Towards Bridging the Data Exchange Gap Between Atomistic Simulation and Larger Scale Models

David Reith, Mikael Christensen, Walter Wolf, Erich Wimmer and Georg J. Schmitz

Abstract Materials properties are rooted in the atomic scale. Thus, an atomistic understanding of the physics and chemistry is the foundation of computational materials engineering. The MedeA computational environment provides a highly efficient platform for atomistic simulations to predict materials properties from the fundamental interactions effective at the nanoscale. Nevertheless, many interactions and processes occur at much larger time and length scales, that need to be described with microscale and macroscale models, as exemplified by the multiphase field tool MICRESS. The predictive power of these larger scale models can be greatly increased by augmenting them with atomistic simulation data. The notion of per phase-properties including their anisotropies provides e.g., the key for the determination of effective properties of multiphase materials. The key goal of the present work is to generate a common interface between atomistic and larger scale models using a data centric approach, in which the "interface" is provided by means of a standardized data structure based on the hierarchical data format HDF5. The example HDF5 file created by Schmitz et al., Sci. Technol. Adv. Mater. 17 (2016) 411, describing a three phase Al–Cu microstructure, is taken and extended to include atomistic simulation data of the Al–Cu phases, e.g., heats of formation, elastic properties, interfacial energies etc. This is pursued with special attention on using metadata to increase transparency and reproducibility of the data provided by the atomistic simulation tool MedeA.

Keywords Integrated computational materials engineering (ICME) · Materials modeling · Software · Interoperability · Industrial deployment · HDF5 · Multiscale modelling · Metadata · Hierarchy

D. Reith (✉) · M. Christensen · W. Wolf · E. Wimmer
Materials Design SARL, 42 avenue Verdier, 92120 Montrouge, France
e-mail: dreith@materialsdesign.com

G.J. Schmitz
MICRESS Group at Access e.V., Intzestr. 5, 52072 Aachen, Germany

P. Mason et al. (eds.), *Proceedings of the 4th World Congress on Integrated Computational Materials Engineering (ICME 2017)*,
The Minerals, Metals & Materials Series, DOI 10.1007/978-3-319-57864-4_5

Introduction

Integrated Computational Materials Engineering (ICME) offers a unification of various computational approaches and simulation tools addressing multiple phenomena at multiple time and length scales. Such multiscale modeling is vital to accurately describe the processes, structures, properties, and performance of materials [1]. Essentially, all materials properties are rooted in the atomic scale, which defines the shortest length and time scale within the ICME framework. As such, having the ability to compute properties at these scales offers a basis for a fully integrated computational framework able to study properties of materials prior to their actual synthesis. This extends far beyond the currently established practice of many continuum methods, which rely strongly on experimental data.

All atomistic simulation methods use "discrete" models that explicitly relate materials properties to interactions between atoms. At the lowest level, interatomic interactions are defined by the electrons. Describing these requires a quantum mechanical approach based on solving the Schrödinger equation for a many-electron system. In the past decades, density functional theory (DFT) and related methods have established themselves as quantum mechanical modelling workhorses offering a good balance between computational cost and accuracy. Above this quantum mechanical level one can find forcefield methods based on a classical description of the atomic interactions. At this level electrons are not explicitly described and Coulomb interactions are simplified on an abstract atomistic level where charges are attributed to atoms. Besides the atomic charge the interactions between atoms are described by convenient functional forms such as Lennard-Jones potentials, Morse potentials or the forms used by the embedded atom method. Class 2 forcefields include complex coupling terms involving 2-, 3-, and 4-body interactions. Such forcefield approaches are computationally much less demanding than the quantum mechanical approaches, thus enabling atomistic simulations to address and describe the evolution of thousands of atoms during time periods of nanoseconds or the exploration of millions of configurations in Monte Carlo simulations.

A simulation software such as MedeA [2] offers a universal and versatile common data model and a graphical user interface to overcome the inherent methodological differences between these various atomistic approaches, being embedded and integrated within a common, unified computational environment. Figure 1 summarizes the various levels of atomistic simulations. A more detailed insight on status and perspectives of atomistic simulations can be found in the paper of Christensen et al. [3].

The major challenge, which we tackle in the present work, is to bridge the differences between "discrete" atomistic simulation tools and "continuum" based tools. It is important to understand that the issue is not one of mere data exchange, but rather a subtle conceptual difference between the various computational approaches. On a continuum level, the central issue is to accurately describe and classify the state or evolution of a material. Once this has been accomplished within a well-defined classification system every ambiguity has been removed.

