VALIDATING REQUIREMENTS USING GAIA ROLES MODELS

Nektarios Mitakidis, Pavlos Delias, **Nikolaos Spanoudakis**

Technical University of Crete

Outline

- Motivation
- Underlying technology
- The contribution
- A real world case study
- Discussion
- Conclusion

Motivation

- Had to develop a real world system, where
  - a personal assistant agent on a mobile device
  - provides personalized infomobility services
- The engineering team provided a technical solution (system architecture)
  - Broker agent
  - Semantic service matching
  - Added value service providers

And here is the challenge

- At the end or the architectural design (analysis phase)
  - We wanted to validate system (non-functional) requirements such as:
    - The system must respond within 10 seconds
  - We wanted to know how will the system scale
    - respond to increased service demand
    - What could be a risk mitigation strategy in this case?
- Are we ready implement our design?

A possible solution

- Simulation in software engineering can help
  - To forecast execution time
  - Identify bottlenecks
  - Test the response to increasing demand
- Business process models are useful as
  - There are a number of (open source) tools for simulating such models
  - They model the interaction of different business entities

The research question

- to define a process and tool that uses the output of the analysis / architectural design phase and creates a MAS model that can be simulated
- Considering the case of Gaia:
  - In Gaia, such a model is the Roles Model
  - Role activity is expressed in the form of **liveness formulas**

The underlying technology

- Gaia roles model
- Business process models

The Gaia Role Model

- Roles consist of:
- *activities*
- *protocols*
- *permissions*
- *responsibilities*
  - Liveness
  - Safety

Operator	Interpretation
x . y	x followed by y
x | y	x or y occurs
x+	x occurs 1 or more times
x*	x occurs 0 or more times
x ω	x occurs infinitely often
[x]	x is optional
x || y	x and y interleaved

Role Schema: CUSTOMER (CUST)

Description:
Organisation or individual requiring a service quote

Protocols and Activities:
MakeCall, GiveRequirements

Permissions:
generates *customerDetails* // *Owner of customer information*
customerRequirements // *Owner of customer requirements*

Responsibilities
Liveness:
CUSTOMER = (MakeCall. GiveRequirements)+

Safety:
- **true**

GiveRequirements	
CUST	QM
provide service details	customerRequirements

Example from Wooldridge et al., 2000

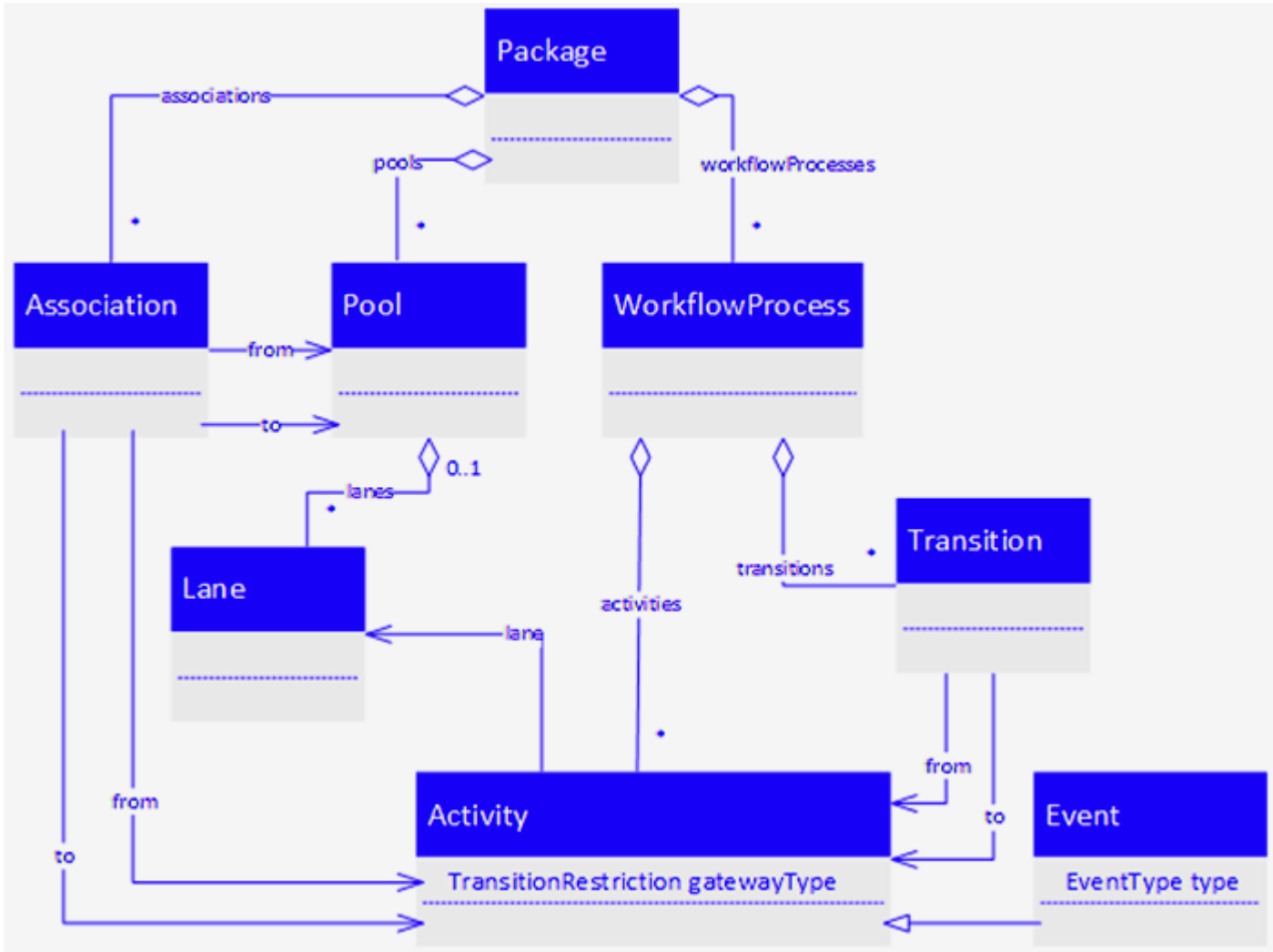
Business Process Modeling Notation (BPMN)

