A Practical Approach to Model-Based Cloud Service Deployment: Using a Smart Interpreter and a Domain Specific Language

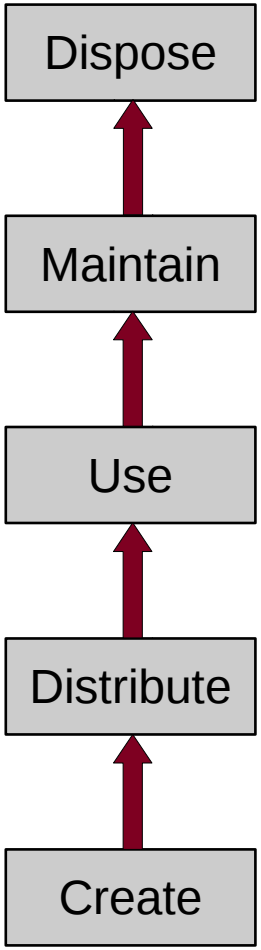
Tim Waizenegger
University of Stuttgart,
Institute of Parallel and Distributed Systems,
Universitätsstr. 38,
70569 Stuttgart, Germany
tim.waizenegger@ipvs.uni-stuttgart.de
http://www.ipvs.uni-stuttgart.de

Agenda

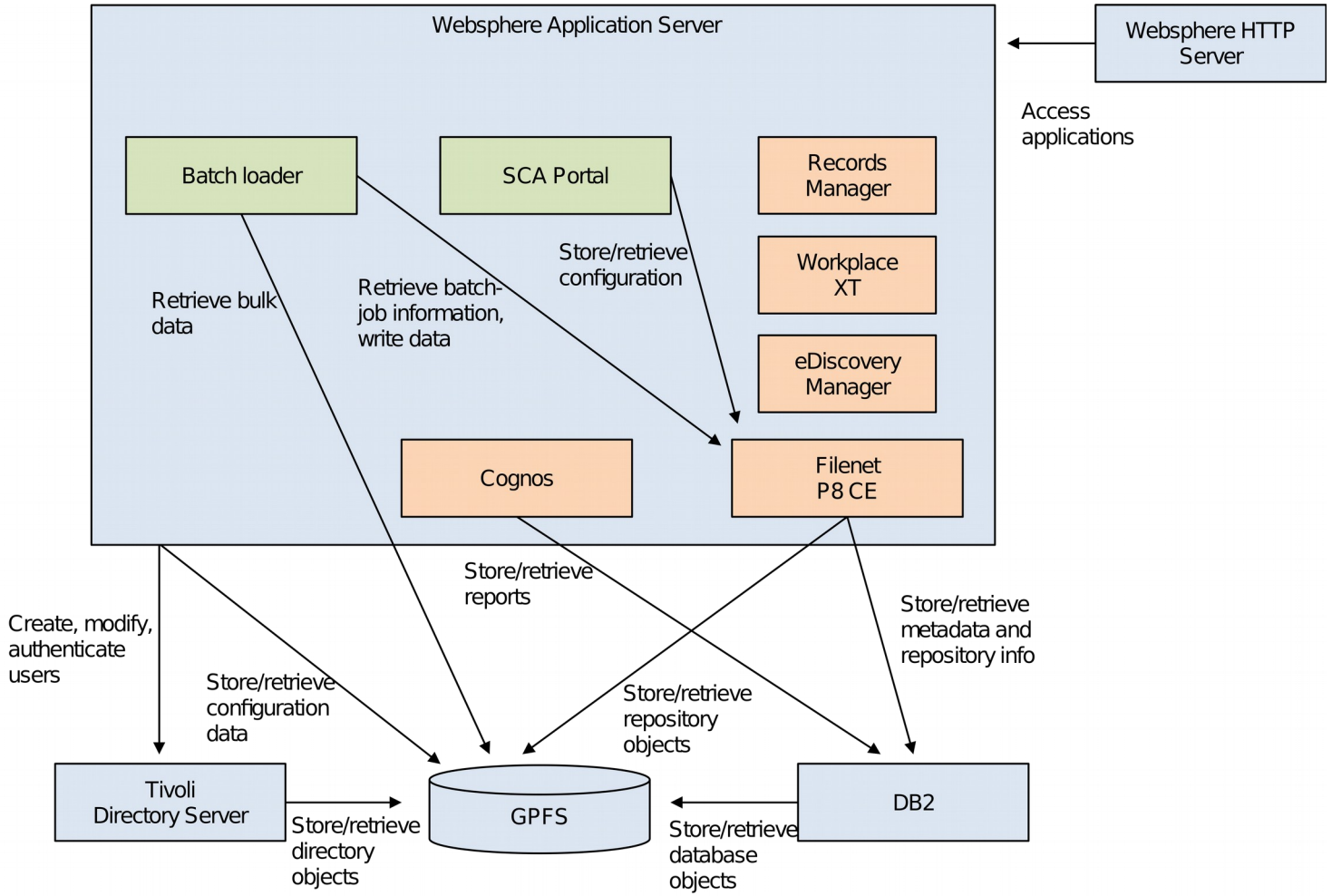
- **Enterprise Content Management**
- Cloud-Service Deployment Automation
- Related Work
- Domain Specific Language Approach

