JOM, Vol. 68, No. 8, 2016
DOI: 10.1007/s11837-016-1960-8

Development of Semantic Description for Multiscale Models of Thermo-Mechanical Treatment of Metal Alloys

PIOTR MACIOŁ[1,2] and KRZYSZTOF REGULSKI[1,3]

1.—AGH University of Science and Technology, Kraków, Poland. 2.—e-mail: pmaciol@agh.edu.pl. 3.—e-mail: regulski@agh.edu.pl

We present a process of semantic meta-model development for data management in an adaptable multiscale modeling framework. The main problems in ontology design are discussed, and a solution achieved as a result of the research is presented. The main concepts concerning the application and data management background for multiscale modeling were derived from the AM3 approach—object-oriented Agile multiscale modeling methodology. The ontological description of multiscale models enables validation of semantic correctness of data interchange between submodels. We also present a possibility of using the ontological model as a supervisor in conjunction with a multiscale model controller and a knowledge base system. Multiscale modeling formal ontology (MMFO), designed for describing multiscale models' data and structures, is presented. A need for applying meta-ontology in the MMFO development process is discussed. Examples of MMFO application in describing thermo-mechanical treatment of metal alloys are discussed. Present and future applications of MMFO are described.

INTRODUCTION

Modern material processing is aimed at improving the cost-effectiveness ratio in the metal industry. A controlled development of microstructures with multi-step processes can lead to achieving desired combinations of properties. Nevertheless, issues related to controlling microstructural processes and related mechanical properties (e.g., impact toughness, yield stress, hardness, etc.) are still challenging for manufacturers.[1] In advanced technologies, all consecutive steps of the manufacturing process are tightly connected, and an efficient way of achieving requested properties must consider all mutual dependencies. Complex processes cannot be developed without extensive use of numerical modeling. The multiscale modeling approach is the one which significantly increases predictive capabilities of numerical models. Multiscale modeling has become a popular tool in many areas of science, such as biology, physiology, material science, chemistry and applied mathematics.

One of the current issues of multiscale modeling is representation of material state. Various methodologies for the generation of microstructural description have been developed.[2] Some are based on digitalization of microstructural images. This approach requires a large number of images, covering the whole range of initial conditions for the numerical model. The stochastic character of microstructures, as well as uncertainty aspects, must be considered, leading to costly and long-lasting investigations before a reliable multiscale model of a process can be used. Alternatively, an initial microstructure description can be developed with numerical methods (e.g., with Cellular Automata).[3] The latter approach is more flexible; however, a reliable projection of material history into a computationally generated description is a demanding task.

The most effective way of improving quality and reliability of multiscale modeling is to keep a history of material treatment from the very beginning. In this approach, the microstructural description is not generated for the particular process but inherited from the previous stages. This approach is one of the main postulates of Integrated Computational Materials Engineering (ICME). The current topic in the ICME community is describing material data. In the context of multiscale modeling, this need was discussed by Yang and Marquardt.[4] More recently, the need of a standardized description of material

(Published online June 2, 2016)

properties was emphasized, and a set of guidelines for such a standard, as well as the main problems to be solved, was presented.[5] The possible application of a VTK-based format of material description, HDF5, was introduced by Schmitz.[6]

Application of multiscale modeling and ICME within the industry is also limited, due to computational demands. Application of detailed numerical models for microstructural modeling everywhere and every time in a macroscopic computational domain leads to a very reliable multiscale model; however, such application is useless due to the required computational resources. Currently, the balance between thoroughness and computational costs of multiscale models is usually controlled by the developers/researchers themselves. An alternative solution, based on Agile multiscale modeling methodology (AM3), was introduced by Maciol et al.[7,8] In this approach, a knowledge-based system (KBS) is applied for runtime control of a multiscale model configuration. Applying AM3 to ICME methodology will decrease necessary computational resources, keeping the quality of results at a sufficient level. However, it makes a transfer of material state description between parts of models even more complex. The specification of inputs and outputs of parts of the integrated model is no longer static but varies during computations. Therefore, the adaptive modeling environment puts high demands on the material description.

