Enabling Interoperability for LS-DYNA Users with Envyo® using the VMAP Standard

Christian Liebold[1], Tolga Usta[1]

[1]DYNAmore GmbH

Keywords: *standardization, simulation process chain, Finite Element data mapping*

1 Abstract

From 2017 until mid' 2020, DYNAmore collaborated with various partners from the manufacturing industry, universities, independent research institutions, and software vendors to develop a software neutral storage format for finite element data. The goal was to define a new standard which allows for the flawless exchange of all the required information, enabling the industrial partners to easily establish closed simulation process chains for production processes, where various software tools with non-consistent data formats have been a barrier in the past. During the project, an interface between the mapping tool Envyo® and the established VMAP standard has been realized and validated with test cases.

In this paper, an overview on the current status of the released VMAP standard [1] will be provided together with an insight into the defined HDF5 [2] based data structure for finite element data storage. Based on the example of Envyo® [3] it will be demonstrated how the C++ based VMAP standard library can be linked to other software solutions as well – either through direct code import or by linking the provided libraries. Small examples will illustrate the VMAP capabilities for simulation data exchange and how Envyo® users can benefit from the standardized format which can be used as input and output in future releases.

2 Introduction to the VMAP project

VMAP is the abbreviation for "Virtual Material Modelling in Manufacturing". The VMAP project started in 2017 as a group of more than 25 different companies, research institutions, universities, and software vendors from various countries, namely: Austria, Belgium, Canada, Germany, the Netherlands, and Switzerland. The goal was to develop a neutral, standardized format for simulation result data storage focusing on a flawless transfer of these data from one software tool to another, considering that different material modeling approaches might be necessary along the simulation process chain. Therefore, a common understanding for the terms and nomenclature being used for various material descriptions within the different software tools represented by the software vendors had to be created in a first step. This task was accompanied and supported by six use cases which should help the consortium to focus on specific aspects on current material modeling problems which are relevant for the consideration of several process steps to improve accuracy in the final usability analysis. These use cases were [4]:

- Blow Forming
- Composites for Lightweight Vehicles
- Injection Molding
  - Impact
  - Foaming
  - Fatigue
  - Creep
- Additive Manufacturing (AM) Plastics
- Hybrid Modeling of Consumer Products
- Composites in Aerospace

The strength of the project was based on the large number of software providing companies supporting the consortium during the discussions and the software implementation process. A list of software vendor companies and the name of the software which shall support the VMAP standard in the future is given in tab. 1.

Table 1: List of participating software companies and their software tools as well as the possible simulation domain [6].

Software	Vendor/developer	Simulation stage
4a Fibermap	4a engineering	Material modeling Mapping solution
ANSA	beta-CAE	Pre- and Postprocessing Mapping solution
Cadmould	Simcon	Process simulation
COMPRO	Convergent	Process simulation
Digimat	e-Xstream	Material modeling Mapping solution
Envyo	DYNAmore GmbH	Mapping solution
LS-DYNA	DYNAmore GmbH	Process simulation Structural analysis
MpCCI	Fraunhofer SCAI	Mapping solution
MSC.Marc	MSC	Structural analysis
Open-FOAM	Open source ESI	Process simulation
PAM-Crash	ESI	Structural analysis
PAM-Form	ESI	Process simulation

As can be seen from tab. 1, various software solutions for either different simulation domains or technologies for the data interpretation and transfer between different software tools and simulation steps have been involved in the projects which underlines the generality of the defined VMAP standard. The DYNAmore GmbH provided their knowledge on LS-DYNA specific definitions of element types, in- and out-of-plane integration rules, material descriptions, used coordinate systems and LS-DYNA specific result data storage. Furthermore, the mapping tool Envyo has been used for first implementations of the developed VMAP standard as a translator (or wrapper) between the LS-DYNA format and the newly defined VMAP standard allowing for fast in-house code adaptions.