Enterprise Content Management (ECM)

- Manage all life-cycle stages of electronic content in an enterprise
- The functionality of these systems varies widely
- Many different software components comprise an ECM system
- The system topology and component configuration is customized
- The components are integrated with other IT-systems



Enterprise Content Management (ECM)



Agenda

- Enterprise Content Management
- **Cloud-Service Deployment Automation**
- Related Work
- Domain Specific Language Approach

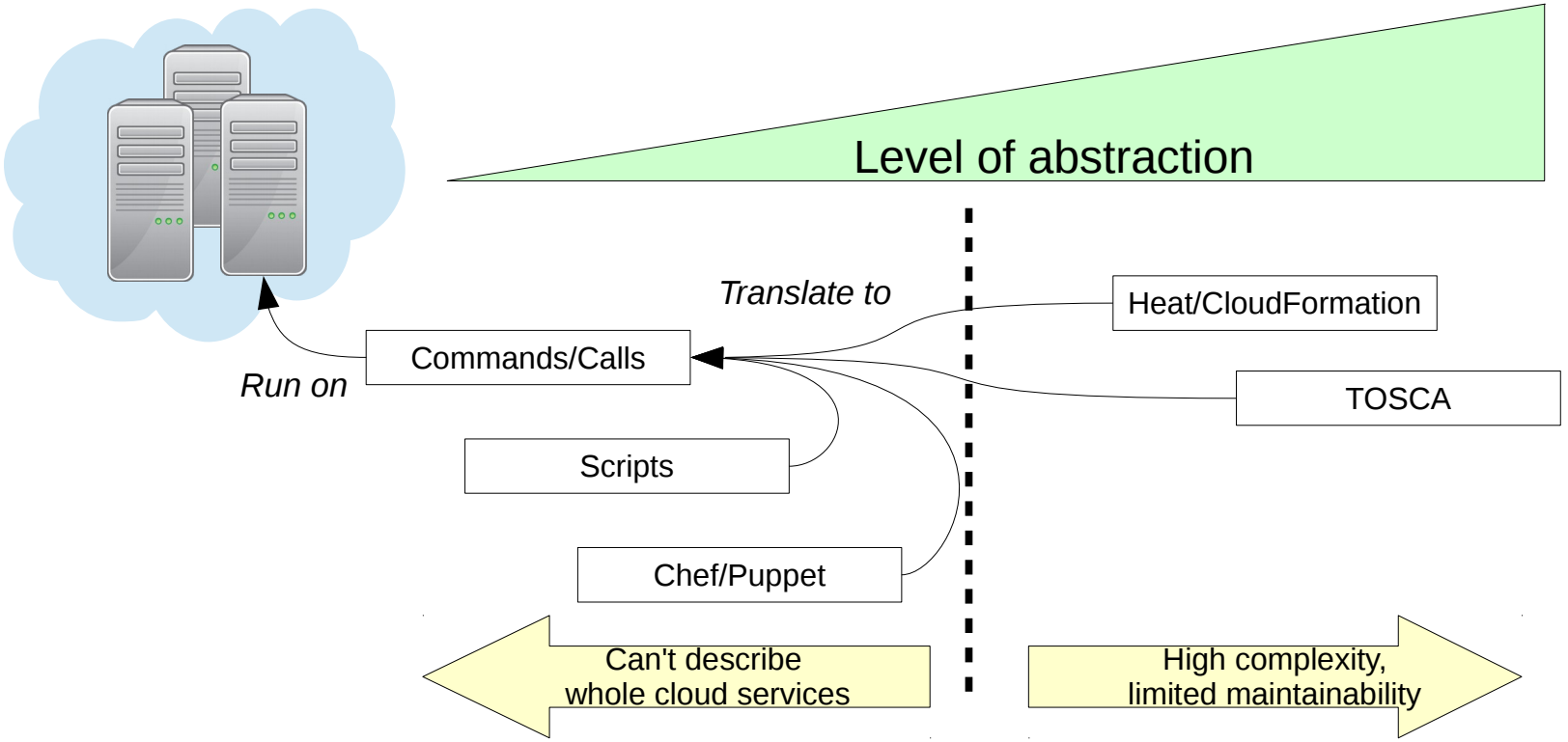
Cloud Deployment Automation – Introduction

