

# *Model Driven Design of Distribution Patterns for Dynamic Web Service Compositions*

Ronan Barrett and Claus Pahl

School of Computing, Dublin City University

Dublin, Ireland

Lucian M. Patcas and John Murphy

School of Computer Science and Informatics, University College Dublin

Dublin, Ireland

[ronan.barrett@computing.dcu.ie](mailto:ronan.barrett@computing.dcu.ie)



# Motivation

---

- Web service compositions are inflexible, can this inflexibility be reduced?
  - Coding effort is considerable
  - Architecture is fixed
  - QoS dependent patterns are lost
- Goal: Increase flexibility by explicitly modeling the compositional aspects
  - No coding effort
  - Model based development
  - Flexible architecture
  - Make QoS dependent patterns visible

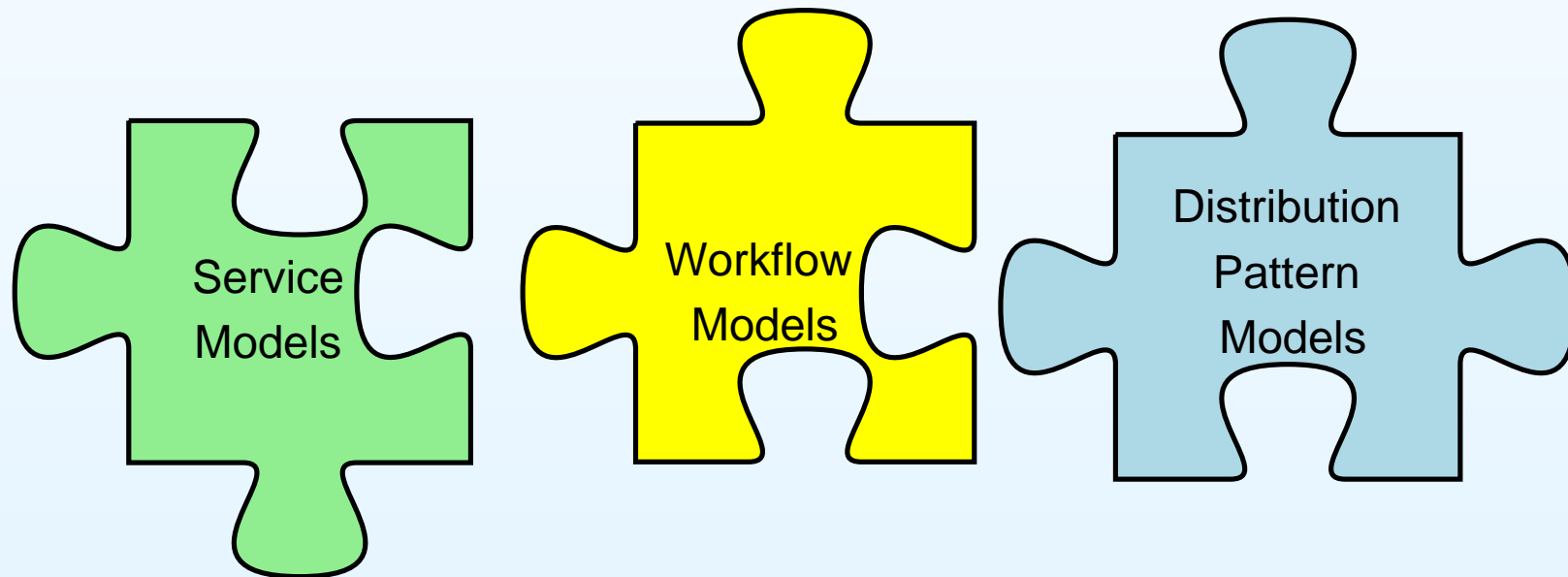
# Outline

---

- Modeling aspects
- Modeling approach & transformation technique
- Tool implementation
- Future work
- Conclusions/Questions

# Modeling Aspects

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



---

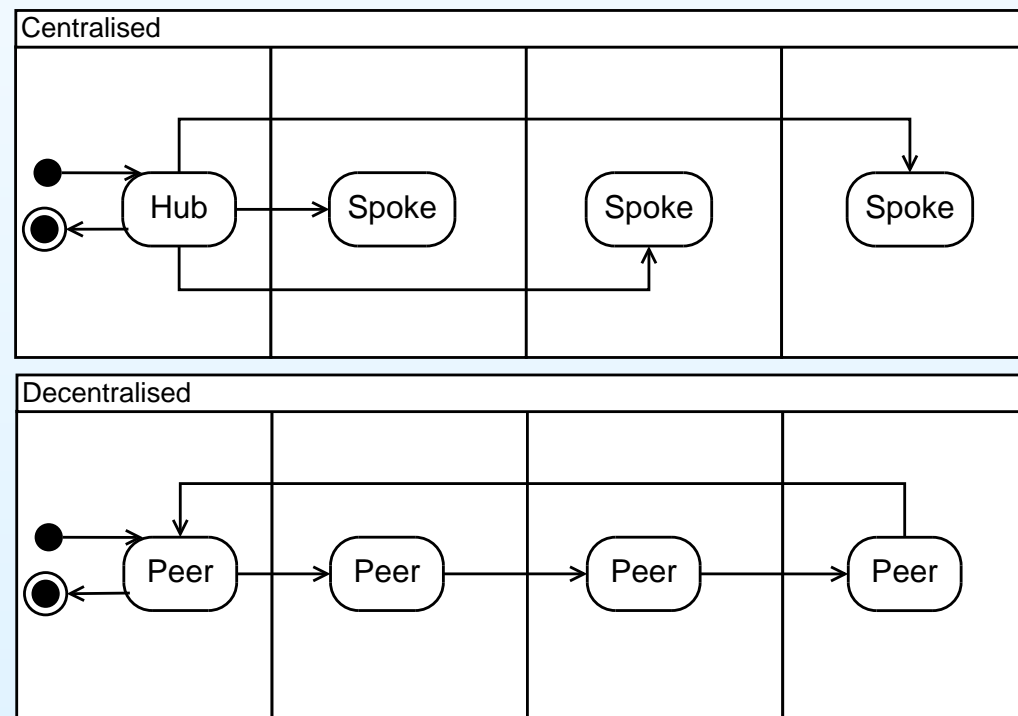
\*Armstrong

†Grønmo & Solheim

# Distribution Patterns

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

- Patterns express varying QoS metrics
- Patterns independent from workflow
- Architecture is flexible