- BPMN is a language for modeling business processes
  - An OMG Standard
  - Allows for simulation through a plethora of tools
- The XML Process Definition Language (XPDL version 2.1)
  - standard supported by the Workflow Management Coalition (WfMC)
  - used today by more than 80 different products to exchange process definitions

BPMN model elements

- **Events,** illustrated as circles.
- **Activities**
  - **Tasks** (rounded rectangles)
  - **Gateways** (diamond shapes)
- **Transitions** (solid line and arrowhead)
- **Message Flows** (dotted line and arrowhead)
- **Swimlanes** in which different participants are shown in their *Pools* or *Lanes*

XPDL meta-model



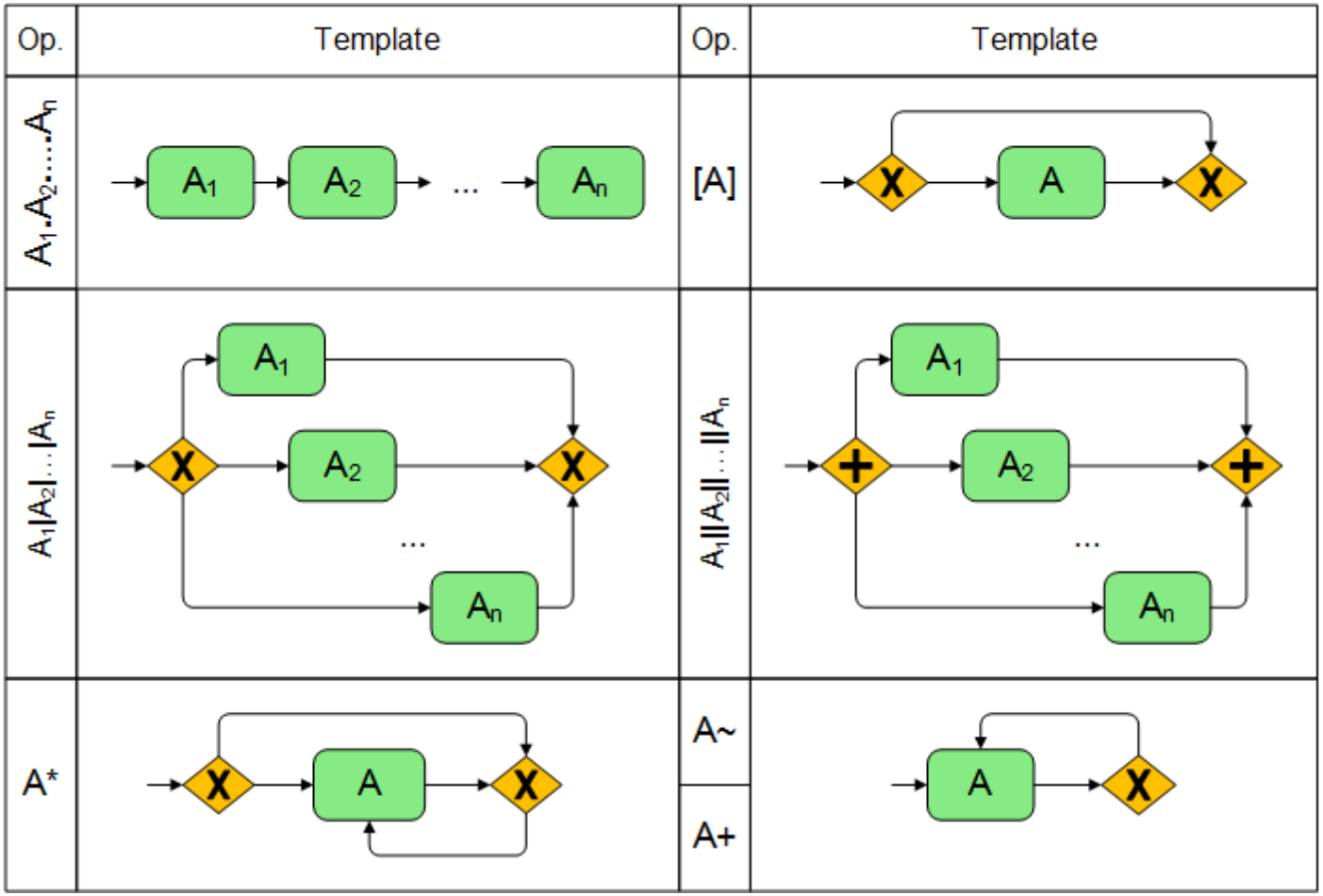
The contribution

- We set to
  - Transform roles' liveness to process models
  - Integrate the different roles process models to a common MAS process model
- It is not a trivial task:
  - Select target model: XPDL, BPMN 2, BPEL, etc
  - operators transformation is not straight-forward
    - the semantics are quite ambiguous
    - a number of features are available
