OnceDeplyer:

A Model-based application Deployment Framework

Chen Wei
Technology Center of Software Engineering
Institute of Software
Chinese Academy of Sciences

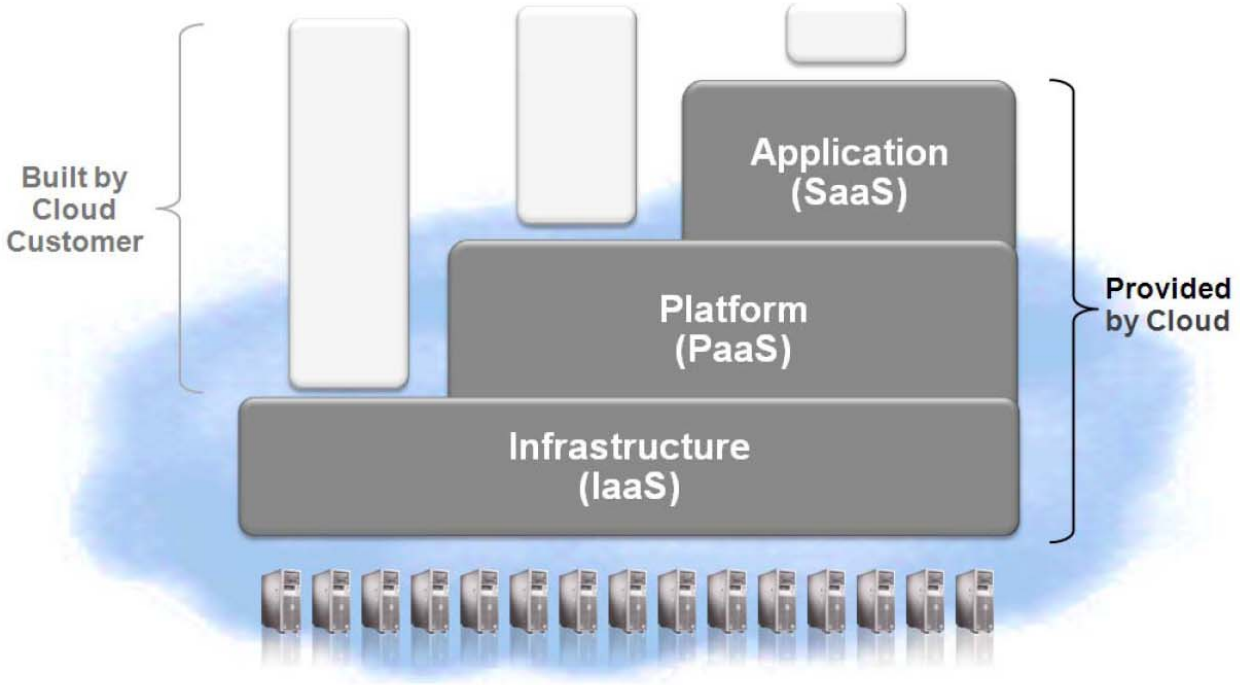
Motivation

Cloud System Deployment & Configuration

Manual work → Automation

?

