

AIM@SHAPE

Advanced and Innovative Models And Tools for the
development of Semantic-based systems for
Handling, Acquiring, and Processing knowledge
Embedded in multidimensional digital objects

IST NoE No 506766

Deliverable D1.2.3.1



Ontology for Product Design – 2nd version

Circulation:	¹ CO
Partner(s):	IGD (Leader), IMATI, INPG, INRIA, ITI, SINTEF
Authors:	N. Sevilmis, C. Catalano, E. Camossi, V. Cheutet, R. Ferrandes
Contributors:	G. Brunetti, T. Dokken, F. Giannini, J.-C. León, M. Pitikakis, G. Vasilakis, J. Wintz
Version:	02
Stage:	Completed
Date:	Friday, August 18, 2006

¹ Please indicate the dissemination level using one of the following codes:

PU = Public

PP = Restricted to other programme participants (including the Commission Services).

RE = Restricted to a group specified by the consortium (including the Commission Services).

CO = Confidential, only for members of the consortium (including the Commission Services).

Copyright

© Copyright 2006 The AIM@SHAPE Consortium

consisting of:

CNR-IMATI-GE	C.N.R. – Istituto di Matematica Applicata e Tecnologie Informatiche Dept. of Genova, Italy
DISI	Università di Genova – Dipartimento di Informatica e Scienze dell'Informazione, Italy
EPFL	École Polytechnique Federale de Lausanne, Switzerland
FhG/IGD	Fraunhofer Institut für Graphische Datenverarbeitung, Germany
INPG	Institut National Polytechnique de Grenoble, France
INRIA	Institut National de Recherche en Informatique et Automatique, France
ITI-CERTH	Informatics and Telematics Institute – Center for Research and Technology Hellas, Greece
UNIGE	Université de Genève, Switzerland
MPII	Max-Planck-Institut für Informatik, Germany
SINTEF	Stiftelsen for industriell og teknisk forskning ved Norges Tekniske Høgskole, Norway
TECHNION	Technion – Israel Institute of Technology, Israel
UU	Utrecht University, Netherlands
WEIZMANN	Weizmann Institute of Science, Israel

This document may not be copied, reproduced, or modified in whole or in part for any purpose without written permission from the AIM@SHAPE Consortium. In addition to such written permission to copy, reproduce, or modify this document in whole or part, an acknowledgement of the authors of the document and all applicable portions of the copyright notice must be clearly referenced.

All rights reserved.

This document may change without notice.

Document History

VERS.	Issue Date	Stage	Content and changes
01	22 July 2006	60%	2 nd version of the Product Design Ontology
02	7 August 2006	Completed	Finalization of the validation process

Executive Summary

This document contains the deliverable **D1.2.3.1** of the IST NoE AIM@SHAPE.

The deliverable ***D1.2.3.1 – Ontology for Product Design – 2nd Version*** – is intended to provide a second version of the Product Design Ontology, which is part of the ontology development process in the network.

The task leader is **IGD** and has been actively supported by all involved partners.

For a better understanding of this document it is recommended to have read the previous D1.2.3.1 “Ontology for Product Design – 1st Version”.

This document is structured as follows: Section 1 is about the evolution steps taken in the last period. Section 2 sketches the main changes proposed to validate the common ontologies, while section 3 describes the validation process for the structure of the changed PD ontology. Section 4 gives an outlook concerning the future work.

Table of Contents

INTRODUCTION.....	7
1 ONTOLOGY EVOLUTION	8
1.1 Roles of shapes: The ShapeRole and the PDModel concept	8
1.2 SimulationResult Hierarchy	11
2 COMMON ONTOLOGIES	12
3 PD ONTOLOGY VALIDATION PROCESS.....	13
4 FUTURE WORK.....	16

Table of Figures

Figure 1: Shape life-cycle in a typical Finite Element Analysis (FEA) design evolution loop.
.....8

Figure 2: *ShapeRole* Taxonomy and properties9

Figure 3: Relations between *ShapeRole* and other involved concepts 10

Figure 4: The *PDModel* taxonomy and its relations 11

Figure 5: Relations between *Task*, *PDModel* and *ShapeRole* 11

Figure 6: *SimulationResultType* Hierarchy 12

Figure 7: Screenshot of the AIM@SHAPE Search Engine showing the results of CQ 2a) 14

Figure 8: Screenshot of the AIM@SHAPE Search Engine showing the results of CQ 3b) 15

INTRODUCTION

The general objective of the Product Design Ontology (PDO from now on) is to assist the development of shape processing tools for design. In the previous deliverable D1.2.3.1 we have presented the first version of the PDO. The focus of the first version of the PDO lay on the modeling process, tool and shape know-how (in general) relevant to the phases of the product development process, i.e., the free-form shape modeling for styling and the engineering analysis.

For the second version of the PDO we have introduced some new ontological concepts formalizing the role of shapes along the product development process as well as a hierarchy for specifying simulation results. Competency Questions (CQs) that could not yet be answered motivated the structural modifications of the PDO. In doing so, we also considered the Common Ontologies for Shapes and Tools in order to be consistent with the concepts to be shared.