3 Technical aspects of the VMAP standard

As a vendor neutral data storage container, the Hierarchical Data Format (HDF) 5 [2] has been chosen since it is available as an open-source library. Data is stored efficiently through a provided programming interface and allows the user to define his own data structure based on groups, sub-groups, datasets and attached metadata, providing additional information about the datasets or containers. Users may also provide information about the type of data being stored in datasets, so they can be integer, double, string, or other data types – even compound data types are allowed but not very efficient when it comes to in- and output of data. Since information are stored as binary data, it is not easily accessible. Therefore, the HDF group provides an additional graphical user interface (GUI) called HDFView which allows users to investigate the data structure such as they would do it in a regular browser, even allowing them to perform minor changes on the data. All HDF5 data container are stored with the file ending *.h5.

The standard specification document [5] provides the main information about how simulation result data should be stored inside the HDF5 container. Developers may choose to define either their own VMAP I/O routines based on the standard specification document, or to use the VMAP Standard I/O Library Application Programming Interface (API) provided by the VMAP Standard Community. The latter approach is the most common one and was also used by DYNAmore within the project. The VMAP I/O routines have been written in C++, but linkage to other high-level programming languages is possible using the “Simplified Wrapper and Interface Generator” (SWIG) [7], such as shown in fig. 1. The usage of the latter allowed the software developers of the consortium to contribute to the project in their preferred programming language without bothering to much about how their work can be linked to the work of others later in the project since SWIG helps to automatically create the linkage between C++ interfaces and the target programming languages.

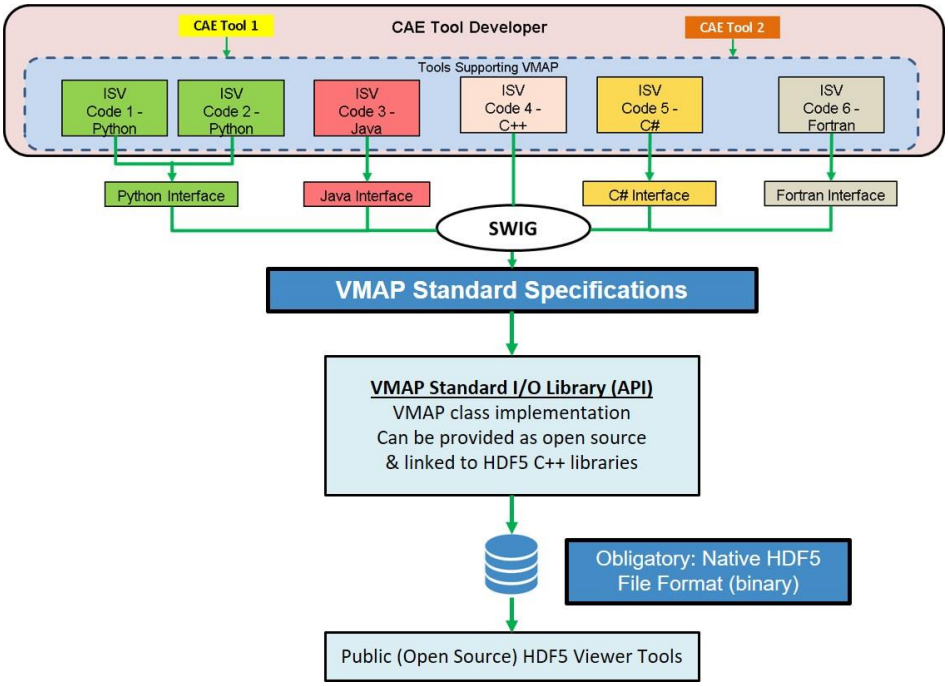


Fig.1: Illustration of the contribution of the various software partners (ISVs) in their preferred programming languages using SWIG and HDF5 [4].

The main files used for the VMAP Standard API are [5]:

- VMAPFile.h: header file which contains the declaration of all in- and output functions of the VMAP Standard API being used to read from or write to the *.h5 file.
- VMAPFile.cxx: source code file for all functions declared in VMAPFile.h
- VMAP.h: header file which contains all arguments defined as struct reference which can be written to and read from the *.h5 file. Furthermore, it contains all necessary constructors and destructors as well as get and set functions to assign values to the attributes of the VMAP Standard Library.
- VMAP.cxx: source code of the aforementioned constructors and destructors as well as definitions for the get and set functions for each struct referenced defined in VMAP.h.
- VMAPH5Tools.h: defines H5 specific tools which are used by the Standard I/O Library.