Error-prone → Correctness



Introduction

- Problems we concern and our solutions

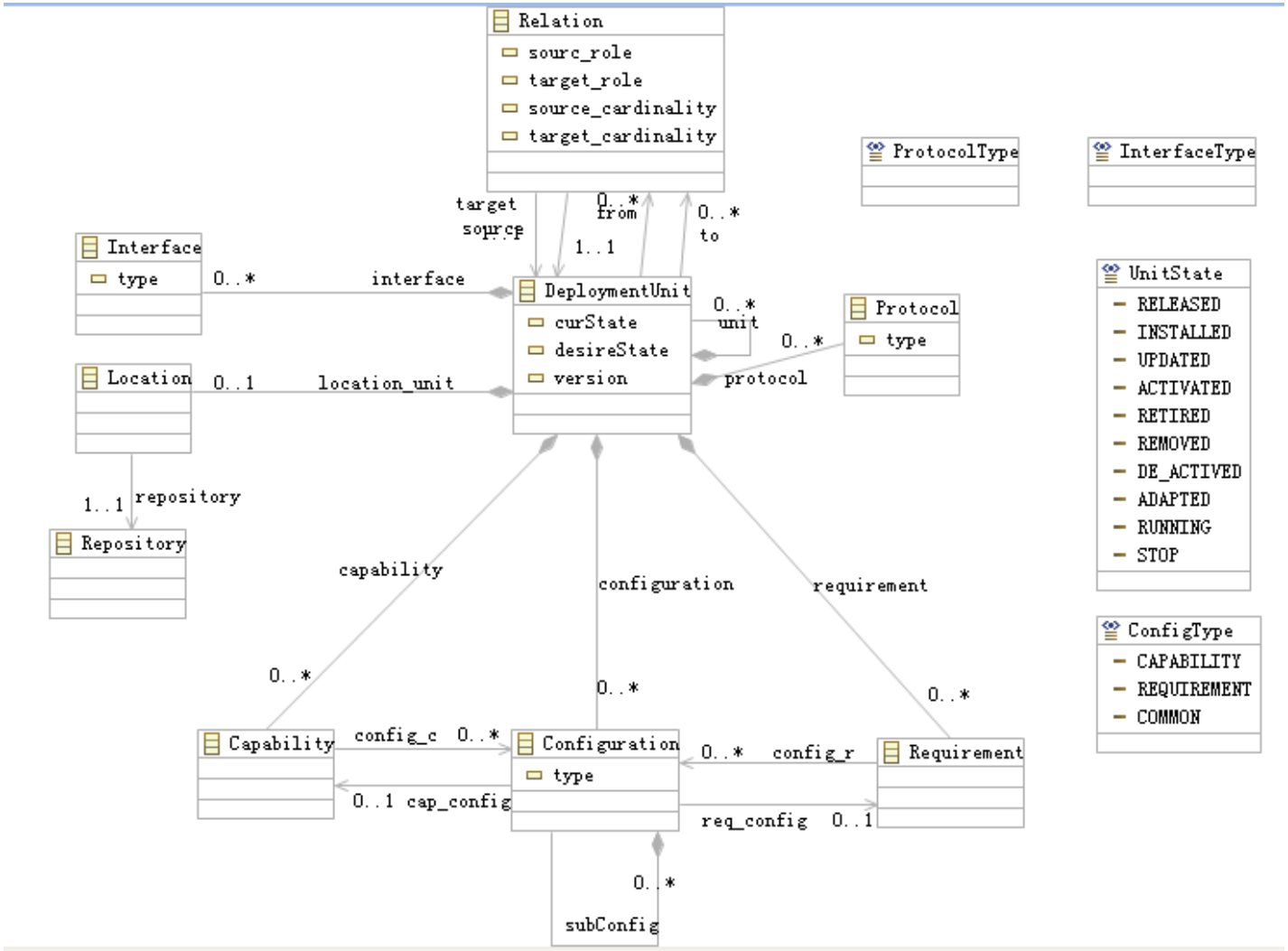
How to provide a **user-friendly environment** to perform deployment, other than in manual or script based manners?	Implementing a **model-based, visualized** framework to promote the automatic level, decrease the human efforts
How to promote the configuration **reliability** and **correctness**?	An automatic method to guarantee the correctness of configuration through constraints validation
How to reduce the cost when migrating an application to cloud?	A semi-automatic approach to generate vendor specific deployment description files
.....	.....

