

# *Semi-Automatic Distribution Pattern Modeling of Web Service Compositions using Semantics*

Ronan Barrett and Claus Pahl

`ronan.barrett@computing.dcu.ie`



School of Computing,  
Dublin City University,  
Dublin, Ireland

# Motivation

---

- Web service compositions are inflexible, can this inflexibility be reduced?
  - Coding effort is considerable
  - Fixed architecture/QoS dependent patterns lost
- General goal: Increase flexibility by explicitly modeling the compositional aspects
  - No coding effort
  - Model based development
  - Flexible architecture/QoS dependent patterns visible
- Specific goal: Reduce modeling effort through automation
  - Use Web service semantics
  - Reduce software architect's workload

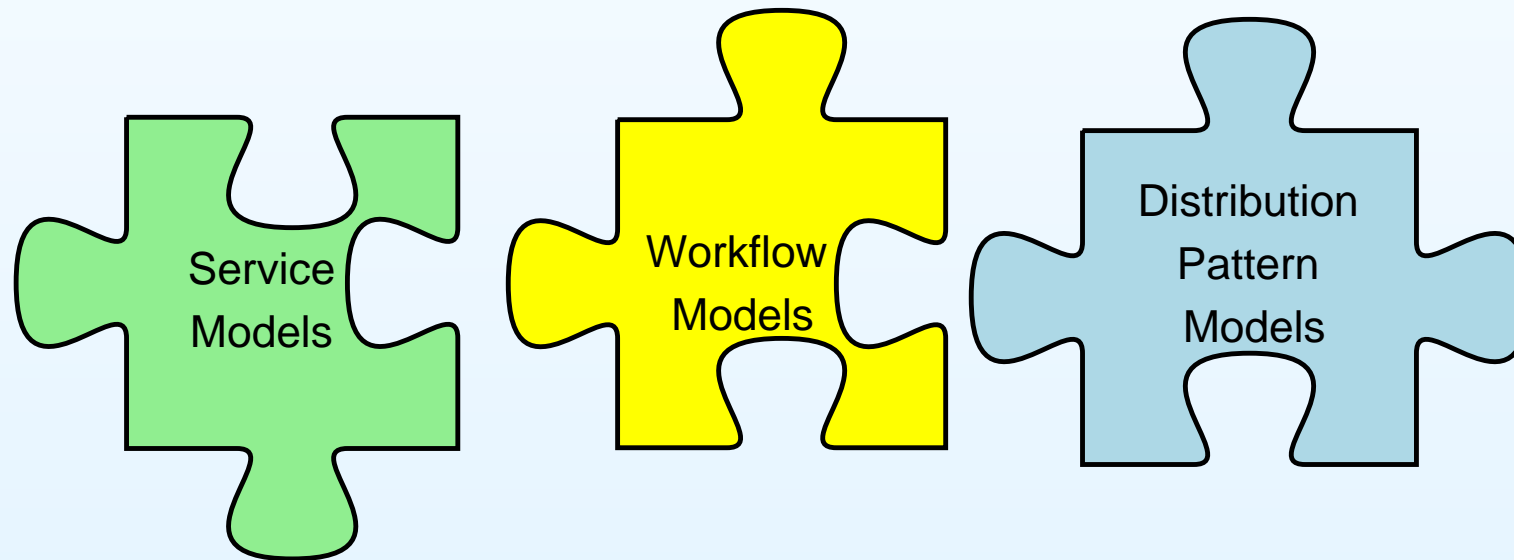
# Outline

---

- Modeling aspects
- Distribution Patterns
- Semantically assisted modeling & transformation
- Conclusions/Questions

# Modeling Aspects

- Service modeling \* e.g. WSDL as UML
- Workflow modeling † e.g. WS-BPEL as UML
- Distribution pattern modeling ‡