    - different tools support different features

Single role transformation algorithm

- Input: a set of liveness formulas
- The transformation algorithm:
  - Initiates a process model with a ***start*** event
  - Gets the first formula and applies the relevant Gaia operator transformation template
  - processes the elements of the right hand side expression of the formula from left to right and if they are expanded in another formula or are complex expressions themselves it again applies the relevant templates
  - Finally, it adds a transition to an ***end*** event

The transformation templates



An example

Role: ComplexProvider

…

Liveness:

CP = ComplexService+

ComplexService = ReceiveComplexServiceRequest. DecideRouteType.
SimpleService. SortRoutes. SendComplexServiceResponse

SimpleService = SendSimpleServiceRequest. ReceiveSimpleServiceResponse

An example

Role: ComplexProvider

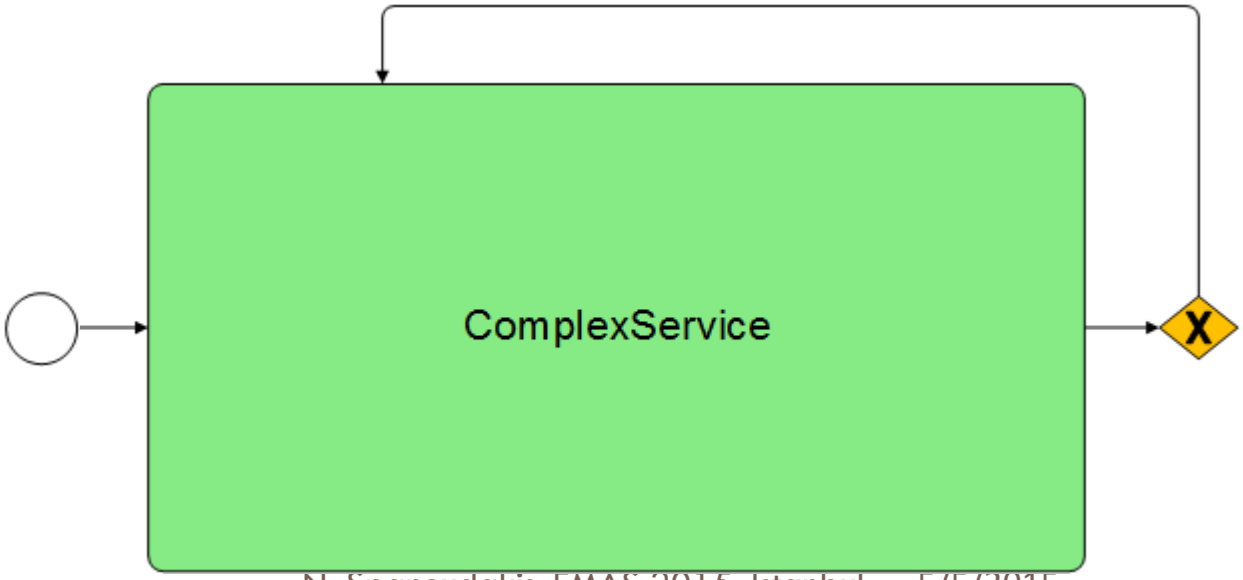
…

Liveness:

CP = **ComplexService+**

ComplexService = ReceiveComplexServiceRequest. DecideRouteType. SimpleService. SortRoutes. SendComplexServiceResponse