Our current work (1)

- Visualized deployment topology design
  - Implemented as an Eclipse plug-in
  - A set of core meta-models are defined
  - abstracting the problem space of deployment, including: applications, environments, relations, capabilities, requirements etc.
  - Features
    - a graphic editor, providing a visual console to design the deployment plan
    - Integrate **design** , **validating**, **transforming** and **process executing**
    - Support Once middleware suite currently: **OnceAS, OnceBPEL, OnceSE**

Our current work (1)

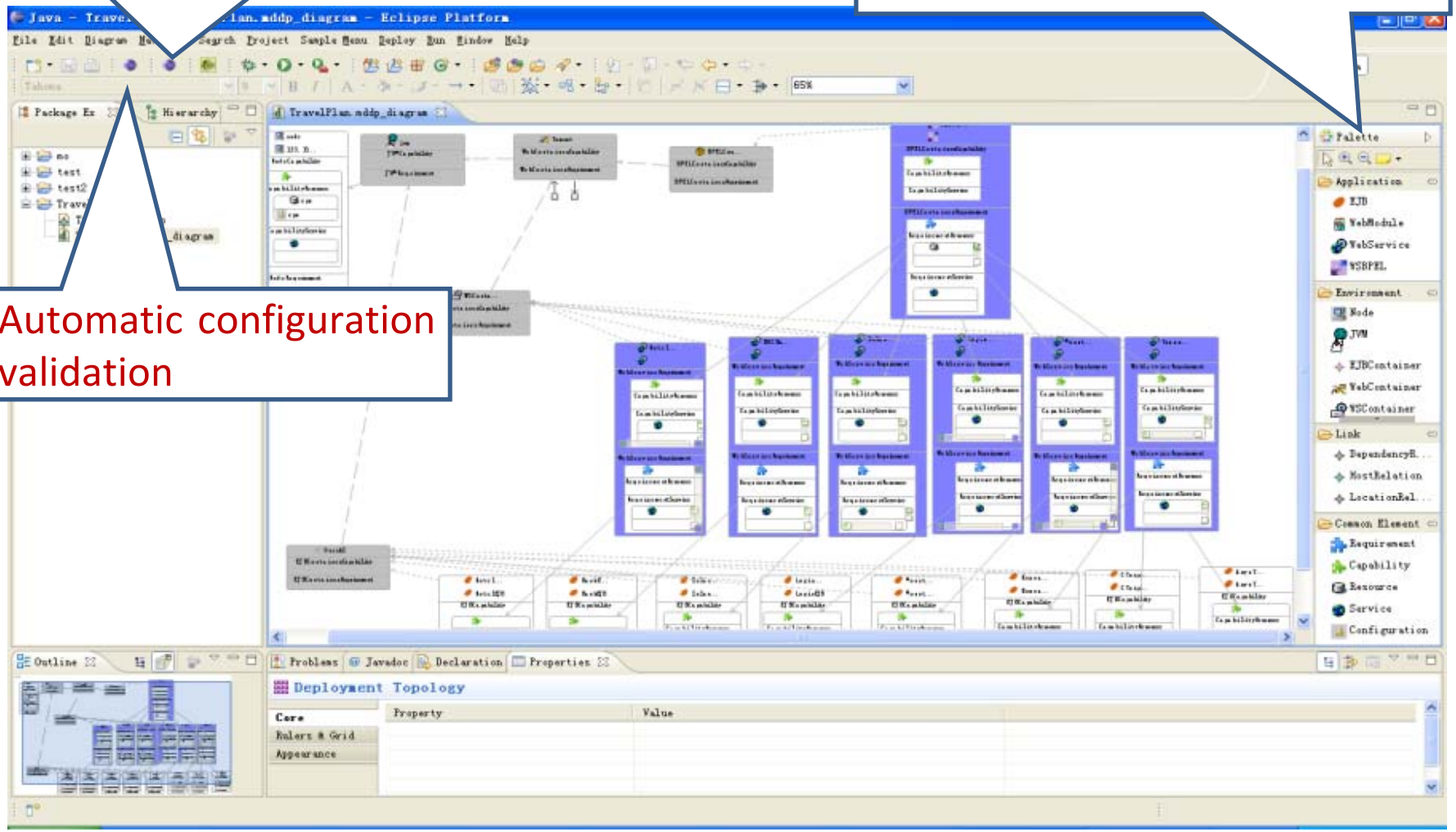
- Visualized deployment topology design



Our current work (1)