One of the postulates of the ICME community is that a material description is sufficient to describe material properties at coarser scales. These properties can be calculated "on-the-fly" with some models, numerical or analytical. This approach is very promising; however, some constraints cannot be neglected. First, in the foreseeable future, it will not be possible to describe engineering material starting from an "ab initio" scale due to both the required amount of data and the stochastic character of materials. In addition, numerical models are very sensitive to the "quality" of inputs. Each numerical model is grounded on some assumptions concerning input and output variables. These assumptions are not always clear. Moreover, violation of these assumptions usually leads to erroneous results. The loss of a Mars Climate Orbiter spacecraft, worth US$125 million, due to a trivial mistake in Imperial/SI unit conversions is one such example. The former constraint requires some generalizations of material data, while the latter tightly binds the reliability of computations with these generalizations. Furthermore, each numerical model requires its own form of generalization. Therefore, it is expected that the pure description of material state could be insufficient in an adaptive multiscale modeling framework. Schmitz et al.,[5] and Konter et al.[9] emphasized the need for additional metadata associated with material description. In this paper, we assume that the common description for both material state and numerical model should be used together.

The approaches of material representation presented above can be used for detailed descriptions of the material state; however, their semantic expressivity is limited. The ambiguity of such descriptions requires additional specifications, which cannot be used directly by computer algorithms. Since it is crucial that the structure of the model is variable, the adaptable structure of an AM3/ICME model provides a sufficient solution. Our goal is to provide a tool for joint semantic descriptions of material state and multiscale models for thermo-mechanical processing (TMP) of metal alloys. With this description, it would be possible to (1) provide a unified, formal and unambiguous description of materials and models that could be shared with other users (researchers), (2) define input and output variables for models, (3) solve problems associated with synonymy (multitude of names for the same designates) and polysemy (usage of identical terms for different values), and ultimately (4) provide a basis for future automatic classification of models and validation of compatibility of data and models.

Adaptive Multiscale Modeling

Numerical models of material processing can be considered from various perspectives. From the point of view of material science, a theoretical description of physical and chemical phenomena inside the material is used. Then, a mathematical description of the phenomena is utilized. Eventually, the model must be described with the use of numerical methods and finally implemented in a code using a programming language. Each of these points of view involves its own knowledge content. In a conventional, single-scale model, material science knowledge is not directly connected with its numerical description. As an example, during modeling of a material deformation process, the relationship between strains and stresses must be provided. Single-scale models use equations or tabularized data for this purpose. The latter are evaluated on the basis of experimental results, and lack any explicit "knowledge." Equations can be partially based on such knowledge, but some empirically fitted parameters are still required. Multiscale modeling introduces stronger linkages between material science and numerical methods. Instead of utilizing equation/data, a fine-scale submodel is involved. Since such a model is believed to reproduce particular phenomena inside a material, material science knowledge is directly involved, at least on a single spatial/temporal scale.

ONTOLOGIES

Ontologies in Computer Science

An *ontology* is a model representing a fragment of domain knowledge, comprising a set of concepts and a set of predicates: properties of concepts being unary predicates and a set of relationships between concepts being binary predicates. An ontology must describe reality on different levels of granularity, maintaining a description of relationships between layers in the hierarchy. Because the levels considered range from the microphysical up to the cosmological, it should be noted that an ontology created in this way is not universal in its range.[10] Ontology should provide knowledge interoperability and reusability. An ontology is not a database schema, but one could say that the ontology serves the same purpose in the case of knowledge repositories as entity diagrams in the case of databases; it is a schema, a model describing a certain field of knowledge, understandable by both computers and people. Creating a unified, formal language of related-concepts definitions and its consequent sharing with a number of users is the first and foremost purpose of ontologies.[11] Issues related to data integration and data interchange among heterogeneous programming artifacts are very common today, due to their importance in complex and distributed systems.[12] Ontologies are capable of solving problems related to the integration of knowledge. Moreover, they could help in classification when available knowledge is incomplete and/or inconsistent. Such tasks have been described in previous authors' publications.[13]