---

\* Armstrong

† Grønmo & Solheim

‡ Barrett & Pahl - IWMEC 2006

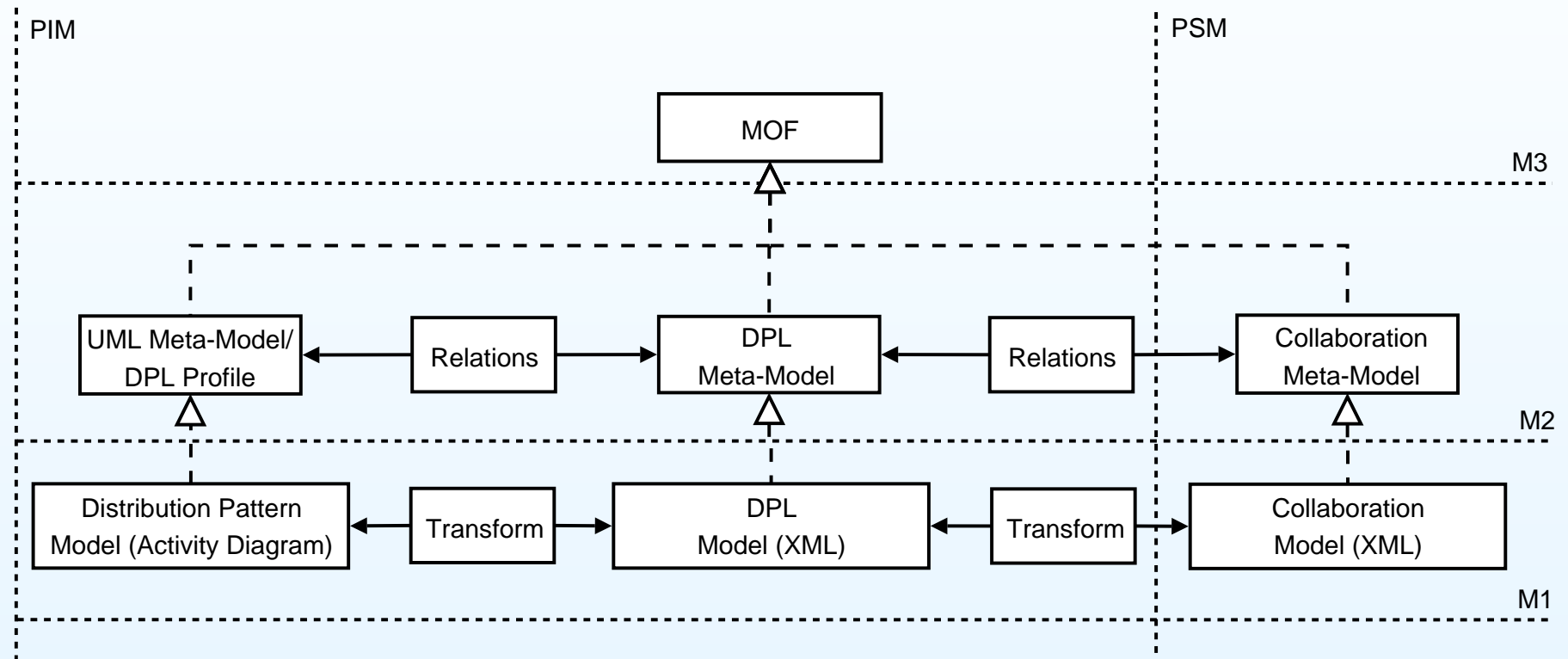
# Distribution Patterns

---

“Expression of how a composed system is to be deployed”

