

Introducing Domain Specific Modeling practices for unambiguous federation scenarios on a diverse pool of organizations

Christos Tranoris PhD, Research Fellow, Electrical and Computer Eng., University of Patras, Greece

Introduction

Federation scenarios usually involve resources offered by a diverse pool of organizations, thus it is imperative introducing practices that result to unambiguous scenario definitions. The work we present in this paper considers the concepts of modeling and meta-modeling to federation scenarios by applying Domain Specific Modeling (DSM) practices. We consider the specific needs of federation, we acknowledge the fact that different federation models already exist and we adopt lightweight practices instead of laborious expensive heavyweight approaches. For example we avoid the adoption of a mega-model like DEN-ng, without excluding it though. We argue that **defining a federation meta-model, focusing on the federation domain and applying DSM practices is necessary** in order to: i) create formal description of an organization with its offered services and resources, ii) have valid, comprehensible and unambiguous configurations that support federation scenarios, iii) simplify the combination of services and resources from third parties that are non-conformant to the meta-model and iv) have common definitions and understanding by the resource federation domain, thus being efficient and practicable. Furthermore we present a Domain Specific Language having the meta-model as its abstract syntax, for defining model entities and federation scenarios between organizations. Additionally, the proposed practice is supported by a tool.

Defining a common federation meta-model

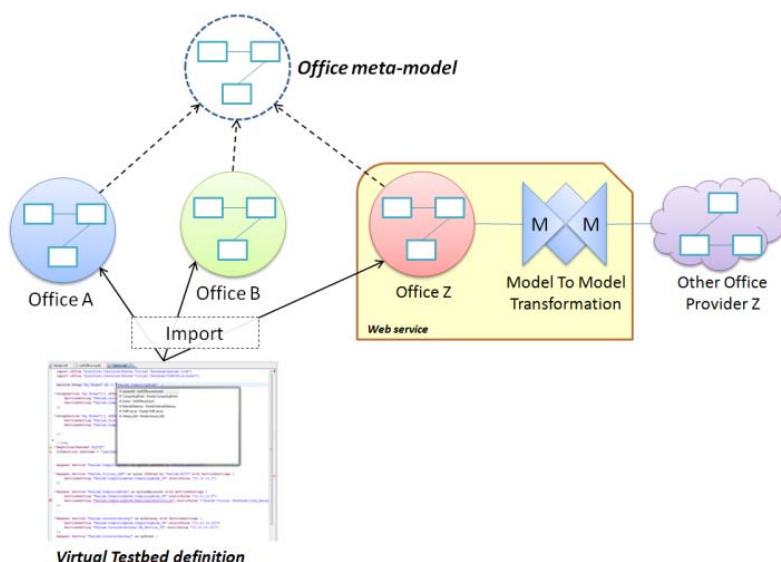


Figure 1 The Office meta-model

The DSM based approach we propose for supporting federation scenarios is illustrated in figure 1. The entity *Office* is defined as an Organization that offers services and resources. An *Office meta-model* defines what an office is, what are the participating entities (i.e. defines what is a resource, defines what is a service, how a service is supported by a resource, service taxonomies, service compositions, SLAs, users, etc) and their relationships. The meta-model is expressed in an object-oriented notation (i.e. UML). For example, as illustrated in figure 1, Office A and Office B are realizations of the Office meta-model. In the meta-model the entity *Virtual Customer Testbed* is also defined. A *Virtual Customer Testbed* (VCT) is a well-defined request of services or resources and their configurations in order to combine them in a single

composition. The VCT includes requests and configurations for services or resources by many Offices (organizations) since they conform to the meta-model. A VCT is defined by a user, the *VCT designer*.

Our approach also utilizes a Software Engineering practice as a solution to simplify the combination of services and resources from third parties that already have a model but are non-conformant to our meta-model. The conformant model of Office Z is a result of a *model-to-model transformation* (statically or dynamically produced) from the non-conformant model of Provider Z. (For example FEDERICA, GENI would be non-conformant Offices if the entity *slice* is not defined in the meta-model). It is worth noting though that the meta-model must include a collection of common definitions by the resource federation domain, in order to be efficient and applicable.