Basically, VMAP.h and VMAPFile.h need to be added to the main program's include paths to allow for the usage of the VMAP library. Following file initiation and creation of the VMAP.h5 – storage file, the user may assign various general information to the version and system group. The version group stores information about the supported VMAP version as well as additional meta-information. The version is stored as a standard "Major-Minor-Path" integer type enumerator. The system group holds information about the used coordinate systems and unit systems as well as used element- and integration types. The two latter ones may be defined by the user himself, but it is beneficial to make use of the provided VMAPElementTypeFactory.cxx database which provides more than 30 predefined element types in 1D, 2D, and 3D and may be combined easily with the VMAPIntegrationTypeFactory.cxx database which allows to assign various integration rules to the elements stored inside the VMAP standard file.

Afterwards, finite element mesh data can be stored in the geometry group which holds information about elements and points for each part. The points group is only depending on the coordinate system block, whereas the element group may be linked to further data such as material, element type, points, and coordinate system. The element types group is again dependent on integration types and points.

Furthermore, state variables may be assigned to the VMAP file using the state variables group. This group holds information about the units being used to calculate these state variables, the coordinate system in which the information is being stored and the location of the storage, mainly: global, nodal, integration point, element, or element face. The already defined material group shall store information

about the material models being used for the calculation. Further dependencies are still missing and will be developed within future work of the VMAP standards community (SC).

Fig. 2 shows the main dependencies of groups and sub-groups inside the defined VMAP standard.