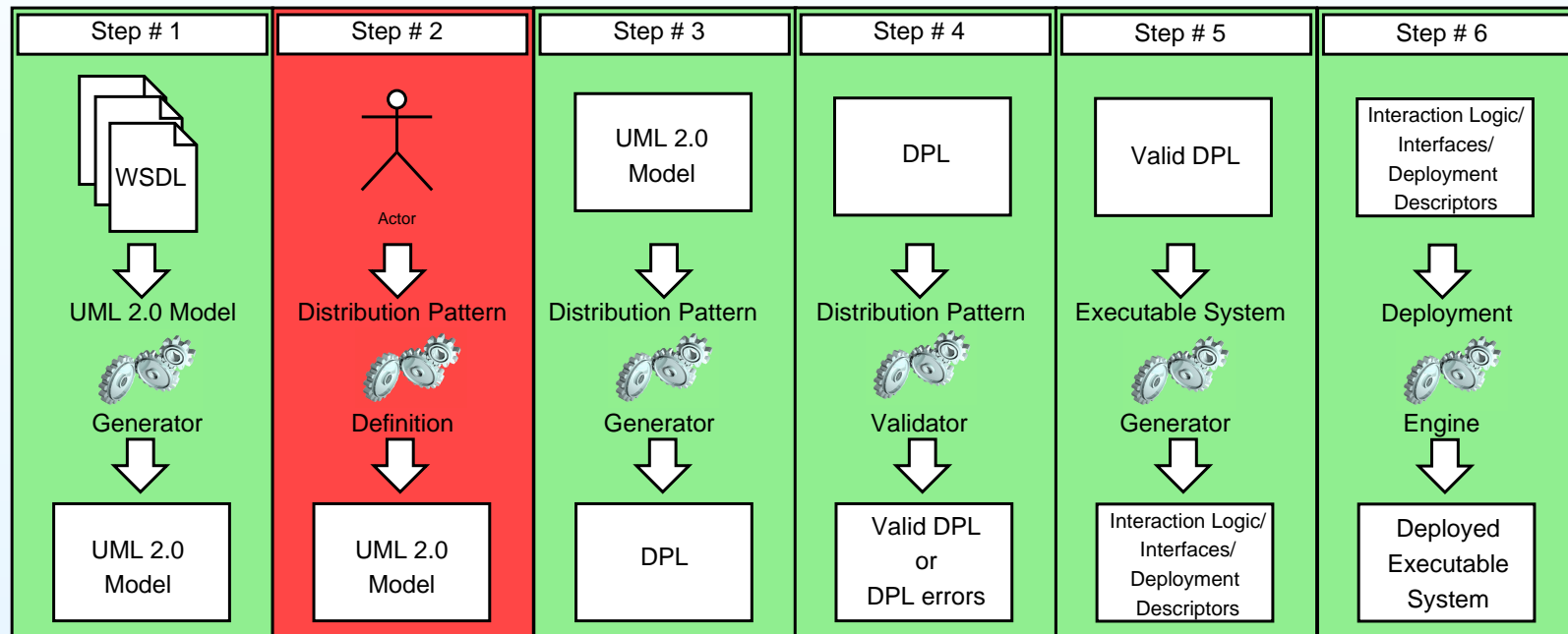
# Distribution Pattern Types

---

- Centralised shared hub
- Centralised dedicated hub
- Decentralised shared hub
- Decentralised dedicated hub
- Ring
- Hierarchical
- Complex