Support automatic deployment execution just need **“one click”**

Support **4** categories modeling notation elements



Automatic configuration validation

Our current work (2)

- Configuration validation
  - Setting up multiple semantic constraint based on templates pre-defined, currently including:
    - **Resources constraint** → $CP_{resource}$ <Capability, ReqSet>
    - **Unique constraints** → CP_{unique} <ID_1,…,ID_k,Scope>
    - **location constraints** → $CP_{location}$ <u_1,u_2,target,locType>
    - **Compatibility constraints**
    - **Equality constraint**
  - Automatic constraint validation based on formal language parser

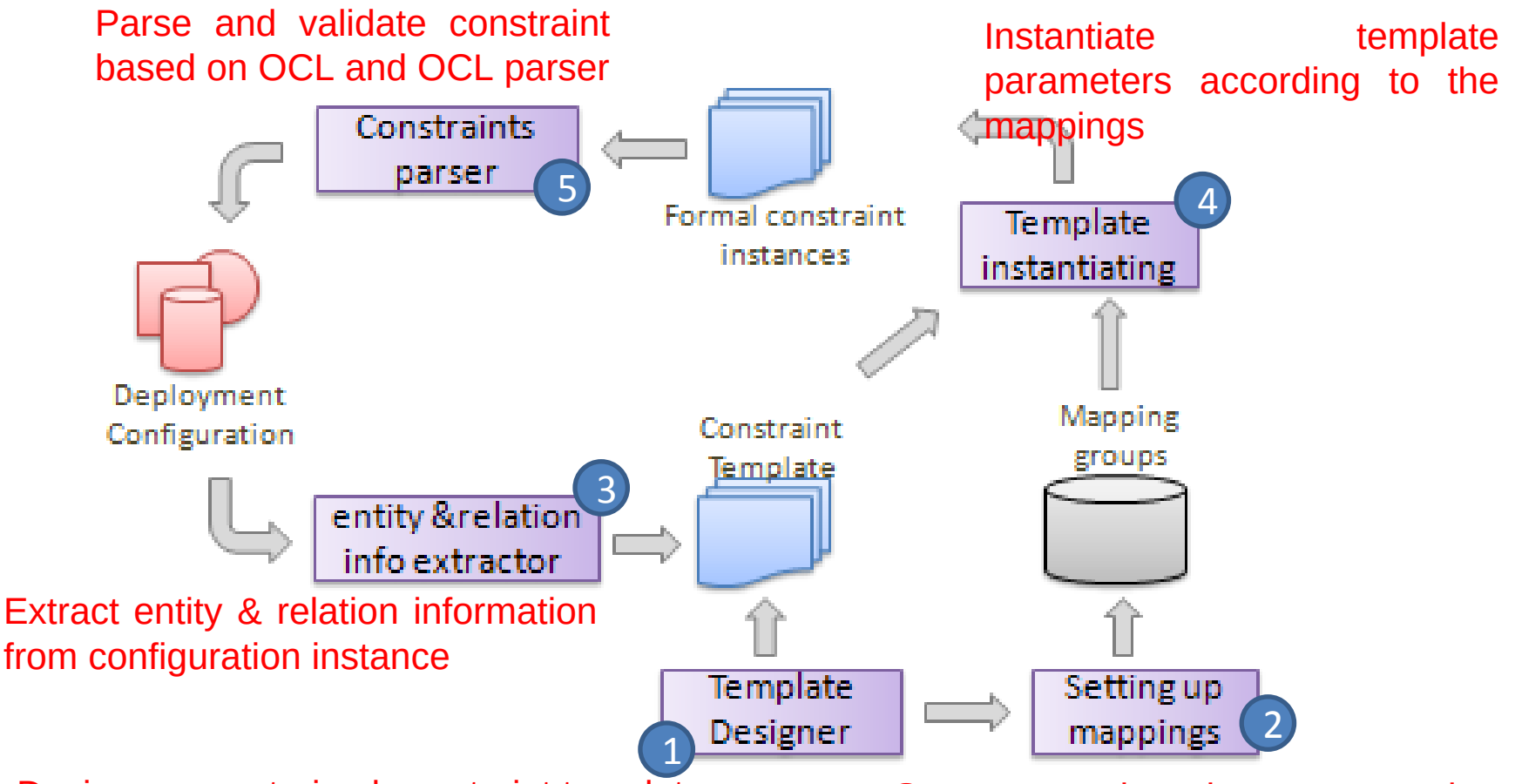
Case
In an application, web services on OnceSE engine must have unique identities