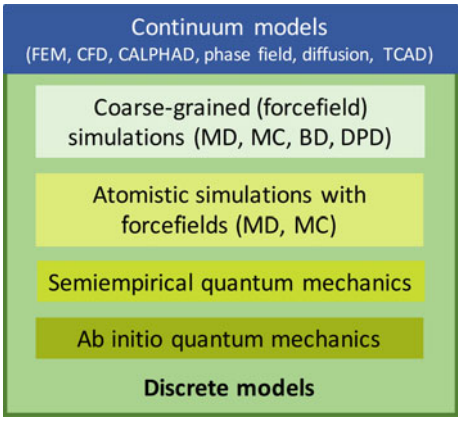

Fig. 1 An overview on various levels of simulations with a focus on discrete atomistic simulation. The abbreviations are: molecular dynamics (*MD*), Monte Carlo (*MC*), Brownian dynamics (*BD*), dissipative particle dynamics (*DPD*), finite element methods (*FEM*), computational fluid dynamics (*CFD*), calculation of phase diagrams (*CALPHAD*), and technology computer aided design (*TCAD*)

Furthermore, many different aspects of the material can be described within a single framework. However, discrete atomistic simulation operates on a calculation centric basis. Each calculation is setup to identify and describe a single property or a class of similar properties. The quality of such a description always depends on the methodological choice and settings with which the calculation has been performed. While the aim is to use sufficiently accurate settings the setup should not be too computationally expensive. Such settings are not only dependent on the applied approach, quantum mechanical or classical, but also materials and property specific. This necessitates multiple calculations using different approaches and settings to describe various properties of a single material.

The paper by Schmitz et al. [4] has created a solid basis on establishing a standard nomenclature and methodology to describe and exchange continuum data of MICRESS microstructures which will be used in the present work. Data exchange is facilitated by using the HDF5 file format [5]. This hierarchical data format is well suited to store and organize data utilizing two types of objects: datasets that can be scalars or multidimensional arrays of a defined type, e.g., float, integer or string, and groups that act as containers holding datasets or other groups. This makes it possible to order data in a hierarchical filesystem-like structure. In addition, each object can be described with attributes allowing for contextual metadata-based description of data. We believe that this flexibility as provided by the HDF5 file format is well suited as a container for atomistic simulation data.

The present work will build upon the proposed notation as suggested in [4]. We expand the definition of descriptors as proposed in [4] by adding another one, which is properties. In this naming scheme, a descriptor is a dataset that describes general information, such as structural data, while a property is a dataset that describes results of calculations. The main difference between both is the amount of

metadata used to describe these variables (see further below). The name of a descriptor or property, which can be either an object or a group, starts with a capital letter. In addition, names may be composed of multiple specifiers, e.g., number of atoms is described by the dataset *NumberAtoms*. Another important rule used is that names followed by a number in brackets are vector components. So *Job(1)* is followed by *Job(2)*.

To control the memory footprint of a HDF5 file we suggest to always define the size of a string or character dataset or attribute to exactly match the required length.

Data Structure

As a first step towards bridging the data exchange gap it is necessary to identify a hierarchical data structure most suitable to describe results from atomistic simulations. This hierarchical structure is calculation centric, or following MedeA's naming paradigm, job-centric. This implies that the data of each job is organized within a group. While such a hierarchy appears to be natural for atomistic simulation tools it is not very accessible to others. Consequently, a strategy is required to make the data more generally accessible. We propose to regroup calculated properties from different jobs by the structures, i.e. the specific and characteristic arrangement of atoms. The job-centric data structure is maintained in parallel,