# Modeling and Transformation Technique



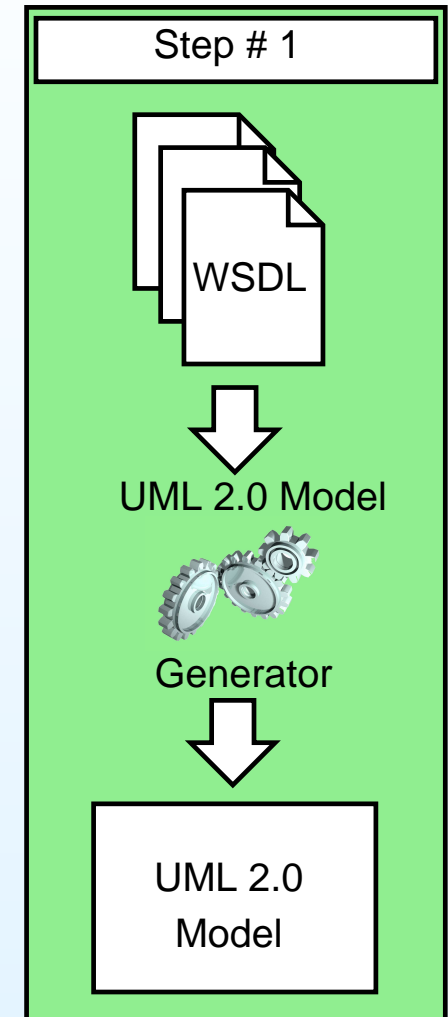
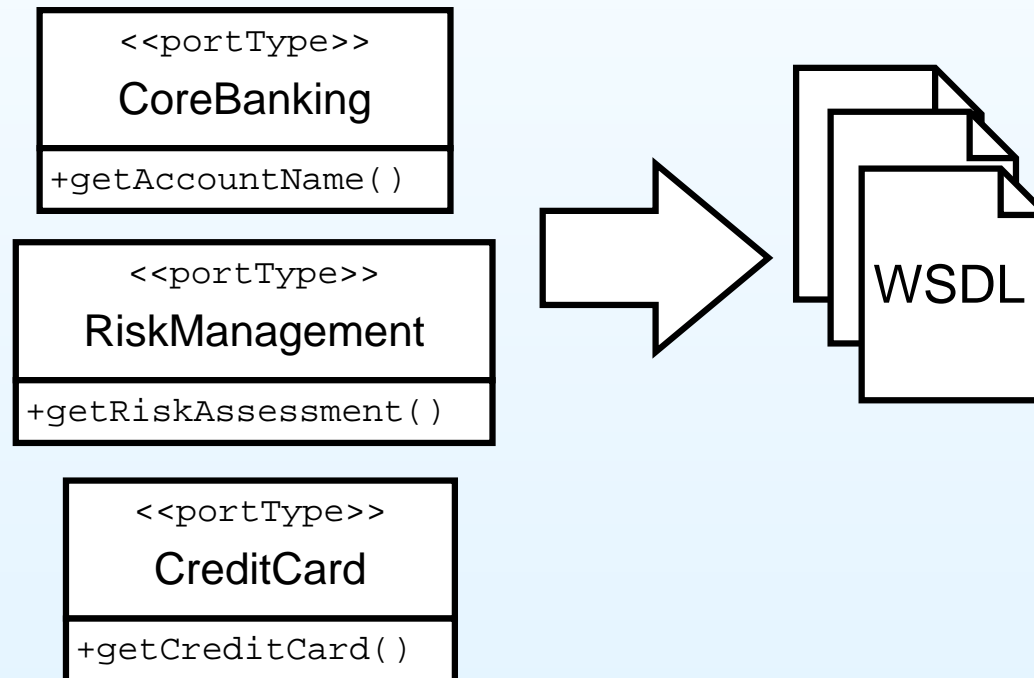
Automatically Generated