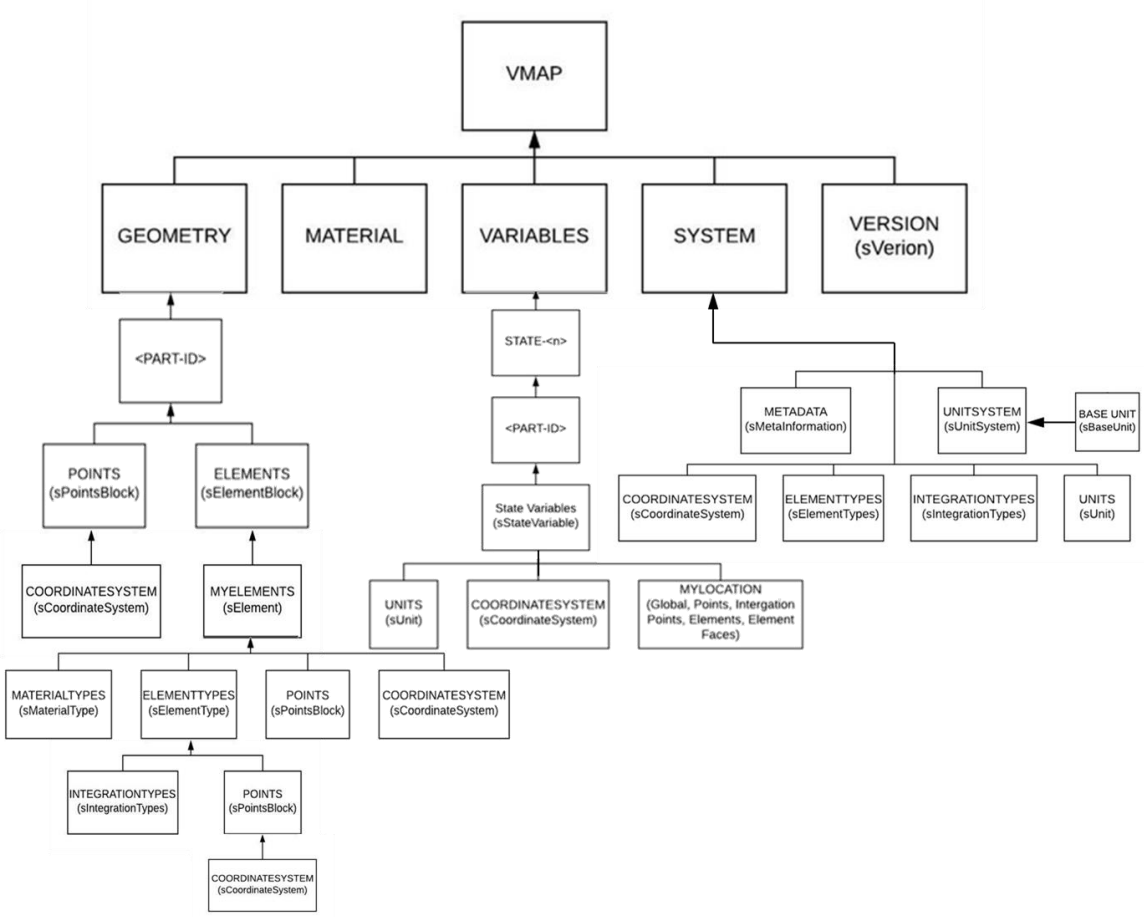


Fig.2: Dependencies of VMAP groups, sub-groups and data sets as defined in the VMAP standard I/O library [5].

4 Implementation of the VMAP standard into Envyo®

Within the VMAP project, Envyo has been used as a wrapper or translator tool between the VMAP standard and the LS-DYNA ascii input and result or dynain files. Therefore, read and write routines had to be adapted in a way that the Envyo internal data structure could be filled with VMAP data and vice versa. Since both VMAP and Envyo are written in C++, a direct integration of the VMAP standard I/O library was possible without going through the SWIG interface language.

As known from the standard mapping options, the main ENVYO – command is used to activate the feature which transfers LS-DYNA dynain-files into a VMAP.h5 file. Tab. 2 shows possible input commands which help to translate the LS-DYNA native ascii input. Thereby, the user may select which parts of the geometry should be transferred, needs to assign the unit system which has been used for the calculation and may activate or deactivate the transfer of various data such as tensorial stress and strain data, history variables, which will be stored inside the VARIABLES group, effective plastic strain (EPS), and fiber orientation tensor data for short fiber reinforced plastic materials and fiber orientation vectorial data for continuous fiber reinforced plastic materials. Furthermore, data can be reduced by the user by defining specific Part-IDs which shall be included in the VMAP.h5 file. Since current implementations do not support the direct reading of element type information provided in the `*SECTION` cards in LS-DYNA input files, the user should also provide element type information following the source PID which should be read. In the provided example below, this is SH13 – which is an LS-DYNA native plain strain element. The `UndeformedSourceFile` equals to the input deck

being used to generate the `dynain` which is in this case the `SourceFile`, since it is read into the VMAP.h5 file.
To generate a LS-DYNA ascii file, the main mapping command becomes `VMAP2DYNA`, and the

Table 2: Envyo input command for reading a LS-DYNA result file into a VMAP.h5 file.

```
$#---------------------------------------
$# Main mapping definition
$#---------------------------------------
ENVYO=DYNA2VMAP
$#---------------------------------------
$# In- and output meshes
$#---------------------------------------
UndeformedSourceFile=UndeformedGeometry.k
SourceFile=dynain
SourceUnitSystem=lb-in-s
MappingResult=VMAP.h5
$#---------------------------------------
$# Source PIDs
$#---------------------------------------
NumSourcePIDs=1
SourcePID#1=1,SH13
$#---------------------------------------
$# Mapping options
$#---------------------------------------
Translate_Stress=YES
Translate_Strain=NO
Translate_HISV=NO
Translate_EPS=YES
Translate_FiberData=NO
```

`SourceFile` will be a VMAP.h5 file. Additional file format information may be provided using `SourceFileFormat=VMAP`, and result data is written to the *.key file referred to in the `MappingResult` command. Otherwise, the user has the same options to reduce data by assigning specific Part IDs for the output to the LS-DYNA ascii file as well as to select the data which should be translated from VMAP into LS-DYNA format.

