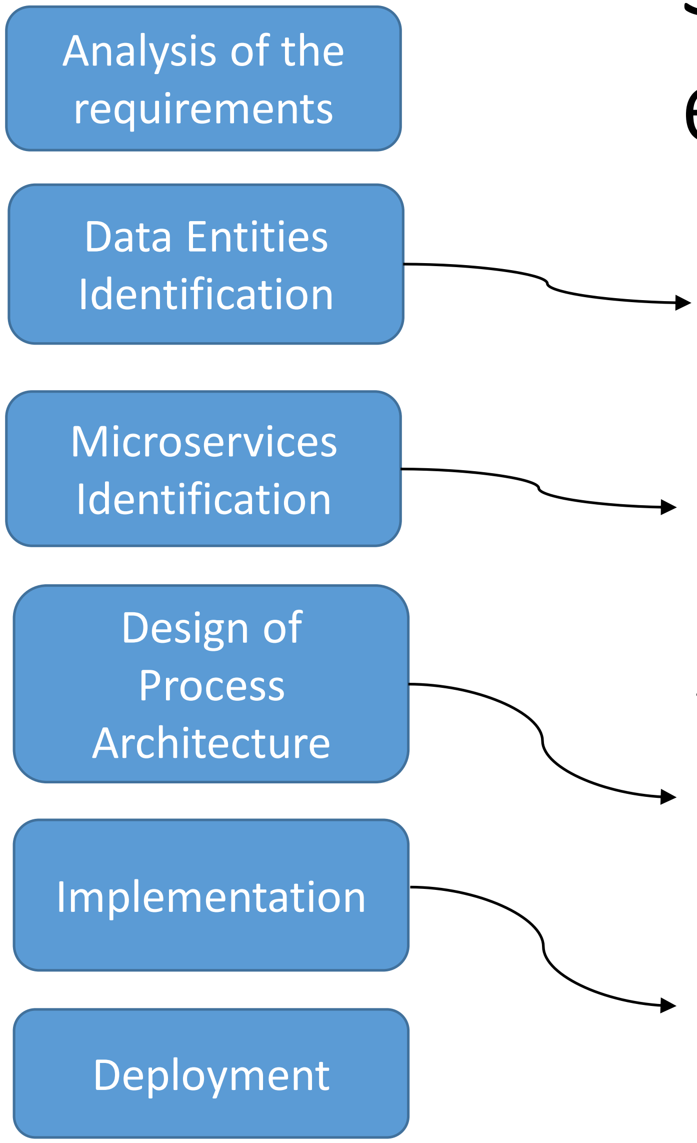
Simplified by using
Jung

Simplified by using
Jolie

Microservices categories

- **Pure computational:** the microservice performs pure computation, it receives a set of parameters and it produces an algorithmic manipulation of them returning a result..
- **Pure data provider:** the microservice just provides functionalities for accessing and manipulating the data stored into a particular data source as a set of files, a database or an electronic device.
- **Pure orchestrator:** the microservice just orchestrates existing microservices by coordinating different calls to them, collecting their replies and refactoring the final response as a composition of all the received data.
- **GUI provider:** the microservices provides also a web interface for enabling human interactions
- **Mixed microservices:** the microservice can combine more than one of the previous characteristics together. Thus it could be both computational and data provider, or it could also be a mix of all the four characteristics: computational, data provider, GUI provider and orchestrator