- **What is Cloud Deployment Automation?**
  - A mechanism for automating the installation and configuration of complex applications on cloud platforms
  - A representation of the components that comprise the application
- **Why do we need this?**
  - Single-tenant applications need to be deployed many times
  - Development and testing benefit from the repeatable and effortless deployment of new instances
  - It enables packaging entire cloud applications
  - Large installations are only managable when the instances are well known

Cloud Deployment Automation – Challenges

- **Challenges and requirements:**
  - Control many different Cloud services (VMs, PaaS offerings…)
  - Enable reusability of the deployment model
    - Through easy customization
  - Keep track of instances to aid with DevOps
    - Software updates
    - Migration
    - Monitoring
  - **Provide an advantage over manual deployment or scripts**

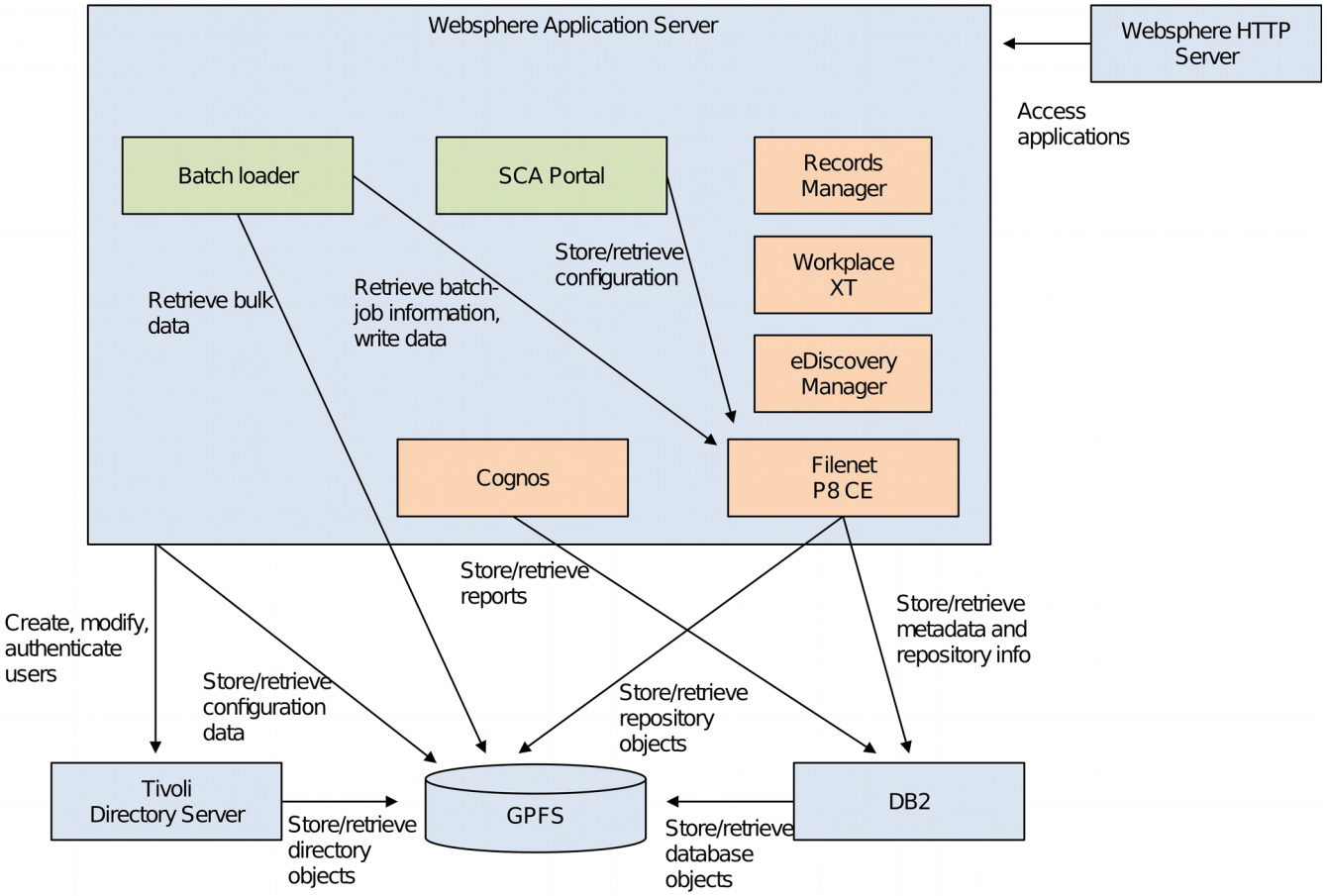
Cloud Deployment Approaches – Overview



Agenda

- Enterprise Content Management
- Cloud-Service Deployment Automation
- **Related Work**
- Domain Specific Language Approach

