Unified Modeling Language description of the object-oriented multi-scale adaptive finite element method for Step-and-Flash Imprint Lithography Simulations

2010 IOP Conf. Ser.: Mater. Sci. Eng. 10 012247

(http://iopscience.iop.org/1757-899X/10/1/012247)

IOP Conf. Series: Materials Science and Engineering **10** (2010) 012247 doi:10.1088/1757-899X/10/1/012247

Unified Modeling Language Description of the Object-Oriented Multi-Scale Adaptive Finite Element Method for Step-and-Flash Imprint Lithography Simulations

Maciej Paszyński[(1)], Piotr Gurgul[(1)], Marcin Sieniek[(1)], David Pardo[(2)]

[(1)]Department of Computer Science
AGH University of Science and Technology,
Al. Mickiewicza 30, 30-059 Cracow, Poland

[(2)]IKERBASQUE (Basque Foundation for Sciences)
and BCAM (Basque Center for Applied Mathematics),
Bizkaia Technology Park, Building 500, E-48160, Bilbao, Spain

E-mail: paszynsk@agh.edu.pl

Abstract. In the first part of the paper we present the multi-scale simulation of the Step-and-Flash Imprint Lithography (SFIL), a modern patterning process. The simulation utilizes the *hp* adaptive Finite Element Method (*hp*-FEM) coupled with Molecular Statics (MS) model. Thus, we consider the multi-scale problem, with molecular statics applied in the areas of the mesh where the highest accuracy is required, and the continuous linear elasticity with thermal expansion coefficient applied in the remaining part of the domain. The degrees of freedom from macro-scale element's nodes located on the macro-scale side of the interface have been identified with particles from nano-scale elements located on the nano-scale side of the interface. In the second part of the paper we present Unified Modeling Language (UML) description of the resulting multi-scale application (*hp*-FEM coupled with MS). We investigated classical, procedural codes from the point of view of the object-oriented (O-O) programming paradigm. The discovered hierarchical structure of classes and algorithms makes the UML project as independent on the spatial dimension of the problem as possible. The O-O UML project was defined at an abstract level, independent on the programming language used.

1. Introduction

The first part of this paper presents the multi-scale simulation of the Step-and-Flash Imprint Lithography (SFIL), a modern patterning process [1,2]. The simulation utilizes the *hp* adaptive Finite Element Method (*hp*-FEM) [6] coupled with Molecular Statics (MS) model [5]. The *hp* adaptation is the most sophisticated version of the mesh adaptive algorithm [6,7], where element are either *h* refined (broken into smaller elements) or *p* refined (polynomial order of approximation is modified over selected elements), or both. The decisions about finite elements that need to be refined, and the kind of refinements are selected by the *hp*-adaptive FEM algorithm. The procedural version of the hp-FEM algorithm was designed and implemented by the group of Leszek Demkowicz [6] in one, two and three spatial dimensions. The *hp*-adaptive codes were designed with the procedural paradigm, and implemented in Fortran 90 language. In this paper we consider the multi-scale problem, with

molecular statics applied in the areas of the mesh where the highest accuracy is required, and the continuous linear elasticity with thermal expansion coefficient applied in the remaining part of the domain. The degrees of freedom from macro-scale element's nodes located on the macro-scale side of the interface have been identified with particles from nano-scale elements located on the nano-scale side of the interface.

The second part of the paper presents the Unified Modeling Language (UML) [7] description of the resulting multi-scale application (*hp*-FEM coupled with MS). We investigated the procedural codes from the point of view of the object-oriented (O-O) programming paradigm. The discovered hierarchical structure of classes and algorithms makes the UML project as independent on the spatial dimension of the problem as possible. The O-O UML project was defined on the abstract level, independently by the programming language being used.

One interesting example of the O-O C++ environment for solving adaptive FEM problems is the PZ project described in [8] and the latest OOPar [9]. The author of [8] describes a lot of unnecessary work he experienced while using multiple almost-identical copies of the low-level code for solving only slightly different problems and came up with a high level solution. His implementation consists of a FEM part and a Matrix part, which are independent of each other. The basic idea behind this solution is to enclose common operations into objects and implement mathematical logic as methods. Although this is quite a comprehensive solution, the coupling is still rather tight, and methods are simply reflecting low-level mathematical steps in the FEM solving algorithms, whereas more abstract approach would be desirable. Another example of a slight misuse of the O-O paradigm is the fact of accessing nodes and elements simply by a numerical, arbitrary ids instead of using much more natural object references.

2. Problem formulation

2.1. Step and Flash Imprint Lithography

The SFIL [1, 2] is a modern patterning process utilizing photopolymerization to replicate the topography of a template onto a substrate. The major processing steps of SFIL presented in Figure 1 include: depositing a low viscosity, silicon containing, photocurable etch barrier onto a substrate; bringing the template into contact with the etch barrier; curing the etch barrier solution through UV exposure; releasing the template, while leaving high-resolution features behind; a short, halogen break-through etch; and finally an anisotropic oxygen reactive ion etch to yield high aspect ratio, high resolution features. Photopolymerization, however, is often accompanied by densification. The average distance between molecules decreases and causes volumetric contraction. Densification of the SFIL photopolymer (the etch barrier) may affect both the cross sectional shape of the feature and the placement of relief patterns.