According to this, in section 1 we describe the evolution steps taken in the last period. Section 2 outlines our cluster contribution to the Common Ontologies. Section 3 addresses the PDO validation process for a quality check of the evolution steps: in fact, here we can verify if the evolved PDO is able to answer more CQs associated with cluster specific scenarios. Finally, in section 4, we describe the future work.

1 ONTOLOGY EVOLUTION

This section describes new ontological concepts that have been chosen and integrated in the PDO, yielding its evolution.

1.1 Roles of shapes: The ShapeRole and the PDMModel concept

As a reminder, we show in Figure 1 the typical shape evolution cycle for Finite Element Analysis (FEA).

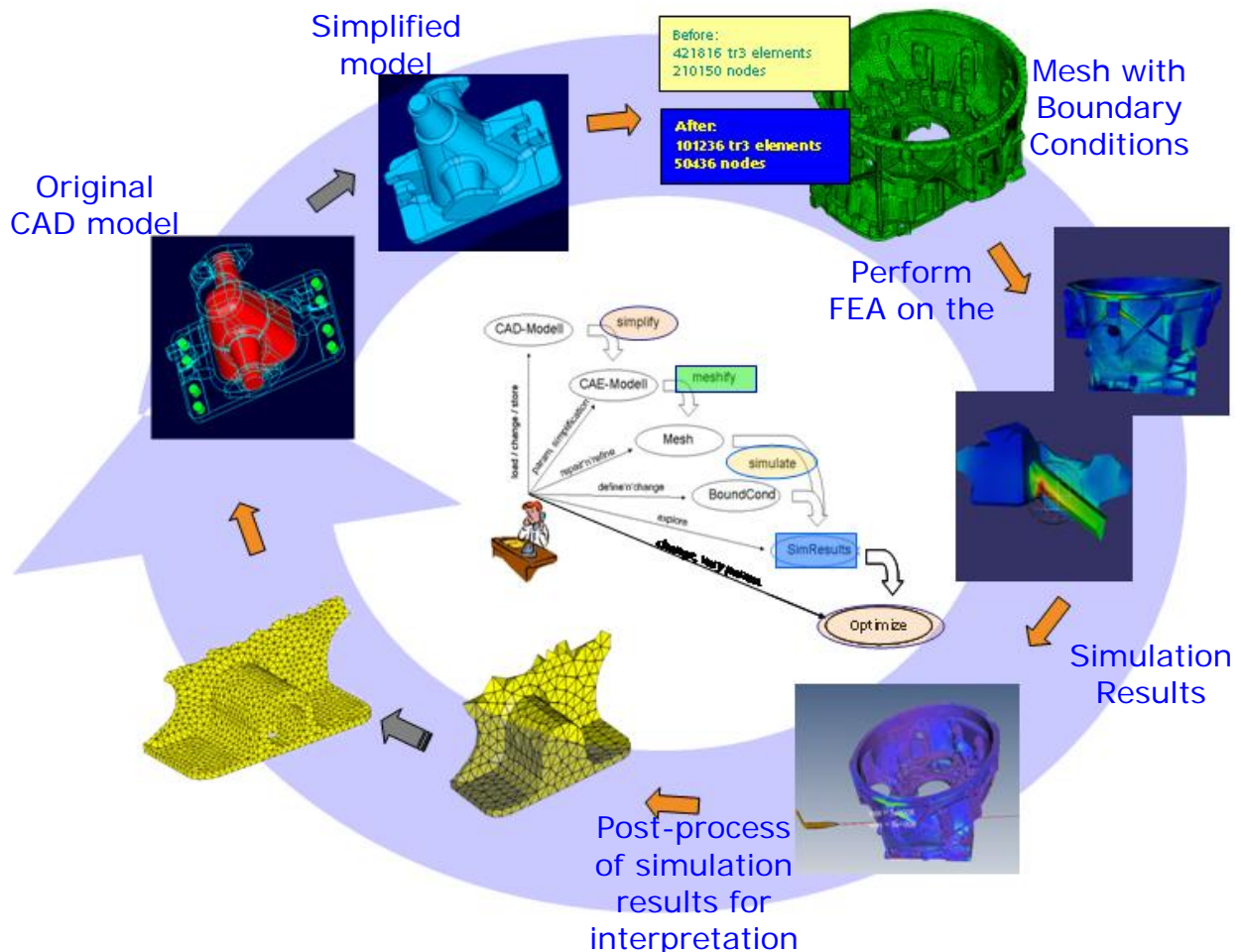


Figure 1: Shape life-cycle in a typical Finite Element Analysis (FEA) design evolution loop.

The starting point is the original CAD model. As simulation is computational expensive we have to simplify the CAD model. Simplification here typically means to reduce the complexity by suppressing design details (de-featuring) that are considered not to influence significantly the simulation outcome. Once having simplified the model, we can go on with the tessellation to generate the corresponding mesh, applying some boundary conditions, performing FEA on this mesh, and finally we can post-process the simulation result for interpretation of the simulation outcome. Simplification can be done either before meshing in the CAD system or after tessellation.