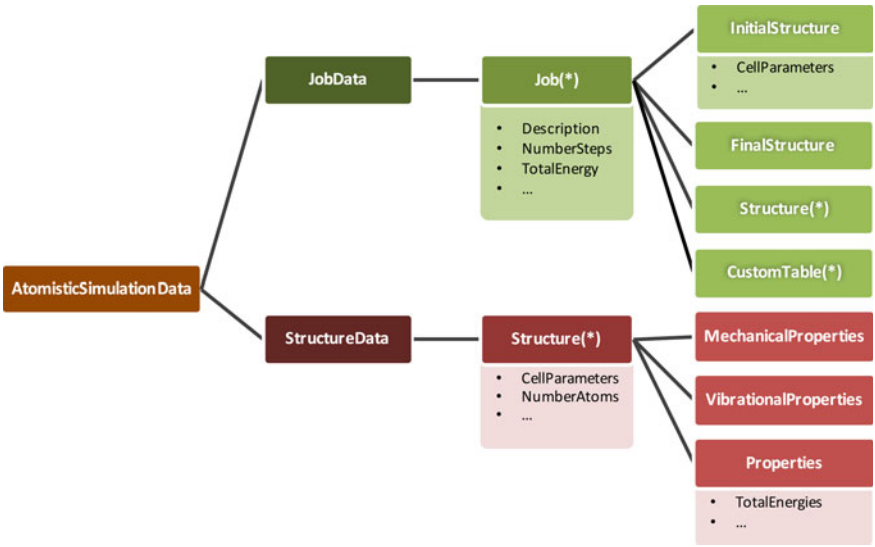

Fig. 2 A possible atomistic simulation data hierarchy of the HDF5 groups is depicted. The group *JobData* contains the primary calculation centric hierarchy. That data can then be mirrored, by using symbolic-links or by copying, into the structure-centric *StructureData* group to allow for an easier access

though. A suitable strategy to deal with this dual concept is to duplicate the job-centric data or to link it into the structure-centric form. Once this is accomplished atomistic simulation data can be easily incorporated into other hierarchies, e.g., the one described by Schmitz et al. [4].

We propose to collect all atomistic simulation data within a group with the name *AtomisticSimulationData*. This will make it easy to identify atomistic results when the HDF5 file is combined with other files that contain data from other sources (see Fig. 2).

The Job-Centric Data Structure

The calculation data are collected in the group *JobData* which is placed in *AtomisticSimulationData* (see Fig. 2). In *JobData* the data from a single calculation procedure, the job, is placed in a group with the name *Job(*)*, with * indicating an index that consecutively increases starting from 1 when additional jobs are added to *JobData*. This index defines the Job ID: *Job(1)* contains all the data from the first calculation procedure while *Job(2)* contains the data from the next one added to the file. The data in a *Job(*)* group can be sorted into general information, properties, structural information, and program specific data. The program specific data may also be a text-file created by the atomistic simulation tool. We will now discuss in detail each of these categories and the associated structure.

General Information

These variables store general information on the calculation setup of a given procedure.

1. *Description*
   The dataset *Description* is a scalar string variable that contains some general description on the calculation procedure. This variable should be user definable.
2. *NumberSteps*
   As a calculation procedure can sometimes contain multiple steps the total number of steps are indicated by this scalar integer.

Properties

Properties are the actual results of the atomistic simulations. The exact makeup of the properties depends on the calculation procedure and on the applied solver, i.e.

the atomistic simulation tool. Focusing on DFT and classical forcefield-based dynamics calculations in MedeA the discussion is narrowed down to calculated properties from the DFT code Vienna Ab initio Simulation Package (VASP) [6, 7] and the molecular dynamics simulation code LAMMPS [8]. However, the discussed definitions can be extended and adapted as required to show results from other solvers as well.

As a procedure can contain multiple steps the properties associated with a given step are indicated by an integer value in brackets at the end of the property name. For example, the dataset *TotalEnergy(1)* contains the total energy as evaluated in the first step while *Pressure(2)* contains the pressure as calculated in the second step. Strictly speaking, this bracketed value is not a vector index as the value in the bracket does not necessarily increase incrementally for any given property. If the pressure is only evaluated in the second calculation step and not in the first one the dataset *Pressure(1)* will not be available.

Since a detailed discussion of all the possible properties would go beyond the scope of the present work an overview on some of the possible properties is given in Table 1.

Structural Information

An accurate description of an atomic structure requires the use of a group of descriptors best organized within a HDF 5 data group in Job(*). As the structure can change during a calculation procedure an initial and, as required, a final structure needs to be described. In addition, the possibility of intermediate structures associated with specific calculation steps should be considered. Therefore, the group *InitialStructure* will contain all descriptors of the initial structure, likewise the group *FinalStructure* those of the final one. Any intermediate structure is defined by descriptors located in *Structure(*)*.

These groups contain a string dataset with the name *Structure.cif* that contains the full structural information in the CIF file format [9]. This allows the user to view the structure with an external program. As with properties a detailed discussion of all descriptors would exceed the scope of this paper. An overview of possible descriptors is given in Table 2. The aim is to use the space group (defined by *SpaceGroupName* and *SpaceGroupID*), the cell parameters (*CellParameters*), the volume (*Volume*), and the Wyckoff positions (*WyckoffPositions*, *WyckoffPositionIDs*, etc.) as structure definition. Note, that some descriptors are only used when required, e.g., ForceFieldAtomType and WyckoffPositionSpins.