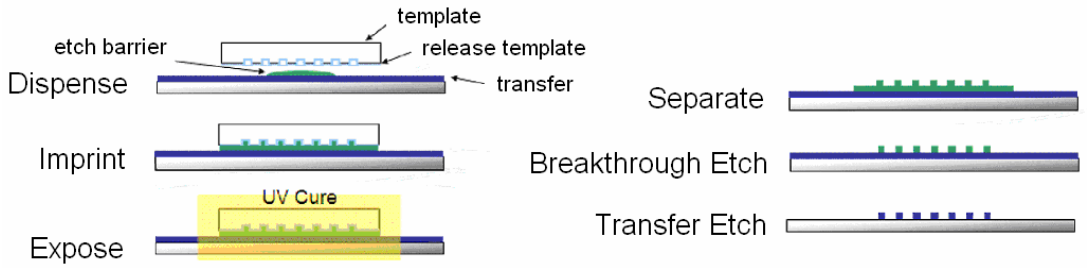

Figure 1. Step and Flash Imprint Lithography process.

2.2. Linear Elasticity with Thermal Expansion Coefficient

The linear elasticity model with thermal expansion coefficient is used to verify the material response of polymerized networks in cured etch-barrier layers that are formed during the *Expose* step.

Strong formulation. Given $g_i : \Gamma_{D_i} \ni x \to g_i(x) = 0 \in R$, θ and α_{kl} find $u_i : \overline{\Omega} \to R$ the displacement vector field such that

$$\sigma_{ij,j} = 0 \text{ in } \Omega \tag{1}$$

$$u_i = g_i \text{ in } \Gamma_{D_i} \tag{2}$$

where σ_{ij} is the stress tensor, defined in terms of the generalized Hooke's law

$$\sigma_{ij} = c_{ijkl}\left(\varepsilon_{kl} + \theta\,\alpha_{kl}\right) \tag{3}$$

c_{ijkl} are the elastic coefficients (known for a given material), ε_{ij}^0 is the initial strain, θ is the temperature, α_{kl} is the thermal expansion coefficients, ε_{ij} is the strain tensor, defined to be $u_{(i,j)}$, the symmetric part of the displacement gradients

$$\varepsilon_{ij} = u_{(i,j)} = \frac{u_{i,j} + u_{j,i}}{2} \tag{4}$$

where u_i is the displacement vector, and $u_{i,j}$ are the displacement gradients.

Weak formulation. The weak formulation is obtained by multiplying (1) by test functions $w_i \in V_i$ and integrating by parts over Ω.

$$-\int_\Omega w_{i,j}\sigma_{ij}\,d\Omega + \int_\Gamma w_i\sigma_{ij}n_j\,d\Omega = 0 \tag{5}$$

Since σ_{ij} is symmetric tensor, then $w_{i,j}\sigma_{ij} = w_{(i,j)}\sigma_{ij}$ (compare [3]), and $w_i = 0$ on Γ. We obtain

$$\int_\Omega w_{(i,j)}\sigma_{ij}\,d\Omega = 0 \tag{6}$$

Finally, we substitute (3) into (6) to obtain

$$\int_\Omega w_{(i,j)}c_{ijkl}u_{(k,l)}\,d\Omega = -\theta\int_\Omega w_{(i,j)}c_{ijkl}\alpha_{kl}\,d\Omega \tag{7}$$

since $\varepsilon_{ij} = u_{(i,j)}$.

Abstract index-free notation. For implementation issues, the most convenient is the following form**:**
Find $\mathbf{u} \in \mathbf{V}$ such that

$$\mathbf{a}(\mathbf{w},\mathbf{u}) = -\mathbf{A}(\mathbf{w}) \text{ for all } \mathbf{w} \in \mathbf{V} \tag{8}$$

where

$$\mathbf{a}(\mathbf{w},\mathbf{u}) = \int_\Omega \boldsymbol{\varepsilon}(\mathbf{w})^{\mathrm{T}}\mathbf{D}\boldsymbol{\varepsilon}(\mathbf{u})d\Omega\,,\ \mathbf{A}(\mathbf{w}) = \theta\int_\Omega \boldsymbol{\varepsilon}(\mathbf{w})^{\mathrm{T}}\mathbf{D}\boldsymbol{\alpha}\,d\Omega \tag{9}$$

Here [5]

$$\boldsymbol{\varepsilon}(\mathbf{z}) = \begin{Bmatrix} z_{1,1} \\ z_{2,2} \\ z_{3,3} \\ z_{2,3}+z_{3,2} \\ z_{1,3}+z_{3,1} \\ z_{1,2}+z_{2,1} \end{Bmatrix},\ \boldsymbol{\alpha} = \begin{Bmatrix} \alpha \\ \alpha \\ \alpha \\ 0 \\ 0 \\ 0 \end{Bmatrix},\ \mathbf{D} = \frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\nu}{2} \end{bmatrix} \tag{10}$$

The numerical simulation presented in Section 3 uses $g_i \equiv 0$, $\theta = 1$ (temperature gradient), $\alpha = -0.06115$ (obtained in [4]), with Young modulus $E = 1$ [GPa] and Poisson ratio $\nu = 0.3$ [1].