Since purposes are different in the product life-cycle, the same shape can play different roles within the shape life-cycle. For instance, the input and output of simulation are 2D or 3D meshes. The first version of the PDO was not able to capture the characteristics of a mesh needed in a FE simulation

process and the information associated to a mesh before and after the FE analysis. In order to distinguish among all these concepts, it is necessary to enrich the shape representation with its role within the shape life-cycle. For this reason, the Product Design cluster introduced two new concepts in order to model the role of shapes along the Product Development Process: *ShapeRole* and *PDMModel*.

The *ShapeRole* concept

The *ShapeRole* concept is designed to be used in relation with shape types that need to be enriched with additional information, intervening in a specific task of the Product Design workflow. Figure 2 depicts the *ShapeRole* concept and its subclasses. In the second version of the PDO, two direct subclasses of *ShapeRole* have been introduced: *SimplificationMesh* and *FiniteElementMesh*. The concept *FiniteElementMesh* is further specialized through the concepts *PreSimulationMesh* and *PostSimulationMesh* in order to distinguish between the two different roles that a mesh can play in the context of simulation.

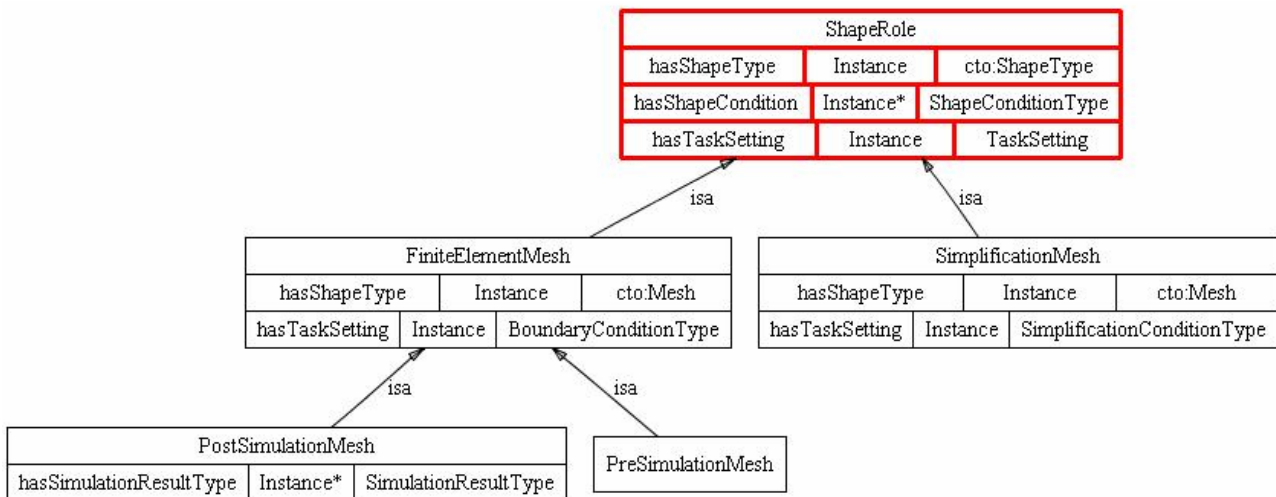


Figure 2: *ShapeRole* taxonomy and properties

Moreover, the *ShapeRole* concept relates to a *ShapeType* with:

- (1) **geometric conditions** that have to be satisfied by a shape type in a certain design phase. For instance, to perform a simulation task, a conformal mesh is required. Thus, in dependence of the task to be performed, a shape type has to fulfil specific geometric conditions.
- (2) **necessary conditions** required to perform a certain task. For example, the boundary conditions are necessary to perform a simulation.

Figure 3 shows the relations between the *ShapeRole* taxonomy and other involved concepts.