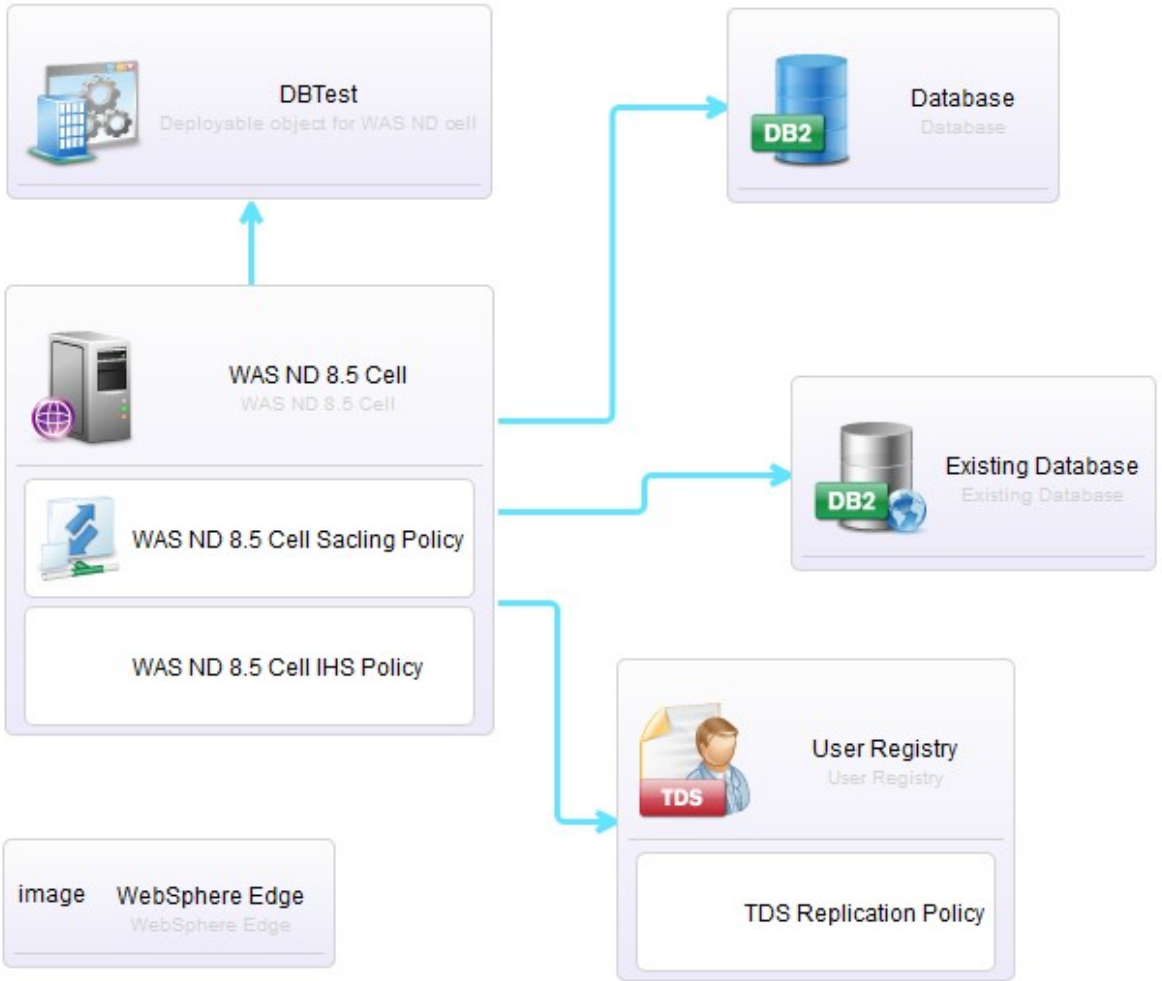
Related Work (1/3) – Ant Scripts



Related Work (1/3) – Ant Scripts

- For each component, Ant-tasks were developed that implemented individual install/config logic
- Each component instance then had Ant-tasks to orchestrate the individual steps
- One global Ant-task would orchestrate all the components
- Problems:
  - Giant mess of config parameters needed to be passed down
  - Low reusability of tasks
  - Not transparent

Related Work (2/3) – IBM Workload Deployer (IWD)



Related Work (2/3) – IBM Workload Deployer (IWD)

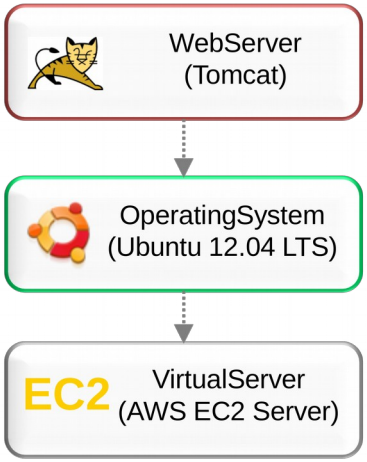
- Model-based declarative deployment
- Components and lifecycle operations need to be defined
- With a graphical tool, the service topology can be defined form a set of components and relationships
- IWD determines an orchestration routine
- Problems:
  - Impossible to express some aspects of custom components
  - Same goes for relationships/lifecycle stages

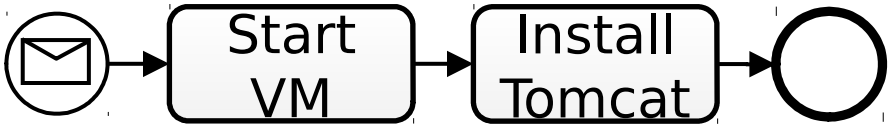
  → not powerful enough for complex application scenarios

Related Work (3/3) – (Open)TOSCA

OASIS

Topology and **Orchestration** Specification for Cloud Applications





Related Work (3/3) – TOSCA

- OASIS Standard, many implementations exist
- Standard specifies mostly the syntax of the topology definition language (doesn't say much about orchestration).
- Interoperability between TOSCA tools can be compared to interoperability between applications using XML
- Very well suited for describing topologies / service layout
- Most implementations only use this aspect and, like IWD, internally generate the orchestration

Related Work (3/3) – OpenTOSCA

- Open Source implemenation of a TOSCA engine
  - Topology modeler
  - Admin-UI
  - Engine:
    - Topology parser / Instance manager
    - Workflow engine (BPEL) for executing orchestration