Ontologies in Material Modeling Science

Modeling of processes in material science entails the necessity of developing complex systems involving various submodels. If submodels of a model are focused on different length/time scales, the model is a multiscale one. There are two possible ways of coupling submodels. They can be incorporated into a consistent set of equations and solved concurrently or split into relatively independent submodels, communicating with others by passing some variables. A detailed discussion is available.[14] E introduced heterogeneous multiscale models (HMM), describing the second of the model families mentioned above. This paper is focused on such heterogeneous models. In such a case, a model can be represented by a set of relationships and variables, which describe dependencies between material properties and its processing. The need to use a unified language to describe individual components of the modeled system—ontologies—has been repeatedly mentioned in several publications, for example,[15] in the fields of chemistry,[4] biology[16,17] and geography.[18]

Currently, a variety of simulation models is known in the field of material science. These models are written using various languages and are designed for a wide range of computational environments. Unfortunately, these models are usually neither interoperable nor annotated in a sufficiently consistent manner to support intelligent searching or integration of available models. Usually, simulation models contain no explicit information on what they represent—they are only systems of mathematical equations encoded in a programming language. Knowledge about a system (process, phenomenon) is implicit in the code; it is an abstract representation of the system utilizing mathematical variables and equations which must be interpreted by a researcher.[19]

MULTISCALE MODELING FORMAL ONTOLOGY CREATION

As mentioned above, a relatively new trend in multiscale modeling concerns adaptiveness of the actual model structure to present conditions. It urges the usage of a large number of submodels, which constitute a base for momentary configurations of the model. A structure of the model is floating; however, each configuration must be "balanced"—each input variable to any active submodel must be provided by at least one other submodel as an output. The above makes model management complex. One adaptive framework was introduced by Macioł et al.[7,8]; however, the object-oriented description of the model within the framework lacks the desired expressivity. Hence, multiscale modeling formal ontology (MMFO) has been developed.[20]

The general concepts of MMFO are based on the core ontology for material processing science,[21] but the application background for multiscale modeling of TMP was created according to AM3. Aside from being a tool for AM3, MMFO should be also treated as a more generic tool—a platform for linking material science knowledge, numerical (multiscale) modeling knowledge, and implementation details.

Objectives

One of the core concepts of AM3 is a multiscale model run-time reorganization based on the material state, the required reliability of computations, and the available computing power. Hence, three aspects must be taken into consideration: (1) compatibility and completeness of the model structure, including material state variables; (2) reliability of submodels in dependence on the material state; and (3) the computational requirements.

Meta-ontology

Describing an adaptive multiscale model structure within the framework is a complex task. The structure can be described on two abstraction levels. The more general one includes abstract classes like *SubModel*, *Phenomena* or *Property*. The respective concretized entities are *CellularAutomataDynamicRecrystalization submodel, DynamicRecrystalization*

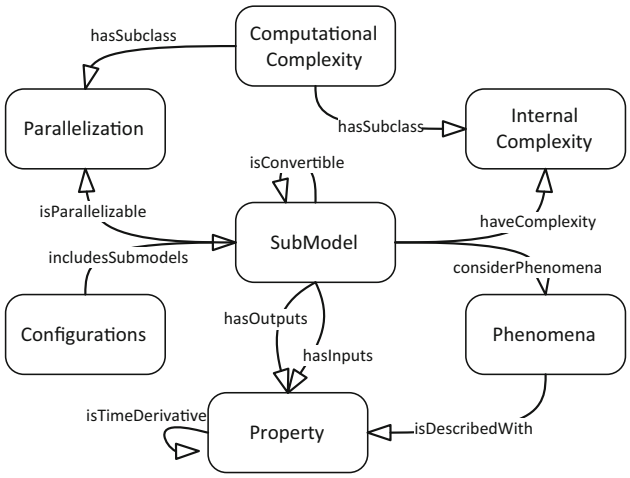

Fig. 1. The meta-ontology of MMFO, including relationships.