Table 1 A list of possible properties sorted by their dimension, scalar or multidimensional array, and the source, general, LAMMPS, VASP or other, is given

	General	LAMMPS	VASP	Other
Scalar	Volume, Density, Pressure, Temperature	TotalEnergy, CoulombEnergy, PotentialEnergy, KineticEnergy, VanDerWaalsEnergy	TotalEnergy	DebyeTemperature, LongitudinalModulus
Multidimensional array	Stress		DOS, BandStructure, FermiSurface,	ElasticConstants, ElasticConstantMatrix, BulkModulus, YoungsModulus, ShearModulus, SoundVelocity, PhononDOS, PhononDispersion

Table 2 Descriptors used to describe structural information are sorted by their dimension, scalar or multidimensional array, and their kind, which can be string, integer or float

	Scalar	Multidimensional array
String	SpaceGroupName, StructureName, EmpiricalFormula, Structure.cif	WyckoffPositionIDs, ChemicalElementNames, ForceFieldAtomType
Integer	SpaceGroupID, NumberChemicalElements, NumberAtoms	
Float	Volume	CellParameters, WyckoffPositions, WyckoffPositionMasses, WyckoffPositionSpins, ForceFieldCharge

Custom Tables

During a calculation procedure custom tables summarizing results may be created. However, HDF5 datasets are homogenous, meaning that they can only contain variables of a single kind, e.g., string, float, etc. Such a homogenous definition does not translate well to a table where different columns might contain data of different kind. To allow for a more flexible table definition we propose to use the group *CustomTable(*)* with the index indicating the calculation step associated with the table contained in a *Job(*)* group. This group then contains the string dataset *Title*, and for each column the datasets *Column(*)* and *ColumnHeader(*)*. With the wildcard * indicating an integer index variable going from 1 to the maximum number of columns defined. *ColumnHeader(*)* will always be string while the kind of the *Column(*)* can vary.

Program Specific Files

A *Job(*)* group may include additional string datasets with the contents of special files used by the atomistic simulation tool. These provide the user with additional information and/or functionality. When using MedeA the *db.backup* string dataset may be included, providing the user with an easy means of reproducing a given calculation protocol on his machine. This *db.backup* dataset just needs to be exported as a file and imported into a MedeA JobServer.

In addition, the dataset *Job.out* provides a text-based summary on the performed calculation and the dataset *Structure.sci*, located in a structure group, contains the full structural information as a MedeA structure file.

Metadata

To track the actual computational setup used to calculate the properties saved in the job centric structure, as described in the previous chapter, we propose to extensively make use of the HDF5 metadata capability. As the quality of calculated properties strongly depends on the actual method and computational setup used, it is important to directly associate each property with the setup. Such a use of metadata increases the transparency as it allows one to reproduce the described property, even if this property has been copied or linked into a different group or file. As with descriptors and properties the list of metadata attributes described in the present work does not necessarily have to be complete and can be extended as required.

General Attributes

Both descriptors and properties use general attributes to track some general job information on the location, job ID and program used.

1. *Program* and *ProgramVersion*
   Both string variables registers which atomistic simulation tool has been used to generate the descriptor or property.
2. *JobID*
   The job ID, that is the integer value in the brackets used for *Job(*)*, is described by this attribute. For example, all datasets and groups located in Job(10) have a job ID of 10.
3. *JobLocation*
   The original location of the dataset is described by this string attribute. If, for example, the dataset or group has been originally written to */ AtomisticSimulationData/JobData/Job(10)* then this information will be given.

Property Attributes

In addition to general attributes properties always contain the following additional information.

1. *Unit*
   The unit of the property is described by this string variable. It can, for example, have a value of *kJ/mol* or *GPa*.
2. *Solver* and *SolverVersion*
   The solver used to calculate the described property is registered by both string variables.
3. EmpiricalFormula
   The empirical formula of the structure, for which the calculation has been performed, is registered by this string variable. This attribute can for example have the value *CuAu2*.
4. SpaceGroupName
   The space group of the structure is registered with this string using the Herman-Mauguin notation which is known as the international notation [10]. An example for its value is $Fm\overline{3}m$.

Other property attributes are more specific and depend on the calculation setup and used solver. An overview on these can be found in Table 3.