System integration engineering process



Analysis of the requirements

Data Entities Identification

Microservices Identification

Design of Process Architecture

Implementation

Deployment

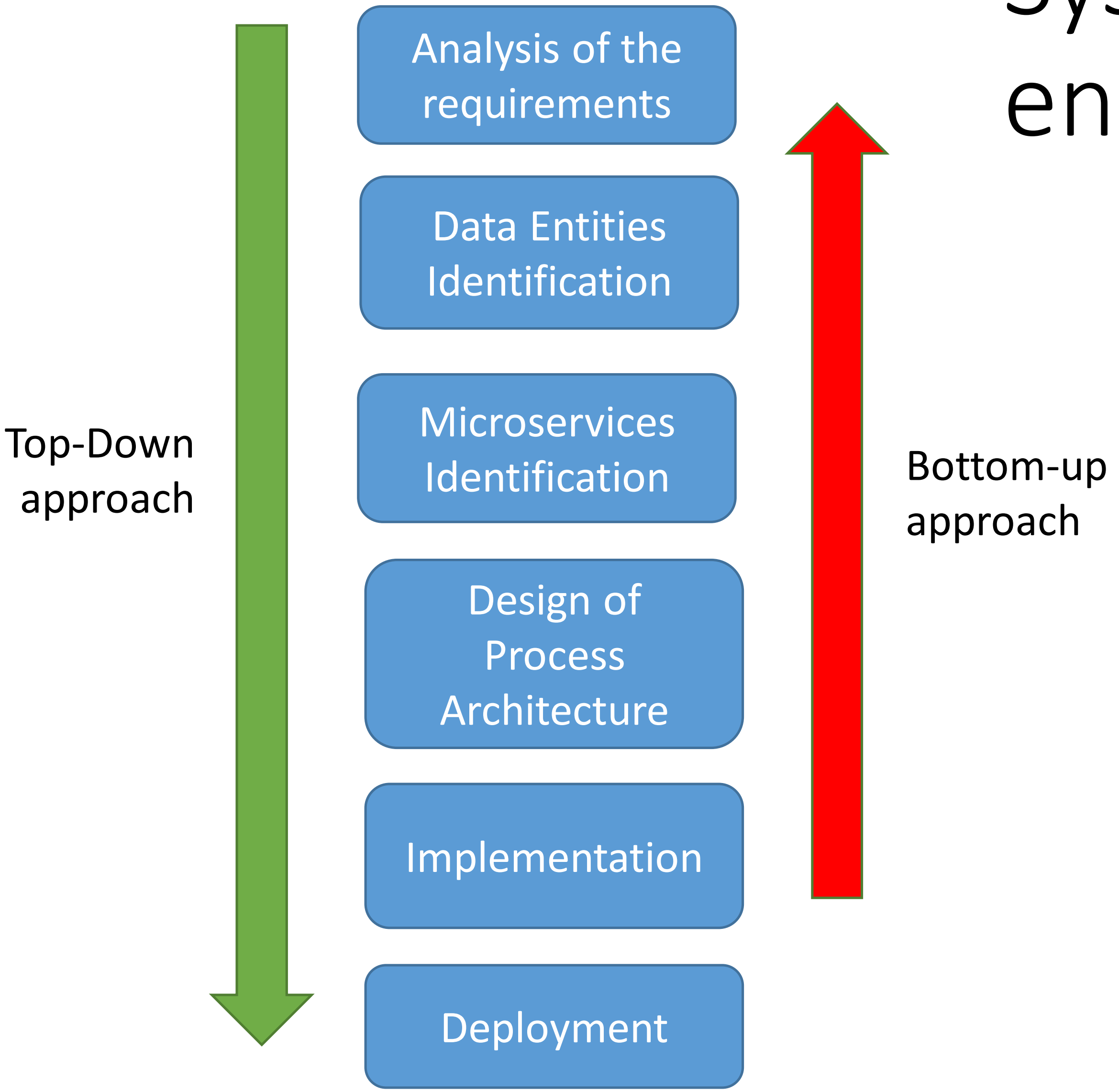
Defined in terms of Jolie types

Jolie interface identification. Microservices can be also selected from those that are already running

Jolie interface identification. Definition of workflows. At the present we use Sequence Chart Diagrams as tools for defining interactions. But we are working for using integrated choreographies