SimpleService = SendSimpleServiceRequest. ReceiveSimpleServiceResponse



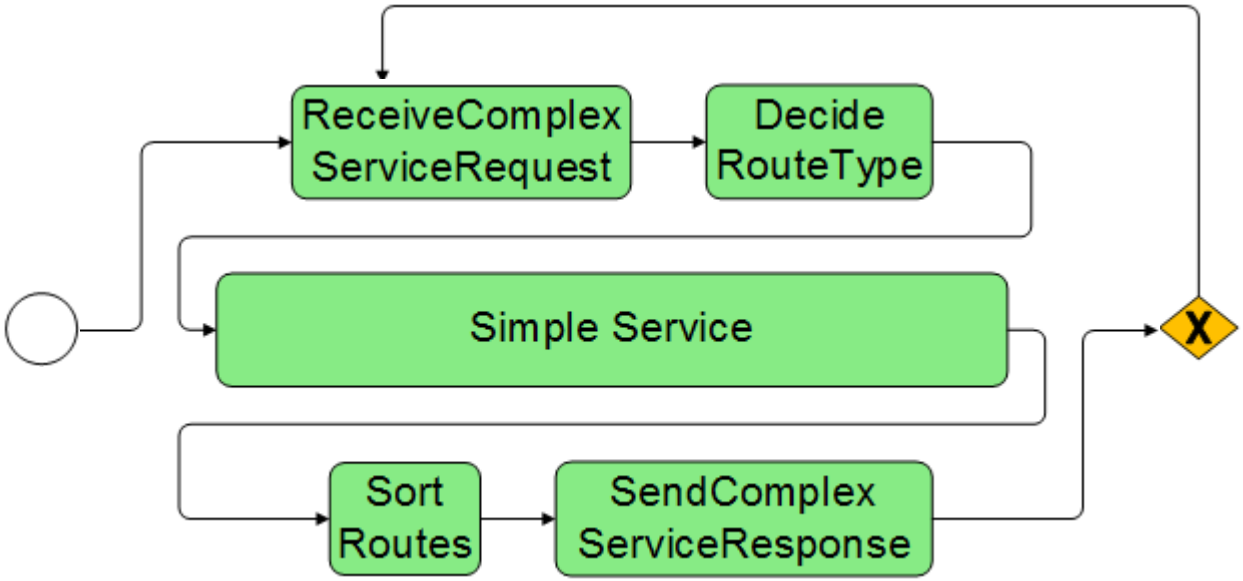
An example

Role: ComplexProvider

…

Liveness:

CP	= **ComplexService+**
ComplexService	**= ReceiveComplexServiceRequest. DecideRouteType. SimpleService. SortRoutes. SendComplexServiceResponse**
SimpleService	= SendSimpleServiceRequest. ReceiveSimpleServiceResponse



An example

Role: ComplexProvider

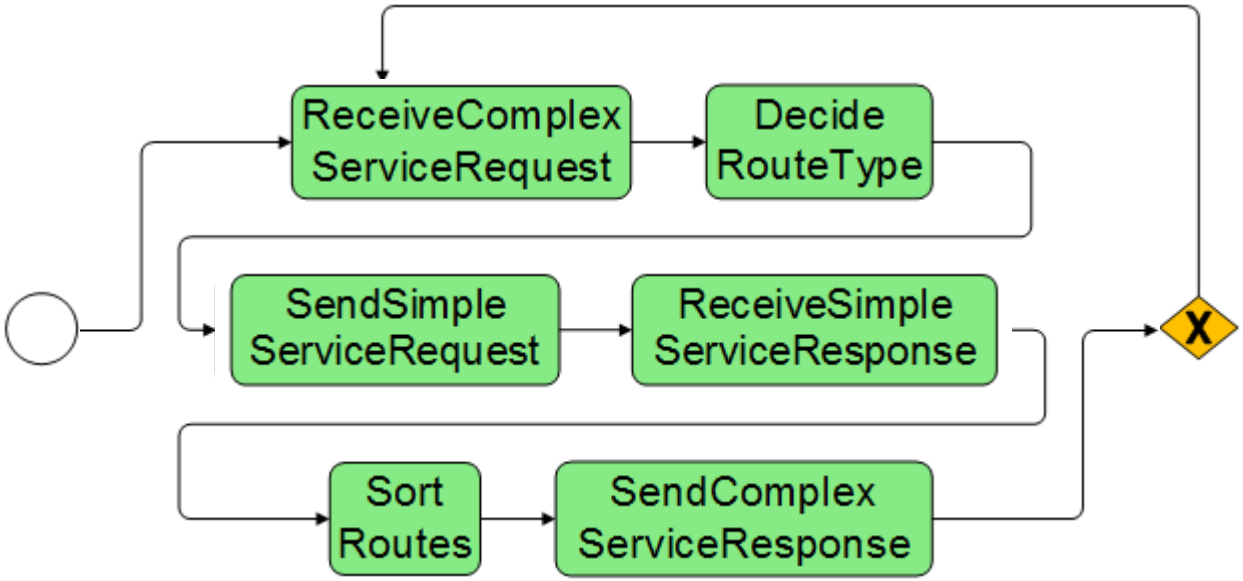
…

Liveness:

CP = **ComplexService+**

ComplexService = **ReceiveComplexServiceRequest. DecideRouteType. SimpleService. SortRoutes. SendComplexServiceResponse**