Finally, as part of our approach we *propose a resource agnostic definition, thus supporting Offered Service-oriented definitions* (without excluding though the support of mixed services and resources requests). The mapping between an offered service and an offered physical or logical resource (which implements the service) is defined by the entity of a Contract. A ResourceServiceContract entity maintains a relationship between one Offered Resource and one Offered Service. For instance, there is one contract that a specific Offered Service (i.e. Computing Service) is implemented by an Offered Resource (i.e a Virtual Machine at Univ. of Patras), under certain availability and cost. Of course the Office has multiple contracts for the same offered services under different availabilities.

Defining the Virtual Customer Testbed and support tooling

For the definition of a VCT, we have adopted practices of DSM, where the systematic use of textual or graphical Domain Specific Language (DSL) is involved. A DSL is defined as a specification language that offers, through appropriate notations and abstractions, expressive power focused on, and usually restricted to, a particular problem domain. For the language definition an abstract syntax (the meta-model), a concrete syntax and semantics are needed. All of these are captured in a solution workbench, which in this case is Eclipse, used both as a development but also as a deployment environment.

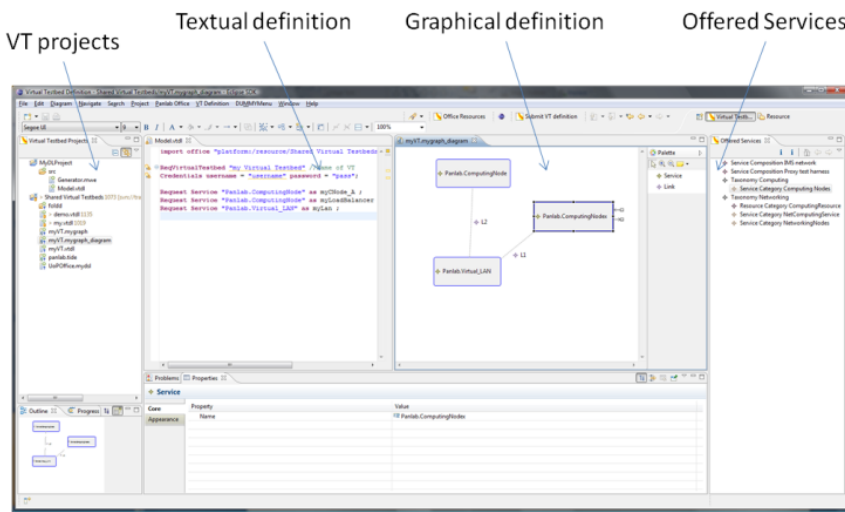


Figure 2 The Virtual Customer Testbed definition workbench

Services (regardless of organization) or Offered Resources of specific organizations and providers. The VCT designer may define configuration values and the configuration workflow, (if it is necessary) when this workflow is not easily derived by the system. As shown in figure 2 the VCT designer works on a project in order to define the VCT. This paper focuses mostly on the textual definition, which is capable of providing unambiguous definitions and helps the VCT designer to define simple or complex VCTs avoiding design and configuration ambiguities. The editor is based on the textual modeling framework (TMF) of Eclipse and specifically the XText framework, which helps defining rich editors by a definition of a specific syntax.