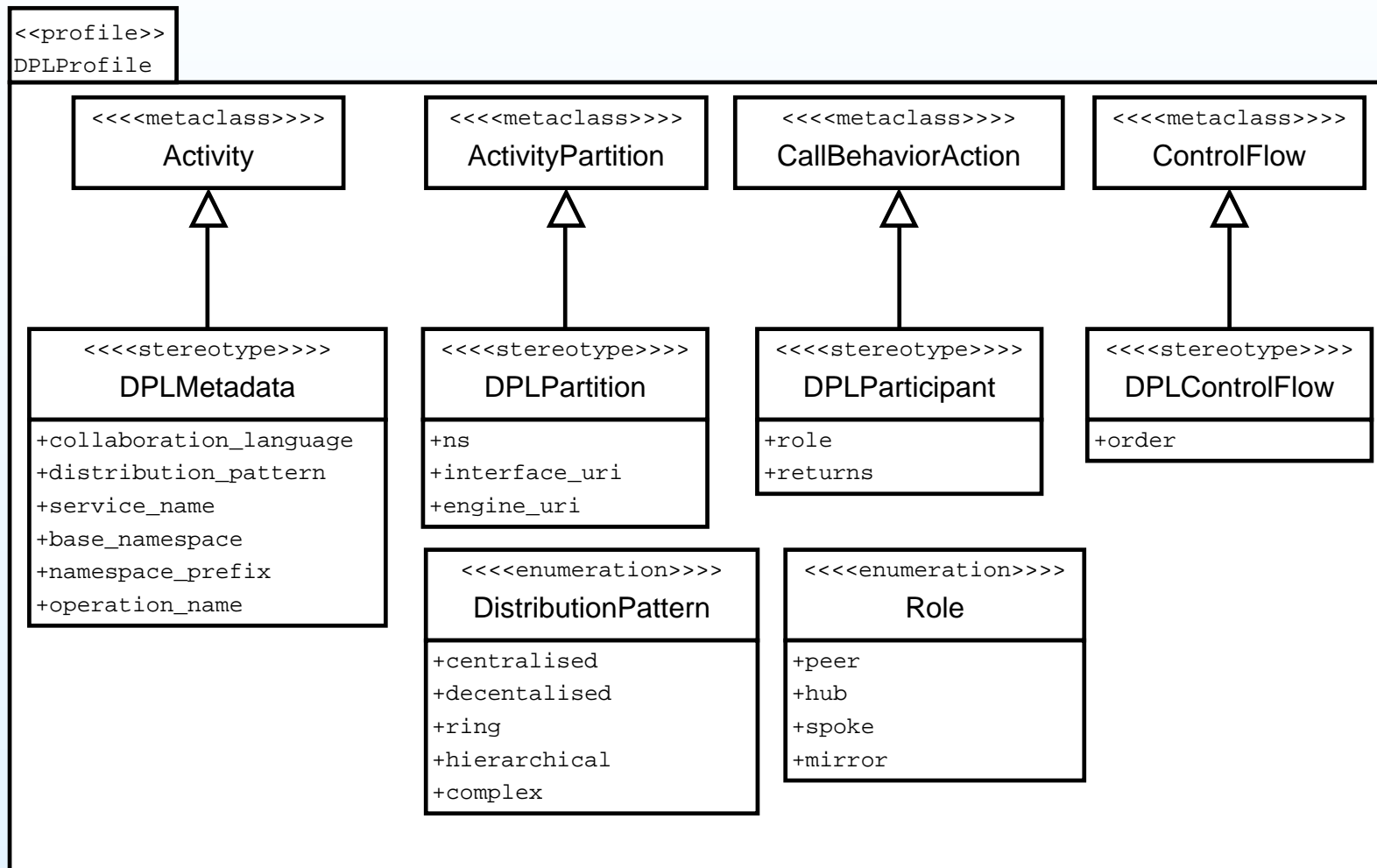
Manually Defined

# Step 1a - From Interface To UML Model

- Web service interfaces as input
- Generate UML activity diagram