SimpleService = **SendSimpleServiceRequest. ReceiveSimpleServiceResponse**



An example

Role: ComplexProvider

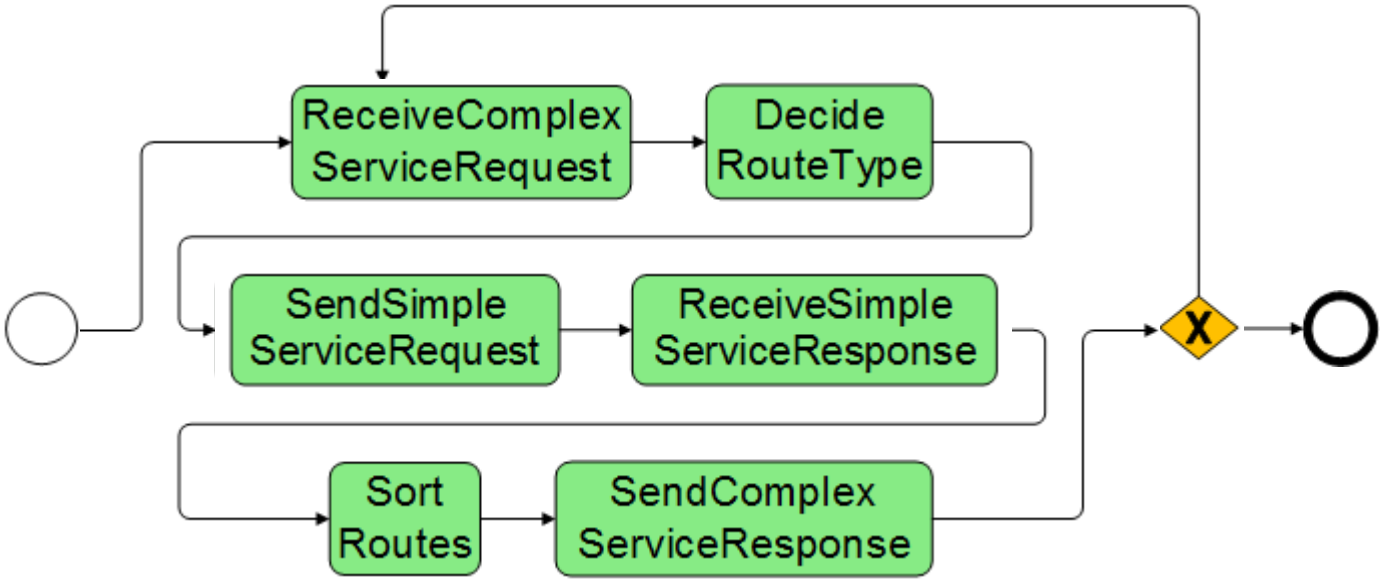
…

Liveness:

CP = **ComplexService+**

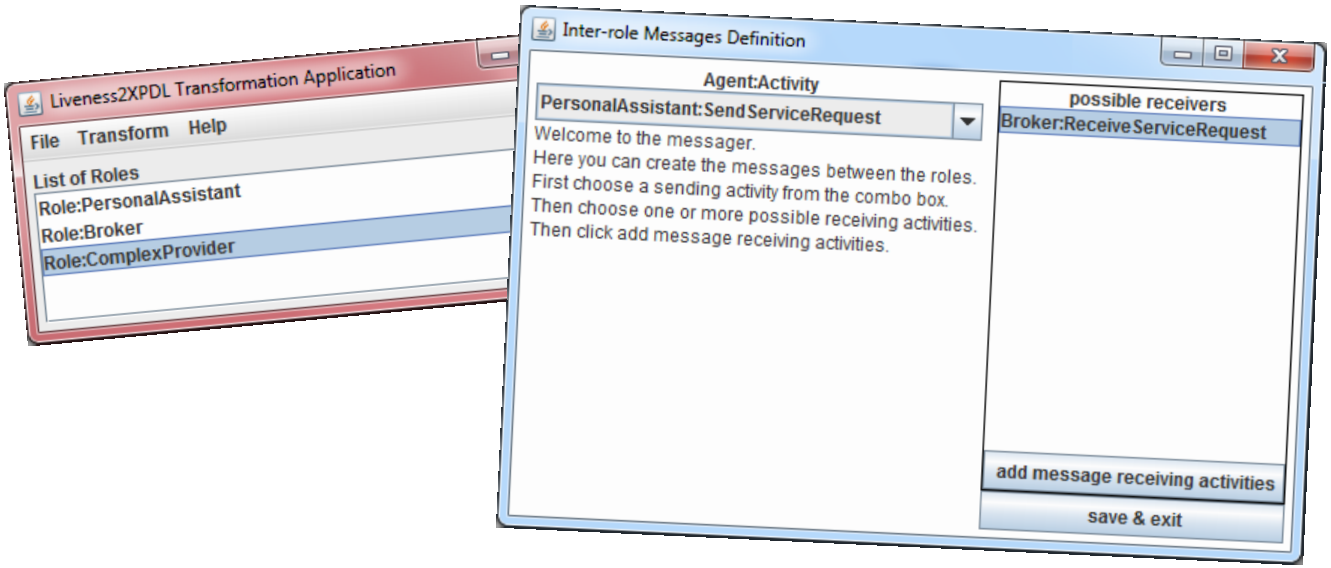
ComplexService = **ReceiveComplexServiceRequest. DecideRouteType. SimpleService. SortRoutes. SendComplexServiceResponse**

SimpleService = **SendSimpleServiceRequest. ReceiveSimpleServiceResponse**



The MAS case

- What about when more than one roles exist?
- Integrate them by connecting message send and receive activities with the relevant message flows



A case study

- The scenario
  - In our system a personal assistant asks for a service to a broker
  - The broker
    - provides simple requests itself
    - uses complex providers, which provide added values services (personalized)
  - Complex service providers use the broker for accessing simple services