CP_{unique}<r_1.SourceID, r_2. SourceID, r_3. SourceID, r_4. SourceID, r_5. SourceID,r.TargetID>

```
context WebService
inv checkID: WebService.allInstances()->forAll(n1 | n1 <> self implies n1.ID <> self.ID)
```

Our current work (2)

- Configuration validation

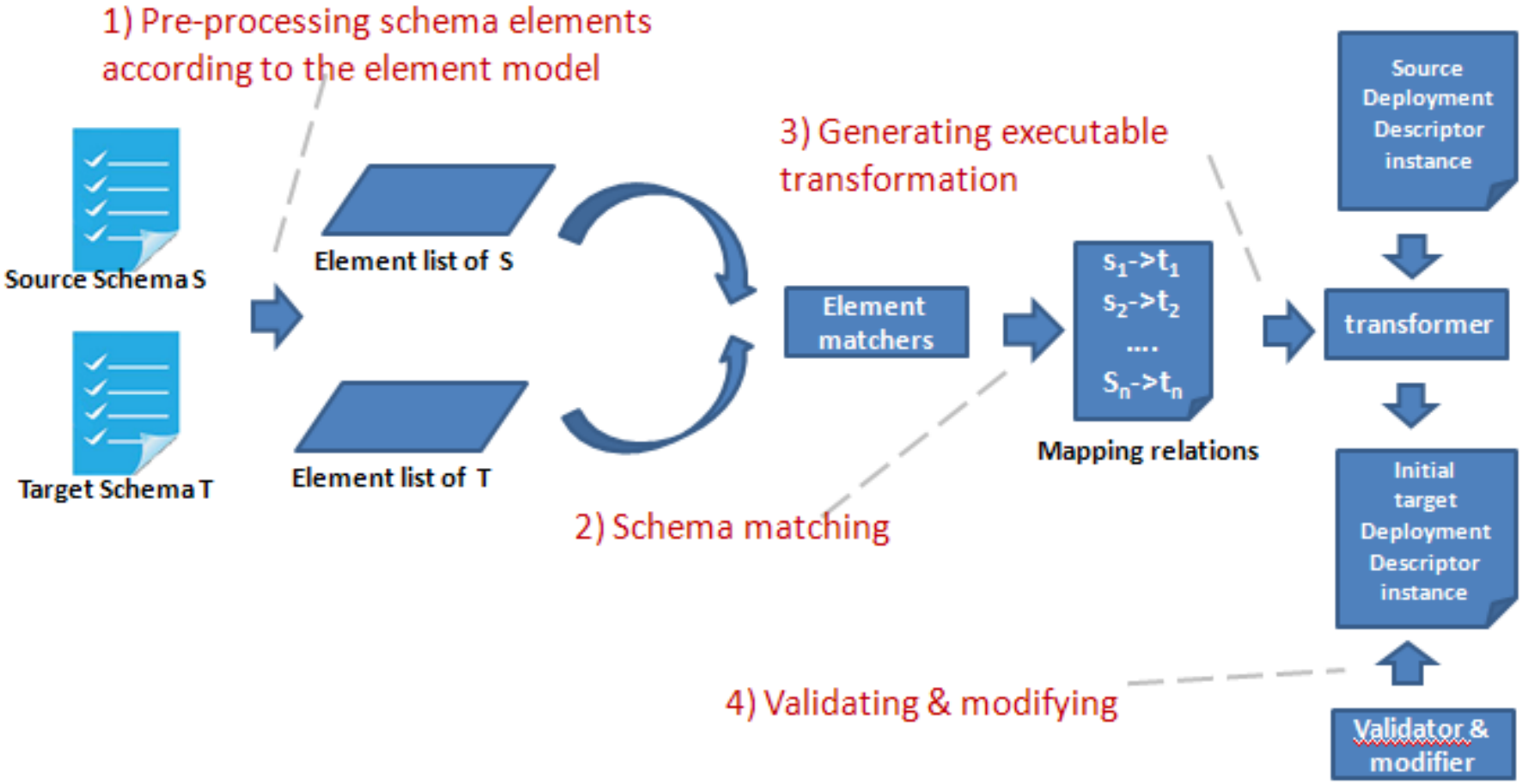


Our current work (3)

- Deployment package customization
  - Deployment packages are vendor specific, customization is inevitable in publishing, migration, and updating