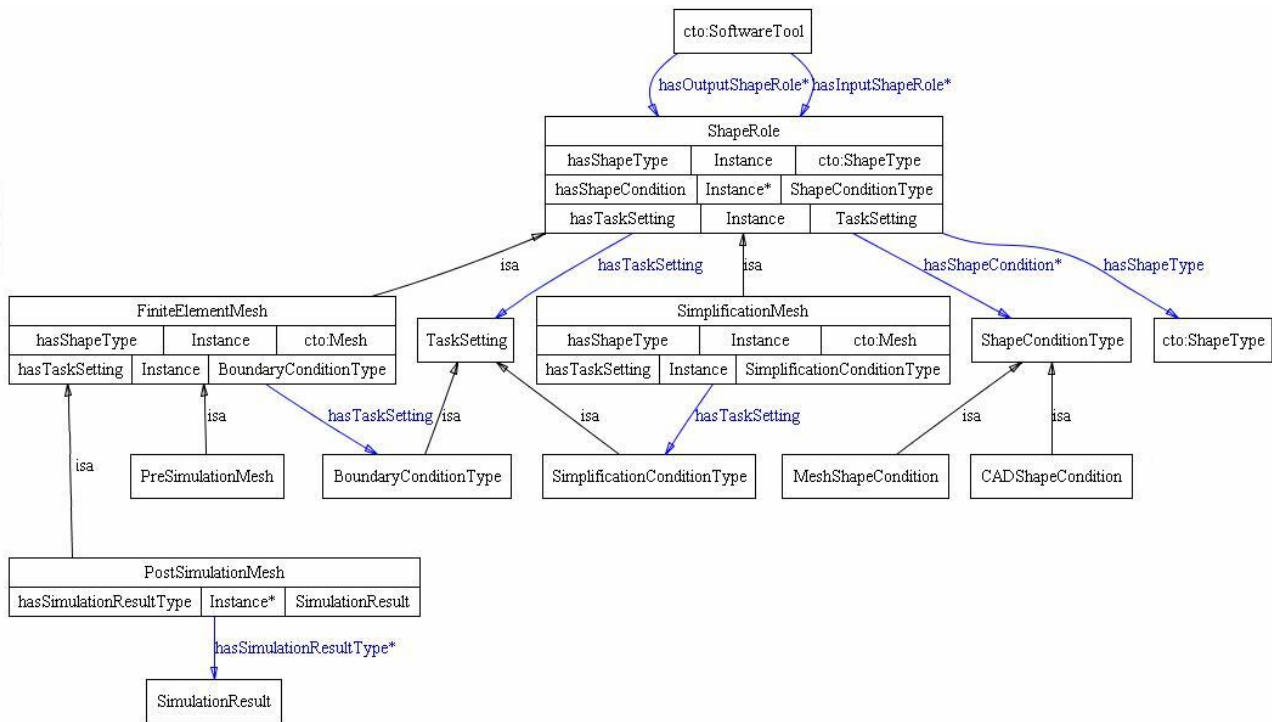


Figure 3: Relations between *ShapeRole* and other involved concepts

The concepts *ShapeRole* and *ShapeConditionType* are related through the relation *hasShapeCondition*; the concepts *ShapeRole* and *TaskSetting* are related through the relation *hasTaskSetting*. The relation *hasShapeCondition* is used for specifying the **geometric conditions**, whereas the relation *hasTaskSetting* is used in order to define **necessary conditions**. In particular, *TaskSetting* has currently two subclasses, *BoundaryConditionType* and *SimplificationConditionType*, and it may be further specialized with more task conditions.

To maintain the consistency of the different roles, some property restrictions have been added: a *FiniteElementMesh* is restricted to have a mesh as *ShapeType* and boundary conditions associated via the *hasTaskSetting* value.

PostSimulationMesh has an additional property in comparison with a *PreSimulationMesh*, that is *hasSimulationResultType*. In fact, a mesh resulting from a simulation analysis is always equipped with the simulation results that need to be interpreted.

To search for tools that apply and/or provide for a given shape role, we introduced two relations between the classes *SoftwareTool* of the Common Tool Ontology and *ShapeRole*: *hasInputShapeRole* and *hasOutputShapeRole*.

The *PDMModel* concept

The *PDMModel* (Product Design Model) concept has been introduced to model the shape role of a specific shape (that is an instance of *ShapeRepresentation*). In this way, we can capture for example that a shape is a simplification of a given CAD model according to certain conditions (associated to the corresponding *ShapeRole*).

At the moment, the class has two subclasses, which are *SimplificationModel*, where the shape role is a *SimplificationModel* and *SimulationModel*, having a *FiniteElementMesh* as *ShapeRole* (see Figure 4).

Due to the ongoing reorganization of the Common Shape Ontology (see section 2), the information related to features has been moved into *PDModel*. In detail, we added a class *FeatureInfo*, which includes a *FeatureType* and the number of that feature present in the model; *PDModel* has a (multiple) property *hasFeatureInfo* (see Figure 4).

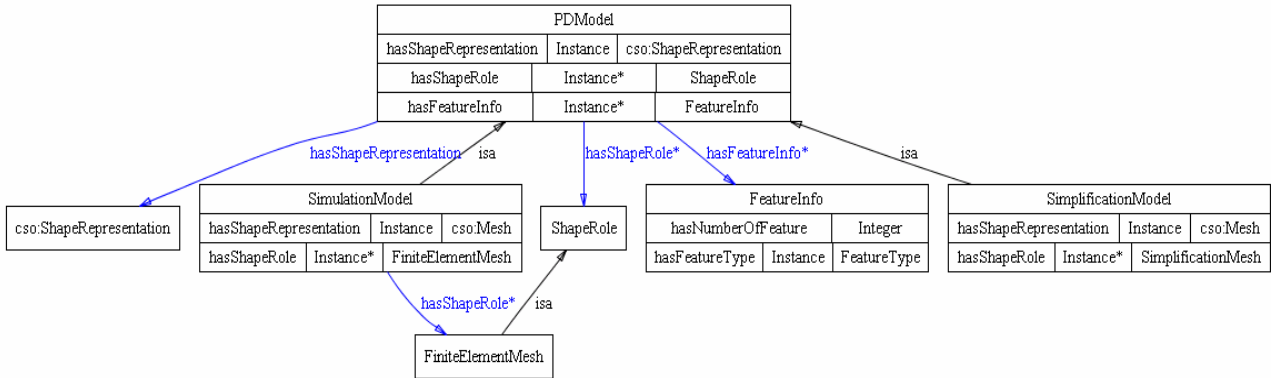


Figure 4: The *PDModel* taxonomy and its relations

Finally, *ShapeRole* and *PDModel* are related to *Task* with the relations *hasInputShapeRole*, *hasOutputShapeRole*, *hasInputPDModel*, and *hasOutputPDModel* (see Figure 5).