After the analysis phase…

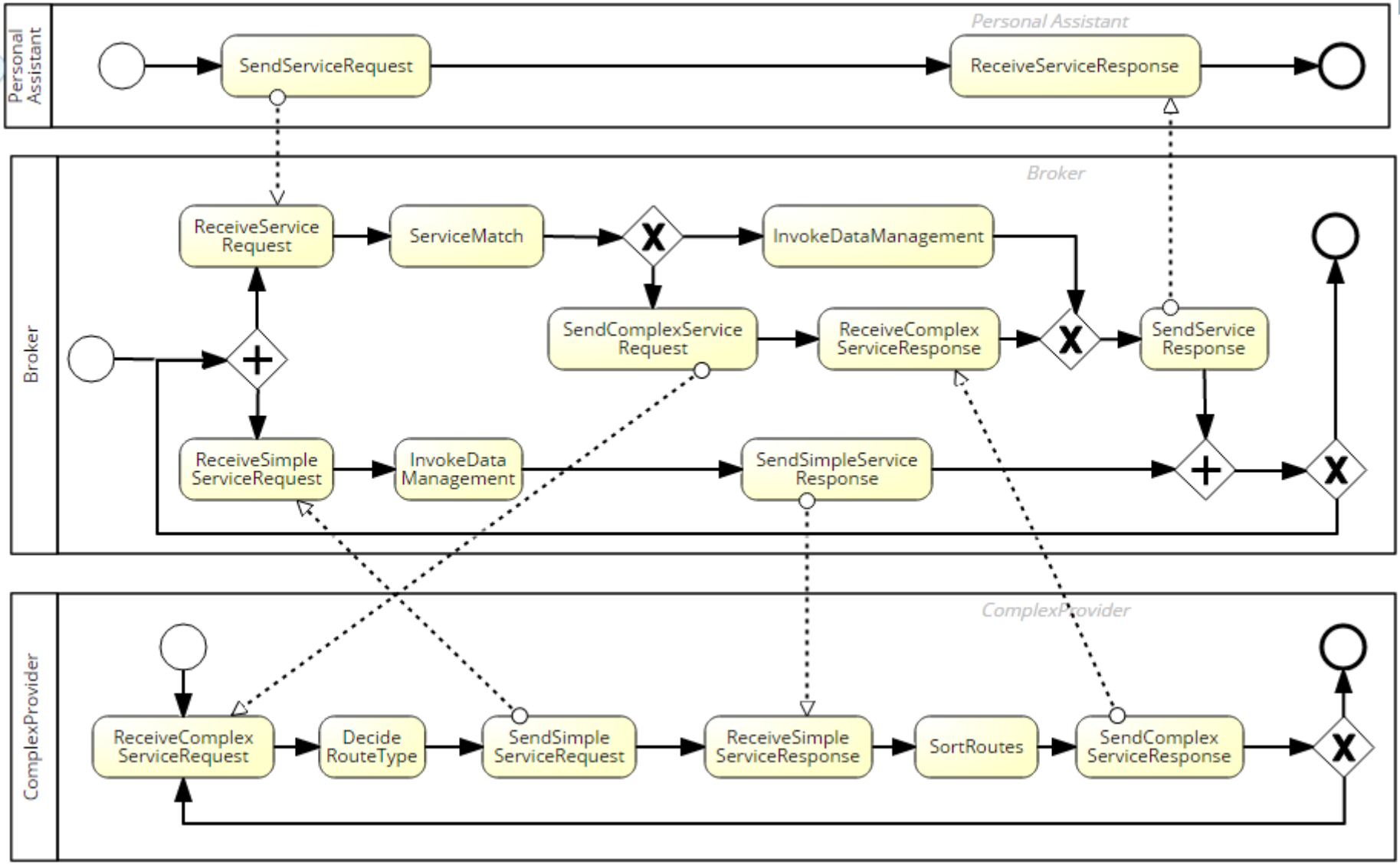
- We want to validate the following system non-functional requirements:
  - The personal assistant requests happen every 30 seconds in average
  - The broker must respond within 10 seconds
- Determine how will the system scale
  - i.e. respond to increased service demand
  - Complete our risk analysis with a mitigation strategy

More Role Models

Role: ***Personal Assistant (PA)***
Protocols:
request for services: service requester (SR)
Liveness:
Personal Assistant = *SendServiceRequest. ReceiveServiceResponse*

Role: ***Broker (BR)***
Protocols:
request for services: service requester (SR),
request for services: service provider (SP)
Liveness:
Broker = (ServicePAs || ServiceCP)+
ServicePAs = ReceiveServiceRequest. ServiceMatch. (InvokeDataManagement |
(SendComplexServiceRequest. ReceiveComplexServiceResponse)). SendServiceResponse
ServiceCP = ReceiveSimpleServiceRequest. InvokeDataManagement.
SendSimpleServiceResponse

The BPMN model



Definition of activities

- The XOR gateway after the service match activity is set to a 50% for each flow option (simple or complex service)
- Tasks definitions (based on prototypes):