  - Implemented a package customization approach based on schema-matching method
  - leveraged XSLT to generate and edit vender specific deployment descriptors
    - Current support middleware vendors: Once, JOnAS, TongWeb, Apusic
    - Package type current support: Web application package(war), EJB package

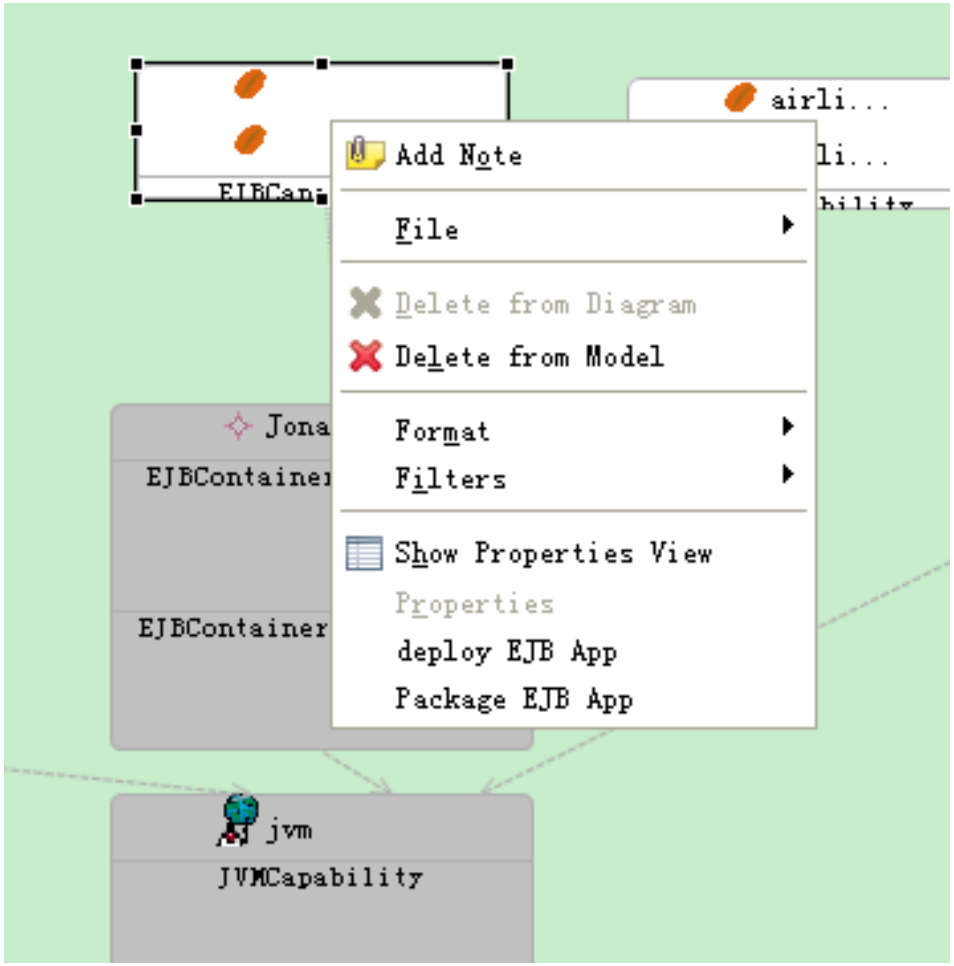
Our current work (3)

