The hierarchical data format HDF5
- a Basis for an Emerging Standard in ICME Settings

G.J. Schmitz

2nd International Workshop on Software Solutions for ICME
Barcelona, Spain, 12th to 15th of April 2016
Overall objective: create interoperability for workflows

What is HDF5?
- Applications of HDF5
- HDF as a Best Practice
- Why to use HDF5 in ICME?
- Main Features

HDF Hierarchical data structure
- Material microstructure description
- HDF based workflow: a solution for interoperability between models
Process workflow structure

/processes

Processes

process#1 (Casting)

process#2 (Forming)

process#3 (Annealing)

models

Model #1 (Continuum)

Model #2 (Mesoscopic)

Model #3 (Atomistic)

Scales/Models

Outputs

Inputs

Solver data

/Component

-Part #

RVE #

Figures: courtesy B. Patzak
Component workflow structure

RVE tracks complete history of a material point, at various scales and across processes. RVEs are included in the hierarchy starting from the component/global level.

Figures: courtesy B. Patzak
Objective: Interoperability

Metadata: Formal scheme to organize and structure data in order to establish interoperability between different types of models and different data. Metadata also allow for tracking of additional information like data about data origin, error bars, and many, many others.

Specific implementation of interoperability and metadata schemes:

Several possible solutions:

- XML
- HDF
- MongoDB
- Simple ASCII files

Topic of this talk: the widely endorsed HDF5 based file format.
HDF software was first developed in 1987 at the National Center for Supercomputing Applications (NCSA) at the University of Illinois.
**What is HDF5?**

https://www.hdfgroup.org/

- A free and open source (BSD license) general purpose platform for storing, managing, archiving, and exchanging data
- Extensive facilities for data and metadata association, hierarchies, and annotation
- A self describing file format that is portable across operating systems and architectures, and that supports flexible user defined types
- A software library for high I/O performance, parallel I/O and out of core data access (partial I/O), which supports compression and other custom filters
- High quality documentation

**HDF5 is best practice in numerous applications!**
Applications of HDF5

**Computation**
- Advanced Parallel Numerical Simulation
- grid computing
- Machine Learning
- quantum Monte Carlo

**Earth observation**
- satellite meteorology
- Satellite Climate Monitoring
- weather radar
- Atmospheric radiation
- Atmospheric Physics & Dynamical Meteorology
- Ozone Monitoring

**Biography**
- DNA sequencing
- bioinformatics

**Geophysics**
- carbon and water fluxes modeling
- land cover maps
- groundwater hydrology
- Biological Oceanography
- fire detection
- geophysics
- seismic hazard analysis

**Medical**
- design of drugs and vaccines
- structure determination of subcellular complexes

**Physics**
- Synchrotron Science
- Neutron and X-Ray Scattering
- Accelerator/Plasma Science
- Gas Dynamics
- optical fibers
- Adaptive Optics
- Astronomy
- AeroAcoustics
- Magnetohydrodynamics
- Electromagnetics,
  **Gravitational Waves**
- Remote Sensing
- Scanning Near Field Optical Microscopy

**Electronics**
- photonics

**Financial Engineering**

**mechanical engineering**
- Aircraft Emissions
- Remotesensing
- airplane engine simulation data

2nd International Workshop on Software Solutions for ICME Barcelona April 2016
Length scales from light-years down to atto-meters
Time scales from billions of years to microseconds
Use of HDF5 in different software tools

ESI world-leading provider in Virtual Prototyping: HDF5 used as data container for their new solver result database format called ERF (ESI Result Database Format)

ITK from KITWARE Inc. uses HDF5 in medical image processing.

MATLAB, one of the core products from The Mathworks, stores its data in the HDF5 format.

CFD General Notation System (CGNS)
HDF5 is standard for storage and retrieval of computational fluid dynamics (CFD) analysis data.

CFD post-processing
● H5Fed (HDF5 based Finite Element Data Library) defines groups and datasets as well as offers an API to read and write hierarchically structured meshes, both on serial and parallel computers. It provides adjacency relations for all topological entities of a mesh, both upward and downward. Continued within DUNE project.