2.3. Molecular Statics Model

We consider a regular rectangular 3D grid with interacting particles. Each particle interacts with all its 26 neighbours. For each pair of interacting particles α and β, we can distinguish their initial configurations $\mathbf{p}_\alpha, \mathbf{p}_\beta$ and (unknown) equilibrium configurations $\mathbf{x}_\alpha, \mathbf{x}_\beta$

The following molecular statics models are considered:

- *Linear model assuming small deformations and quadratic potentials*. In this model the force between pair of interacting particles α and β is given by

$$\mathbf{F}_{\alpha\beta} = k_{\alpha\beta}\left(r_{\alpha\beta} + \Delta r_{\alpha\beta} - r_{\alpha\beta}{}^{o}\right)\frac{\left(\mathbf{p}_{\beta} - \mathbf{p}_{\alpha}\right)}{\left\|\mathbf{p}_{\beta} - \mathbf{p}_{\alpha}\right\|} \tag{11}$$

where $k_{\alpha\beta}$ is the spring stiffness coefficient, $r_{\alpha\beta} + \Delta r_{\alpha\beta} = \left\|\mathbf{x}_{\beta} - \mathbf{x}_{\alpha}\right\|$ is the length of the spring in the equilibrium configuration, $\mathbf{x}_{\alpha}, \mathbf{x}_{\beta}$ represents the (unknown) equilibrium configuration of particles, $r_{\alpha\beta}{}^{0}$ is the length of the unstretched spring, $\mathbf{p}_{\alpha}, \mathbf{p}_{\beta}$ represents the initial configuration of particles. The small deformations are assumed in this model, which implies the direction of the interparticle forces along the initial spring alignments $\mathbf{p}_{\beta} - \mathbf{p}_{\alpha}$.

- *Non-linear model allowing for large deformations, with quadratic potentials.* In this model the force between pair of interacting particles α and β is given by

$$\mathbf{F}_{\alpha\beta} = k_{\alpha\beta}\left(r_{\alpha\beta} + \Delta r_{\alpha\beta} - r_{\alpha\beta}{}^{o}\right)\frac{\left(\mathbf{x}_{\beta} - \mathbf{x}_{\alpha}\right)}{\left\|\mathbf{x}_{\beta} - \mathbf{x}_{\alpha}\right\|} \tag{12}$$

The large deformations are observed here, which implies the direction of the interparticle forces along the resulting spring alignments $\mathbf{x}_{\beta} - \mathbf{x}_{\alpha}$. The molecular statics problem consists in finding the equilibrium configuration of particles satisfying

$$\sum_{\beta} \mathbf{F}_{\alpha\beta} = \mathbf{0} \tag{13}$$

for α=1,...,*N* (total number of particles). More mathematical details are presented in [5].

2.4. Coupling between macro-scale and nano-scale models

The macro-scale and nano-scale models are coupled by identifying the particles located on the interface of the nano-scale domain with the corresponding nodes of the FEM mesh, located on the interface of the macro-scale mesh. In such a case, we solve the molecular statics equations inside the nano-scale domain, and the linear elasticity with thermal expansion coefficient discretized by FEM inside the macro-scale domain, while the interface between macro- and nano-scales is treated in a special way. The nodes of the FEM mesh, from the point of view of the nano-scale model are identified with particles represented by their positions $\mathbf{x}_{\alpha}$, while from the point of view of the macro-scale model, the nodes are understood in the classical way like degrees of freedom, representing the displacements $\mathbf{u}_{\alpha}$. It implies the additional coupling equations

$$\mathbf{u}_{\alpha} = \Delta\mathbf{p}_{\alpha} = \mathbf{x}_{\alpha} - \mathbf{p}_{\alpha} \tag{14}$$

for each particle (FEM mesh node) α located on the interface between nano- and macro-scales.
In practice, it is not necessary to add the coupling equations (14) to the system, but only to aggregate the FEM and MS equations to the same global matrix.

3. Numerical results

3.1. Deformation of the feature inside the template

The first computational problem consists of computing the displacement field of the polymer inside the template, after the photopolymerization process. We consider the multi-scale problem, with molecular statics applied close to the boundary between the template and the polymer, and the continuous linear elasticity with thermal expansion coefficient applied in the internal part of the

domain. The multi-scale model has been implemented within the procedural version of 3D hp-adaptive FEM [6]. The results of the simulation are presented in Figure 3. The multi-scale structure of the domain, with the macro-scale model used in the interior and the nano-scale model used in the external part of the domain is presented on left panel in Figure 2. The deformation of the external nano-scale model is presented in the middle panel, while the deformation of the internal continuum model put inside the nano-scale model is presented in the right panel.

