



Programming Microservices with Jolie

Saverio Giallorenzo

Hi



Saverio



Post-doc at **Computer Science and Engineering Department** of the **University of Bologna**

Topics:

- Choreographic Programming;
- Jolie and Microservices;
- Session Types;
- Process Calculi.

Why a language?

- ❖ To give to programmers a proper vocabulary to express:
 - ❖ APIs \Rightarrow Interfaces
 - ❖ Communication Patterns \Rightarrow Behaviours
 - ❖ Architectures \Rightarrow Deployments
- ❖ And to avoid distractions (objects \neq services)

The Jolie Way

APIs



```
interface MyInterface {  
  OneWay: sendNumber( int )  
}
```

```
include "MyInterface.iol"  
outputPort B {  
  Location:  
    "socket://localhost:8000"  
  Protocol: sodep  
  Interfaces: MyInterface  
}
```

```
main  
{  
  sendNumber @ B ( 5 )  
}
```

```
include "MyInterface.iol"  
inputPort B {  
  Location:  
    "socket://localhost:8000"  
  Protocol: sodep  
  Interfaces: MyInterface  
}
```

```
main  
{  
  sendNumber( x )  
}
```

The Jolie Way

Deployment

```
interface MyInterface {  
  OneWay: sendNumber( int )  
}
```

```
include "MyInterface.iol"  
outputPort B {  
  Location:  
    "socket://localhost:8000"  
  Protocol: sodep  
  Interfaces: MyInterface  
}
```

```
main  
{  
  sendNumber @ B ( 5 )  
}
```

```
include "MyInterface.iol"  
inputPort B {  
  Location:  
    "socket://localhost:8000"  
  Protocol: sodep  
  Interfaces: MyInterface  
}
```

```
main  
{  
  sendNumber( x )  
}
```

The Jolie Way

Behaviour

```
interface MyInterface {  
  OneWay: sendNumber( int )  
}
```

```
include "MyInterface.iol"  
outputPort B {  
  Location:  
    "socket://localhost:8000"  
  Protocol: sodep  
  Interfaces: MyInterface  
}
```

```
main  
{  
  sendNumber @ B ( 5 )  
}
```

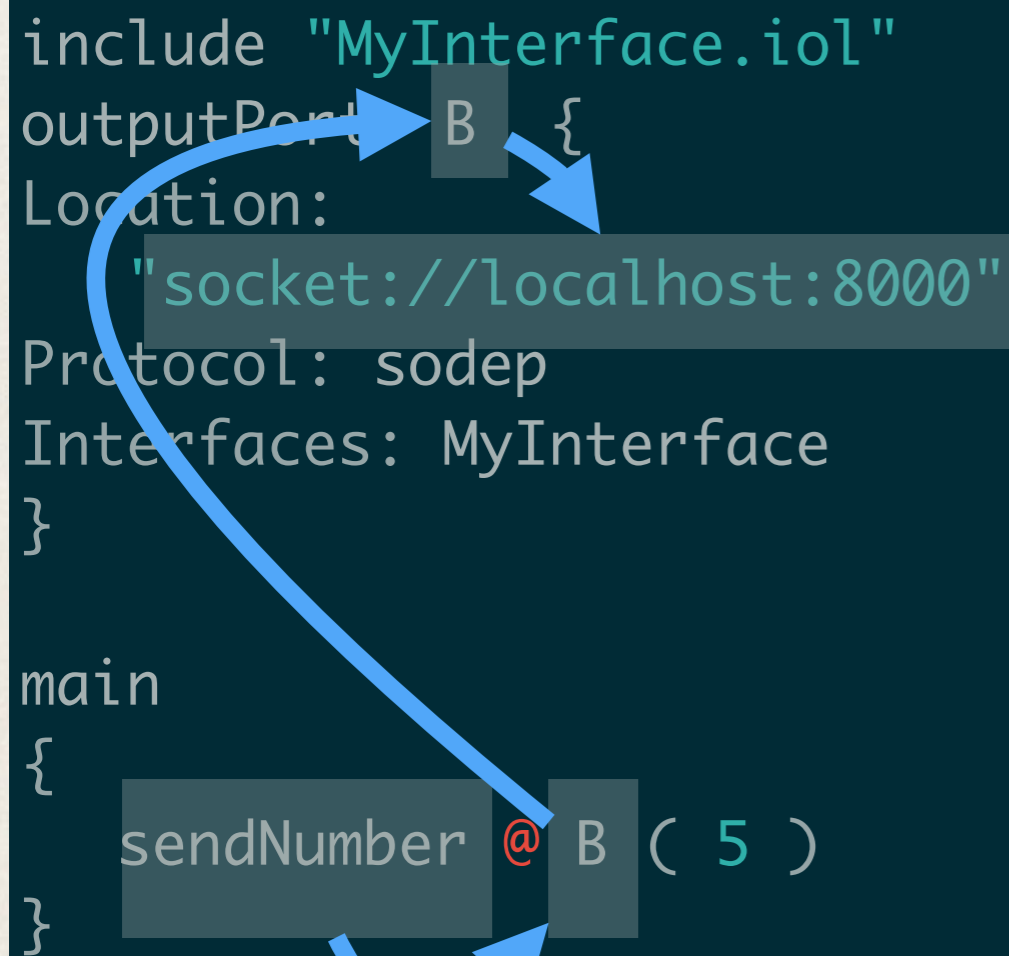
```
include "MyInterface.iol"  
inputPort B {  
  Location:  
    "socket://localhost:8000"  
  Protocol: sodep  
  Interfaces: MyInterface  
}
```

```
main  
{  
  sendNumber( x )  
}
```