phenomenon, and *GrainSize* property. Similarly, two levels of ontology are considered. The more general one (meta-ontology) involves abstract classes and abstract object properties. Meta-ontology, as an abstract root of the case-specific detailed ontology, can be perceived as an intermediate stage of ontology creation. This approach is comes from philosophy.[22] In MMFO, the meta-ontology describes fundamental concepts and relationships between the abstract classes (Fig. 1).

The meta-ontology is very coarse and has no real descriptive value for AM3-based models. However, the meta-ontology is a guideline for development of concretized classes and relationships between them.

There is a practically unlimited number of configurations, phenomena, properties, and submodels. From the modeling framework perspective, entities (objects) can be incorporated into a single class if all their relationships are equal. Each of the subclasses, as well as the object "subproperties," are derived from the meta-ontology entities. Subclasses derive from abstract classes of the meta-ontology, and subproperties derive from object properties linking abstract classes of the meta-ontology. These subclasses and subproperties are components of the ontology. All hierarchies of classes are briefly described below.

Hierarchies of Classes

Property is a root for measurement of some physical or chemical states, for example temperature, pressure, stress, etc. The properties are quantitated with the data property *Unit*. A given property can be represented using various units. In this case, a generalized *Property* has subclasses for each unit (e.g., Fahrenheit, Kelvin, Celsius degrees). The subclasses are *disjoint-with* the other subclasses. Since conversions between all temperature representations are direct, a family of object subproperties is defined, where all members are inheriting from the Convertible object property (Fig. 2).

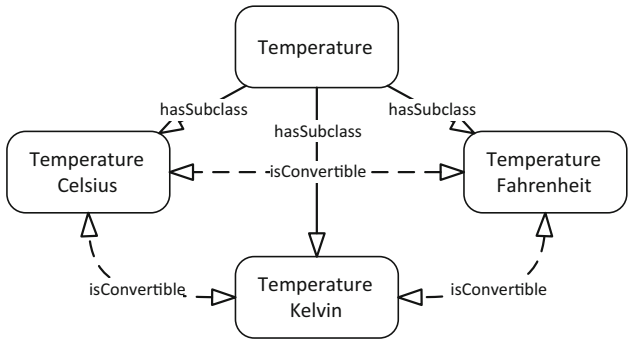

Fig. 2. The hierarchy of *Temperature* property's subclasses.

Each subclass of the *Property** hierarchy has a *Representation* data property, being a scalar, a vector, or a tensor. For example, the class *Temperature* is a scalar (as well as the deriving subclasses) and the class *Velocity* is a vector.

The subclasses of *Phenomena* represent physical or chemical processes. This is the closest hierarchy to material science—*Phenomena's* subclasses are linked to concepts from the material science domain. Some *Phenomena* subclasses are branches; for example, *DynamicRecrystallization* is a subclass of *Recrystallization*. Each *Phenomena* class must be *isDescribedWith* at least one *Property*. In other words, the subclasses of *Property* (e.g., *HeatGeneration*) are measures of *Phenomena's subclasses* (e.g., *PlasticWork*). *isDescribedWith* is a many-to-many relationship. Although the instances of these concepts represent the quantity value units, the using of an ontology or approach that more closely conforms to ISO 80000 such as QUDT could therefore be justified; these same concepts, however, are derived from phenomena that are not easily described with the use of existing models.

The *SubModel* hierarchy classes represent particular implementations of numerical models, capable of modeling particular *Phenomenas*. Subclasses of *SubModel* are related to the areas of numerical methods and software development. The *SubModel* subclasses can be simple equations, tabularized data, single- or multiscale numerical models, external numerical software, etc.