HDF5 Features

https://www.hdfgroup.org/

HDF5 suite includes:

- A versatile data model that can represent very complex data objects and a wide variety of metadata.
- A completely portable file format with no limit on the number or size of data objects in the collection.
- A software library that runs on a range of computational platforms, from laptops to massively parallel systems, and implements a high-level API with C, C++, Fortran 90, and Java interfaces.
- A rich set of integrated performance features that allow for access time and storage space optimizations.
- Tools and applications for managing, manipulating, viewing, and analyzing the data in the collection.
HDF5 Features

- **Free and open source**: open Format - HDF5 is widely supported in many programs, including open source programming languages like R and Python,
- **Platform-independent** binary data storage with optional compression

- **Hierarchical data ordering**
- **Self-describing tags** and datasets (including metadata, no additional metadata documents)
- "Data slicing", or extracting only portions of the dataset as needed for analysis

- Supports **complex data relationships** and dependencies through Datasets (multidimensional arrays) and Groups (container structures)
- Different types of datasets can be contained within one HDF5 file.
- Supports Large, Complex Data: HDF5 is a compressed format that is designed to support large, heterogeneous, and complex datasets.

- One file can describe data from an **entire workflow**
Why use HDF5?

http://www.hdfgroup.org/why_hdf/
Why HDF for Materials Modelling?

- different models (discrete/continuum):
  - electronic, atomistic, mesoscopic, continuum

- complex and heterogeneous data:
  - from "positions of atoms" up to multicomponent, multiphase, polycrystalline microstructures comprising defects

- Ease of use: inspectable by non-computer specialists similar to a windows directory

- Several applications of HDF known in experimental materials data

- Materials reveal a hierarchical structure. Need for structured data

- Much more...see this conference
The European Materials Modeling Council

Hierarchy in materials (continuum models)

integral variables

phase fractions $f_{\text{fcc}}$, $f_{\text{Al}_2\text{Cu}}$

+ state variables:
  avrg. composition $c$
  avrg. temperature $T$
  avrg. pressure $p$

characteristic scales

e.g. spacing $\lambda$, grain size $d$

local variables

order parameter $\phi_{\text{fcc}}$, $\phi_{\text{Al}_2\text{Cu}}$

lattice orientation $o(x)$

dislocation density $n_d(x,t)$

…

+ local state variables
  local composition $c(x,t)$
  local temperature $T(x,t)$
  local pressure $p(x,t)$
Hierarchy in materials (discrete models e/a/m)

**integral variables**
- avg. NumberDensity
- avg. Mass/MassDensity
- avg. composition c
- avg. vel./ temperature T
- avg. pressure p
- avg. defect density
- ...

**local variables**
- positions of atoms
- velocities of atoms
- types of atoms
- ...

**characteristic scales**
- e.g. interfaces, grains, lattices, …
discrete (e/a/m) → continuum

Information exchange is important at ALL and even across ALL hierarchical levels!!
need for 
a hierarchical  
data structure !!
Hierarchical Structure – a file directory within a file

http://neondataskills.org/HDF5/About/
Inspecting HDF5 files with HDFview

- Intuitive inspection
- similar to Windows explorer
- hierarchical structure
- extendable/amendable
- easy handling of metadata
Handling Metadata in HDF5

- Orientation(EulerAngles)
- Orientation(Quaternions)
- PhaseID
- FieldData
  - AtomPercent(1)
  - FeatureID
  - Orientation(EulerAngles)
  - PhaseID
- Volume_Fraction(1)
- Volume_Fraction(2)
- RVEData
- RVEDComposition
  - ChemicalElementNames
  - Composition(AtomPercent)
  - Number(Atoms)
- NumberChemicalElements
- RVEDefectStructures
- RVEDiscretisation
- RVEGeometry
- RVEDentifier

Table:

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Type</th>
<th>Array Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Units</td>
<td>moles</td>
<td>String, length = 20</td>
<td>1</td>
</tr>
<tr>
<td>DataOrigin</td>
<td>This is a trial to define the origin of the data...</td>
<td>String, length = 300</td>
<td>1</td>
</tr>
<tr>
<td>test</td>
<td>0</td>
<td>32-bit Integer</td>
<td>1</td>
</tr>
</tbody>
</table>

Number of attributes = 3

DataOrigin = This is a trial to define the origin of the data...
DREAM.3D (http://dream3d.bluequartz.net/)

- Collection of data analysis tools (filters) for the construction of customized workflows (pipelines) to analyze data.
- Provides a flexible, extendable, and open-format data structure simplifying data exchange between collaborators.

- Reconstruction of 3D microstructures from EBSD serial sectioning.
- Generation of synthetic microstructures.
- Algorithms available to extract various statistics like ODF, texture, neighborhood relations.
- Reconstructed volumes can be exported as industry standard STL files, ParaView files (.xdmf, .vtk), Abaqus (.inp).
- DREAM.3D stores all data as HDF5 files by default.
HDF5 customized to the description of materials!

**Available for:**
- Crystallographic data per phase (simple)
- Geometric information on objects (size, shape, centroids, neighbours, ...)
- Orientation information on objects (Euler angles)
- RVE statistics/texture
- ...many more

**Desirable for:**
- **composition data** to be included / EDX type data
- Thermodynamic data to be included
- Defect data to be included (dislocations etc)
- **data for properties** / elasticity/plasticity data
- 4D data (i.e. time dependant microstructures)
Effective properties of the RVE being derived from knowledge about microstructure and per phase properties are stored in the MATERIAL_Properties container of the RVE data.

Origin of data:
- experimental
- Literature
- e/a/m models
- etc
Microstructure data for a given phase: thermodynamic, kinetic, crystallographic properties.

Material properties for each phase may be integrated into the ensemble data:

- by reading from a material file (e.g. Abaqus) for each phase
- by extracting (post-processing) such properties from discrete models. (multiscale linking)
Ensemble properties can be used to generate the effective properties of the RVE,

High level homogenization structure from lower scale models

Implementation through HDF data structure:

storing homogenized properties at different hierarchical levels in the same data structure.

phase containers containing all properties of a phase
Information flow between material models

Basic scheme of a simulation workflow. **Linking of models:**

- Along the **time/history**: from initial conditions to output/final conditions
- Across the **scales** via exchange of **boundary** conditions and **effective** values

**HDF5 file**

providing all necessary I/O ports

**scale**

**time/process**
Towards a Metadata Scheme for the Description of Materials
-the Description of Microstructures-


- to appear in „Science and Technology of Advanced Materials“ (under review)

- scheduled to appear in spring 2016 on-line as open access

- HDF5 template file comprising the defined descriptors will be provided along with publication
Conclusions

- HDF is established as a “best practice” in many areas of application
- HDF has been presented as a possible choice for data storage/implementation level for interoperability
- HDF hierarchical data structure allows for a description of an entire workflow in a single data file
- All processes and models in the workflow can operate on the shared component/material data structure
- The HDF data structure allows homogenization of material properties as it contains both local and integral data
Benefits of HDF hierarchical data structure

- Store and retrieve information at the desired level of detail
- Allows integration of information from all types of sources/models (also experimental!)
- Selected parts of the file can be exchanged individually
- Meets the hierarchical structure of materials

HDF may become a hub for information exchange between models (electronic/atomistic/mesoscopic/continuum)
The Hierarchical Data Format HDF5 - a Basis for an Emerging Standard in ICME Settings?
Georg J. Schmitz

HDF File-based interoperability for multiscale modelling
Fabio Sacconi and Stefano Bellocchio

On the usage of HDF5 in the DAMASK crystal plasticity toolkit
F. Roters, C. Zhang, P. Eisenlohr and P. Shanthraj

HDF5: A New Approach to Interoperability in Finite Element Tools
A. Bhaduria, C. Agelet, M. Chiumenti and M. Cervera