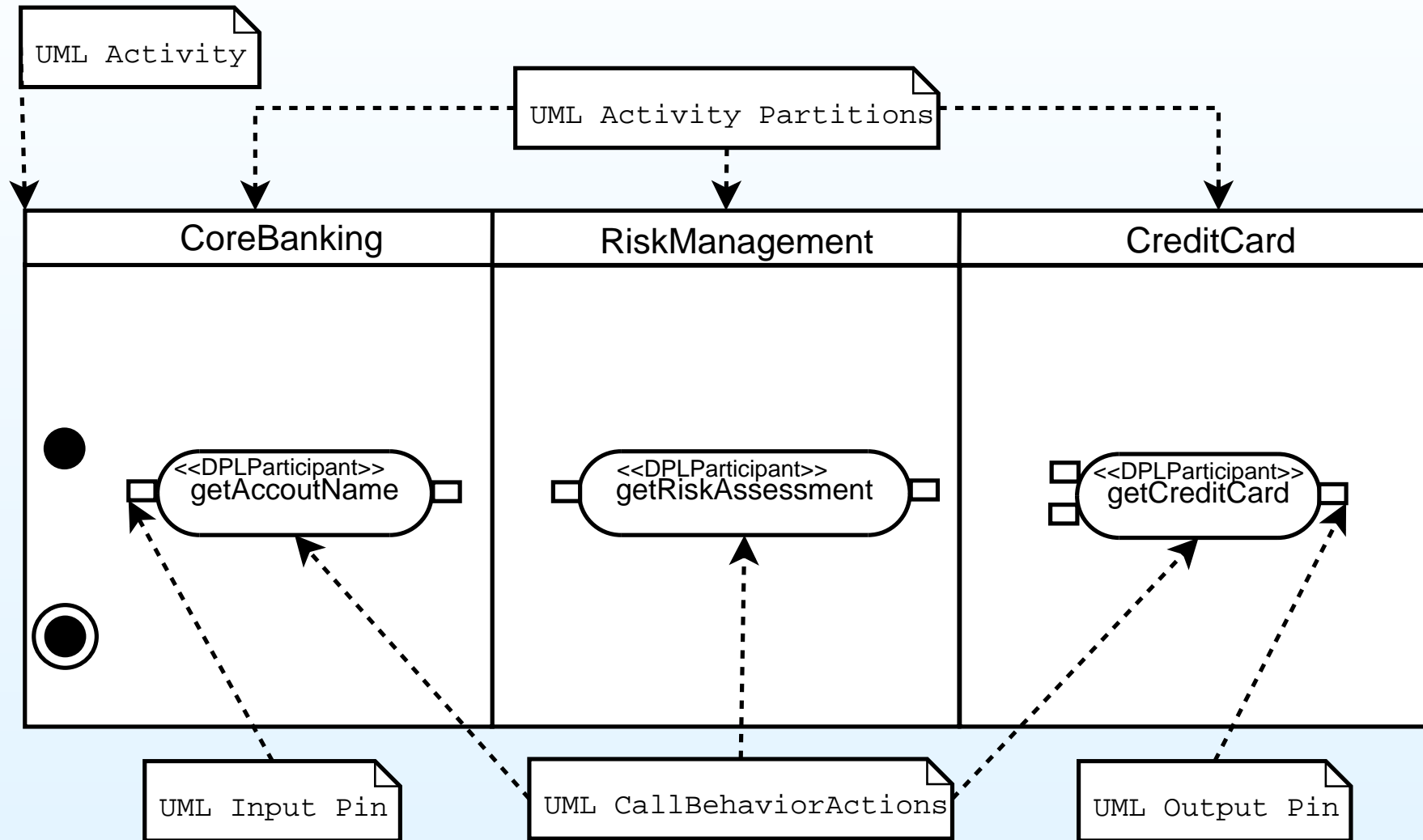


# Step 1b - Automatically apply UML Profile



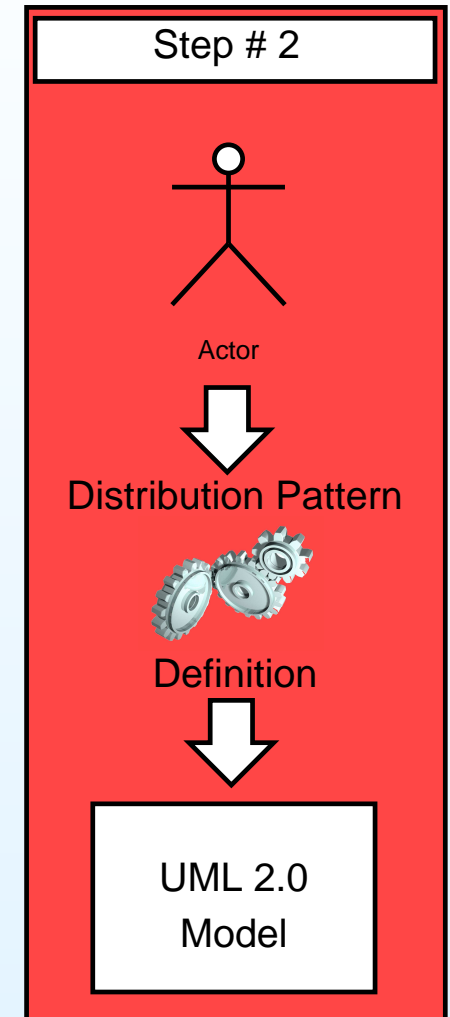
\* InputPin and OutputPin elements omitted for space reasons