A comparison between the resulting VMAP.h5 file and the dynain input file is provided in fig. 3. Color-coded frames shall help to understand the relationship between what is written to the VMAP.h5 file (left) and how these data are represented in the dynain result file. Information related to nodes and elements from `*NODE` and `*ELEMENT` are stored inside the GEOMETRY group, listing everything related to Part ID 1 in subfolders. Inside ELEMENTS, MYELEMENTS is a dataset which holds identifiers (element-IDs), the referred element type and coordinate system (in this case, global), no information referring to materials or sections, but connectivity, which directly represent the nodal IDs defining the element. The nodes corresponding to the connectivity are listed inside the POINTS group, under MYIDENTIFIERS, with a corresponding list of nodal coordinates provided in MYCOORDINATES. Additional information such as thicknesses stored at nodal points or angles referring to the element coordinate system (e.g. `*ELEMENT_SHELL_BETA`) would be stored inside the VARIABLES group. In this example, the VARIABLES group mainly holds information about the current stress state inside the STRESS_CAUCHY group and the plastic strain inside STRAIN_PLASTIC_EQUIVALENT. Those are highlighted in light red (effective plastic strain) and green (stress tensor variables). Information such as the used integration rule, which in this case is a plane strain shell element with four in-plane integration points is stored inside the SYSTEM block in the ELEMENTYPES and INTEGRATIONTYPES datasets. The five history variables written to the dynain file would be stored inside the VARIABLES block as well. The nodal displacement is calculated internally by Envyo from the provided `UndeformedSourceFile` and the `dynain` file.

