Architetture Software a Microservizi

Corso di Laurea Magistrale in Informatica

Enterprise SOA modeling

Davide Rossi

Dipartimento di Informatica – Scienze e Ingegneria

Università di Bologna

Enterprise SOA

- No conceptual difference between E-SOA and Microservices
- Yet there is a clear distinction between the two approaches

Enterprise SOA

- Top-down modeling approach
- Single tech stack (mostly Java EE)
- SOAP (and WS-*) and queues
- Clear governance
- ESB

Governance

SOA governance is related to Corporate governance/IT governance; it is about **policies definition** and **enforcement**

OASIS SOA RM about policies management:

- policy assertion (verifiable/measurable)
- policy owner
- policy enforcement (technically or by review)

Policies

Policies can be related to:

- design-time
- runtime

examples: lifecycle (design and documentation, versioning, retirement, ...); security; testing; performance

Top-down SOA design approach

Focus on business structure and needs; technology comes last

Pros:

- business-relevant solution
- clear governance
- better at addressing risks

Cons:

- extensive planning and strict policy enforcement
- all-or-nothing solution (but only from a design perspective)
- slow solution delivery

Rationale: the case for SOA modeling

- Business is the problem domain
- IT is the solution domain
- We need to relate solutions with problems
- We need models

Model-based top-down approach

- Business characterization (modeling)
- Services identification
- Services specification (modeling)
  - Structural perspective (interfaces, dependencies)
  - Behavioral perspective

Business modeling

- Process-based approach
- Business Process Modeling
- BPMN

Service Identification

- Whichever SOA design approach you take there is a step in which services have to be identified
- A SLR[1] on the topic lists 105 relevant studies on service identification methods
- We will show something about SOMA

[1] Huergo, Rosane S., et al. "A systematic survey of service identification methods." Service Oriented Computing and Applications 8.3 (2014): 199-219.

Service modeling

- Structural / behavioral viewpoints
- SoaML
- Non-standard UML profiles
- RESTful API modeling
  - OAS (OpenAPI Specification a.k.a. Swagger)
  - RAML

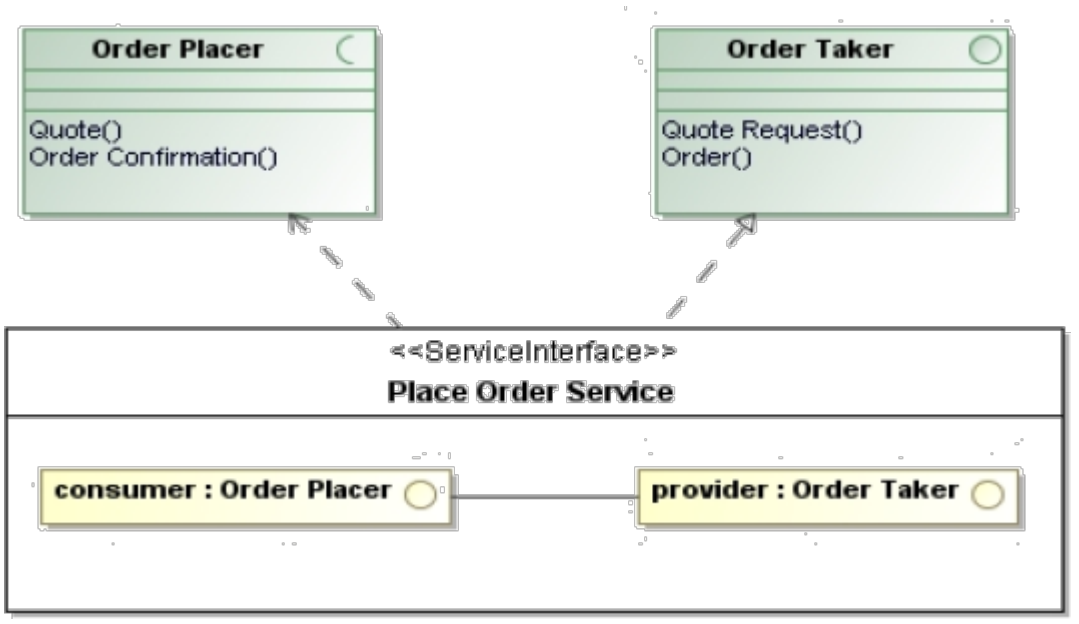
SoaML

- Specifying services and functional capabilities, capabilities of consumers, protocols and rules for use
- Identifying services, dependencies and requirements links
- Defining the policies associated to services