# Step 1c - From Interface To UML Model

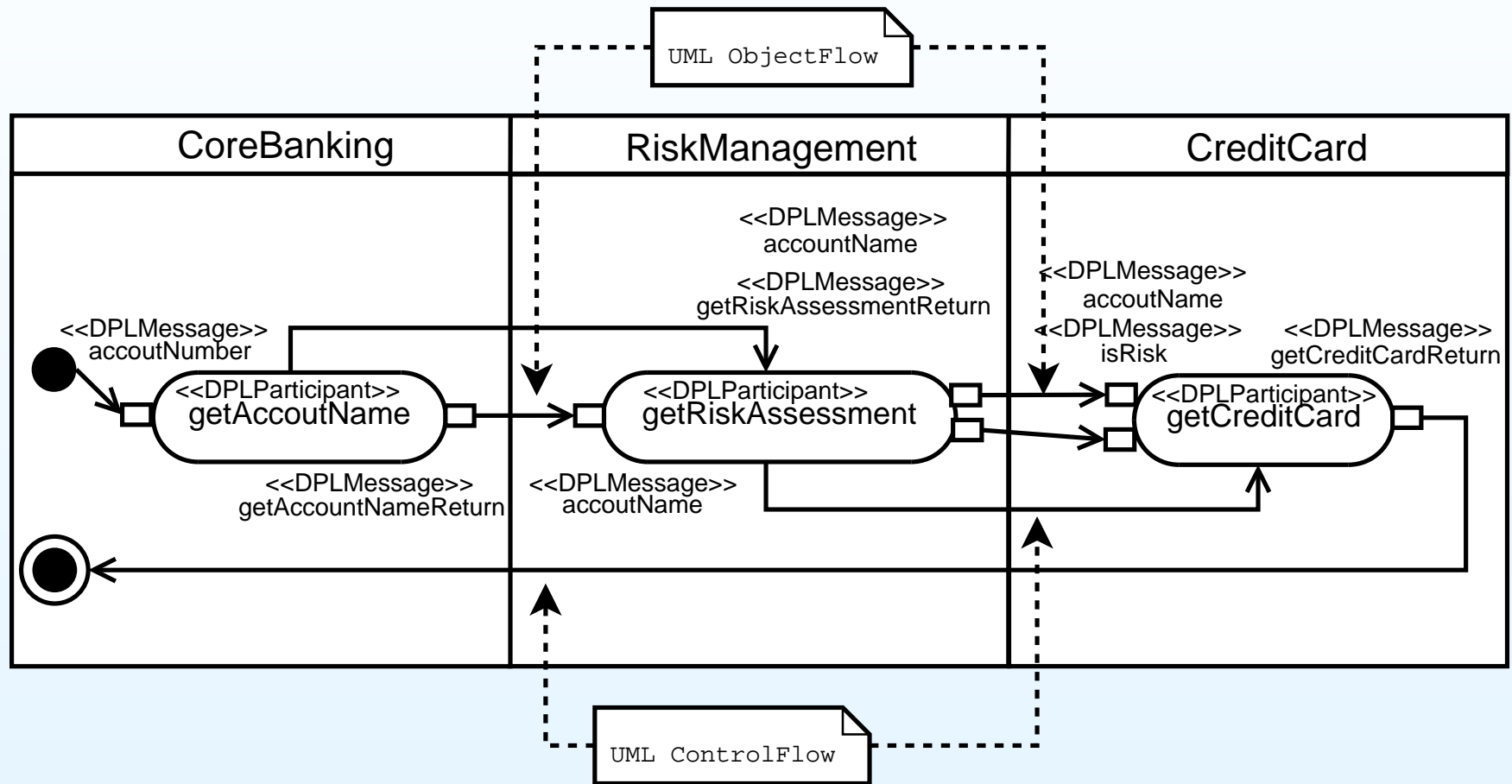


## Step 2a - Distribution Pattern Definition

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

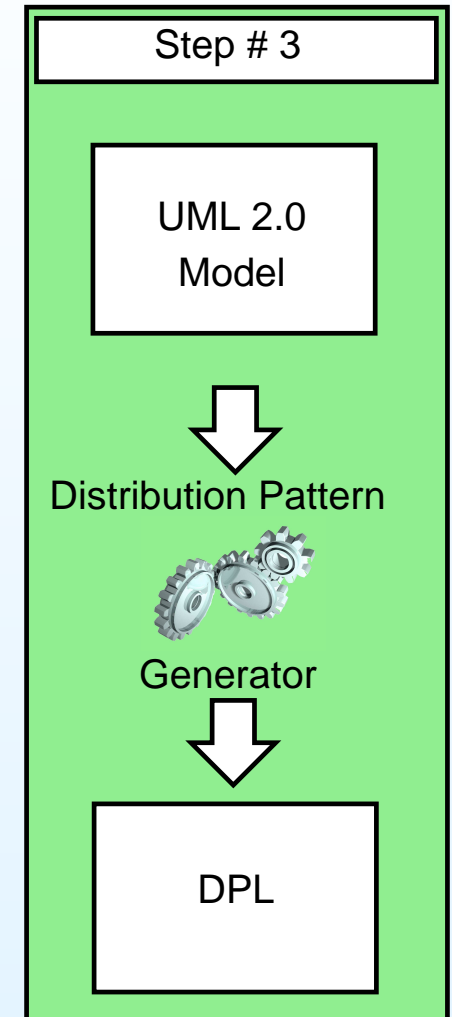


# Step 2b - Distribution Pattern Definition



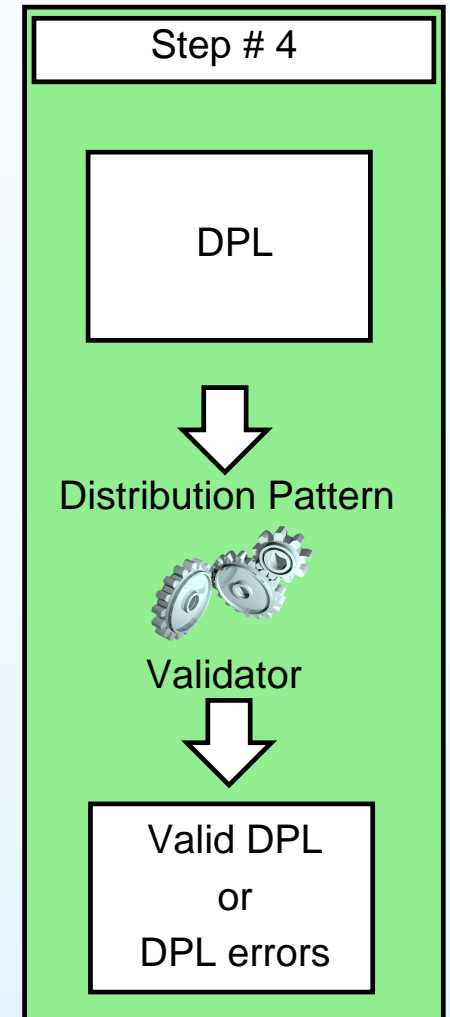
## Step 3 - From Model to DPL

- Novel Distribution Pattern Language (DPL)
- $DPL \equiv (UML \text{ Model} + UML \text{ Profile})$
- No reliance on UML
- Distribution pattern as XML
- Restricted by XML Schema
- Can be validated at build time



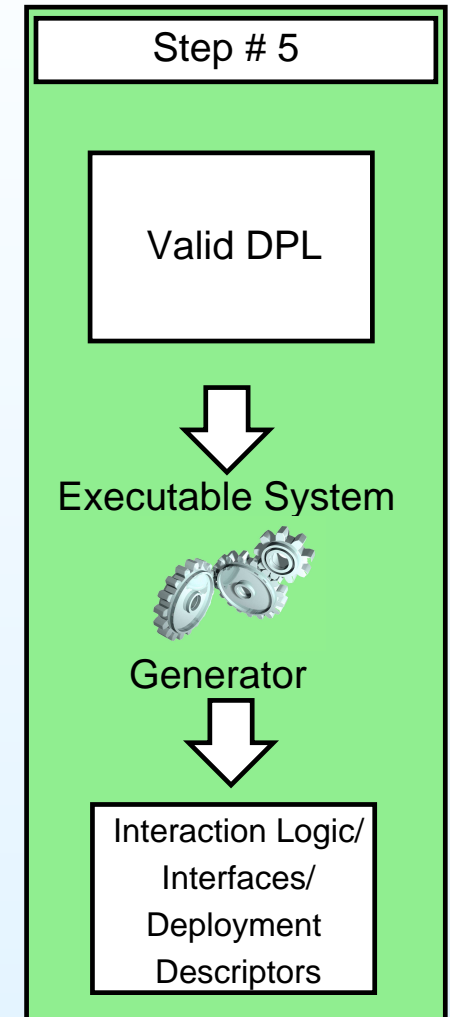
## Step 4 - Model Validation

- Validate pattern instance against XML Schema
- Validation ensures
  - Connections are compatible with pattern
  - Profile values are compatible with pattern
- If errors are found return to Step 2