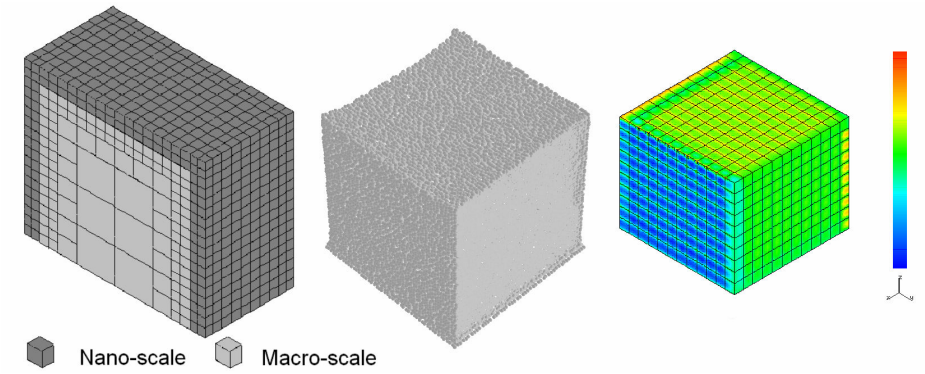

Figure 2. Multi-scale computations of the deformation of the polymer inside the template: **(left panel)** multi-scale domain; **(middle panel)** deformation of the nano-scale external domain; **(right panel)** deformation of the macro-scale internal domain (norm of the displacement vector field)

3.2. Deformation of the feature outside the template

The second numerical problem concerns the shrinkage of the feature after removal of the template. It is assumed that the polymer network has been damaged in one part of the feature – in the bottom part, in one half of the horizontal cross-section. Thus, the interparticle forces are weaker in that part of the feature. This problem has been solved first by using pure nano-scale approach, with non-linear model allowing for large deformations, with quadratic potentials (12). The resulting equilibrium configuration of polymer network particles is presented on left panel in Figure 3. The damage has been modeled here by assuming smaller values of the spring stiffness coefficients $k_{\alpha\beta}$. This implies the use of a non-linear model, to be solved by the Newton-Raphson method with GMRES solver.

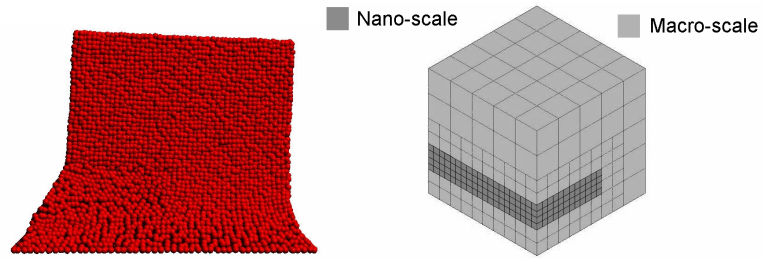

Figure 3. (Left panel) Results of the non-linear model allowing for large deformations, with quadratic potentials – side view; **(right panel)** 3D domain. The light grey colour denotes the macro-scale domain with FEM model; the dark grey colour denotes the nano-scale domain with MS model

The problem has been solved again by using the multi-scale approach. The part of the mesh with undamaged polymer has been modeled by the macro-scale approach, with adaptive Finite Element Method, as linear elasticity with thermal expansion coefficient. The part of the mesh with the damaged polymer, denoted on right panel in Figure 3 by grey color, has been modeled by the nano-scale approach with linear model assuming small deformations and quadratic potentials (11), with smaller values of the spring stiffness coefficient $k_{\alpha\beta}$. The usage of the linear model assuming small

deformations allows to utilize the direct solver from adaptive FEM package [6], and it seems to be acceptable in that part of the mesh. The results – the x, y and z components of the vector displacement field are presented in Figure 4. The damage of the polymer, modeled by weakening the inter-particle forces results in a slight lean of the feature, as illustrated in left panel of Figure 3 for the nano-scale model, and in Figure 4, for the macro-scale model. The displacement fields are similar in both nano-scale and macro-scale simulations.

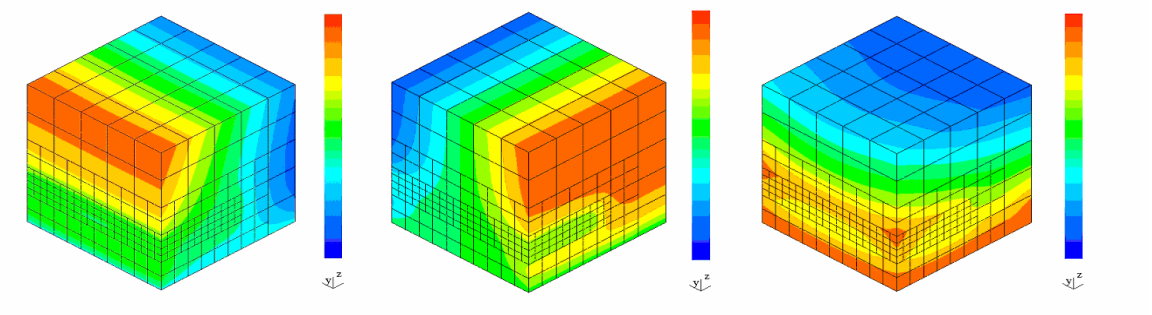

Figure 4. (Left panel) z-components of the displacement vector field; **(middle panel)** y-components of the displacement vector field; **(right panel)** z-components of the displacement vector field