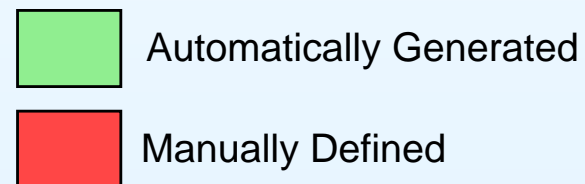
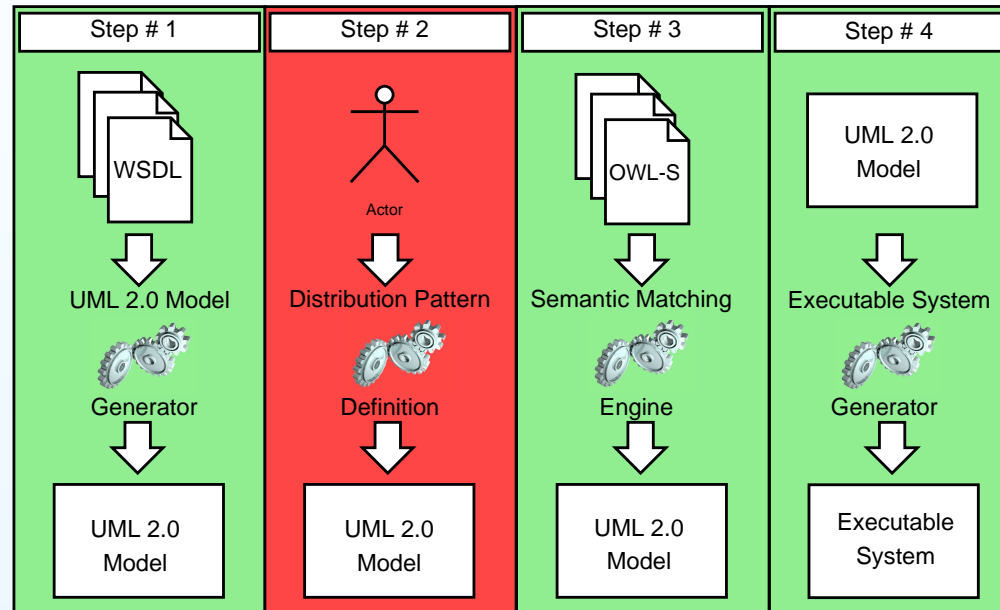
- Patterns express varying QoS metrics
- Patterns independent from workflow
- Architecture is flexible
- Pattern categories
  - Core patterns
  - Auxiliary patterns
  - Complex patterns

# Modeling Approach



- Based on OMG's MDA approach
- Formal approach based on meta-model relations
- Distribution patterns modeled using UML Activity diagrams

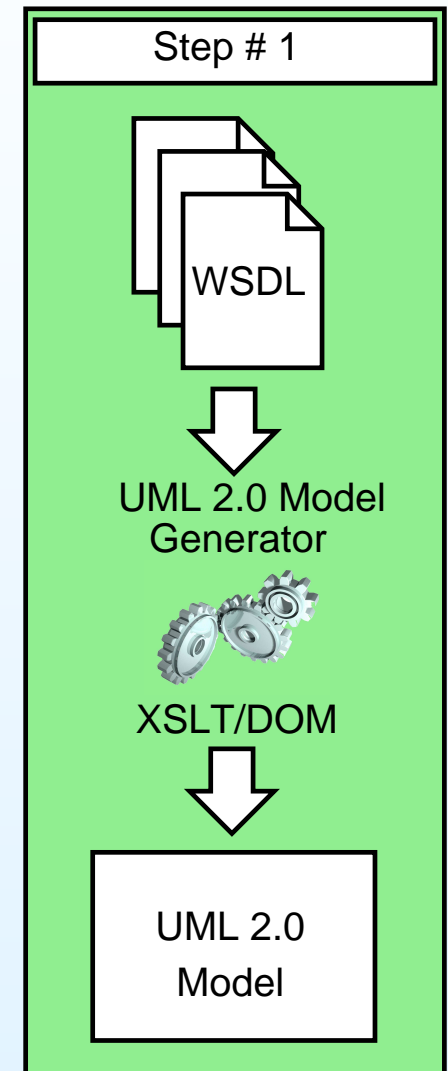
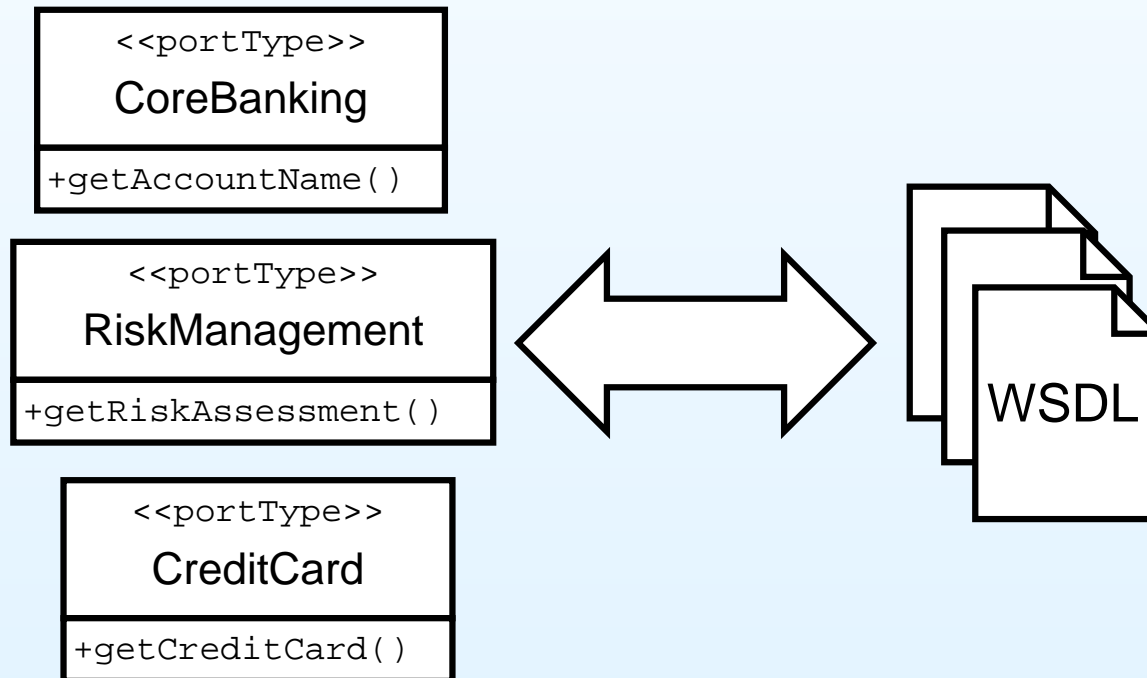
# Semi-Automated Modeling and Transformation