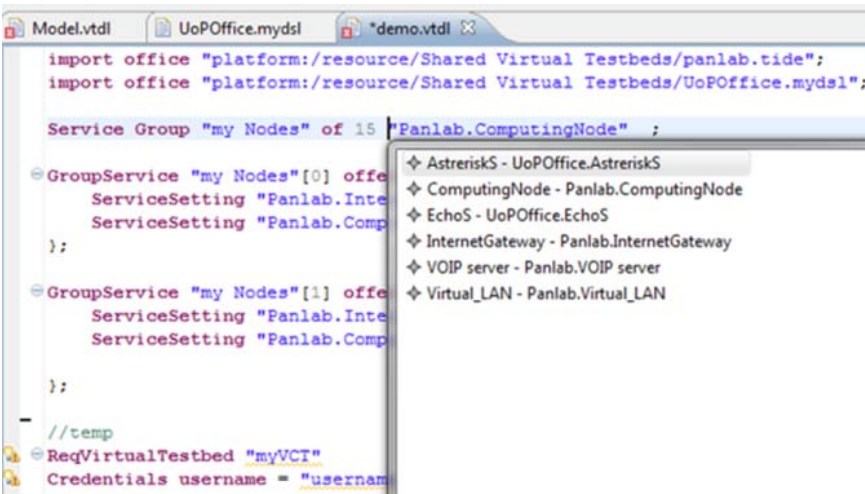


Figure 3 The VTDL editor

Moreover, in the syntax, we define a way for helping the user on selecting Offered Services and Offered Resources, with importing definitions by different Offices. This is implemented by Import statements (see figure 3 top), allowing for creating federation scenarios from services of different organizations. For example in Figure 3 we present a federation scenario for defining a VCT with offered services and offered resources from both Panlab and University of Patras. This is possible of course if the imported office models conform to the same office meta-model, as the concept illustrated in figure 1. Additionally, the defined syntax helps configure the requested Services.

As discussed, in order to utilize a non-conformant organization's model, a Model-to-Model (m2m) transformation should be implemented. This is accomplished in many ways. For example a conformant model is created by applying the well known Query/View/Transformation (QVT) operation to the non-conformant model in order to create a conformant model. This transformation is done, for example, dynamically by a webservice when the VCT designer requests to import services from a non-conformant organization. Another way is to maintain a static conformant description of the non-conformant model.

We define a DSL for Virtual Testbed definition, called *Virtual Testbed Definition Language* or **VTDL**, where the Office meta-model is the abstract syntax of the language. VTDL comprises a concrete textual syntax and a graphical notation, both synchronized for the same result. The graphical approach is used for an overall "birds-eye view" and the textual for more complex scenarios. All editors implement instances of the same model, so it is possible to synchronize, both graphical and textual syntaxes, since they realize a common model.

The definition of the VCT consists of request statements for Offered Services (regardless of organization) or Offered Resources of specific organizations and providers. The VCT designer may define configuration values and the configuration workflow, (if it is necessary) when this workflow is not easily derived by the system. As shown in figure 2 the VCT designer works on a project in order to define the VCT. This paper focuses mostly on the textual definition, which is capable of providing unambiguous definitions and helps the VCT designer to define simple or complex VCTs avoiding design and configuration ambiguities. The editor is based on the textual modeling framework (TMF) of Eclipse and specifically the XText framework, which helps defining rich editors by a definition of a specific syntax. Figure 3 displays the implemented editor where we define the meta-model (for the abstract syntax) and implement the concrete syntax and validation mechanisms during defining a VCT in VTDL. Syntax highlighting, context assistance, validation errors and warnings are some of the features. This VCT definition framework is called Testbed Integration Development Environment (TIDE) installed as Eclipse plugins.

Moreover, in the syntax, we define a way for helping the user on selecting Offered Services and Offered Resources, with importing definitions by different Offices. This is implemented by Import statements (see figure 3 top), allowing for creating federation scenarios from services of different organizations.