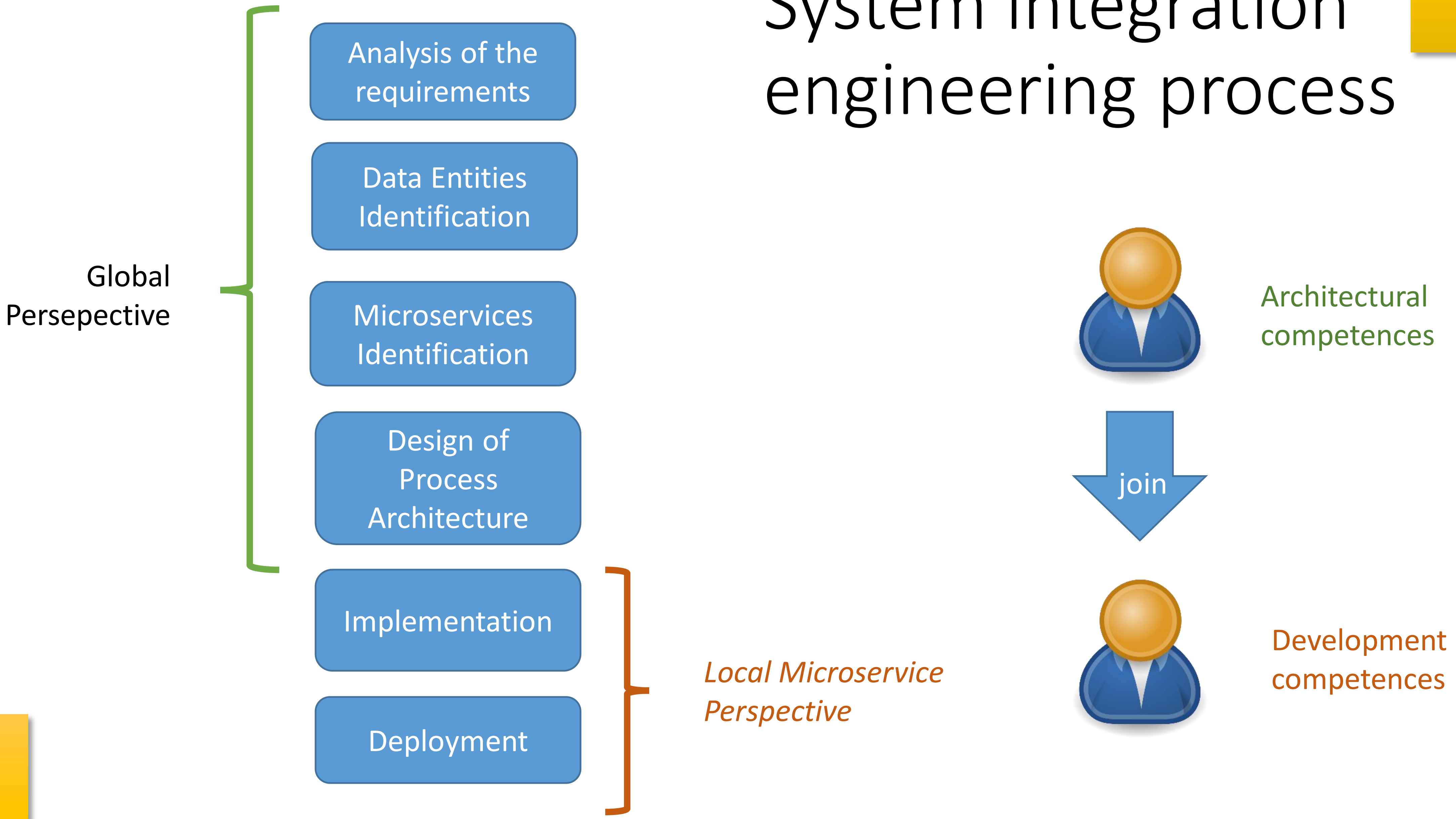
Done in Jolie

System integration engineering process


Top-Down approach
Analysis of the requirements
Data Entities Identification
Microservices Identification
Design of Process Architecture
Implementation
Deployment
Bottom-up approach

Thanks to the knowledge reduction introduced by Jolie, we are able to empower the bottom-up approach together with the top-down one.

System integration engineering process



Global Persepective

Analysis of the requirements

Data Entities Identification

Microservices Identification

Design of Process Architecture

Implementation

Deployment

Local Microservice Perspective

Architectural competences

join

Development competences

Teams and knowledge

Global knowledge and localized microservice knowledge must be managed together within the same team

A real scenario

- At the present, Jung based solutions are adopted in more than ten different production environments for dealing with archiving documentation flows from SAP to third party applications.
- In a case of a specific customer we can discuss the impact our approach had to their projects from the point of view of time to market, costs and human resources organization.

Project	Direct competitor estimation	Jolie based microservices approach
DMS Project	60 days	15 day
Maintenance Management	40 days	10 day
Marketing API	20 days	7 days
DMS Project (upgrade)	1 day ~ 3 days	3 hrs ~8hrs

3 people
Team

≅75%

Conclusions

- Distributed systems could help in providing a high level of flexibility to an IT infrastructure also for SMEs
- System integration could play a fundamental role for economic growth
- We need to work on approaches which reduce the required knowledge
- We worked for reducing the required knowledge.
  - Jolie as a unique programming language for dealing with microservices development facilitates the communication among the team and its artifacts like types and interfaces can also used during the design phase as implementation guidelines.
  - Jung is an administrative suite which help teams in the management of the Jolie artifacts and microservices by allowing navigation of interfaces and deployment of microservices.
- We shown how we obtained successful results in a case history

…finally, let us solve a puzzle

- How many microservice antipatterns did you find?