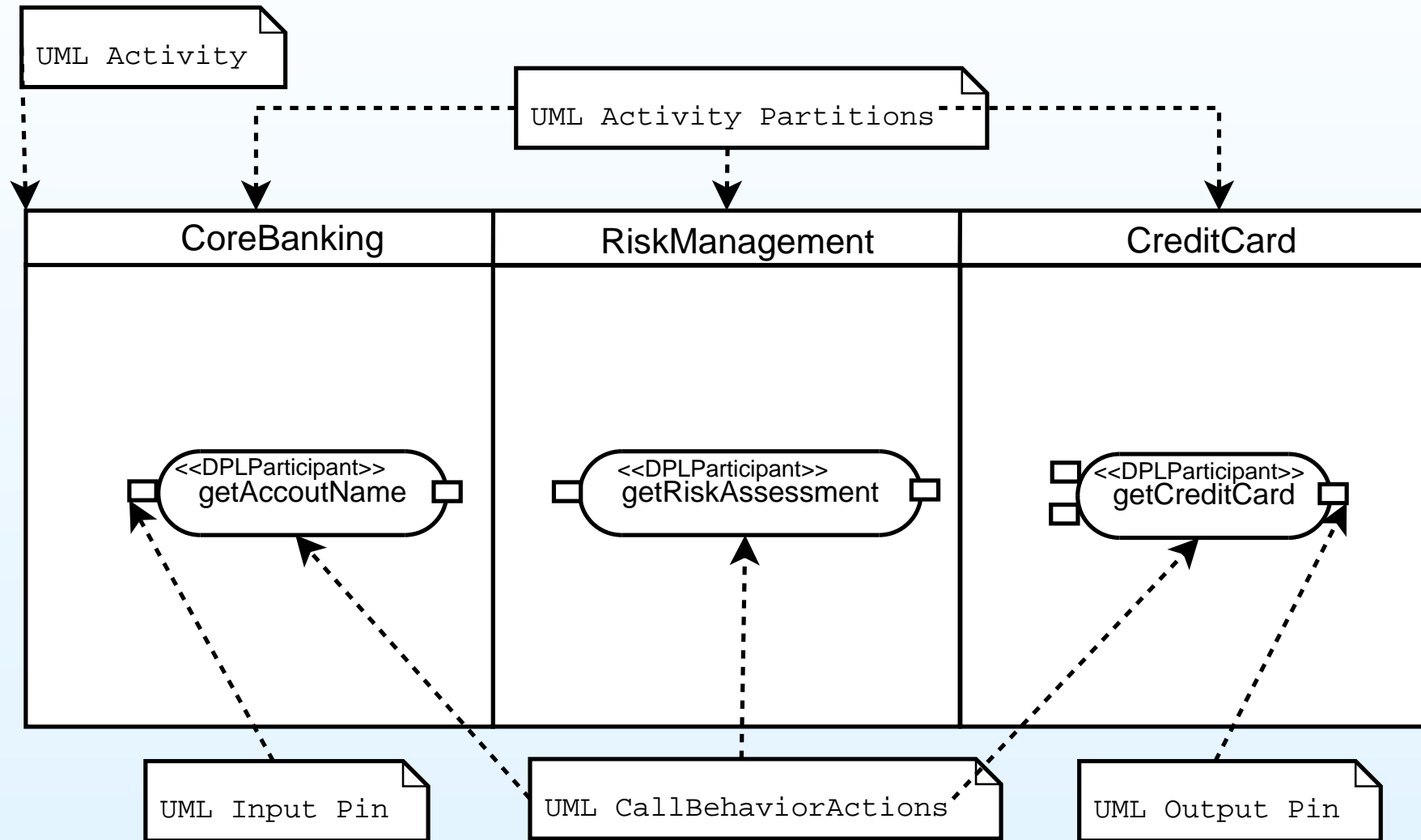
- Semantics reduce manual modeling effort
- Can reduce modeling errors
- Assists in executable system generation