Specification of Offered Services and Offered Resources

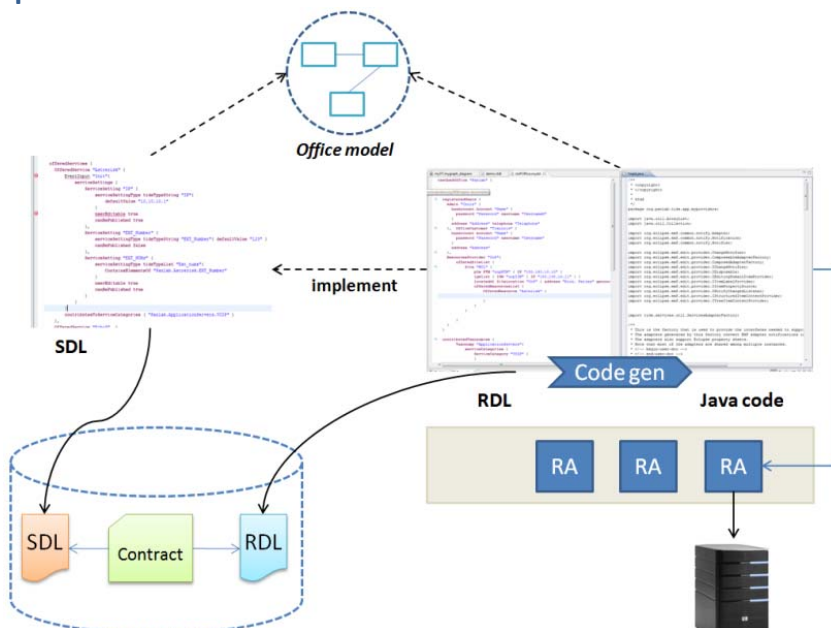


Figure 4 Specifying an Offered Resource

While the meta-model is used for defining an office, what an offered service is and how it relates to other entities, there is a need to define Offered Services themselves, having of course the meta-model as its abstract syntax. Towards this, we define a human editable syntax, a Service Description Language (SDL) in order to specify an Offered Service of an office (i.e. its taxonomy, description interfaces, constraints, etc). The SDL specification is applicable to federation scenarios by simply importing it to the VCT description. For example a Computing Service is specified in SDL.

The same concept is needed for describing an Offered Resource that is offered by a provider which should implement an Offered Service. We define a Resource Description Language (RDL) that has the meta-model as its abstract

syntax. As an example, one defines in RDL a Computing Resource at the University of Patras which implements the Computing Service of the Office. Both an SDL and an RDL are associated under a Contract and stored in the office repository. Moreover, a contract implies more: The Offered Resource is compliant with the contractual Service (at least implements the settings [*interfaces*] of the Service). This concept is illustrated in figure 4.

Going a step further, we introduce mechanisms to transform (with code generation) the RDL specification to the entity *Resource Adaptor* (RA). The RA implements the well-known "*adapter design pattern*": an adapter (the RA) allows Adaptees (the Resources) to work together that normally could not because of incompatible interfaces, by providing its interface to compatible clients while using the original interface. Adaptee is the underlying resource. Potential clients are workflow provisioning engines that are used to configure the resources. Thus, an RA provides a common, compatible interface to workflow provisioning engines.

Conclusion and acknowledgments

In this work we propose an approach that adopts Domain Specific Modeling practices for federation scenarios. The approach considers a federation meta-model and how it is used in order to create a formal description of valid, comprehensible and unambiguous configurations that support federation scenarios. These federation scenarios are specified by using the Virtual Testbed Definition Language (VTDL). Additionally, we present a tool that supports the approach.

Parts of the work presented here have been designed and developed in the context of PII (<http://www.panlab.net>) an EU funded 7th Framework project.

About the author

Christos Tranoris, Research Fellow, Electrical and Computer Engineering, University of Patras, Greece, email: tranoris@ece.upatras.gr, tel: 0030 6936985680.

Christos Tranoris received his Diploma degree in 1999 and holds also a PhD since 2006 from the Electrical and Computer Eng. of Univ. of Patras in the area of software processes on the modeling and design of industrial applications. Since 1995 he worked also on the private sector as a software engineer. During his PhD worked for several IST and Greek founded projects. Published over 15 papers in journals and conferences and holds one patent, a member of IEEE, ACM and the Technical Chamber of Greece. He is member of the Network Architectures and Management Group of the Electrical and Computer Engineering Department of University of Patras, which carries out research in the areas of Future Internet, Peer-to-Peer and Network Management while it is currently participating in related EU projects, i.e. PII, VITAL++ and Autol.