Thermodynamics and HDF5: Towards a Metadata Keyword Scheme for the Description of Thermodynamic Properties
S. Petersen
Database concepts and ICME links for atomistic simulations

Joerg Neugebauer

Implementation, benefits and handling of formal metadata schemata as enablers for Platform interoperability - From Data Structures to Modelling Platforms
T.F. Hagelien, A. Hashibon and S. Amini

Role and Descriptions of Chemical Element Distributions in Microstructure Simulation

B. Böttger and J. Eiken

Organization of materials data for efficient ICME implementation
A. Salem, J. Shaffer, R. Kublik and D. Satko

Towards a Metadata Keyword Scheme for the Description of Materials – Keywords for the Description of Microstructures –
Georg J. Schmitz, Bernd Böttger, Markus Apel, Janin Eiken, Gottfried Laschet, Ralph Altenfeld, Ralf Berger, Guillaume Boussinot and Alexandre Viardin

The Distributed and Unified Numerics Environment (DUNE)

Oliver Sander

Creating a Common Environment for Storing, Sharing and Working With Digital Microstructure Data: Thoughts and Lessons Learned
Michael Groeber, Michael Jackson and Sean Donegan
HDF software was first developed in 1987 at the National Center for Supercomputing Applications (NCSA) at the University of Illinois.
Integration of e/a/m data and continuum data structure

**Geometry/structure:**
lattice vectors, position, atomtype, material name, composition, strain

**Properties**
- **Mechanical**
piezoelectric coefficients, elasticity
- **Electronic**
  *Bulk*: Band gap, Band gap bowing, Valence band offset
  *Band structure*: effective mass, band energy, alpha, deformation potentials

Total energy of the system from the DFT calculation (*Hamiltonian*)

**Solver parameters:**
TB parameters, \(k\cdot p\) (Luttinger) parameters, internal strain parameters, VFF parameters
Process description

- 4D data, i.e. time-dependent microstructures, could be thought of as a sequence of HDF5 files.
- All spatially resolved data could be described in vtk only.
- Description of phase data, RVE effective properties etc. through HDF5 structures (with vtk describing All spatially resolved data being something like a subset).
- Description of Workflows and temporal evolution of materials through combinations of HDF5 (and other data and file types) and probably best achieved on the basis of xdmf (with HDF5 being a subset).
## Hierarchy of models

<table>
<thead>
<tr>
<th>integral models ( p = f(t) )</th>
<th>intermediate ( p = f(t, \lambda) )</th>
<th>comprehensive models ( p = f(t, x) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>thermodynamics + relaxation</td>
<td>CAFE</td>
<td>Monte Carlo Potts model</td>
</tr>
<tr>
<td>Scheil-Gulliver</td>
<td>needle crystal</td>
<td>phase field / phase field crystal</td>
</tr>
<tr>
<td>JMAK kinetics</td>
<td>precipitation/ripening (LSW)</td>
<td>cellular automaton</td>
</tr>
<tr>
<td>volume averaging</td>
<td>envelope dynamics</td>
<td>level set</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>lattice Boltzmann</td>
</tr>
</tbody>
</table>

increasing fidelity and number of variables
What is HDF5?
- Applications of HDF5
- Use of HDF5 in software tools
- HDF for FEM
- Features
- Hierarchical data structure

Motivation: Why we need a hierarchical data structure

HDF for microstructures
- Hierarchy in description of materials
- Application: Dream3D
- Example of data structure
- Data flow for Multiscale approach
Benefits

- hierarchical data format with metadata support
- portable (endianness) and flexible file format and a API to store or retrieve multidimensional arrays,
- data model is defined in the file
- parallel I/O implementation (Array data transfer can be collective or independent.)
- Interface to C/C++/Fortran/Java/Python/Matlab/CLI/...
- de facto standard (netCFD also can use HDF as storage format)
- Open visualization tools supporting HDF (Paraview)

Drawbacks

- Openness: For commercial applications, data encryption is often imperative.
- No existing widely accepted HDF standard in FEM community