# Step 1a - From Interface To UML Model

- Web service interfaces as input
- Generate UML activity diagram
- Automatically apply UML Profile

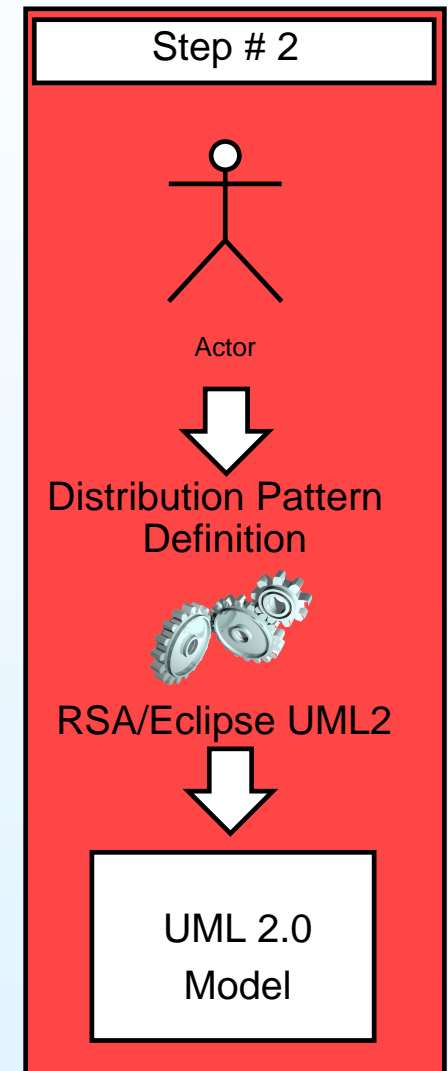


# Step 1b - From Interface To UML Model



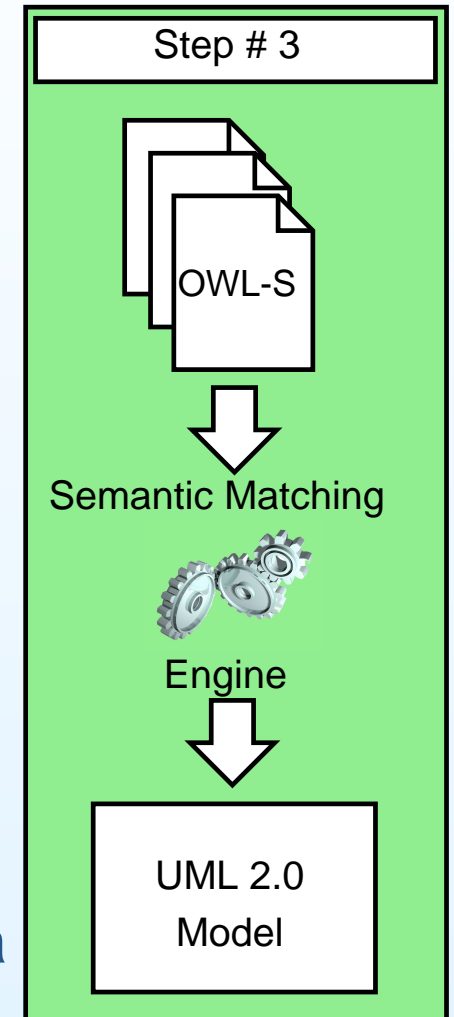
## Step 2 - Distribution Pattern Definition