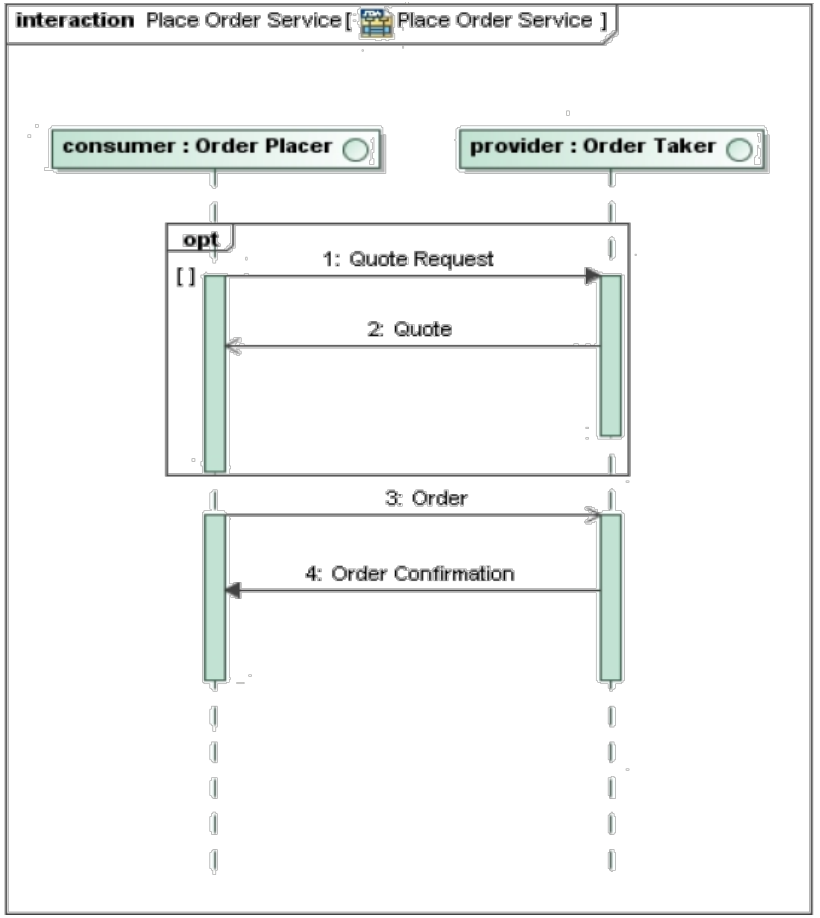
SoaML

- A UML profile and extension
- Supporting both an IT and a Business Perspective on SOA
- Supporting both a contract and an interface based approach to SOA (via ServiceContract and ServiceInterface elements)
- Supporting both top down and bottom-up development