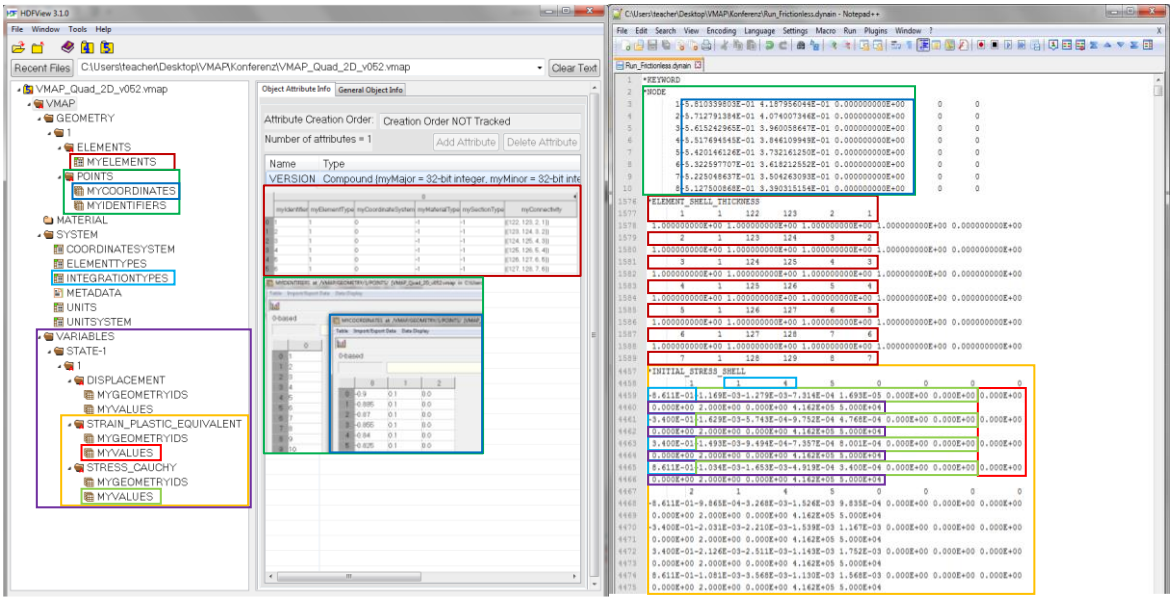


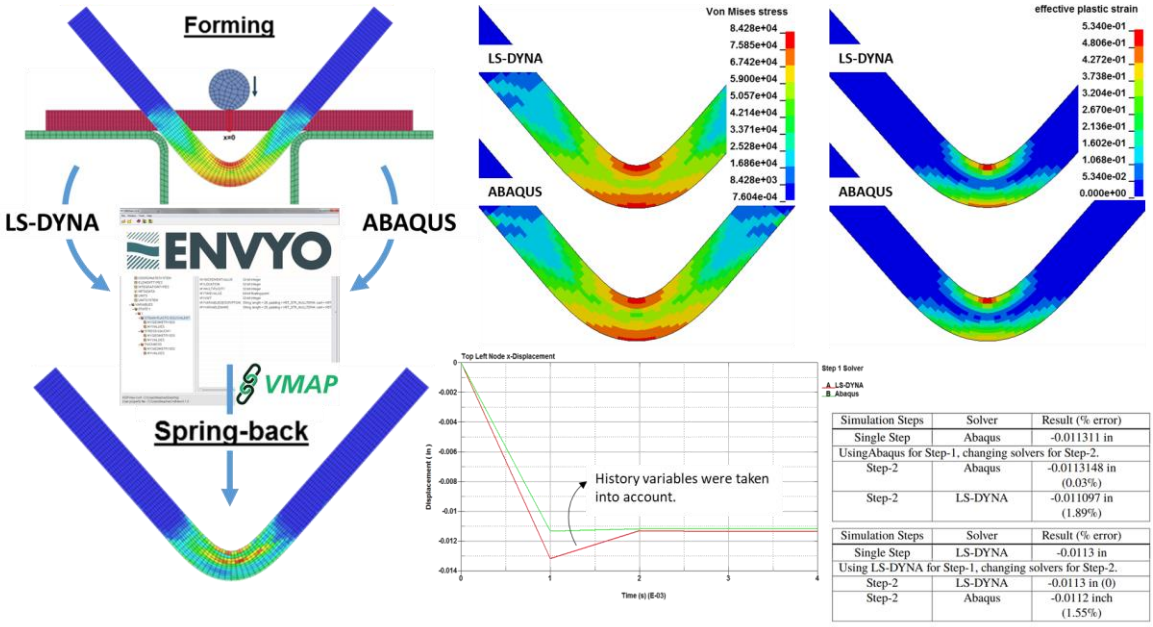
Fig.3: Illustration of relevant data from a dynain file (r) being stored in a VMAP.h5 (l) file using the above provided translation commands (tab. 2) [8].

5 VMAP application examples

As stated above, Envyo is currently used as a translator or wrapper between VMAP and LS-DYNA format specifications and vice-versa. Two examples shall demonstrate the realized capabilities, a metal forming test case and a short fiber reinforced plastics test case. These two and further test cases are available for those interested in VMAP through the project's website [1].

5.1 Metal forming test case

The metal forming test case is used to demonstrate the interoperability capabilities of the defined VMAP standard, using both, Abaqus and LS-DYNA result data from a forming simulation step, transforming each result file into the VMAP format and back into the native solver formats to use the generated data as input for a subsequent spring-back analysis. As shown in fig. 4, the differences in the results are neglectable. In the provided example, no mapping is performed, but the results are exchanged between the different solvers, allowing to evaluate the capability to transform data through VMAP without losing accuracy. As reference, the two-step simulation chain using LS-DYNA for both the forming and the spring-back analysis is chosen. Thereby, the difference in the vertical displacement of the upper left node of the geometry is chosen for evaluation purposes. When running the subsequent spring-back analysis with Abaqus, a difference of about 1.55% compared to the pure LS-DYNA modeling approach is being observed. Running both simulations using Abaqus, results are comparable to the pure LS-DYNA modeling approach, with a minor difference of 0.03% which can be due to numerical differences within the two solvers. Performing the initial simulation with Abaqus and the subsequent simulation with LS-DYNA leads to an offset of 1.89%. The graph displayed in fig. 4 shows a small offset between the LS-DYNA and the Abaqus spring-back simulation. This is due to the consideration of material history variables in the LS-DYNA modeling approach whereas due to the solver change from LS-DYNA to Abaqus, history variables are neglected. Visual inspections of the von Mises stresses after the forming simulation and the effective plastic strains after the spring-back simulation for both solvers used emphasize that there is no loss of accuracy when transferring data through the VMAP standard between different solvers which is crucial for acceptance of the new format within the Computational Aided Engineering (CAE) community.