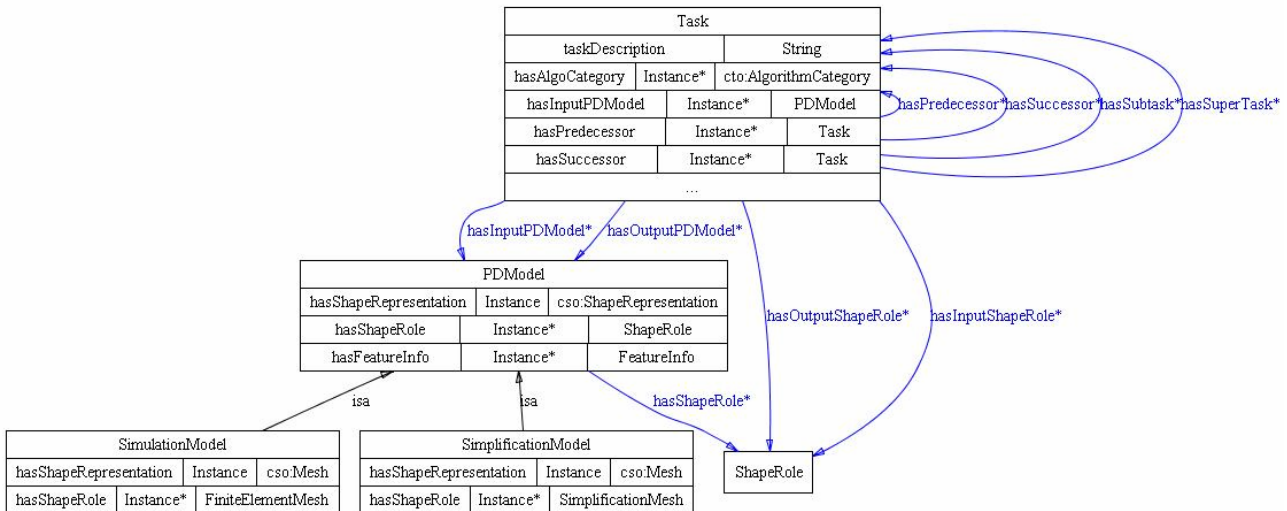


Figure 5: Relations between *Task*, *PDModel* and *ShapeRole*

1.2 SimulationResult Hierarchy

As pointed out in section 1.1 we have illustrated that after a FE simulation a FE mesh has some simulation results associated with it. The concept *SimulationResultType* has been organized in a taxonomy, consistently with the *BoundaryConditionType* hierarchy (see Figure 6). The property *hasResultType* permits to distinguish among a scalar, vector and tensor type of result.