4. Unified Modeling Language description of the multi-scale application

In this section we describe the Unified Modeling Language (UML) project of the object-oriented (O-O) multi-scale framework with hp-adaptive Finite Element Method (hp-FEM) coupled with Molecular Statics (MS) model. The presented UML model has been created by analyzing the procedural version of the hp-adaptive FEM algorithm from the point of view of the O-O paradigm. The resulting UML model is intended to be independent on the spatial dimension of the computational problem N.
The finite element being used is based on the Euler's hierarchical model, where higher dimension objects are defined as compositions of some number of lower dimension objects. The hierarchy of objects assumed on the element level is also utilized on the level of approximation spaces. This is done by using special hierarchical shape functions defined as tensor products of 1D shape functions. The proposed computational algorithms, e.g. making decisions about the optimal refinements, are also designed in the hierarchical manner. It is obtained by a recursive selection of optimal refinements for higher dimension objects with the use of previously selected optimal refinements for the lower dimension objects. The O-O pattern is utilized in the design process to obtain clear, open, reusable, multi-component structure of the adaptive framework. We have also investigated the coupling of the *hp*-FEM to support multi-scale problems, where some parts of the computational domain are modeled by the nano-scale MS model. The method of coupling macro-scale and nano-scale models has been implemented and tested in the procedural version of the 3D hp-adaptive FEM on the multi-scale problem concerning the Step-and-Flash Imprint Lithography (SFIL) simulations. The resulting architecture has been incorporated in the UML project, since the O-O paradigm allows for easy combination of macro-scale and nano-scale models. It should be emphasized that the UML project is defined on the abstract level and it is independent from the programming language used. Most software engineering tools allow for the generation of the structure of the classes from a UML diagram in O-O language selected by the user (e.g. C++/JAVA). The following subsections present basic concepts utilized in the UML project. The O-O paradigm enables to effectively use of the hierarchical structure on the level of data structures as well as algorithms.

4.1. O-O hierarchical data structures

The computational mesh is designed based on the Euler's hierarchical model. Higher dimension objects are composed of several lower dimensional objects, as it is presented in Figure 6. An *edge* consists of two *vertices*, a *face* consists of four edges, and *interior* consists of six faces. Technically

speaking, *Vertex2D* is not the same as *Vertex1D*, since e.g. it contains more coordinates. However, a general spatial dimension independent *Vertex* template can be created, and dimension dependent objects can be created by parameterizing the general template. The *Element* object consists of several *Node* objects, stored at *NodesList* object as it is presented in Figure 6. The *Element1D* consists of two vertices and one edge, the *Element2D* consists of four vertices, four edges and one face, the *Element3D* consists of eight vertices, twelve edges, six faces and one interior. There are also *ParticleElements* classes representing nano-scale elements filled with particles. All vertices, edges and interiors inherit from an abstract *Node* class.

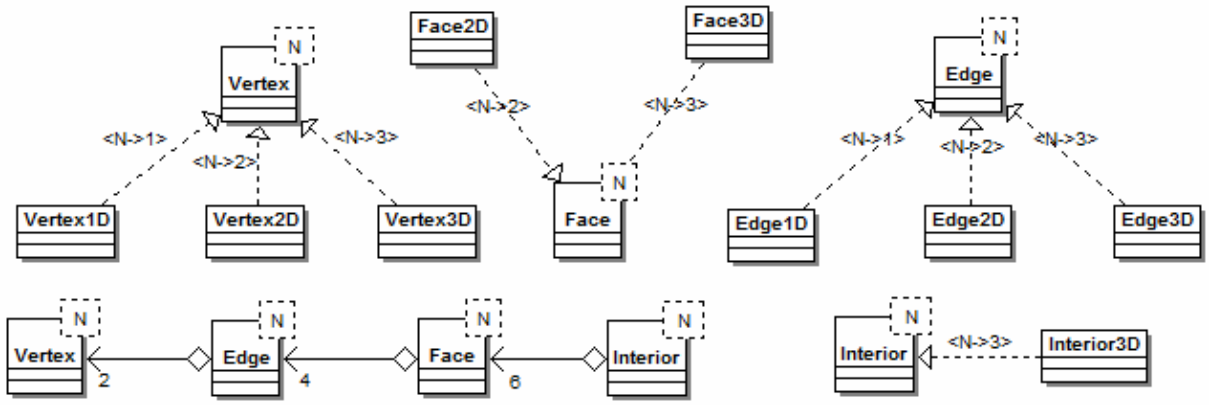

Figure 5. Class diagram illustrating relations between nodes: vertices, edges, faces and interiors.