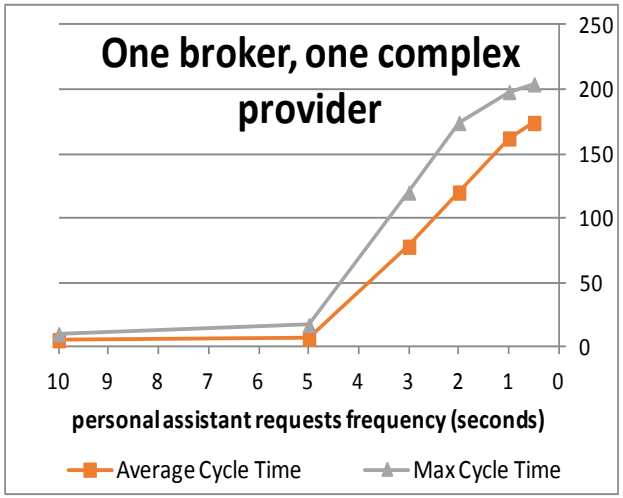
Task Name	Distribution	Mean Value	Standard Deviation	Role
ServiceMatch	Normal	0.254	0.112	Broker
InvokeDataManagement	Normal	2.639	1.113	Broker
SendComplexServiceRequest	Normal	0.007	0.006	Broker
SendServiceRequest	Normal	0.063	0.024	Personal Assistant

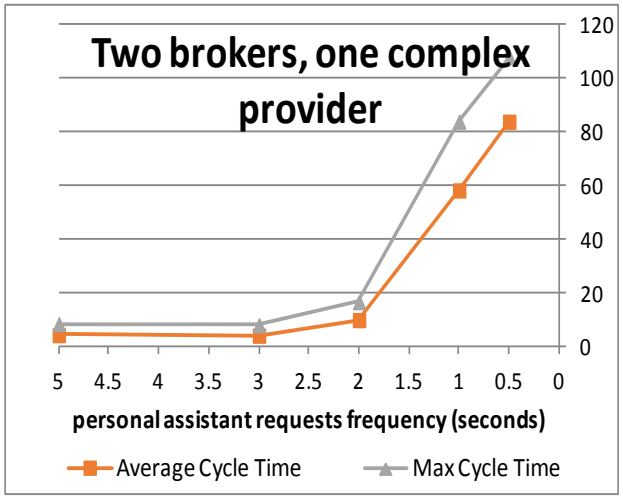
Simulating the scenarios

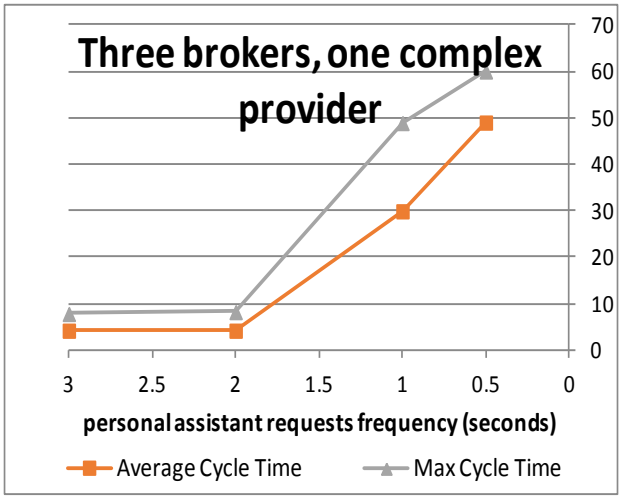
- The basic scenario was simulated with the parameters:
  - PA requests: every 30, 20, 10, 5, 3, 2, 1, and, 0.5 seconds in average
  - Resources: 1-3 brokers and 1-2 CPs

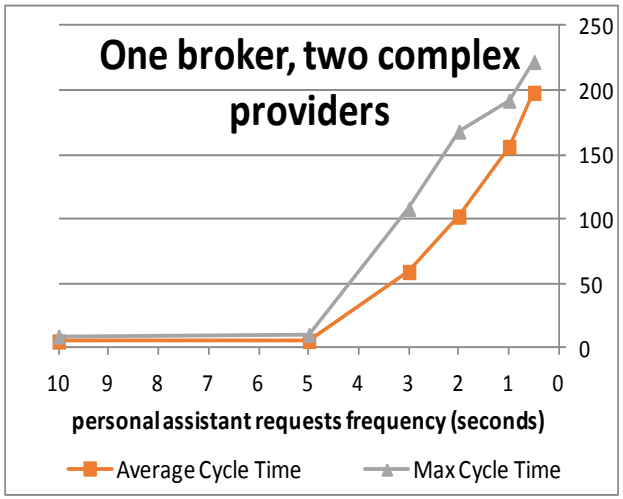
Request Every	Number of Brokers	Number of Complex Providers	Complex Provider Utilization	Broker Utilization	Average Cycle Time	Max Cycle Time	Min Cycle Time
30	1	1	0.96%	15.87%	5.7	10.3	2
20	1	1	1.52%	21.33%	5.2	11.2	2.3
10	1	1	3.33%	44.64%	5.5	10.2	2.2
5	1	1	6.14%	84.54%	6.7	17.2	2.3
3	1	1	7.70%	99.49%	78	120	5.4
2	1	1	7.66%	99.54%	120	174	5.2
1	1	1	7.31%	99.53%	162	198	9.6
30	2	1	1.04%	7%	4	6.7	2
30	1	2	0.52%	14.81%	5.4	9	2.5