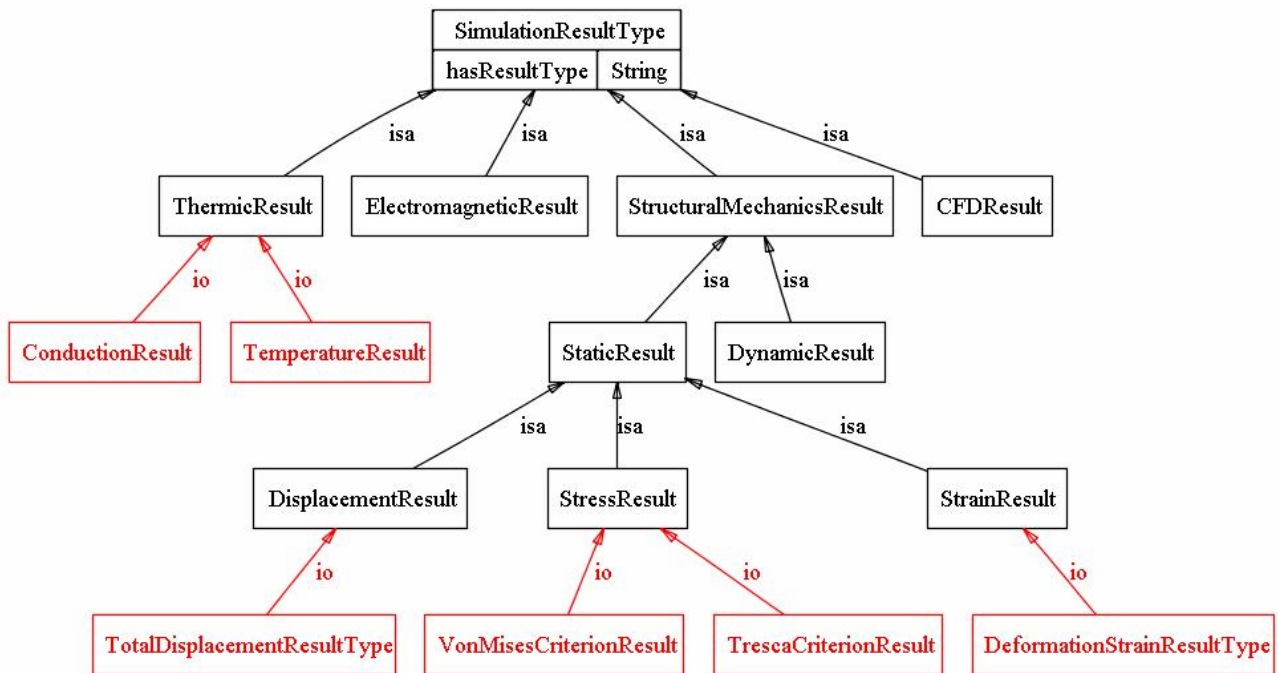


Figure 6: *SimulationResultType* Hierarchy

The first subdivision level of the taxonomy is similar to the *BoundaryConditionType* hierarchy, with the classes *ElectromagneticResultType*, *ThermicResultType*, *StructuralMechanicsResultType* and *CFDResultType*. The *StructuralMechanicsResultType* class has two subclasses: *DynamicResultType* and *StaticResultType*, which is further subdivided into *DisplacementResultType*, *StressResultType* and *StrainResultType*.

2 COMMON ONTOLOGIES

Due to the necessity to align the Shape Repository metadata and the classes and properties of the Common Ontology for Shapes (SCO), more general concepts have been introduced to describe the shapes of interest for the PD cluster in cooperation with the task T1.5. For this reason, the *CADRepresentation* has been removed from the SCO and substituted with *BRep*, *ParametricRepresentation* and *ImplicitRepresentation*, the last two specialized to distinguish curves and surfaces. In these classes, the meaningful metadata related only to the topology and geometry of the shapes have been selected and are currently under validation. As a consequence, all the information present in *CADRepresentation* that is more specific to the cluster domain has been moved to the Product Design ontology, particularly to the class *PDModel* (e.g. see the feature information).

Regarding the Common Ontology for Tools (TCO), the proposal raised during the validation phase to reorganize the relationships among *SoftwareTool*, *Functionality* (*AlgorithmCategory* in the previous version) and *Algorithm* has been supported since it better suits the cluster needs. On the contrary, software tools in the PD domain do not apply only to shape types, but also to shape roles: as explained above, tools often deal with shapes equipped with additional information necessary to perform a specific task. For this reason, in PDO we introduced two new relations between the classes *SoftwareTool* of the TCO and *ShapeRole*: *hasInputShapeRole* and *hasOutputShapeRole*.

3 PD ONTOLOGY VALIDATION PROCESS

In this section we present the list of CQs that have validated the new structure of the PDO. Regarding the FEM scenarios, the CQs we are now able to answer are:

1. Find a mesh model of a given format, whose product category is mechanical part, and is a real object, which has a certain boundary condition type.

This was the CQ as formulated in the previous version of the PDO (see previous D1.2.3.1). With the current version of the PDO, we are able to look for a logical combination of *PDModel* and a mesh model. Thus, we can reformulate the CQ and ask for:

- a) Find a *SimulationModel*, which related mesh fulfils the demanded properties (given format, mechanical part, real object) and which *ShapeRole* is a *FiniteElementMesh* with boundary conditions of the specified type.

2. Find shapes that are simulation results of FEA of a given boundary condition type, that are possibly real-world objects, and that are in a given file format.

Even this CQ, formulated in the previous deliverable D1.2.3.1, can be now interpreted as:

- a) Find a *SimulationModel*, which *ShapeRole* is *PostSimulationMesh* and which has boundary conditions of the specified type.

3. Find the FE analysis tools which execution platform is Windows 2k, which have as output a model with format abaqus (or ansysn, unv) and some information of the simulation like displacement of nodes, strain energy of tetrahedral elements.

This CQ can be answered in two different ways. Implicitly, we can search for a tool with a precise functionality (e.g. *ElectroMagneticAnalysisAlgorithm* and *ThermalAnalysisAlgorithm*) and a given output format and this implies a specific output type. Explicitly, we can search for tools with a given output shape role thanks to the property *hasOutputShapeRole*. These interpretations lead to the following CQs:

- a) Find the FE analysis tools which perform a *ThermalAnalysis*, which execution platform is Windows 2k and which have as output a model with format abaqus (or ansys, unv).
- b) Find the FE analysis tools which execution platform is Windows 2k, which have as output a *PostSimulationMesh* with format abaqus (or ansys, unv) and a *ThermalSimulationResult* type.

4. Find post-processing algorithms or post-processing software tools that work with a given type of simulation results.

This CQ can be answered by means of the property *hasInputShapeRole* linking a software tool with shape role. Analogously, the following CQ can be answered:

- a) Find shapes that are *SimulationResults* of a FEA of given *BoundaryConditionType*, that are possibly real-world objects, and that are in a given format.

Finally, the CQs that involve information about features can be answered passing through the concept of *PDModel*. For example,

5. Find a PDModel of a given format, whose product category is mechanical part, which is a real object and has through holes.

