The CIM, PIM, and PSM models live on level M1. All of them are specified on level M2 by meta-models (CIM-MM, PIM-MM, PSM-MM), dialects of UML, enriching the UML core by profiles containing markup for model elements (stereotypes and tagged values). A meta-model describes properties and structures of each of the CIM, PIM and PSM models accurately. In our case a platform based on Web services is chosen. Web services options for accessing the service repository as well as protocols for communication between services and customers form a platform of SOA implementation.

In principle the MDA process starts with defining CIM or domain models, then these models can be transformed by professionals to PIM and PSM models. These transformations are called vertical transformations. A direct vertical transformation is not always possible because there gaps between the models that are too big to bridge in a single transformation. In these cases, additional transformations are used to, for instance, transform a certain abstract PIM model to a more detailed PIM (horizontal transformation).

Various types of platforms may differ. A common feature of the platform is its service-oriented architecture, and the specific technologies of implementation are Web services and service arrangements (for example, Apache Axis, or Oracle BPEL). As it was noted above, metamodelling is one of the most important concepts of the MDA. Ontologies are considered as CIM model, so conceptual models PIM and PSM also reflect the semantics of the application for the domain corresponding to the semantics of real-world domain (Real World), regardless of the specific application needs.

According to the Model-driven approach proposed, in the analysis phase (conceptual modeling) of any software system the emphasis is placed on the data (i.e., in the domain information or CIM model), rather than in the operations (i.e., the behavior). In this vein, the MDA allows representing models of the real-world using conceptual models that abstract key domain concepts, to represent them appropriately and allows transforming them into code correctly.

Considered approach to service systems design is well adapted to the peculiarities of modern distributed computing environments (such multi-clouds or grid), where Web services can be found in various geographically dispersed repositories (possibly in several equivalent embodiments), and their composition as applications are executed on different computing nodes of the environment, resources of which are freed at the beginning of the next execution of service compositions. As a result, the implemented applications have dynamically variable architecture and variable component compositions.

So instead of traditional usages of programming languages for specifying how a system is to be implemented, the approach in hands allows specifying what system functionality is required and what architecture is to be used.

POSSIBLE UNG CONTRIBUTIONS TO EOSC

The UNG-community is interested to participate in development of Data and services layer of European Open Science Cloud with the following activities:

- Using domain ontology as a meta-model basis for generating a conceptual model for specific applications or information systems.
- Developing wide set of ready-made multidisciplinary invariant computing services to researcher and innovators which support mathematical modeling or mathematical experiments in Science and Engineering.
- Configuration and coordination of services in architecture, based on the services, by using a template “request / response” for an “one-to-one” connection (or synchronous model); using a template “publish/subscribe” for “one-to-many” communication (or asynchronous model); using intelligent agents when each agent has at its disposal some of the knowledge of the business process and can share this knowledge with other agents.
- Platforms and information systems integration by harmonization of used domain ontologies.
- Providing Service models in machine-readable form that allows service systems developers to convert them directly into executable code (CIM-> PIM -> PSM -> UML).
- Contributing to “Digital innovation hub” for supporting innovation with industry/SMEs by the approach which specifies what system functionality is required and what service architecture is to be used instead.

OJECT-ORIENTED DESIGN

RESPONSIBLE EXECUTORS

Software developers, most of the time working with the code.

TECH REQUIREMENTS ANALYSIS STAGE

The requirements are given in natural language for system analysts who convert their technical characteristics in order to enable developers/designers to understand them.

DESIGN STAGE

Refinement of the class structure, describing the architecture (structure components and modules), choosing the building blocks.

IMPLEMENTATION STAGE

Encoding in a particular programming language (Java, C, ++, C #). Development is done by one (virtual or physical) team, which creates functions, classes, modules.

TESTING AND SUPPORT

Testing is usually performed in the same organization on the basis of source code and functional specifications. Test scripts are defined by developers/testers. Result: project-specific solutions for selected domain with relatively short using time, which are difficult to reuse.

MODEL-ORIENTED SOA DESIGN

Software architects, most of the time working jointly with the domain experts in ontology and models.

Domain analysis is done by the service provider that allows application developers to focus only on discovering and composing services which meet the business/technical characteristics.

Select or build ontologies of the application domain to application itself.

Repository services discovering.

Service models are provided in machine-readable form that allows systems developers to convert them into executable code (CIM -> PIM -> PSM -> UML).

New services can be dynamically created using the existing ones.

Testing is divided between the service provider, the broker and the client with little or no interaction between them. Test scripts can be generated on the basis of service metadata and specifications.

Result: Cross-project solutions which are serving multiple domains and are shared by many designers and organizations.