The Jolie Way

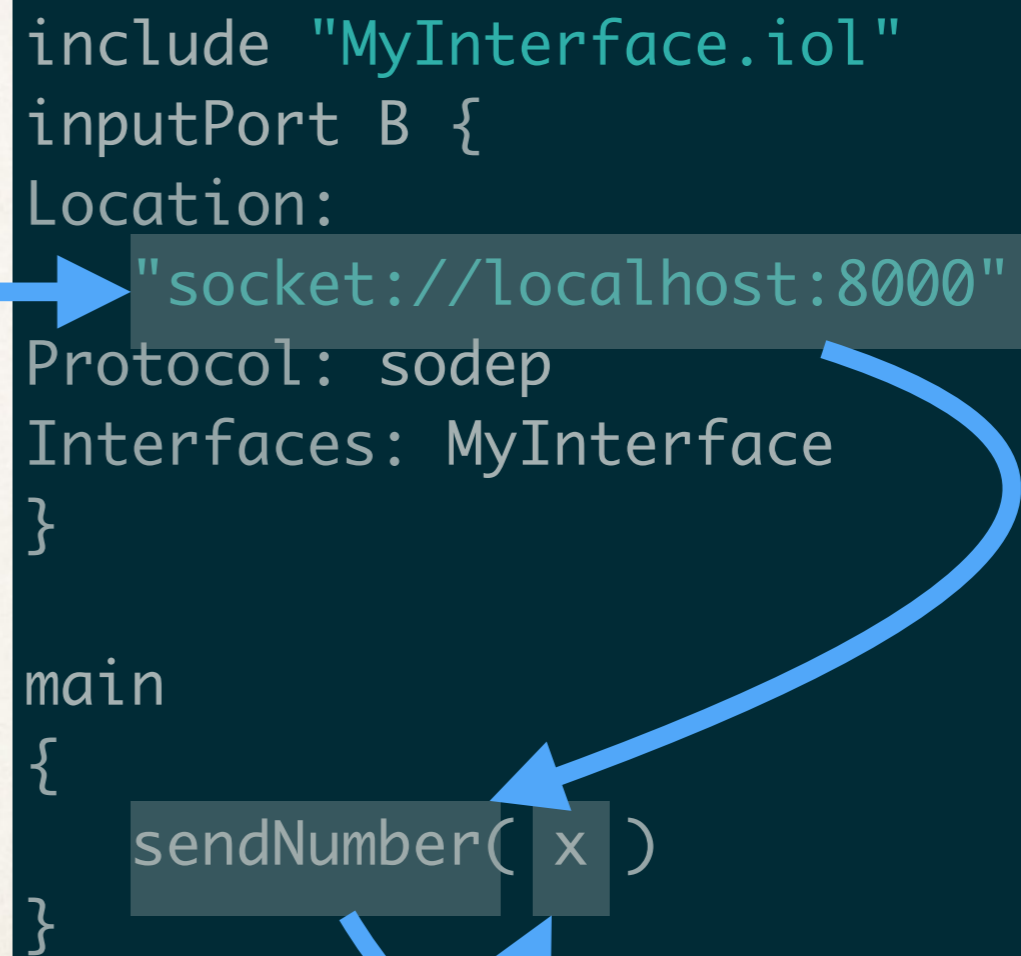
```
interface MyInterface {  
  OneWay: sendNumber( int )  
}
```

```
include "MyInterface.iol"  
outputPort B {  
  Location:  
    "socket://localhost:8000"  
  Protocol: sodep  
  Interfaces: MyInterface  
}  
  
main  
{  
  sendNumber @ B ( 5 )  
}
```