- Deployment package customization



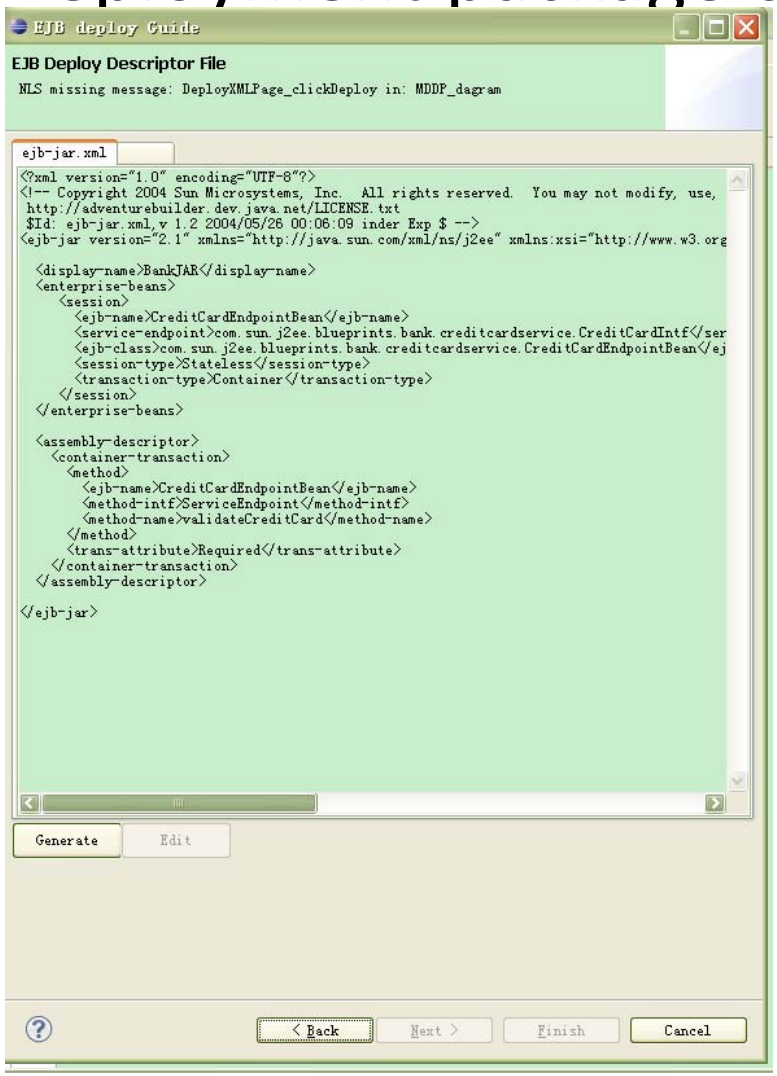
Our current work (3)