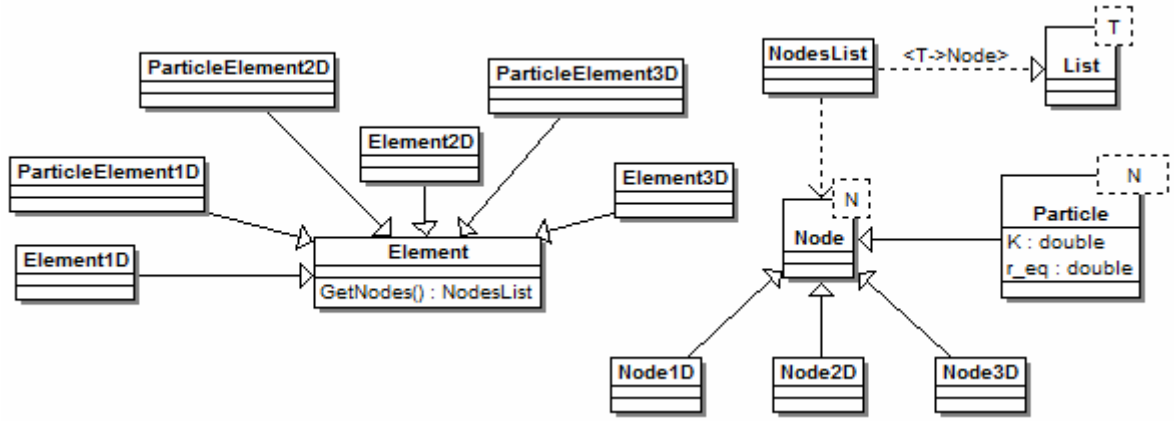

Figure 6. Class diagram illustrating general relation between *Element* and *Node* classes.

The solution of the variational problem is approximated using *global shape functions*. Global shape functions are associated with nodes and have supports over one or several neighboring elements. A restriction of a global shape function into one element is called a *local shape function*. For the purpose of the UML project we utilize special kind of hierarchical local shape functions [6], where higher spatial dimension functions are tensor products of several lower dimension functions. The basic 1D shape functions are defined by the following recursive formulae:

$$\kappa_1(\xi) = 1-\xi\,,\ \kappa_2(\xi) = \xi\,,\ \kappa_3(\xi) = (1-\xi)\xi\,,\ \kappa_n(\xi) = \kappa_{n-1}(\xi)(\kappa_2(\xi) - \kappa_1(\xi)) \qquad (15)$$

We distinguish κ_1 and κ_2 as *VertexLocalShapeFunction1D*, and κ_n n>=3 as E*dgeLocalShapeFunction1D*. The *VertexLocalShapeFunction2D* are defined as tensor products of two *VertexLocalShapeFunction1D*

$$\phi_1(\xi_1,\xi_2) = \kappa_1(\xi_1)\kappa_1(\xi_2) = (1-\xi_1)(1-\xi_2)\,,\ \phi_2(\xi_1,\xi_2) = \kappa_2(\xi_1)\kappa_1(\xi_2) = \xi_1(1-\xi_2) \qquad (16)$$

$$\phi_3(\xi_1,\xi_2) = \kappa_2(\xi_1)\kappa_2(\xi_2) = \xi_1\xi_2\,,\ \phi_4(\xi_1,\xi_2) = \kappa_1(\xi_1)\kappa_2(\xi_2) = (1-\xi_1)\xi_2$$

The *EdgeLocalShapeFunction2D* are defined as tensor products of two *VertexLocalShapeFunction1D*.

$$\phi_{5,j}(\xi_1,\xi_2)=\kappa_{2+j}(\xi_1)\kappa_1(\xi_2)\ j=1,...p_1\text{-}1;\ \phi_{6,j}(\xi_1,\xi_2)=\kappa_2(\xi_1)\kappa_{2+j}(\xi_2)\ j=1,...p_2\text{-}1; \quad (17)$$

$$\phi_{7,j}(\xi_1,\xi_2)=\kappa_{2+j}(1-\xi_1)\kappa_1(\xi_2)\ j=1,...p_3\text{-}1;\ \phi_{8,j}(\xi_1,\xi_2)=\kappa_1(\xi_1)\kappa_{2+j}(1-\xi_2)\ j=1,...p_4\text{-}1$$

where p_k is the polynomial order of approximation over an edge. The *FaceLocalShapeFunction2D* are defined as tensor products of two *EdgeLocalShapeFunction1D*

$$\phi_{9,ij}(\xi_1,\xi_2)=\kappa_{2+i}(\xi_1)\kappa_{2+j}(\xi_2)\ i=1,...,p_h\text{-}1,\ j=1,...,p_v\text{-}1 \quad (18)$$

The relations are expressed in the class diagram presented in Figure 7. The local shape functions are again created by instantiating template *Local Shape Function*, as described in Figure 9.

4.2. Hierarchical algorithms

In the proposed UML project, the basic objects are created by instantiating general templates. Thus, their methods must be defined within these templates as spatial dimension-independent. The exemplary dimension-independent algorithm for computing a value of the shape function is listed below. *N* stands for the spatial dimension, and *Value* stands for either a double precision or a complex arithmetic type.

```
template <int N, class Value>
class LocalShapeFunction{
LocalShapeFunction1D mLocalShapeFunction1D[N];
Value GetValue(Point<N> P){
   value=1.0;
   for(int i=1; i<=N; ++i)
       value*=mLocalShapeFunction1D[i].GetValue(P[i]);
```

We have analyzed algorithms utilized in [6, 7] in the existing procedural version of 1D, 2D, and 3D *hp*-FEM. Most algorithms are actually designed in a hierarchical manner, since they call procedures implemented for lower spatial dimension objects. Let us discuss the issue on the *h* adaptation algorithm example. The process of refining e.g. a 2D face consists of two processes of refining 1D edges and then linking them to make a new faces. The process of refining an edge is consequently described as the process of making new vertices and linking them into new edges. Newly created nodes are linked on father / sons lists from existing nodes, as described in Figure 8. The active finite elements are dynamically reconstructed after the refinement process is finished. In addition to the classes already introduced, we have created the *Particle* class, which is a special kind of node storing the nano-scale particle utilized during modeling of the inter-particle interactions in the nano-scale. The particles are collected within element objects. Thus, there exist macro-scale elements collecting multiple *Nodes*, and nano-scale elements collecting multiple particles, as it has been illustrated in Figure 6.