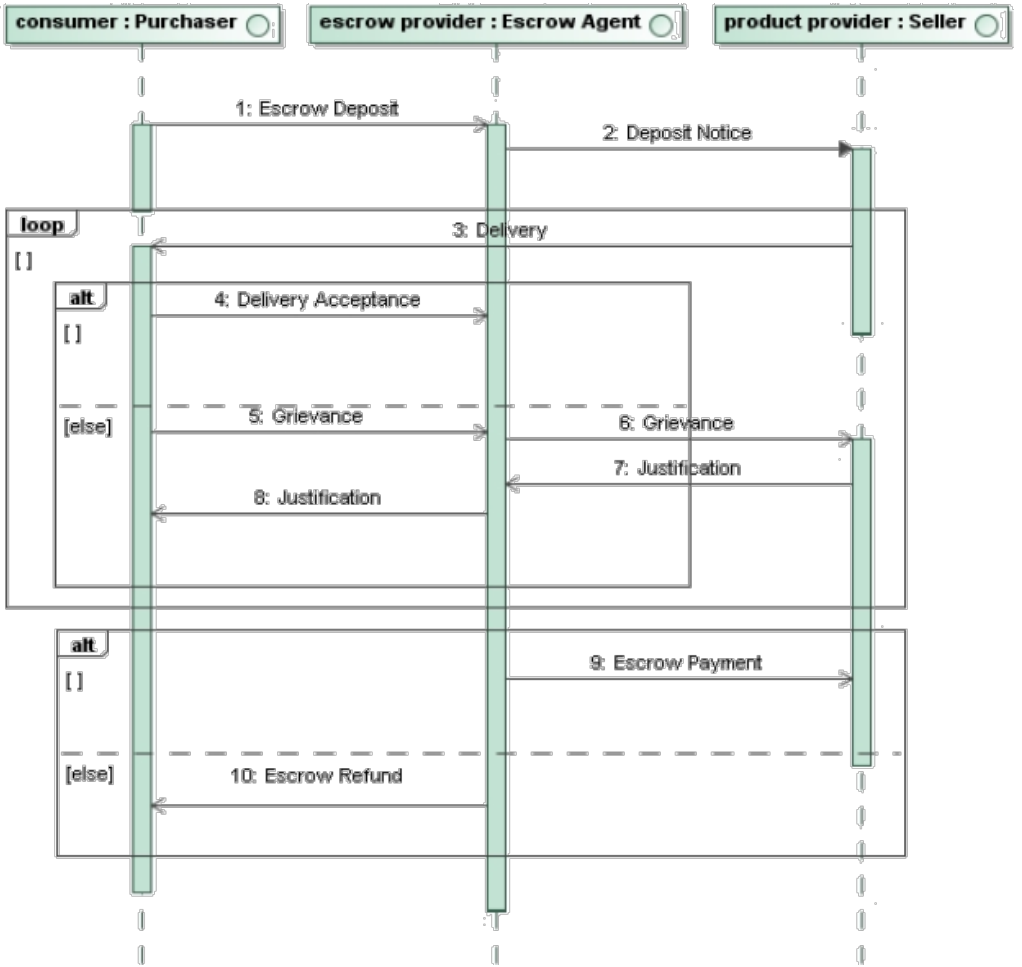
Service interface definition



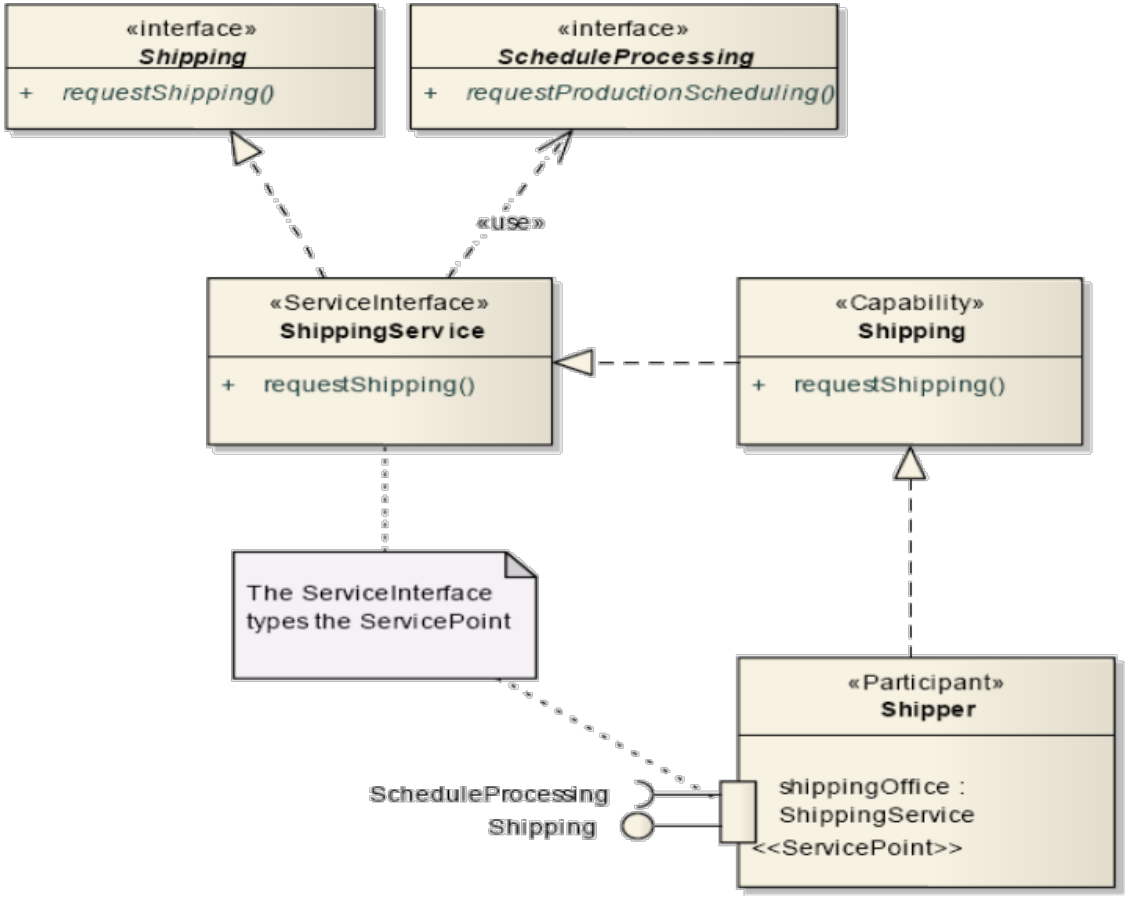
Choreography via owned behavior



Choreography via owned behavior



Capabilities



Alternatives to SoaML

- Various UML profiles
- Address structural / behavioral aspects
- Can assume other architectural pattern/styles (e.g. Layers)

Service layers

- In E-SOA services are often organized in layers:
  - Business process or task services
  - Entity services
  - Utility or generic services
- Beware: this is not really a standard layered architecture since task services can invoke utility services directly

Business process or task services

- Represent a whole process and wraps an *external capability* of the organization
- Can be internally composed by several lower-level services
- Can directly or indirectly entail human activities
- Can be implemented by a program or using a process engine