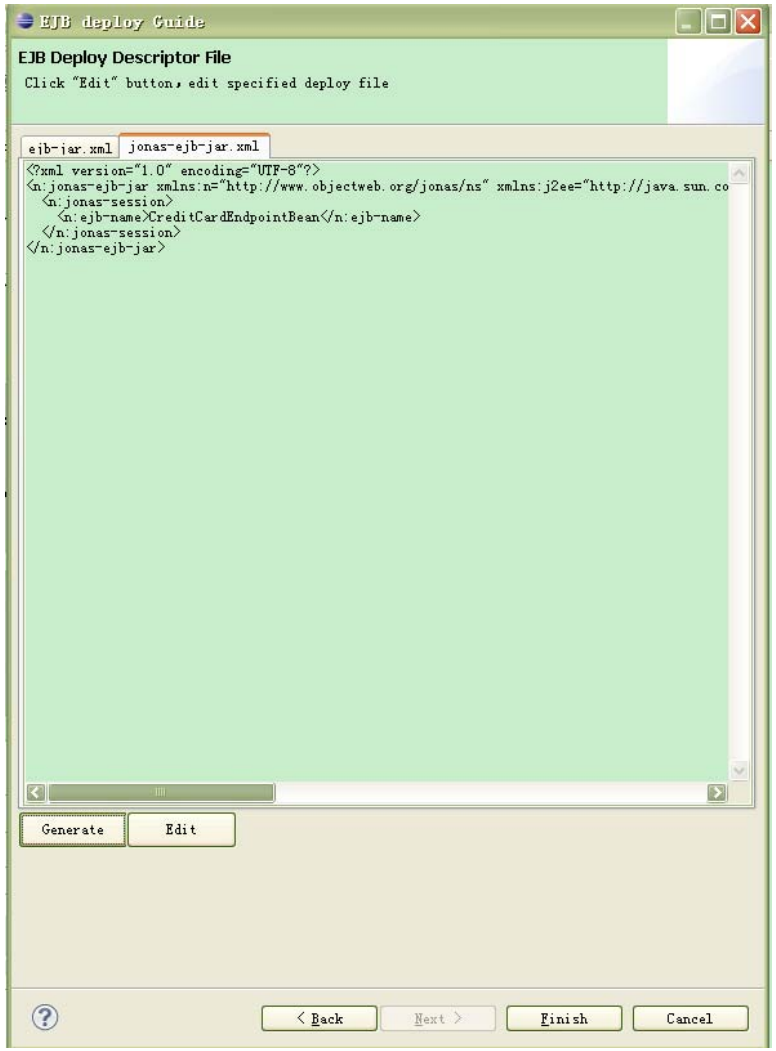
- Deployment package customization



Our current work (3)

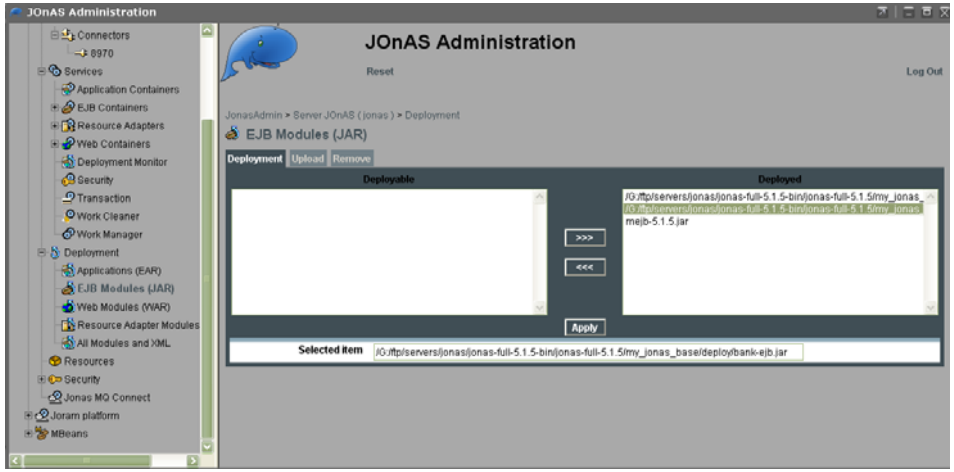
- Deployment package customization





Our current work (3)

- Deployment package customization



Future Work

- Configuration optimization
  - Support NFP expression and modeling
  - Support multi-objective optimization and trade off based on some optimization algorithm, concerning
    - Cost: resources must be paid
    - quality: services level must be met
- Configuration adjustment and resolution
  - Exploring heuristic rules about configuration adjustment and confliction resolution, when constraint violations occur

Thanks! Q & A