A single *SubModel subclass* may serve more than one *Phenomena* subclass. For example, a complex model of recrystallization can be used for computing both static and dynamic recrystallization processes. It is defined with relationships constituting subproperties of *considerPhenomena* object property (e.g., *PrecipitationsEquation considerPhenomena Time*). Each *SubModel* is characterized by its computational complexity.

*"Subclass of Property" is used interchangeably with "one of Properties"; the same applies to the other classes of the meta-ontology.

SubModels have their input and output sets of properties defined (*hasInputs* and *hasOutputs* object properties). The input sets can be eventually empty (e.g., constant value of Young coefficient in *ConstantYoungModulus* submodel). The output set must contain at least one *Property* subclass. From the perspective of an overall model correctness, balancing of required inputs and provided outputs is essential. Since a given *SubModel* can become "active" at any time, availability of input variables cannot be checked during the design stage. The validation must be carried out at the same time as changes are made to the *SubModels* configuration.

Optimization of computational resources usage is one of the main tasks of AM3. Hence, description of computational complexities of submodels is included in MMFO; however, it is not considered in this paper, due to a lack of direct connection to material data.

The ultimate goal of MMFO is to describe available configurations of submodels. Such a configuration includes a set of *SubModels*, interconnected with subproperties deriving from *configurationIncludesSubModels*. Since most of the configurations will be very similar, varying by only a few *SubModels*, subclasses of *Configuration* are organized in hierarchies. An example is shown in Fig. 3. During computations, an AM3 multiscale model must be consistent with one of the pre-defined *Configurations*. Any change in model conditions (material state, available computational resources) may move the model from the current to any other *Configuration*. Since particular numerical models are individuals, more than one real numerical model can be used in the same *Configuration*, hence not every change in the numerical models involved affects the present *Configurations* of the multiscale model. The allowed *Configurations* of the model are the leaves of the hierarchy, while *abstract* classes are the root and branches.

The classes and relationships introduced above may be extended. There are many relationships in material science and numerical methods. If more of them are to be considered, the description of *SubModels* configurations would become more sophisticated. One of the simplest exemplary possibilities is the introduction of the *isTimeDerivative* object property. For example, the *Velocity* property *isDerivative* of *Coordinates* with respect to *Time*. By introduction of this knowledge, the system would be able to provide *Velocity* automatically, not only if it is available directly but also when *Coordinates* and *Time* are available for the current and previous steps.

Examples

Two real multiscale models were described with MMFO: the model of aluminum alloy thermo-mechanical treatment, based on the Sherstnyev three-internal-variable model (3IV)[23,24] and the model of aluminum alloy deformation with precipitation kinetics considered.[7]

The IV3 model allowable configurations are the leaves of the *Dislocations3IVModel* branch (Fig. 4a). The common part of all configurations is defined using *Dislocations3IVModelIncludes* and contains *Deform2DTMC*, *ShearModulusConst,* and *StressDislocationsSubgrainEquation* submodels. Depending on the existence of deformation (the existence is defined as a threshold value of *EffectiveStrainRate* property), two alternative *Configuration* subclasses (dynamic and static) are available. Both inherit also the submodels included with *dislocations3IVModelIncludes* object property.

There are two possible *Configurations* of precipitation kinetic models (Fig. 4b). The difference lies in solving the precipitation kinetic problem with the external MatCalc software or the simplified, equation-based submodel. In this case the common part includes *AthermalStressWithDislocations*, *Deform2DTMC*, *FlowStressThermalAthermal*, *ShearModulusInTemperature*, *StressStrainRateCurve*, *ThermalStressEquation*, *DislocationsRandomEquation*, and *DislocationsWallEquation*. The alternative introduces one of the precipitation kinetic submodels: (1) *PrecipitationModelMatCalc*, including *PrecipitationsMatCalc* and (2) *PrecipitationModelEquation*, including *PrecipitationsEquation*.