To test the CQs and to validate the new structure of the PDO we have used the semantic-based AIM@SHAPE Search Engine. As an example of the queries we are able to answer, we present in Figure 7 and Figure 8 two screen shots of the AIM@SHAPE Search Engine: Figure 7 shows the results of the CQ 2a) and Figure 8 shows the results of the CQ 3b).

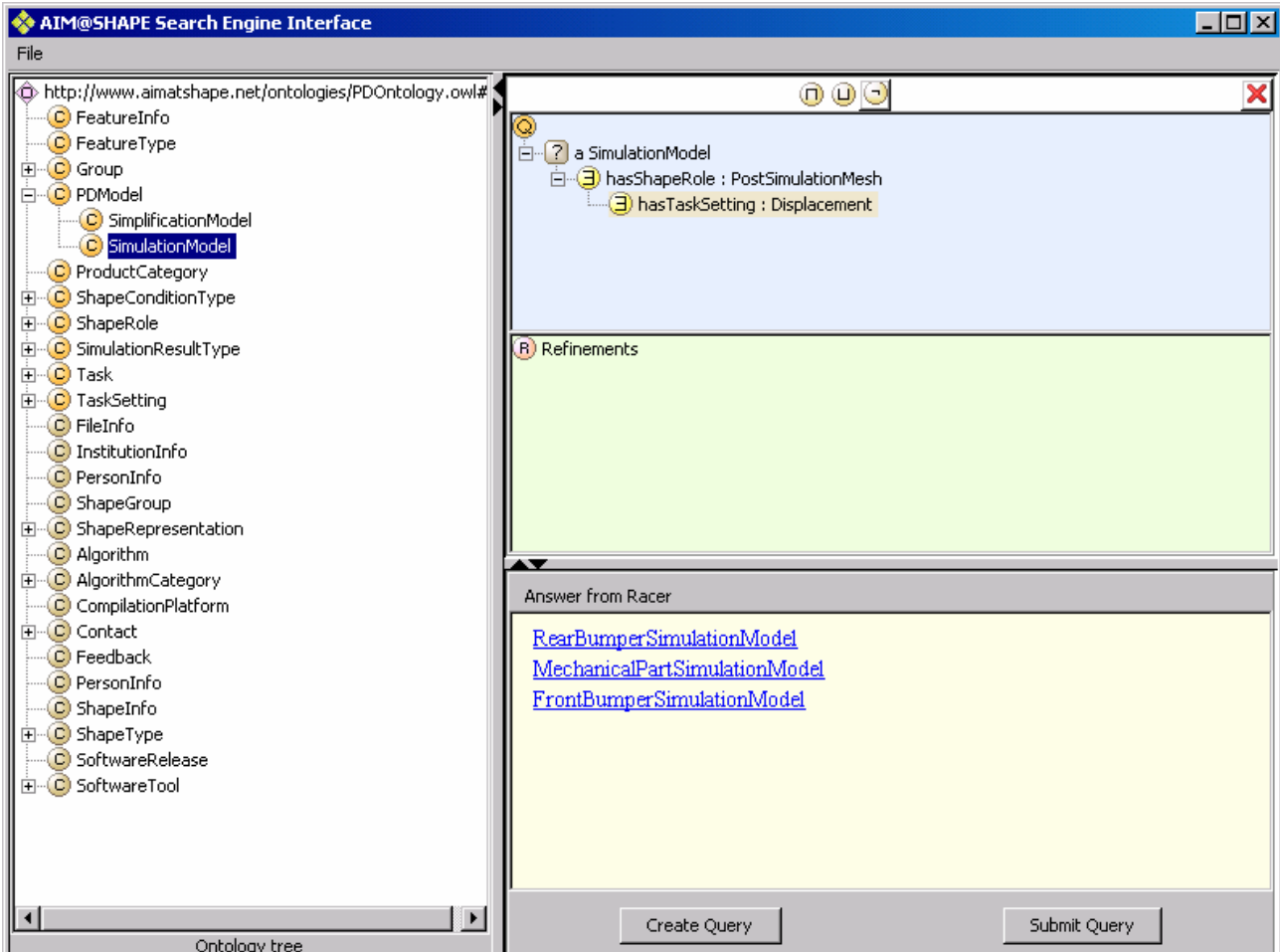


Figure 7: Screenshot of the AIM@SHAPE Search Engine showing the results of CQ 2a)