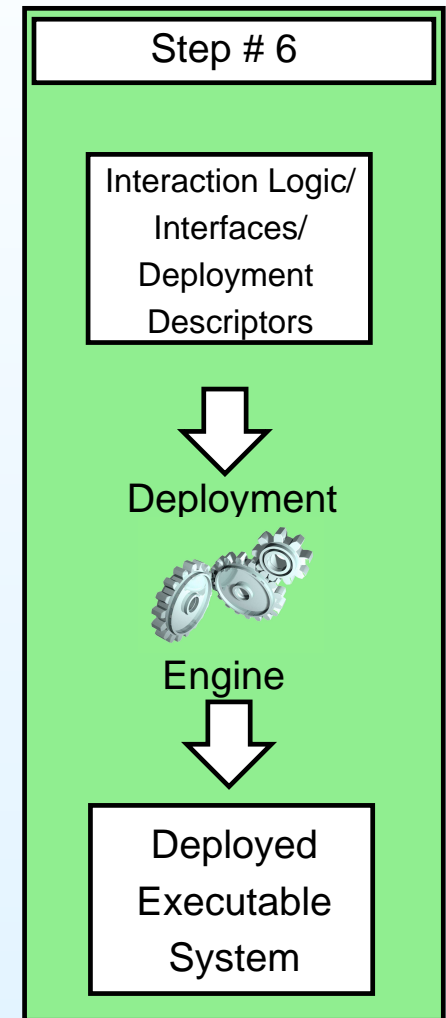
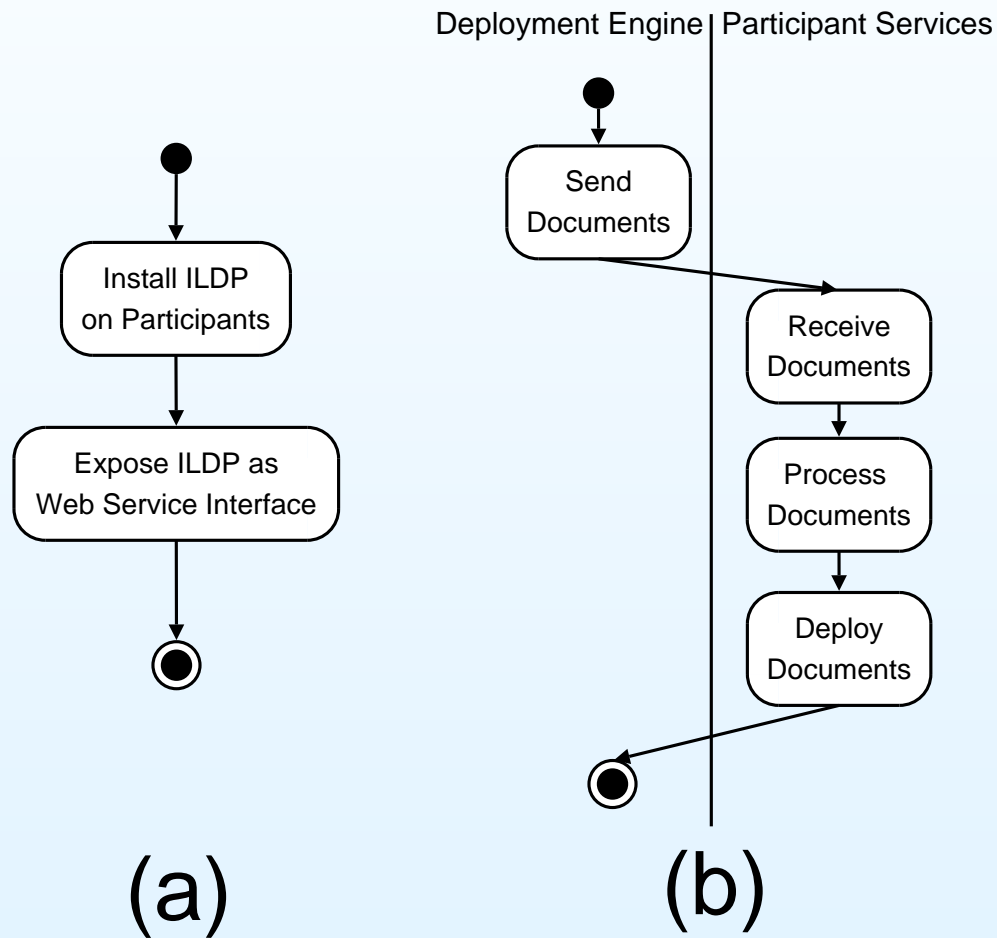
## Step 5 - DPL to Executable System

- Valid DPL taken as input
- Generates
  - Interaction logic document(s)
  - Interface document(s)
  - Deployment descriptor(s)
- System is ready for deployment
  - Participants must have workflow engine



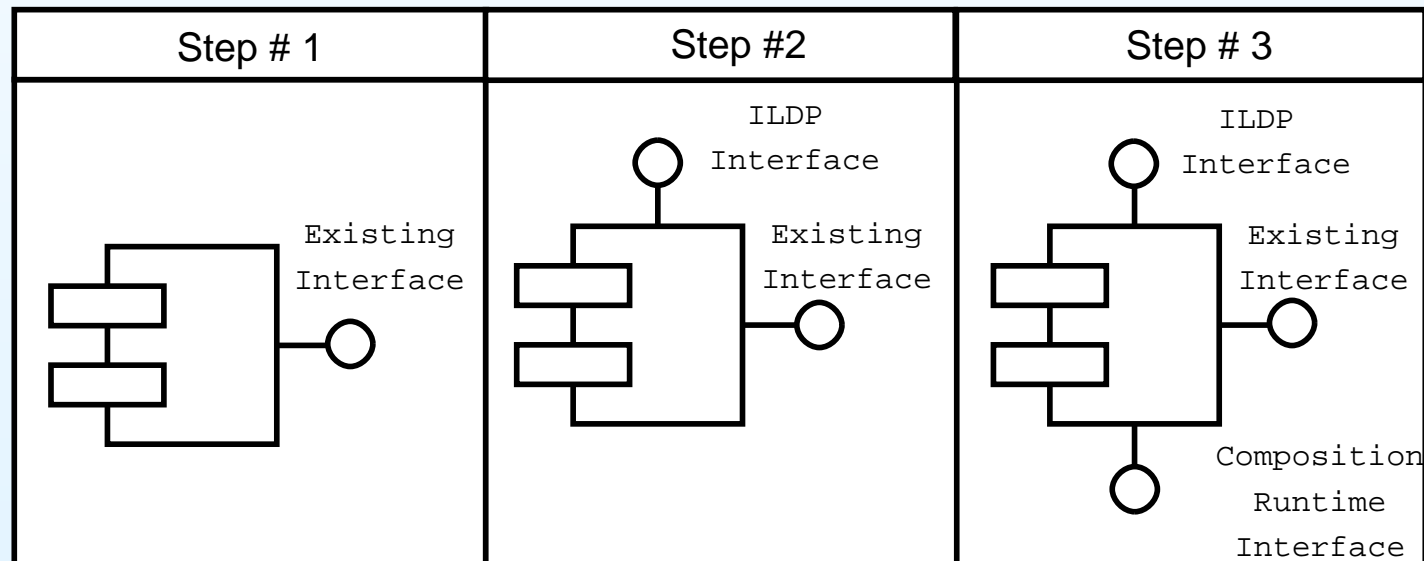
# Step 6a - Automated Deployment

- Interaction Logic Document Processor(ILDP)