- Manually apply values to UML Profile
  - Select distribution pattern
  - Select collaboration language
  - Select roles for actions
- UML 2.0/XMI 2.0 tool support
  - Rational Software Architect
  - Eclipse UML 2

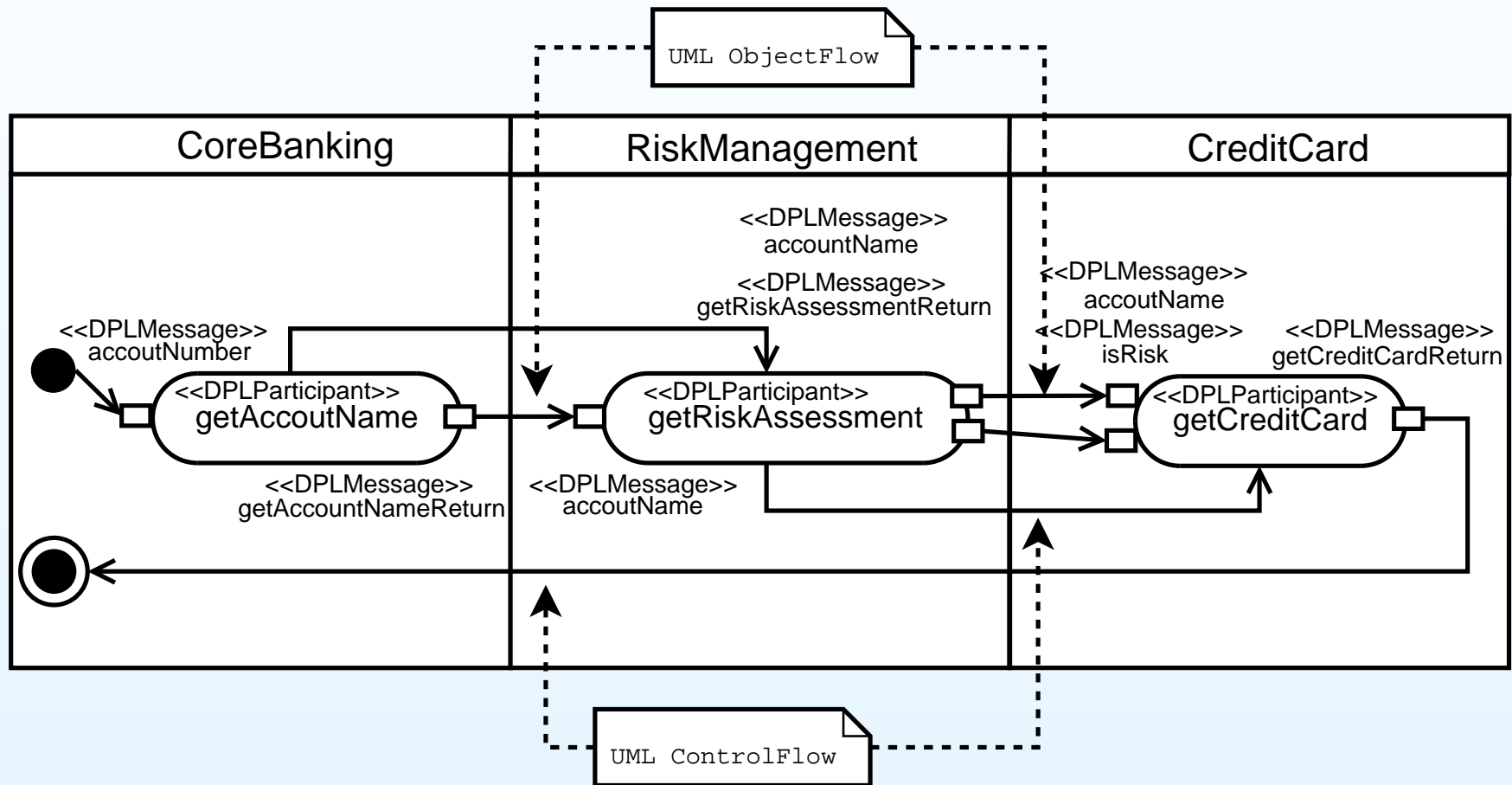


## Step 3 - Semantic Matching and Integration

- Previously performed manually
- Use semantic matching to apply connections
  - Applies ObjectFlows
  - Applies ControlFlows
  - Applies initial/final nodes
- Achieved using OWL-S descriptions
  - Atomic process models describe inputs/outputs
  - Matches made based upon ontological definitions
  - Output pins connected to input pins
  - Service execution order dependent on data flow

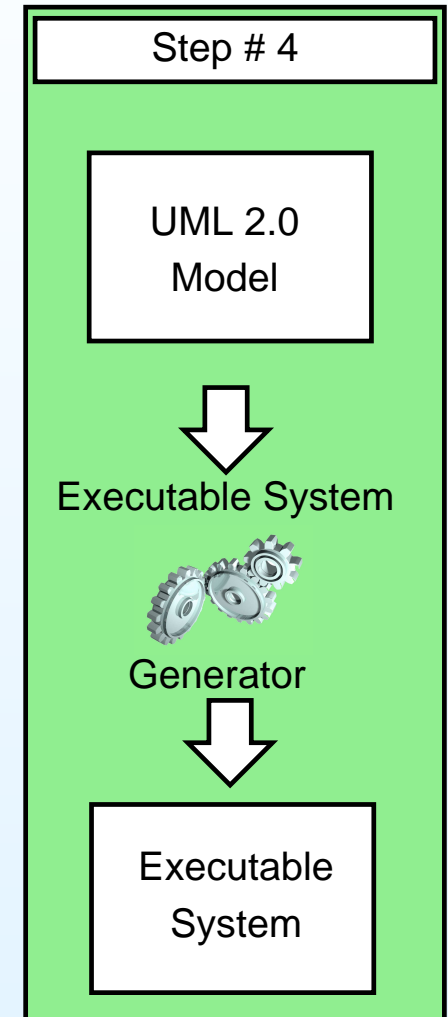