4.3. Node-based solver

The next essential part of the multi-scale application is an efficient multi-frontal solver. We have developed a node-based solver suitable for the adaptive multi-physics computations, utilizing the hyper-matrix, working on the level of *Nodes*, independent on the spatial dimension and the polynomial order of approximation. The new solver has been described in [10] based on the previous version [11]. The remaining parts of the model are shown below.

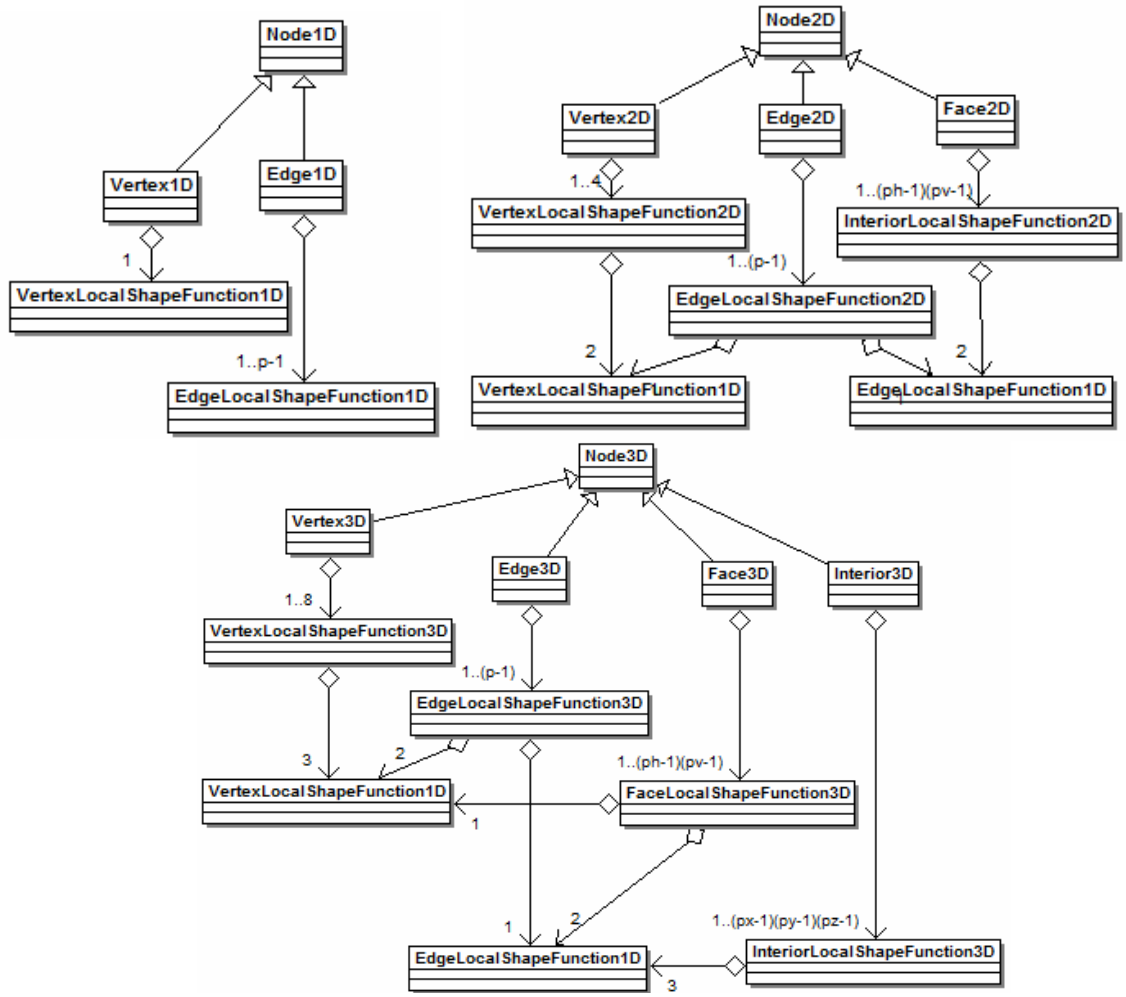

Figure 7 Class diagrams illustrating general relations between local shape function and node classes.