Table 3 Solver dependent metadata attributes for property datasets and groups are sorted by their kind, which can be string, integer or float, and by their source. The source is the program or solver used to calculate a given property and in the present work can be VASP, LAMMPS or other

	VASP	LAMMPS	Other
String	Functional, ExchangeCorrelationFunctional, KMesh, KIntegrationScheme, Precision, Magnetism, Potentials, Projection, CalculationType	ForceField, SimulationTimeUnit, TimeStepUnit, InitialTemperatureUnit, FinalTemperatureUnit, InitialPressureUnit, FinalPressureUnit, Ensemble	Strain
Integer	SmearingFunctionOrder		
Float	KSpacing, SmearingWidth, CutoffEnergy, ElectronicIterationsConvergence, Pressure	SimulationTime, TimeStep, InitialTemperature, FinalTemperature, InitialPressure, FinalPressure, CellConstrains	InteractionRange, AtomDisplacementSize

Structure-Centric Data Structure

The next step is to increase accessibility of the available data by transferring it into a structure-centric form. That is, by placing it in the *StructureData* group located in the *AtomisticSimulationData* group. The data consisting of descriptors and properties of each structure are located in the subgroups *Structure(*)*, with the wildcard * indicating an index that consecutively increases from 1 when additional structures are added to *StructureData*.

The properties themselves can be sorted groups within the *Structure(*)* group. For example, the *ElectronicProperties* group contains electronic properties such as the electronic density of states, the *VibrationalProperties* group contains vibrational properties such as phonon dispersion, the *MechanicalProperties* group contains mechanical properties such as elastic constants, and other properties can be placed in the generic *Properties* group.

Summary and Conclusion

In the present paper we have outlined a strategy on how to bridge the data exchange gap between atomistic and continuum simulation tools. A basis for our discussion has been the work done by Schmitz et al. [4] which describes microstructure data definition within a HDF5 file. However, due to a subtle difference on how these two approaches create and collect data, another strategy is required for atomistic simulation. We start from a calculation centric data definition, that is more suitable for atomistic simulations, and then identify a data structure by which calculation results can be made more accessible to others. Another important ingredient of our approach is to make excessive use of metadata to keep track of the calculation setup used to obtain a property.

The present approach is well suited for atomistic simulations, where a computational procedure is applied to a uniquely defined initial structure, i.e. an arrangement of atoms defined by their element type and coordinates, resulting in a set of computed properties. This requirement is fulfilled for most DFT calculations and forcefield-based molecular dynamics simulations. Other approaches such as Gibbs ensemble Monte Carlo simulations may require an extension of the present concept. In such simulations, the number of particles may change during the course of a simulation and hence the definition of "initial structure" needs to be extended. Another generalization will be needed, if the initial structure is actually an ensemble of structures, for example a set of models of amorphous structures, where the computed properties is a statistical average obtained from the entire ensemble of initial structures. Nevertheless, we believe that the present concept can serve as foundation for building a bridge between discrete models and continuum approaches.

References

1. Committee on Integrated Computational Materials Engineering, National Materials Advisory Board, Division on Engineering and Physical Sciences, National Research Council, *Integrated Computational Materials Engineering: A Transformational Discipline for Improved Competitiveness and National Security* (National Academies Press, 2008), p. 132
2. MedeA, Materials Design, Inc., Angel Fire, NM, USA, 2016
3. M. Christensen, V. Eyert, A. France-Lanord, C. Freeman, B. Leblanc, A. Mavromaras, S. J. Mumby, D. Reith, D. Rigby, X. Rozanska, H. Schweiger, T.-R. Shan, P. Ungerer, R. Windiks, W. Wolf, M. Yiannourakou, E. Wimmer, Software platforms for electronic/atomistic/mesoscopic modeling: status and perspectives. Integr. Mater. Manuf. Innov. **6**, 92–110 (2017)
4. G.J. Schmitz, B. Böttger, M. Apel, J. Eiken, G. Laschet, R. Altenfeld, R. Berger, G. Boussinot, A. Viardin, Towards a metadata scheme for the description of materials—the description of microstructures. Sci. Technol. Adv. Mater. **17**, 411 (2016)
5. The HDF group, http://www.hdfgroup.org. Accessed Feb 2017
6. G. Kresse, J. Furthmüller, Efficient iterative schemes for ab initio total-energy calculations using a plane-wave basis set. Phys. Rev. B **54**, 11169–11186 (1996)
7. G. Kresse, D. Joubert, From ultrasoft pseudopotentials to the projector augmented-wave method. Phys. Rev. B **59**, 1758–1775 (1999)
8. S. Plimpton, Fast parallel algorithms for short-range molecular dynamics. J. Comput. Phys. **117**, 1–19 (1995)
9. S.R. Hall, F.H. Allen, I.D. Brown, The crystallographic information file (CIF): a new standard archive file for crystallography. Acta Crystallogr. **A47**, 655–685 (1991)
10. D.E. Sands, Crystal systems and geometry, in *Introduction to Crystallography* (Dover Publications, Inc., Mineola, New York, 1993), p. 165