Choreographies for Microservices

Ivan Lanese
Computer Science Department
University of Bologna/INRIA
Italy

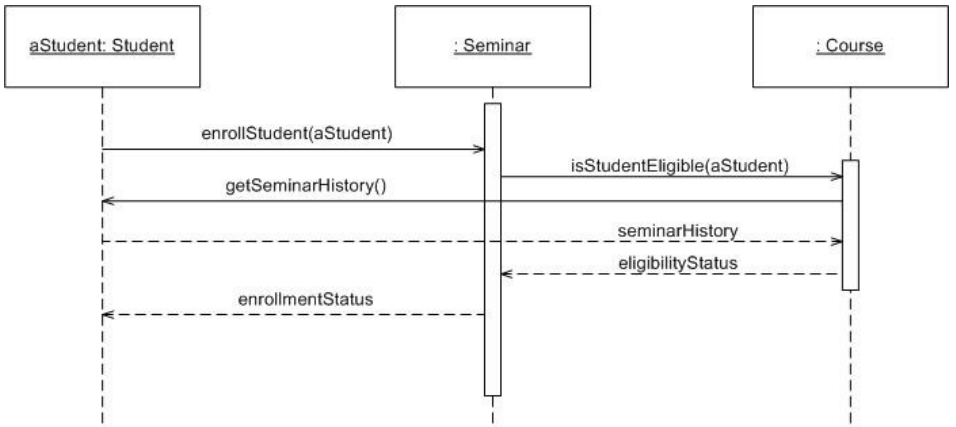
Joint work with Saverio Giallorenzo

Choreographies: the idea

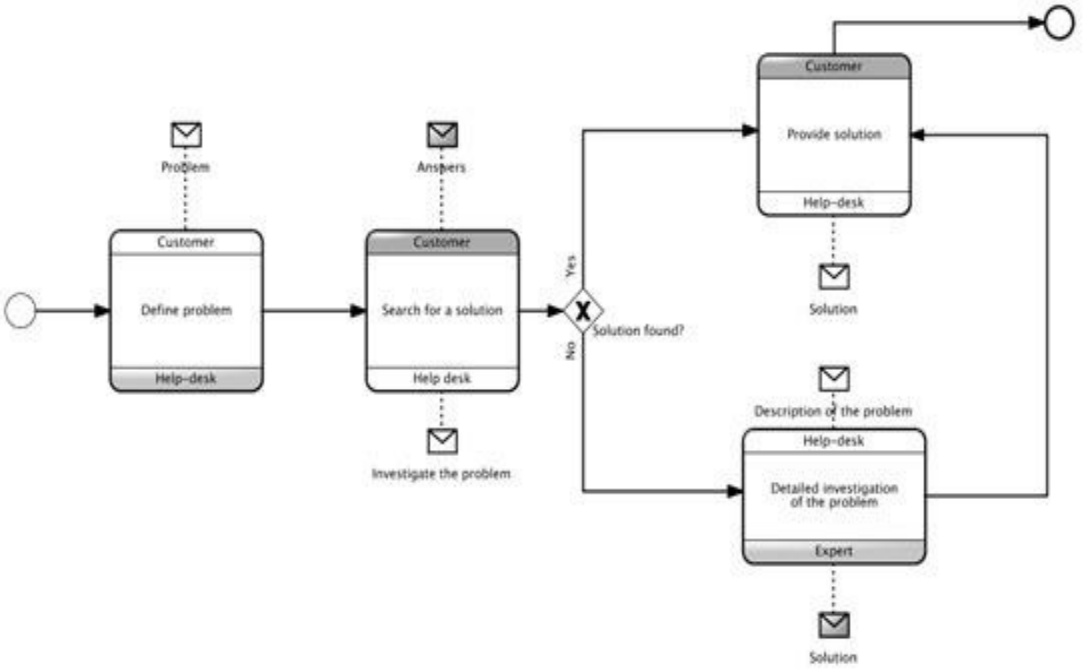
- Modeling a whole distributed system in a single artifact
- Having interaction as key element
- An interaction is the communication of a message between two entities
  - One sending the message
  - One receiving the message

Choreographies: examples

- Used at many abstraction levels
  - Documentation
  - Types
  - Programming language