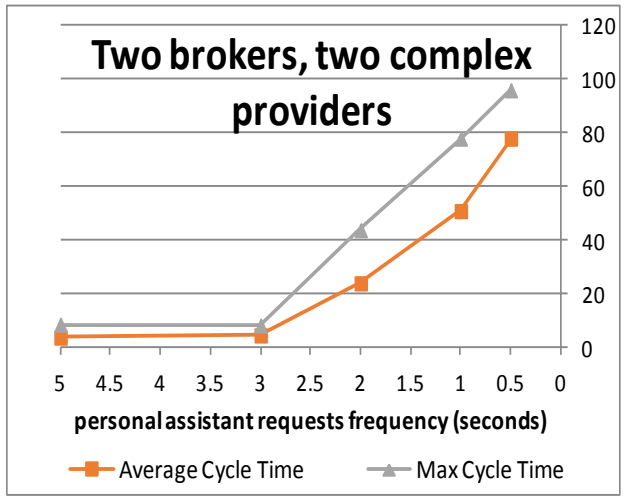
Results

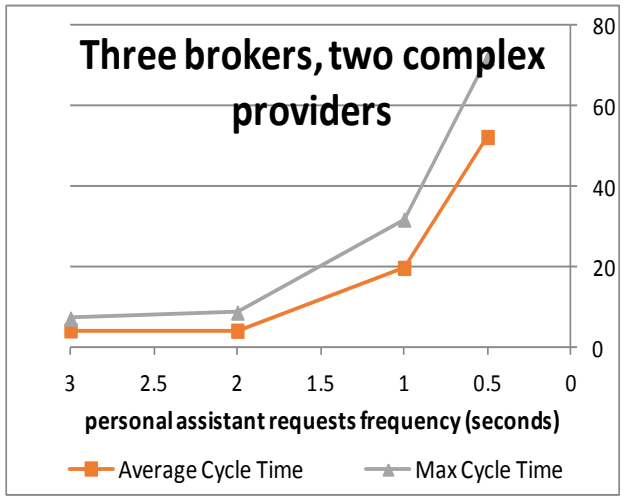

One broker, one complex provider
250
200
150
100
50
0
10 9 8 7 6 5 4 3 2 1 0
personal assistant requests frequency (seconds)
Average Cycle Time
Max Cycle Time


Two brokers, one complex provider
120
100
80
60
40
20
0
5 4.5 4 3.5 3 2.5 2 1.5 1 0.5 0
personal assistant requests frequency (seconds)
Average Cycle Time
Max Cycle Time


Three brokers, one complex provider
70
60
50
40
30
20
10
0
3 2.5 2 1.5 1 0.5 0
personal assistant requests frequency (seconds)
Average Cycle Time
Max Cycle Time


One broker, two complex providers
250
200
150
100
50
0
10 9 8 7 6 5 4 3 2 1 0
personal assistant requests frequency (seconds)
Average Cycle Time
Max Cycle Time


Two brokers, two complex providers
120
100
80
60
40
20
0
5 4.5 4 3.5 3 2.5 2 1.5 1 0.5 0
personal assistant requests frequency (seconds)
Average Cycle Time
Max Cycle Time


Three brokers, two complex providers
80
60
40
20
0
3 2.5 2 1.5 1 0.5 0
personal assistant requests frequency (seconds)
Average Cycle Time
Max Cycle Time

Discussion

- complexity of our algorithm is low: $O(n^2)$
- A number of existing works allow for
  - mapping of BPMN diagrams to a normalized form checking for certain structural properties (Endert et al., 2007)
  - improving a process model representing the behavior of agents (Szimanski et al., 2013)
  - transforming BPMN models to agent-oriented models in the Prometheus methodology (Khanh Dam and Ghose, 2010)

What about Message Send and Receive Activities?

- Protocols need to be expressed as a series of activities (connected with Gaia operators) for each participant
- Simple original Gaia protocols such as "GiveRequirements" may be used
- In the Gaia2JADE process and ASEME the protocols model is also compatible

Concluding…

- We can use the Liveness2XPDL tool to
  - Determine if a system analysis model meets specific requirements
  - Determine how the system would scale
  - Decide on risk mitigation strategies
  - Identify errors in system conception and propose strategies for resolving them
  - Optimize the system regarding resource (agents) allocation and utilization

Future directions

- Cater for
  - better message-event handling and
  - message broadcasting and collecting
- Extend the WADE toolkit for automatic JADE agents generation
  - WADE is a software platform based on JADE that provides support for the execution of tasks defined according to the workflow metaphor (jade.tilab.com).
- Human-agents teams collaboration

Thank you

- You can download the Liveness2XPDL tool from

https://github.com/ASEMEtransformation/Liveness2XPDL

- Send us your comments

nikos@amcl.tuc.gr