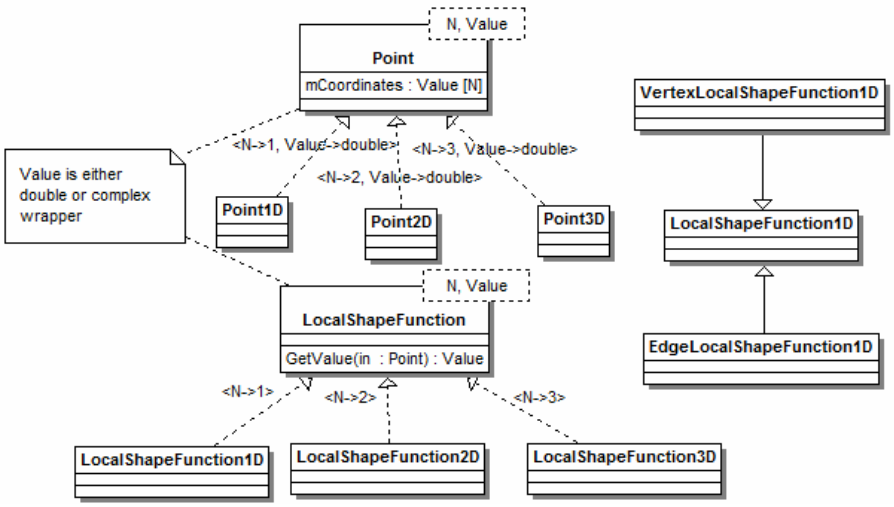

Figure 8. Class diagram illustrating relations between father and son nodes in the refinement tree

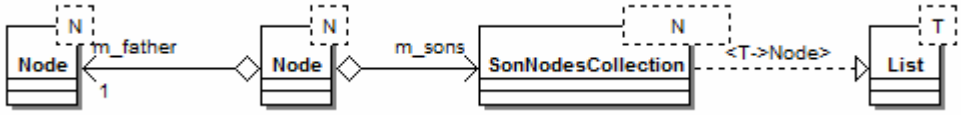

Figure 9. Class diagram illustrating relations between father and son nodes in the refinement tree

5. Conclusions

The paper presented the formulation and numerical simulations of the multi-scale SFIL problem within the 3D *hp*-FEM procedural application [6]. The paper also presents the O-O UML project developed based on the analysis of the resulting multi-scale procedural application (macro-scale *hp*-FEM coupled with nano-scale MS). The discovered hierarchical structure of classes and algorithms makes the UML project as independent on the spatial dimension of the problem as possible. The O-O UML project was defined on the abstract level, independently of the programming language being used. In order to include the nano-scale into the model, we replaced the macro-scale elements with nodes by the nano-scale elements filled with particles, and incorporated nodes and particles from the macro/nano-scales interface. The future work will include the development of a O-O template multi-scale dimension-independent code based on the presented UML model.

6. Acknowledgements

The work reported in this paper was supported by Polish MNiSW grant no. NN 501 120 836.

References

[1] Colburn M E, Suez I, Choi B J, Meissi M, Bailey T, Sreenivasan S V, Ekerdt J E, Willson C G 2001 Characterization and modeling of volumetric and mechanical properties for SFIL photopolymers *Journal of Vacuum Science and Technology* **B 19 6**.

[2] Bailey T C, Colburn M E, Choi B J, Grot A, Ekerdt J G, Sreenivasan S V, Willson C G 2001 Step and Flash Imprint Lithography: A Low-Pressure, Room Temperature Nanoimprint Patterning Process. Alternative Lithography. Unleashing the Potentials of Nanotechnology. C. Sotomayor Torres, Editor, *Elsevier* .

[3] Hughes T.J.R. 2000 The Finite Element method. Linear Statics and Dynamics Finite Element Method Analysis, *Dover.*

[4] Paszyński M., Barbasz B., Schaefer R. 2007 Efficient Adaptive Strategy for Solving Inverse Problems *Lecture Notes in Computer Science* **4487** 342-349.

[5] Paszyński M, Romkes A, Collister E, Meiring J, Demkowicz L, Willson C G 2005 On the Modeling of Step-and-Flash Imprint Lithography using Molecular Statics Models. *ICES Report* **05-38**, *The University of Texas in Austin.*

[6] Demkowicz L, Kurtz J, Pardo D, Paszynski M, Rachowicz W, Zdunek A 2007 Computing with hp-Adaptive Finite Elements Vol. II *Chapman & Hall.*

[7] Rumbaugh J, Jacobson I, Booch G 2004 The Unified Modeling Language Reference Manual, *Addison-Wesley Professional 2nd edition.*

[8] Philippe T., Devloo B. 1997 PZ: An object oriented environment for scientific programming *Computer Methods in Applied Mechanics and Engineering* **150** 1-4 133-153.

[9] Philippe R., Devloo B. 2008 OOPar: An Object Oriented Environment for Implementing Parallel Algorithms *11th IEEE Int. Conference on Computational Science and Engineering.*

[10] Paszyński M., Pardo D., Paszyńska A. 2010 Parallel Multi-Frontal Solver for Multi-Physics *p* Adaptive Problems, accepted to International Conference on Computational Science May 2010, Amsterdam, *Proceedia Computer Science.*

[11] Paszyński M., Pardo D., Torres-Verdín C., Demkowicz L., Calo V. 2010 A Parallel Direct Solver for the Self-Adaptive *hp* Finite Element Method, *Journal of Parallel and Distributed Computing* **70** 270-281.