# Step 3b - Semantic Matching and Integration



## Step 4 - Model to Executable System

- Generates
  - Interaction logic document(s)
  - Interface document(s)
  - Deployment descriptor(s)
- System is ready for deployment



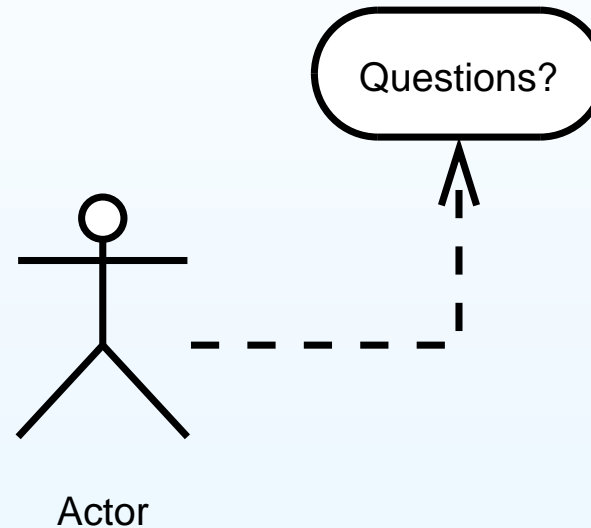
# Conclusion

---

- Motivated new service modeling aspect
  - Distribution patterns
- Presented four step modeling approach
  - Standards based
  - Extensible
- Illustrated use of semantics
  - Enables semi-automation of modeling approach
  - Reduces modeling effort & possibly errors
  - Assists in executable system generation

# Questions

Go raibh maith agat!



Open Source tools used to make this presentation.....



Funded by the Irish Research Council for Science, Engineering and Technology: funded by the National Development Plan.