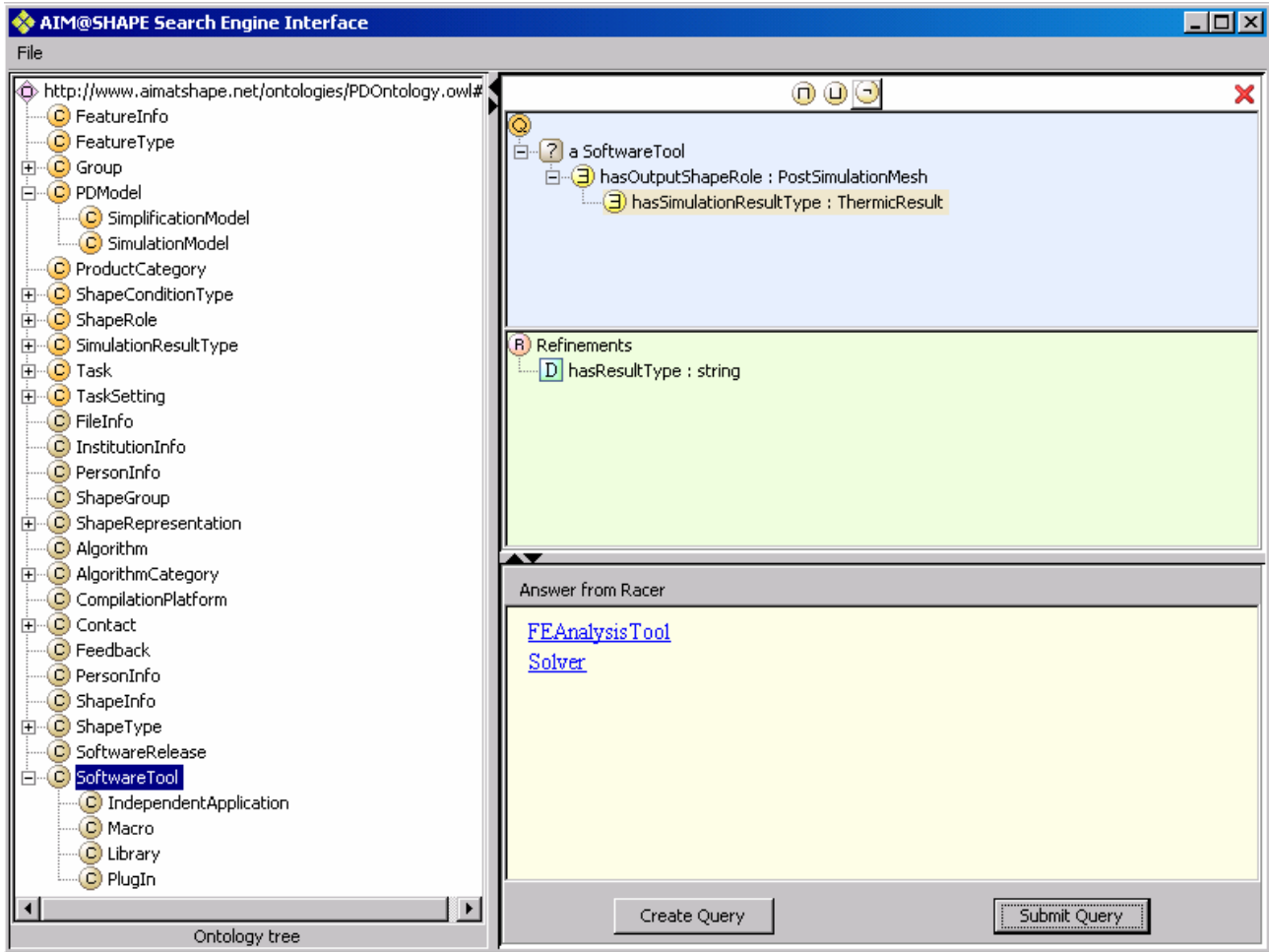


Figure 8: Screenshot of the AIM@SHAPE Search Engine showing the results of CQ 3b)

4 FUTURE WORK

In this phase of the work we mainly focused on the simulation task, formalizing the entities involved in the process. In particular, the shape role, the geometric and the task-oriented conditions have been deeply investigated and described in order to address the scenarios “FEA model preparation and analysis of the details effects” and “Acquisition of test data for FEA post-processing” (see previous D1.2.3.1).

We also reorganized and enriched some entities of the Common Ontologies in order to better suit our cluster view.

Some concepts are not completely developed; the following issues have to be addressed:

- When the *ShapeRole* is a *SimplificationMesh*, the additional information needed is related to some conditions which can drive or constrain the simplification process. For instance, some geometric constraints could be edge size or ratio, minimum triangle angle, etc. We could also think of mechanical criteria driving the simplification process, e.g. volume or area variation, etc. The concept of *SimplificationConditionType* has been introduced, but it will be developed further on.
- To be able to recover a certain *ShapeRole*, a *ShapeType* needs to verify some conditions. Therefore, the need of formalizing the concept of conditions related to a shape emerged, and the class *ShapeConditionType* has been introduced. At the moment, this class has two subclasses: *CADShapeCondition* and *MeshShapeCondition*. To give an example, a mesh needs to fulfil some additional conditions (e.g. conformity) to fit the concept of *FiniteElementMesh*. Further work is still needed for modeling this aspect.

Within the cluster we debated a lot about a suitable grouping mechanism. The formalization proposed in the common shape ontology is consistent with the DSW requirements, but it is not fully compliant with the PD cluster view. Thus, we should investigate how to extend the common *ShapeGroup* class for our needs. The assembly gives an example of a typical CAD group that needs to be shared among the network partners.

Finally, the scenario “Quality check of CAD-models” has not been tackled yet.