  → OpenTOSCA does not generate an orchestration process, it must be provided (i.e. imperative orchestration)

- Problems:
  - No tool support for orchestration process development (yet)
  - Variability / interaction of orchestration process with different topologies unclear

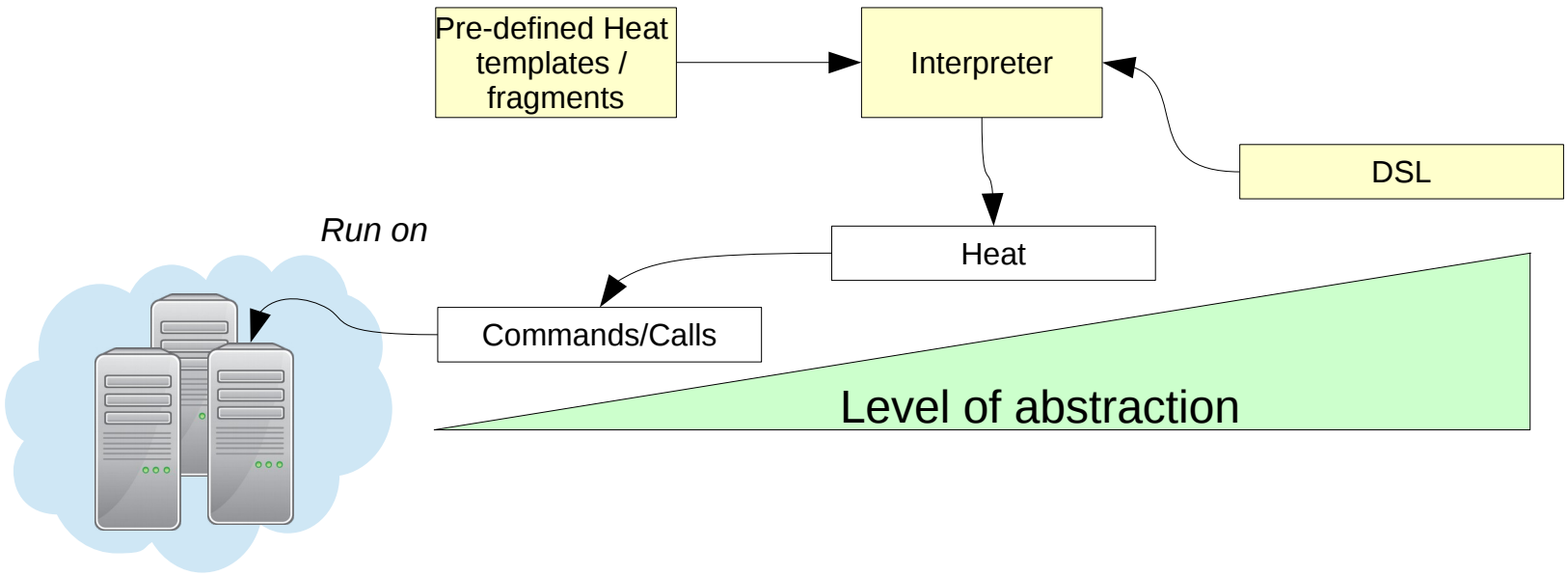
Agenda

- Enterprise Content Management
- Cloud-Service Deployment Automation
- Related Work
- **Domain Specific Language Approach**

DSL-Approach – Motivation

- Conclusions from related work:
  - Model based systems are well suited for representing complex topologies
  - Models provide an understanding of the topology
  - Domain specific approaches can work very well in their application domain (see IWD and Websphere)
  - Low level capabilities are needed to represent all aspects of custom components