The diagram shows a Jolie actor with an output port named 'B'. The port's location is 'socket://localhost:8000' and its protocol is 'sodep'. It implements the 'MyInterface' interface. The main function of the actor is to call 'sendNumber @ B (5)', where the '@' symbol indicates a message being sent to the output port 'B'.

```
include "MyInterface.iol"  
inputPort B {  
  Location:  
    "socket://localhost:8000"  
  Protocol: sodep  
  Interfaces: MyInterface  
}  
  
main  
{  
  sendNumber( x )  
}
```



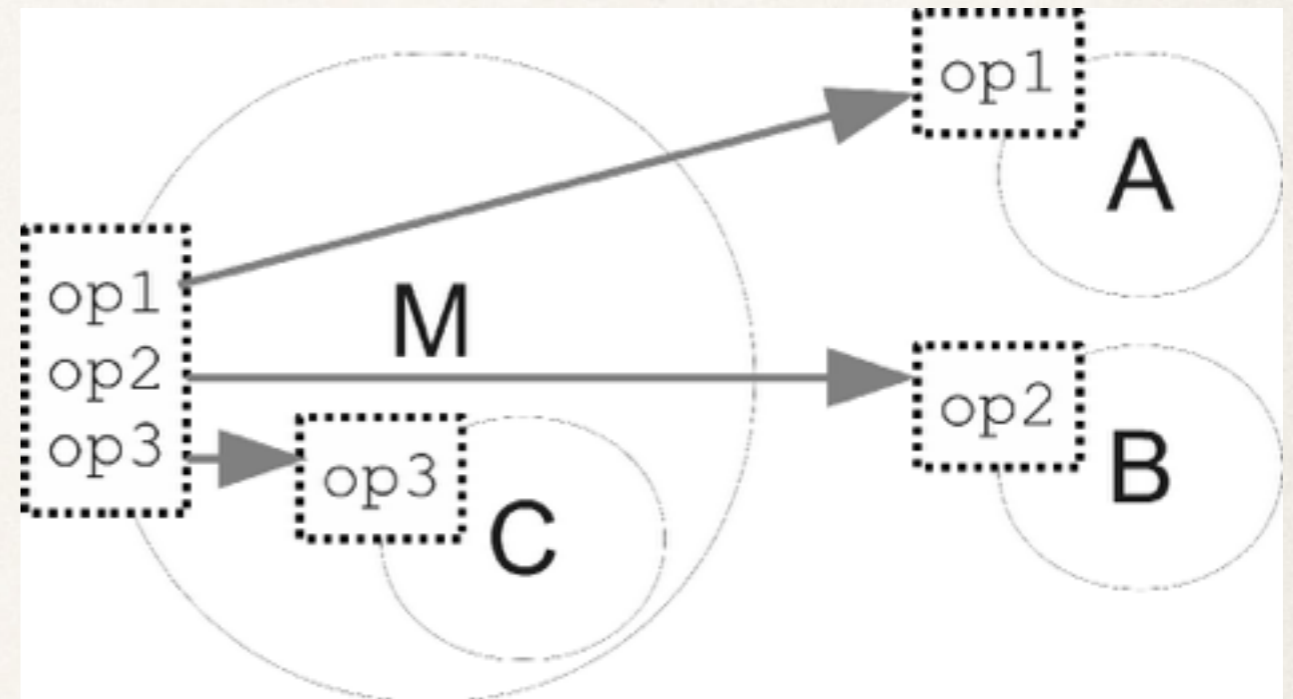
The diagram shows a Jolie actor with an input port named 'B'. The port's location is 'socket://localhost:8000' and its protocol is 'sodep'. It implements the 'MyInterface' interface. The main function of the actor is to call 'sendNumber(x)', where 'x' is a variable representing the received message.

Demo



We just scratched the surface

```
outputPort A {
  Location: "socket://A_URL:80/"
  Protocol: sodep
  Interfaces: MobileInterface
}
outputPort B {
  Location: "socket://B_URL.com:80/"
  Protocol: soap
  Interfaces: DesktopInterface
}
outputPort C {
  Interfaces: BlindInterface
}
embedded {
  Java: "blindServer.service" in C
}
inputPort M {
  Location: "socket://myApp.com:80/"
  Protocol: http
  Aggregates: A, B, C
}
```



Others:

- Redirection
- Courier

Contacts

<https://github.com/jolie/jolie>

“This *is* the programming language
you are looking for”



Contacts



Jolie

<http://www.jolie-lang.org>

Thanks@Audience(greetings)