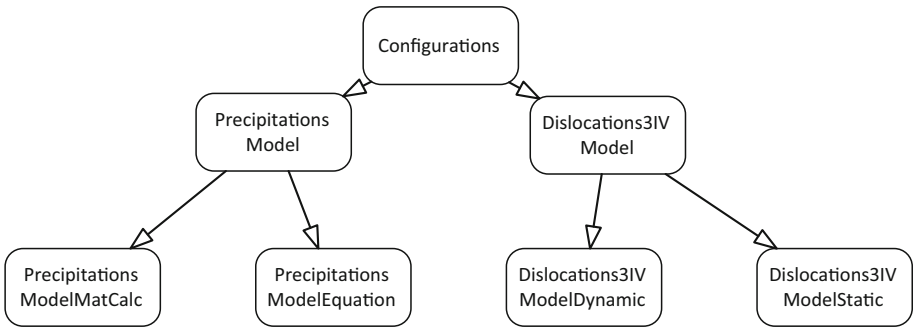

Fig. 3. An exemplary hierarchy of *Configuration* subclasses.

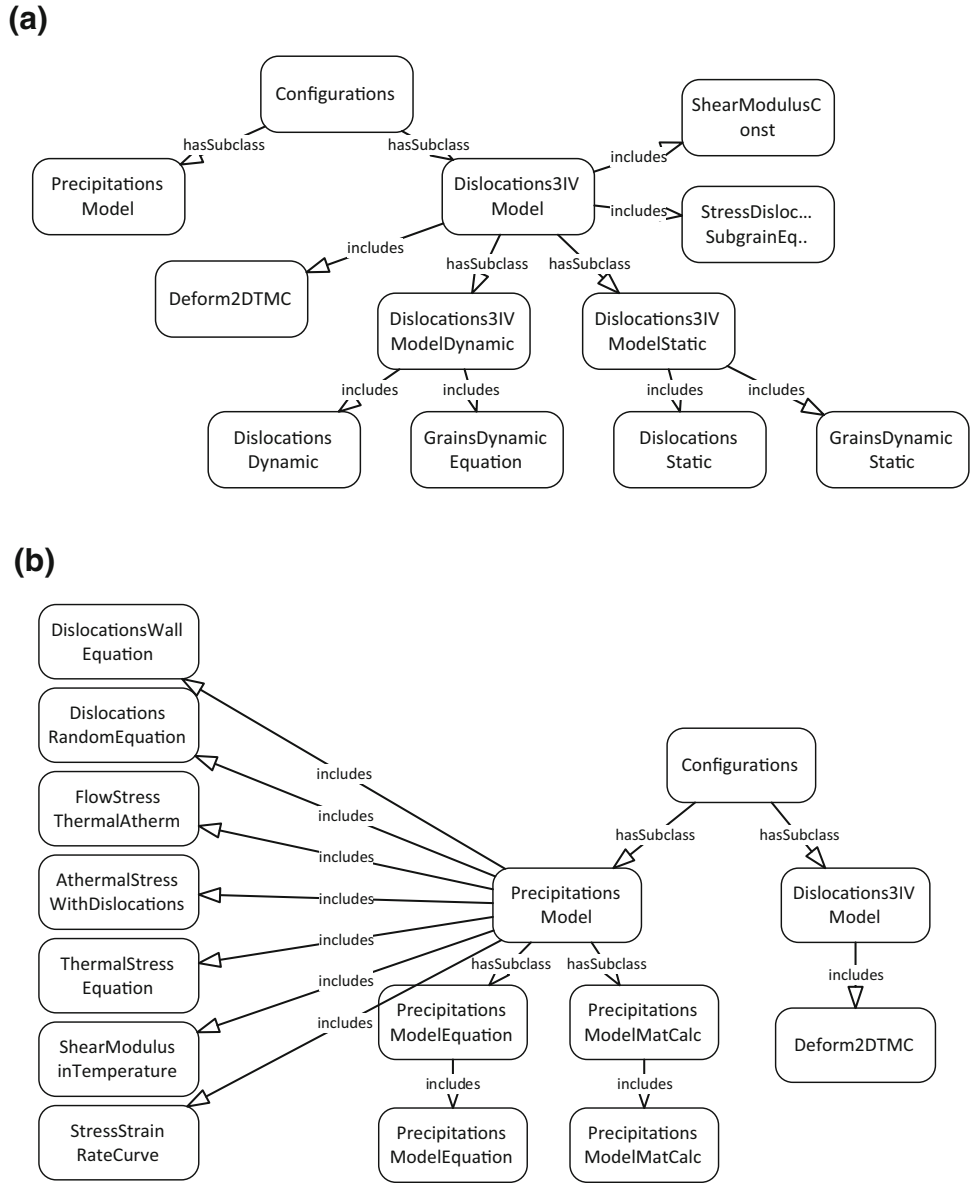

Fig. 4. *Configurations* of exemplary model: the three-internal-variable (a) and precipitation kinetic model (b).

As mentioned before, some *SubModel* subclasses may be represented by more than one individual. For example, there are two *PrecipitationsMatCalc* individuals: *AA6082AsCast* and *AA6082Prepared*, as well as three *Deform2DTMC* individuals: *AA6082FullProcessingModel*, *AA6082coldRolling*, *AA6082coldSpike*. Since switching between alternative individuals does not affect the *Configuration*, it is valid for each of six possible processes (three *Deform2DTMC* processes multiplied by two *PrecipitationsMatCalc* initial states).

PRESENT AND FUTURE APPLICATIONS

The main reason why MMFO has been developed is to support researchers in the design of multiscale models, mainly the ones based on the AM3 framework. A flexible, run-time-adjustable structure of a model makes data management difficult. From this point of view, the main objective of MMFO is to support the compatibility assurance of all simultaneously active submodels. The compatibility of submodels is defined as a 'balance' between requested outputs and provided inputs. A configuration is deemed valid if each input that may be requested by any submodel is provided as an output by at least one other submodel. There is an additional requirement that there are no closed loops (i.e. when a submodel A requests data from a submodel B, the latter cannot request the data back from the former) leading to deadlocks. Presently, that problem is solved on a programming level within the AM3 framework. This solution works reliably, but the validation is conducted during the