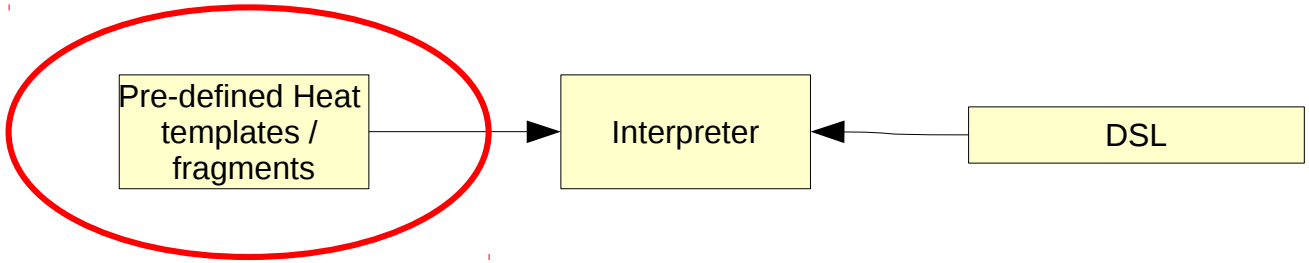
DSL-Approach – Classification



DSL-Approach – Template Catalog

- OpenStack Heat templates provide a good mechanism for defining individual components
- Powerful mechanism for providing scripts and low-level artifacts for deploying components
- We build a catalog of components and topology fragments

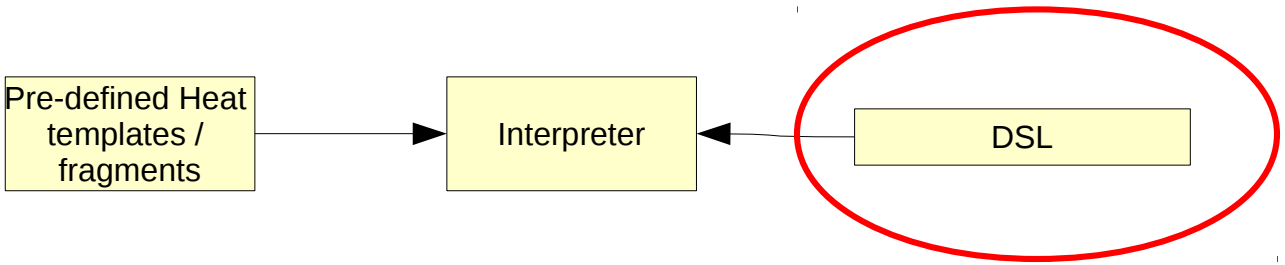
→ **re-use of existing Heat infrastructure and tools**



DSL-Approach – Domain Specific Language

- The DSL provides a syntax and semantic for describing ECM applications
  - Syntax: we use TOSCA → „internal DSL“
  - Semantic: we define TOSCA types for ECM components and relationships
- High level DSL is accessible and understandable

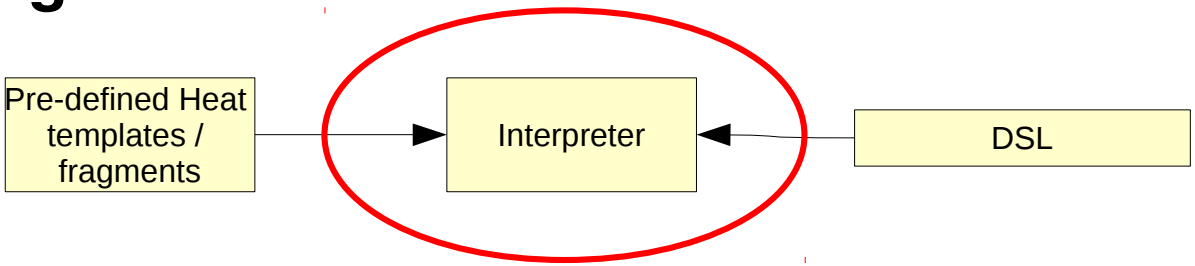
→ **re-use of TOSCA topology syntax and modeling tools**



DSL-Approach – Interpreter

- The interpreter analyses and checks the service description written in the DSL
- Based on rules and the DSL input, it constructs a Heat template for deploying the desired ECM service
- The rules/templates create a known decision space

→ **we can test and validate all possible output topologies**



Thank You!

Tim Waizenegger
University of Stuttgart,
Institute of Parallel and Distributed Systems,
Universitätsstr. 38,
70569 Stuttgart, Germany
tim.waizenegger@ipvs.uni-stuttgart.de
http://www.ipvs.uni-stuttgart.de