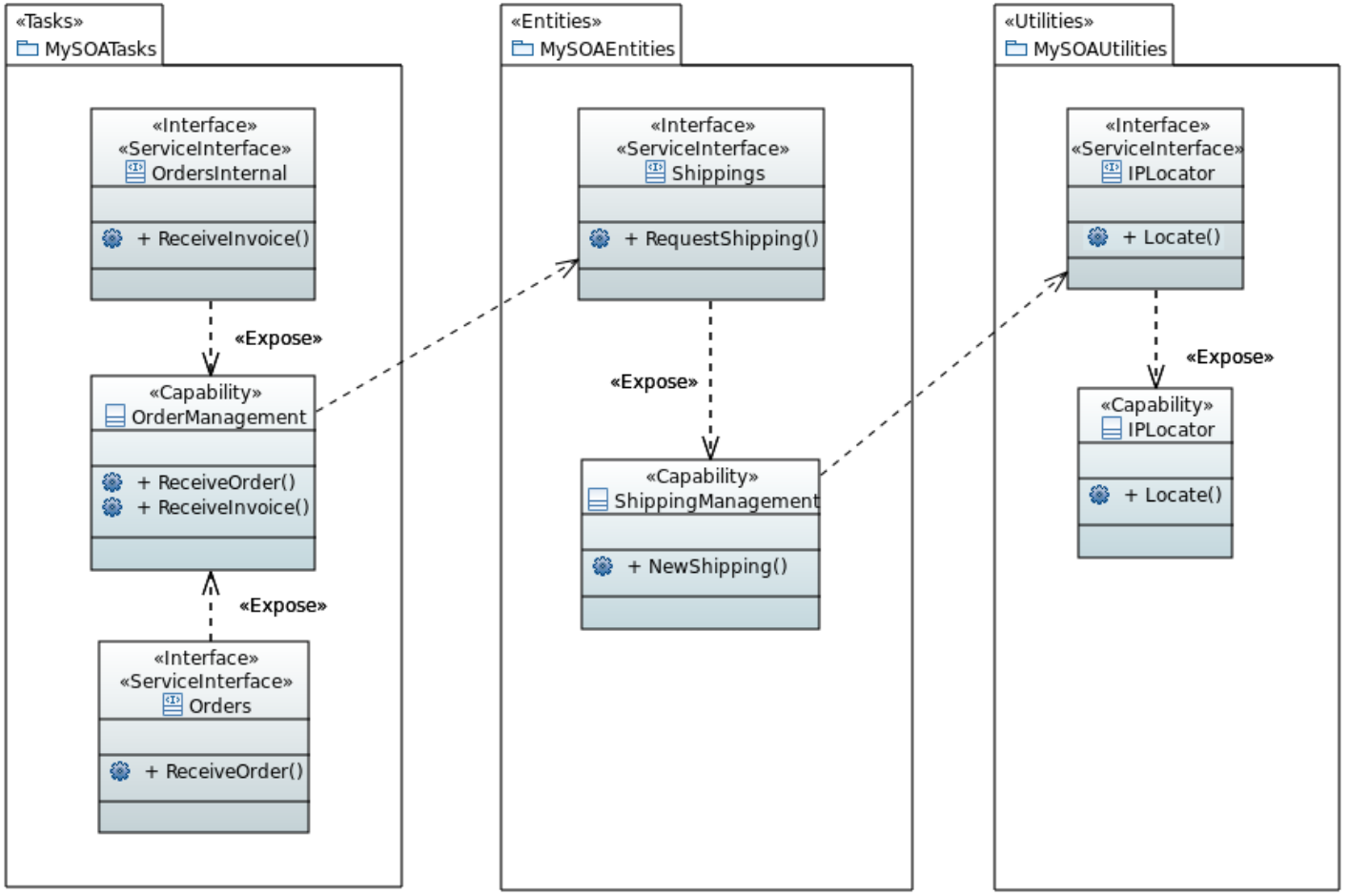
Entity services

- Make single capability accessible
- Are usually process-agnostic (but domain aware!) and thus reusable
- Can make use of utility services
- Can wrap an existing enterprise application function

Utility services

- Support functions not directly related to the business domain
- Totally business agnostic and highly reusable

TinySOA UML profile



Modeling methods

- OASIS
  - SOA-RM (Reference Model)
  - SOA-RFA (Architecture Foundation for SOA)
- OpenGroup
  - SOA Ontology
- Service-oriented Modeling Framework (SOMF)
- Platform-independent Model for SOA (PIM4SOA)

Modeling methods

- OMG SOA Modeling Language (SoaML)
  - A UML profile and extension NOT a full method
- IBM Service-oriented Modeling and Architecture (SOMA)
  - A RUP-related method

SOMA

- Service-oriented Modeling and Architecture is IBM proprietary method for identifying services, specifying them, and making decisions regarding how to realize them.
- Is largely RUP-inspired
  - Use case-driven

SOMA: practices

Four practices

- Use Case Driven Business Modeling
- Service Identification
- Service Specification
- Service Realization