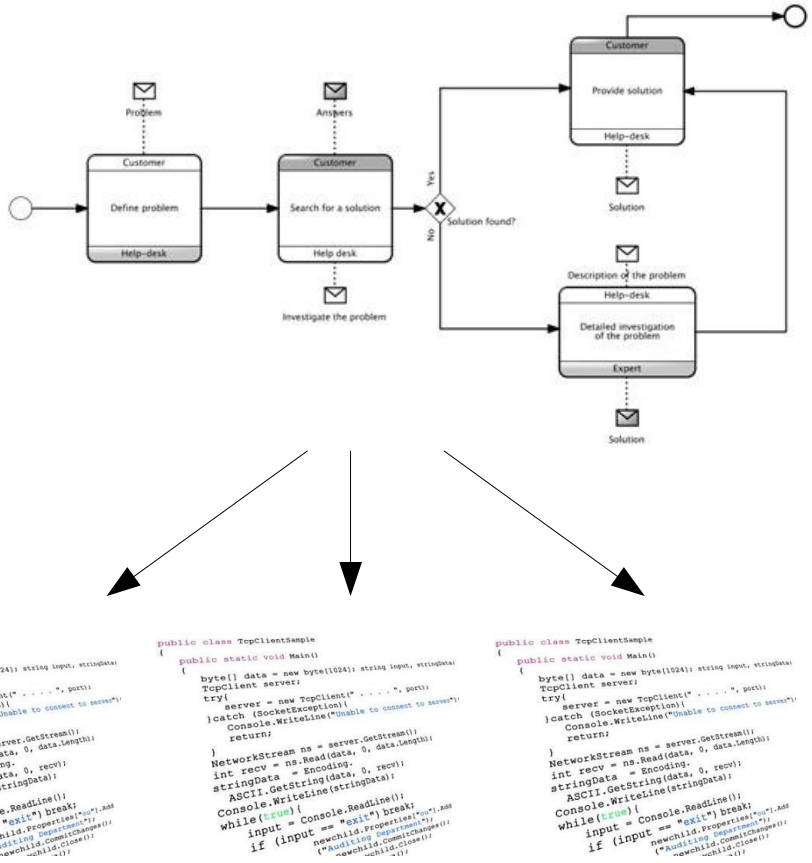

```
aioc {
 continue@u1 = "y";
 while( continue == "y" )@u1{
  scope @u1 {
   msg@u1 = "Hello World"
  } prop { N.scope_name = "hello_world" };
  sendMsg: u1( msg ) -> u2( msg );
  {
   r@u2 = show( msg )
   | continue@u1 = getInput( "Continue? (y/n)" )
  }
 }
}
```



Choreographies: advantages

- Global view of the expected behavior

- Local behaviors implementing the global view can be automatically generated
  - Projection on each participant
  - Correctness ensured by mathematical proofs
  - Deadlock freedom, ...



- Generating the choreography from local descriptions is also possible

Smart projections

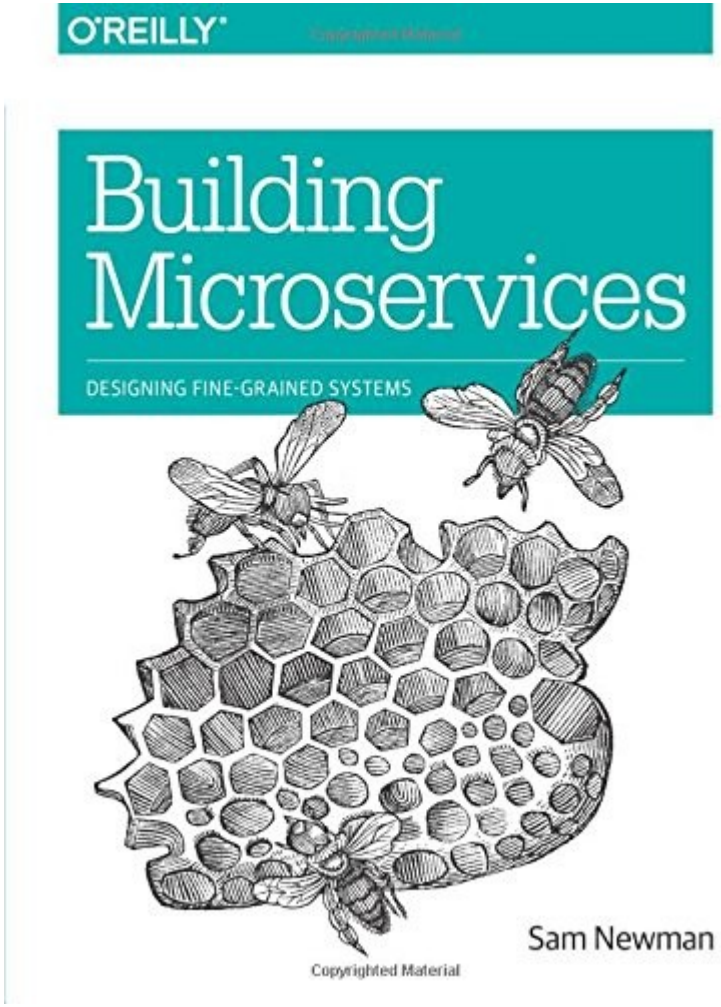
- Projection can add all what is needed to ensure the abstract specification is satisfied
  - Synchronizations
  - Error management
  - Security mechanisms

- The programmer has no need to consider these aspects

Choreographies for evolution

- If the system needs to evolve, the choreography is changed
- Local code is re-generated
- New local code replaces the old one
  - Techniques for coordinated hot swap exist

Oh, yes, we have to speak about microservices



Microservices

- Not a unique agreed definition, but...
- Microservices are autonomous entities communicating via message passing
- Microservices are small
  - Describe a single functionality, built by a small team, easily disposable, ...
- Microservices collaborate to reach a common (complex) goal

The puzzle analogy

- The smaller the pieces, the more difficult the puzzle

- How to solve the puzzle?

The puzzle analogy

- Look at the global picture

- Choreographies are the global picture for the microservice puzzle

Summary

- Choreographies are a suitable tool to manage the complexity of communication in microservice systems
- But a lot of work is still needed to make the approach practical

Future work

- Choreographies need further development
  - Choreography as a programming language not yet fully developed
  - Compositionality

- Application of choreographies to microservices raises new questions
  - Choreography-based development process
  - Choreography refinement
  - Interplay between choreographies and deployment

End of my part

Thanks!

Questions?