compilation phase and cannot be configured during the design stage. Moreover, it is extremely non-user-friendly. MMFO is expected to solve this problem. Definition of submodels using ontology-dedicated tools is much easier than plain programming. Unfortunately, the ontology tools are not able to validate the model structure; they can only describe it. Hence, the next research objective is to design tools capable of automatic analysis of the model structure and converting the ontology based model into a C ++ source code. Further works will also involve the development of a tool for automatic validation of configurations' completeness.

CONCLUSION

Application of MMFO makes possible the semantic description enhancement of the material description and multiscale model. It allows for validating the model in terms of completeness of input variables passed between submodels. It also introduces a convenient tool for communication between material scientists, numerical modelers, and programmers. It is also useful in the management of large knowledge bases, which are required to describe transitions between various configurations of AM3-based multiscale models.

Another issue addressed during MMFO development for the AM3 framework is the need for continuous enhancement of the ontology. Since the set of modeled phenomena and submodels is not restricted, MMFO has been split into two levels: *meta-ontology* as a guideline for continuous extension of the *ontology* itself.

The future works, aimed at the development of additional tools for linking MMFO with a programming language (automatic code generation for the backbone of multiscale model), automation of input/output sets balancing and supporting rules creation for the knowledge-based system will significantly increase the present capabilities of MMFO.

ACKNOWLEDGMENT

Financial support of National Science Centre, Grant No. 2011/01/D/ST8/04984, is gratefully acknowledged.