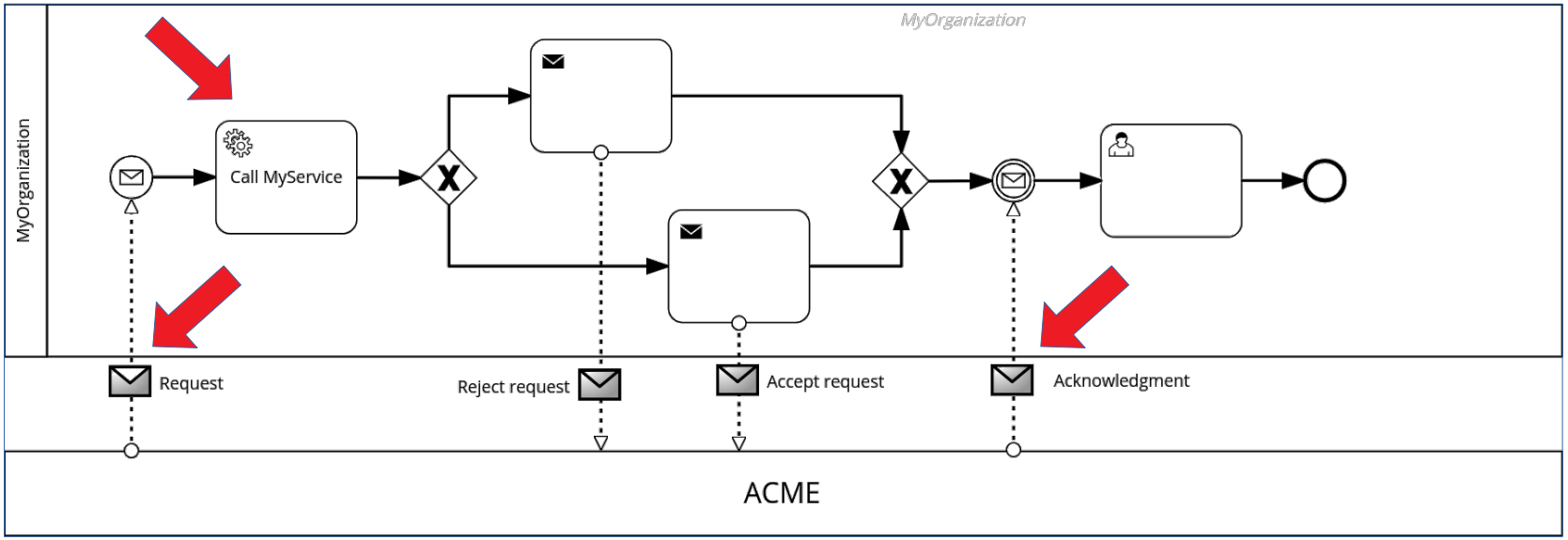
SOMA: simplified workflow

- Define business use case realizations using BPMN
- Identify capabilities from BPMN diagrams elements (processes, lanes or tasks) to create service candidates
- Refine the candidate services
- Define the service interfaces that expose each capability
- Define the data that are exchanged and refine the service interfaces

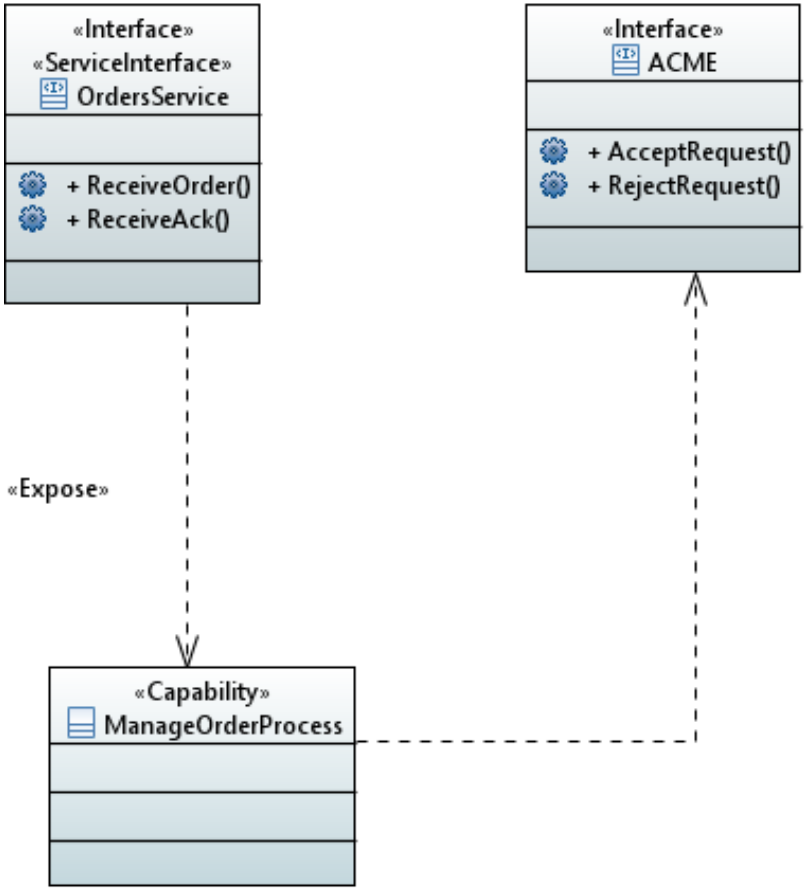
BPMN services identification

- A capability can be a pool in a collaboration
  - The service interface will possibly expose an operation for each incoming message
- A capability can be a service task
  - Systems used to implement the capability can be identified by lanes or using specific pools (that usually do not generate initiating messages)