Simulation Steps	Solver	Result (% error)
Single Step	Abaqus	-0.011311 in
UsingAbaqus for Step-1, changing solvers for Step-2.		
Step-2	Abaqus	-0.0113148 in (0.03%)
Step-2	LS-DYNA	-0.011097 in (1.89%)

Simulation Steps	Solver	Result (% error)
Single Step	LS-DYNA	-0.0113 in
Using LS-DYNA for Step-1, changing solvers for Step-2.		
Step-2	LS-DYNA	-0.0113 in (0)
Step-2	Abaqus	-0.0112 inch (1.55%)

Fig.4: Illustration of the metal forming testcase, transforming LS-DYNA and Abaqus forming results into the VMAP format through Envyo (l) and performing a spring-back analysis after retrieving Abaqus and LS-DYNA native file formats from that file. Visual comparison of the v. Mises stress after the forming simulation (u.m.) and of the eff. plastic strain after the spring-back analysis (u.r.). The graph at bottom, middle, shows the resulting displacement of the top-left node in the spring-back simulation run with both, Abaqus and LS-DYNA over time – and the tables at the bottom, right, show the offset in displacement in percent, using a two-step approach with LS-DYNA used for both simulation steps as reference [8].

5.2 Short fiber reinforced plastics test case

The second test case illustrated here is related to short fiber reinforced plastics. In the provided example, results from Cadmould provided in a VMAP.h5 format have been interpreted by Envyo and written to a dynain file which supports orientation initialization for the commonly used LS-DYNA material model `*MAT_ANISOTROPIC_ELASTIC_PLASTIC` (`*MAT_157`) [9]. Fiber orientation in this case is read from the ORIENTATION_FIBRE group. This test case is interesting because it demonstrates the advantage of a standardized CAE result output format for both, software developers and users. Current implementations of Envyo were not able to read and write native Cadmould result data since they are commonly provided in a binary output file which needs to be investigated thoroughly to be interpreted correctly. A standardized result file format such as VMAP significantly simplifies that process.

A visual inspection of the results provided in fig. 5 show the accuracy of the transferred Cadmould data to LS-DYNA. The distribution to the right and left of the specimen is strongly influenced by the position of the injection gauges, so a spread of the fibers into the various directions is clearly visible. A little further away from the injection gauges, fiber orientations perpendicular the direction of flow are visible and are caused by the so-called skin-core effect. At the center of the specimen, a weld-line becomes clearly visible due to the meeting of the two flow-fronts. These results are expected from such a geometry and injection gauge position, and therefore the interoperability through the VMAP standard has been proven for short fiber reinforced plastic materials.

For visualization purposes, a single step analysis has been performed with LS-DYNA and the history variables stored in the d3plot file have been used to generate the vector plot using the Post → Vector → Hist. Var. cosine functionality. Note that the orientation histories in this case are stored in LS-DYNA with reference to the element coordinate system.