OPEN ACCESS

REFERENCES

1. M. Pietrzyk, L. Madej, L. Rauch, and R. Golab, *Arch. Civ. Mech. Eng.* 10, 57 (2010).
2. L. Madej, J. Wang, K. Perzynski, and P.D. Hodgson, *Comput. Mater. Sci.* 95, 651 (2014).
3. P. Macioł, J. Gawąd, R. Kuziak, and M. Pietrzyk, *Int. J. Multiscale Comput. Eng.* 8, 267 (2010).
4. A. Yang and W. Marquardt, *Comput. Chem. Eng.* 33, 822 (2009).
5. G.J. Schmitz, A. Engstrom, R. Bernhardt, U. Prahl, L. Adam, J. Seyfarth, M. Apel, C.A. de Saracibar, P. Korzhavyi, J. Ågren, and B. Patzak, *JOM* 68, 70 (2015).
6. G.J. Schmitz, *JOM* 68, 77 (2015).
7. P. Macioł, R. Buerau, C. Poletti, C. Sommitsch, P. Warczok and E. Kozeschnik, *Esaform 2015* (2015).
8. P. Macioł, A. Krumphals, S. Jędrusik, A. Macioł, and C. Sommitsch, *V International Conference* on Computational *Methods* for *Coupled Problems* in *Science* and *Engineering COUPLED Problems 2013*, ed. S. Idlesohn, E. Papadrakakis and B. Schrefler (Ibiza, 2013), pp. 1237–1248.
9. A. W. A. Konter, H. Farivar, J. Post, and U. Prahl, *Jom* 68, 59 (2015).
10. S. Kluska-Nawarecka, K. Regulski, M. Krzyżak, G. Leśniak, and M. Gurda, *Arch. Metall. Mater.* 58, 927 (2013).
11. S. Kluska-Nawarecka, D. Wilk-Kolodziejczyk, J. Dajda, M. Macura, and K. Regulski, *Arch. Metall. Mater.* 59, 927 (2014).
12. S. Kluska-Nawarecka, D. Wilk-Kołodziejczyk and K. Regulski, *Semantic Methods* for *Knowledge Management* and *Communication SE—2*, ed. R. Katarzyniak, T.-F. Chiu, C.-F. Hong and N. Nguyen (Berlin, Heidelberg: Springer, 2011), pp. 13.
13. S. Kluska-Nawarecka, D. Wilk-Kołodziejczyk, K. Regulski and G. Dobrowolski, *Intelligent Information* and *Database Systems SE—6*, ed. N. Nguyen, C-G. Kim and A. Janiak (Berlin, Heidelberg: Springer, 2011), pp. 52.
14. W. E, *Principles of Multiscale Modeling* (Cambridge: Cambridge University Press, 2011).
15. W. Sun, Q. Ma, and S. Chen, *J. Mech. Sci. Technol.* 23, 3209 (2010).
16. D. Beard, M. Neal, N. Tabesh-Saleki, C. Thompson, J. Bassingtwaighte, M. Shimoyama, and B. Carlson, *Ann. Biomed. Eng.* 40, 2365 (2012).
17. J. Cooper and J. Osborne, *Procedia Comput. Sci.* 18, 712 (2013).
18. A. Sorokine, T. Bittner, and C. Renschler, *Geoinformatica* 10, 313 (2006).
19. J.H. Gennari, M.L. Neal, B.E. Carlson, and D.L. Cook, in *13th Pacific Symposium on Biocomputing 2008 (PSB 2008), 4–8 January* (Kohala Coast, HI, 2008), pp. 414–425.
20. P. Macioł, http://home.agh.edu.pl/~pmaciol/wordpress/wp-content/uploads/mmfo.zip. Accessed 10 Feb 2016.
21. K. Regulski, G. Rojek, and J. Kusiak, *Key Engineering Materials*, Vols. 622–623 (Switzerland: Trans Tech Publications, 2014), pp. 978.
22. A. Thomasson, *The Continuum Companion to Metaphysics*, ed. R. Barnard and N. Manson (New York: Continuum International, 2012).
23. P. Macioł, R. Bureau, and C. Sommitsch, *Key Eng. Mater.* 611–612, 1356 (2014).
24. P. Sherstnev, C. Melzer, and C. Sommitsch, *Int. J. Mech. Sci.* 54, 12 (2012).