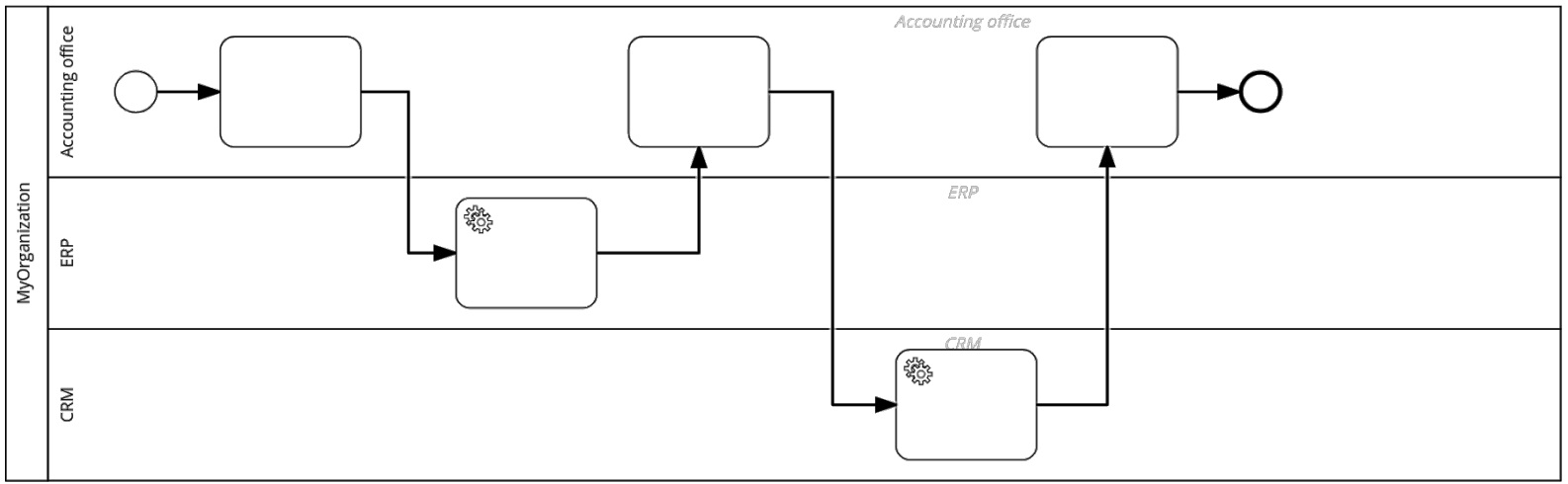
BPMN services identification

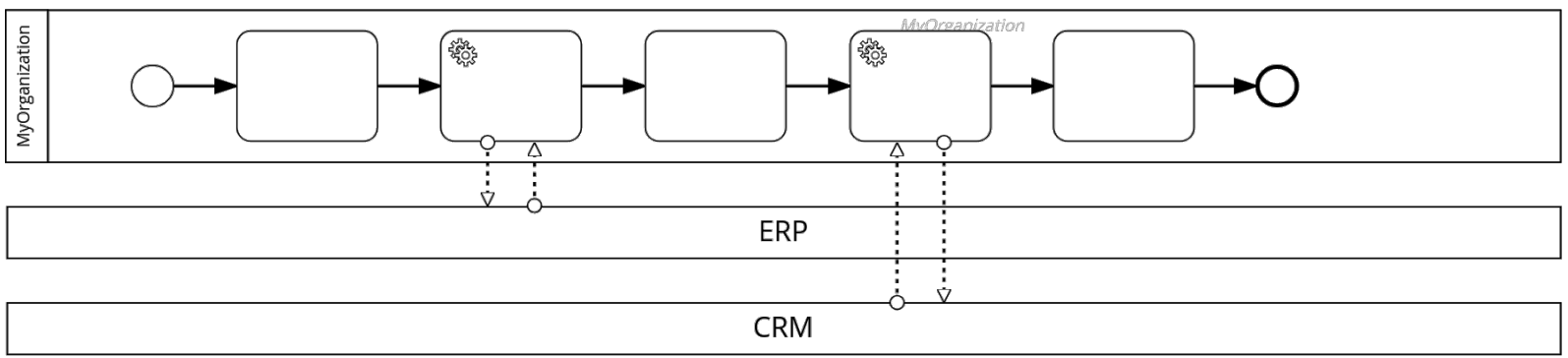


Process as a service



Systems and services modeling options

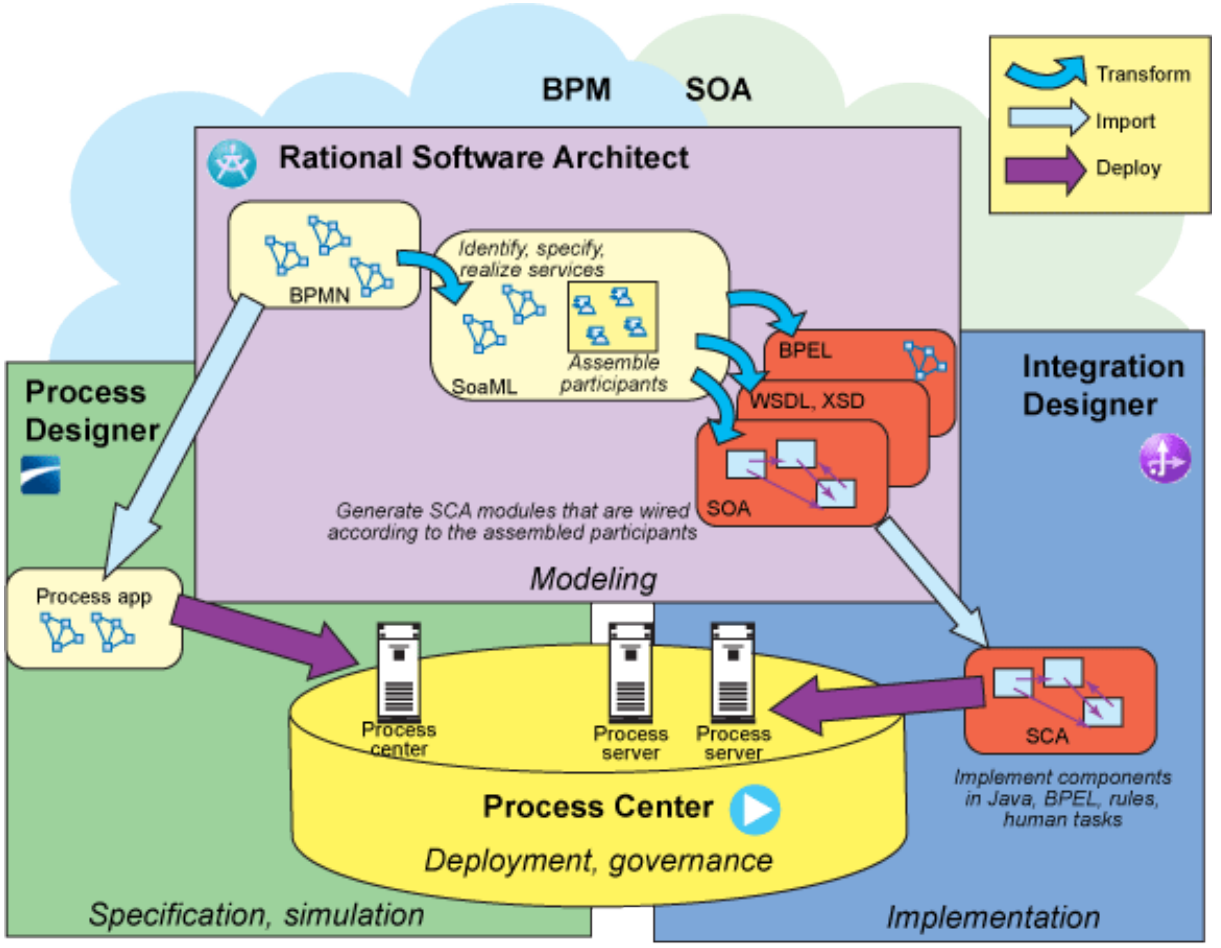




SOMA: simplified workflow (opt)

- Create contracts (using sequence diagrams)
- Participants in contracts trace to BPMN elements
- Add service and/or request ports to participants (opt: add provided and requested interfaces to ports)
- Tie providers and consumers with service channels
- Define the high-level service architecture describing how participants work together to collectively realize a business process

SOMA



[Source: IBM]