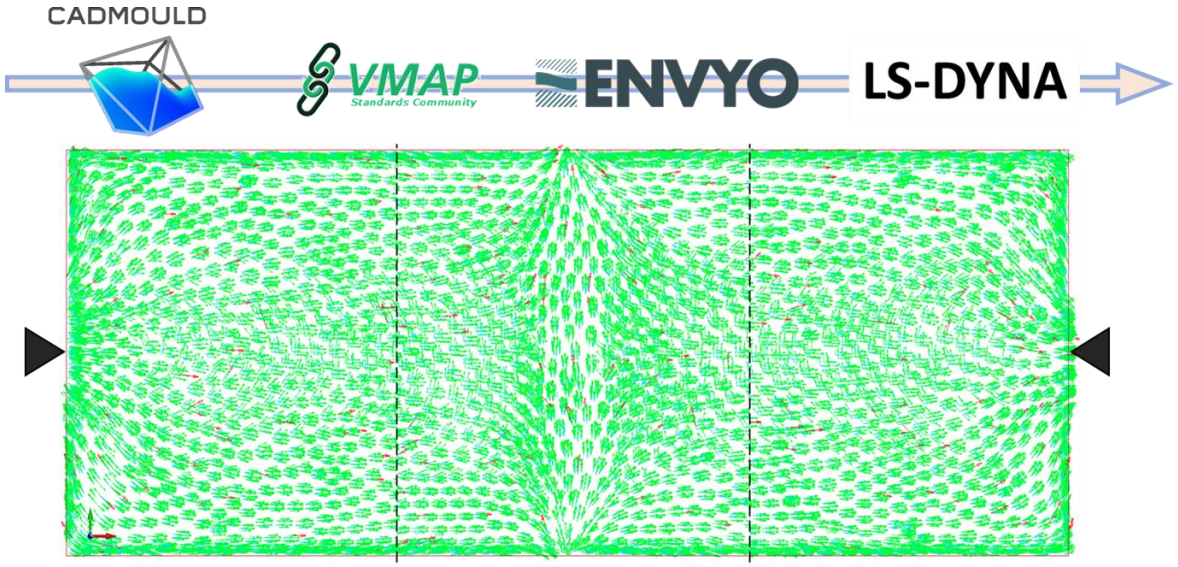


Fig.5: Fiber orientation visualization transferred from Cadmould to LS-DYNA `*MAT_157` using the VMAP standard [8].

6 Conclusion and Outlook

Within the VMAP project, a standardized result data storage format has been defined, based on a HDF5 binary data container. The high number of software vendors within the project consortium ensured a huge degree of acceptance within the community and knowledge transfer right from the start of the project. Therefore, the resulting standard is vendor-neutral, so that everybody may benefit through its usage. As has been shown in this paper, the VMAP standard I/O library developed during the project run-time is easy to use and implement into other software codes as well, even if other programming languages besides C++ are being preferred by the developers. Furthermore, the developed standard can be adjusted to the users' needs. The standard's specifications have been well defined and the implementation guidelines are easy to follow and available to public through the VMAP website [1].

Two test cases have been chosen to demonstrate the data transfer between different solvers through the VMAP standard without losing any accuracy or information. Currently, Envyo may be used by LS-DYNA users to read and write VMAP data. Future developments will allow for a direct mapping between VMAP file formats, so that Envyo will be an interesting mapping tool for all CAE engineers, regardless of the finite element solver they are using.

Following the VMAP project, the newly created VMAP standards community which is also supported by DYNAmore and most of the project members will take care of further developments of the VMAP standard and enhance the current I/O library based on results gained in follow-up projects. A preliminary focus might be on the enhancement of the material data and meta-information storage allowing the transfer of these information between the different solvers. Compared to the storage of geometry and mesh information for finite element analysis, this task is much more sophisticated since it is not only defined by clear mathematical descriptions, but other physically motivated assumptions as well which might be referred to by the different CAE tools with different terms and wording. Another interesting topic which should be dealt with in the future is the storage of data gained in other simulation disciplines, such as Computational Fluid Dynamics (CFD) or multi-physics simulations. In addition to that, first steps towards the integration of data gained through material testing shall be made so that basically all data generated and relevant during a component's lifetime could be stored within one data storage container.

7 Acknowledgements

This work is funded by ITEA, a transnational and industry-driven R&D&I program in the domain of software innovation.

Furthermore, this research and development program is funded by the Federal Ministry of Education and Research (BMBF), Germany under the supervision of the project execution organization in Berlin (PT-DLR). The author is responsible for the content of this presentation.

8 Literature

[1] www.vmap.eu.com
[2] www.hdfgroup.org/solutions/hdf5/
[3] DYNAmore GmbH, “Envyo® User’s Manual”, GER, 2021.
[4] Gulati, P., Duffet, G.: “VMAP – General Information” – Version no. 1.0.0, Dec. 2020.
[5] Gulati, P.: “VMAP Standard Specifications” – Version no. 1.0.0, Dec. 2020.
[6] Liebold, C., Haufe, A., Vinot, M., Holzapfel, M., Dittmann, J., Böhler, P., Fritz, F., Finckh, H.: “Lessons learned from Developing a Digital Prototype within the ARENA2036 Environment and Improvements with the new VMAP Standard”, NAFEMS WC ’19, Québec, CA, June 2019.
[7] www.swig.org
[8] T. Usta, C. Liebold, A. Haufe: “Implementation insight and demonstration of the advantages of the VMAP standard for CAE engineers”, 1st Intl. Conference on CAE Interoperability, online, Oct. 2020.
[9] Nutini, M., Vitali, M., Benanti, M., Formolo, S.: “Polypropylene Composites under impact: anisotropy, mapping and failure criteria in simulations, and validation on a part for building and construction industry”, 12th European LS-DYNA Conference, Koblenz, GER, May 2019.