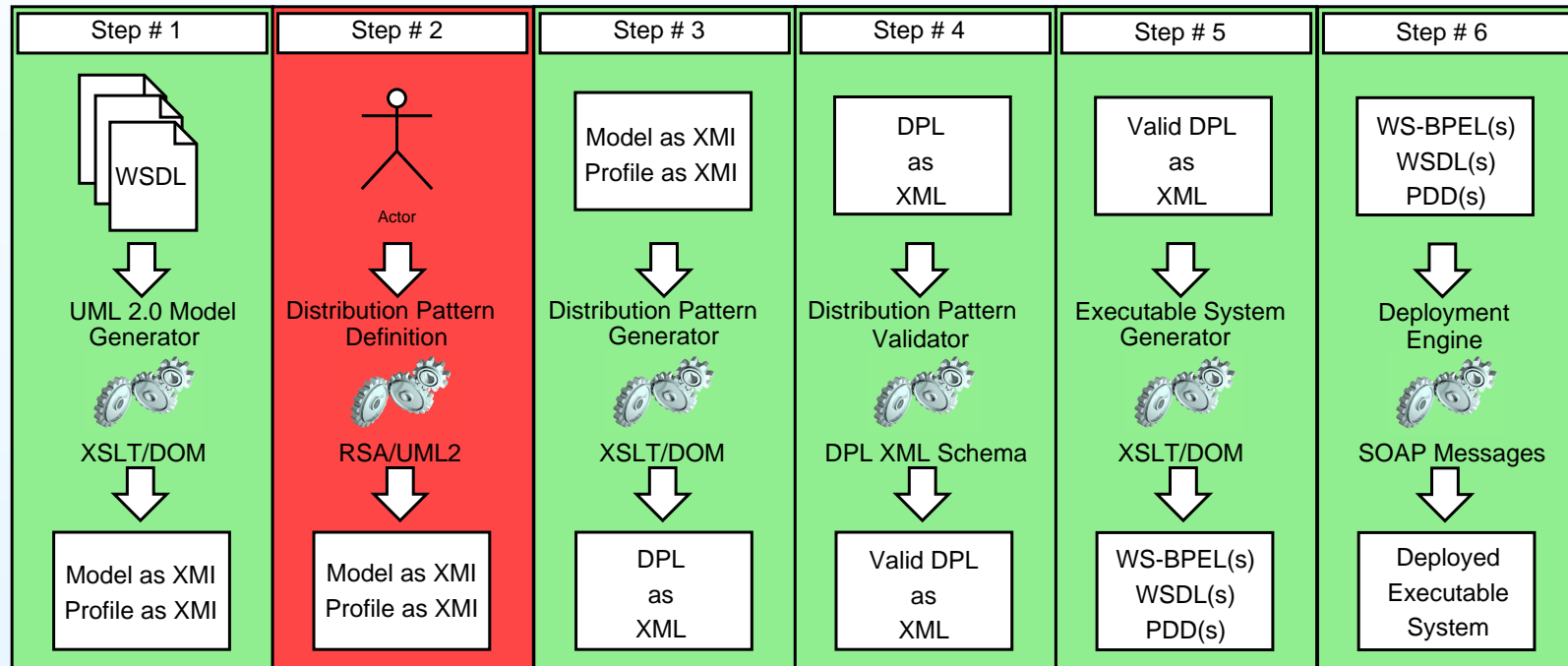


## Step 6b - Automated Deployment

- Interaction Logic Document Processor(ILDP)
  - Enables automated deployment of executable system
  - Executable artifacts taken as input
  - Artifacts installed & deployed
  - Reduced deployment effort



# Tool Implementation - TOPMAN



## Future Work

---

- Complete work on TOPMAN tool
- Develop plugin for AndroMDA
- Extensions to modeling approach
  - $\pi$  calculus
  - Architecture Description Languages (ADL)
- Unify all three modeling aspects into the tool
- Document distribution pattern metrics

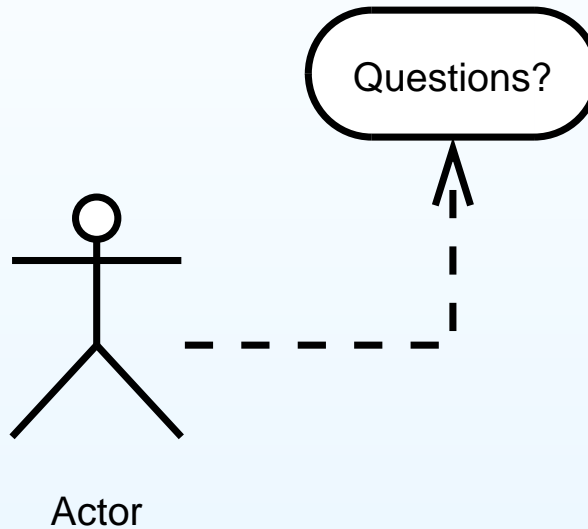
# Conclusion

---

- Motivated new service modeling aspect
  - Distribution Patterns
- Presented modeling approach
  - Standards based
  - Extensible
- Presented six step modeling technique
  - UML profile extension
  - Introduced dynamic deployment facility
- Implementation
  - Compatible with existing modeling tools
  - Generates executable system

# 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.

