Relating Structural Test and FEA Data with STEP AP209

R.Lanza[a,b,*], J.Haenisch[a], K.Bengtsson[a], J.C.Johnson[c], M.Jahadi[c], T.Rølvåg[b]

[a] *Jotne EPM Technology AS, Grenseveien 107, 0663 Oslo, Norway*
[b] *Norwegian University of Science and Technology, Richard Birkelands vei 2B, Trondheim, Norway*
[c] *Lockheed Martin Corporation, 1 Lockheed Boulevard, Fort Worth, TX 76108, USA*

Abstract

This paper proposes a method for incorporating FEA data and structural test data into one common standardized data model. This is done by taking advantage of the extent of the STEP ISO-10303 [1] standard, specifically Application Protocol 209e2 [2]. The model keeps the data traceability between the two phases, analysis and testing, such as sensor locations and finite elements, test and FEA load cases, and test and FEA results.

We also present an introduction to STEP and AP209e2, and discuss how it can be used in a Simulation Data Management environment.

Keywords: STEP ISO 10303, FEM Analysis, Structural Testing, Data Exchange, Simulation Data Management

---

1. Introduction

Simulation and structural testing plays a big role in the development of complex products. As Moore's Law continues, higher computational power and storage becomes available. This has led to an ever-increasing amount of simulations, especially with analysis optimization becoming more common. The higher computational power allows engineers to perform more complex and precise analyses with denser mesh than ever before, and if properly done, results in more optimized and safer products.

The problem arises when all this data must be stored for reuse in the near future or archived for longer term. The large amount of data means finding information becomes more difficult. Files in different formats, for different applications are spread over multiple locations, and working on projects across companies turn out to be complicated. A solution for this is often declared to be Simulation Data Management (SDM) and Product Data Management (PDM) which are growing in popularity. This makes organizing simulation and CAD data together with other engineering information more efficient. One of the industries where the above is significant is the aerospace industry, which is also behind the work of this paper.

*Corresponding author. Tel.: +47 452 04 992
Email addresses: remi.lanza@jotne.com (R.Lanza), jochen.haenisch@jotne.com (J.Haenisch), terje.rolvag@ntnu.no (T.Rølvåg)

Still, with SDM, users are often locked to proprietary formats of the software initially used for their simulations and designs, causing certain complications when different partners use different software. SDM is not the main focus of this paper, but as we will see, AP209e2 (AP209e2 is the second edition of AP209, and will from now on be referred to as AP209) is not only used as a file format but can also be the backbone data model in any software (including Data Management tools).

Added to the complexity of simulation data, we also have structural test data. When safety is of high importance, a complex analysis may require a physical validation. This can be everything from testing the capacity of a certain composite part to a complete full-scale airplane test. The result is however more data to organize. A typical (and simplified) scenario involving structural testing is as follows:

1. A simulation is performed and results are saved in the CAE software's native format.
2. Based on the results, actuator and sensor locations are chosen for the structural test.
3. Tests are performed and loads and results are exported from the test equipment to yet another format.

Companies then often have their own internal work-flows to be able to compare the two results. Performed manually or by scripts, a set of definitions are required:

1. Sensor distribution in the FE model frame of reference.
2. Sensor orientations in the FE model frame of reference.
3. Relation between corresponding test cases and analysis load cases.
4. Sensor mapping to channel IDs from the test equipment.
5. Information about applied filtering techniques on applied loads and sensor result data.

With this information, the corresponding virtual and physical results can be managed. After transforming the results to matching orientations, a comparison can be done. The results are typically output to Excel sheets or report documents.

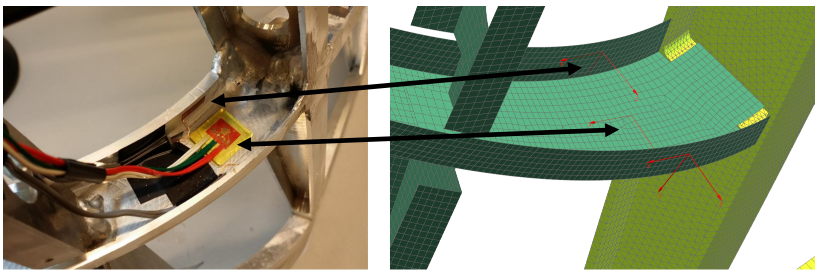

Figure 1: Mapping sensor locations and orientations to FEM model.

Finally, some of the data that has to be stored are the following:

1. FEM analysis files
2. FEM result files
3. Structural test output files
4. FEM-Structural test definition files
5. Comparison / correlation results
6. Reports

In certain industries there exist strong regulations on data retention of products. This is the case for the aerospace industry. As an example, the Federal Aviation Administration (FAA) in the United States, requires that *'Type design data must be retained and accessible for the lifespan of the product. It is possible that technical support for the original software will be terminated during the product lifespan, so your procedures manual must explain how access to the data will be retained or transitioned to a new software system.'* [3].

The goal of this paper is to validate that the AP209 data model have the capabilities to represent the above information, as well as keeping the traceability between the different fields. Thus, enabling the storage of the complete data set in a neutral and archive-friendly format.

Figure 2 presents an overview of most of the data which we want to represent in AP209, and how it all relates together internally in a model.

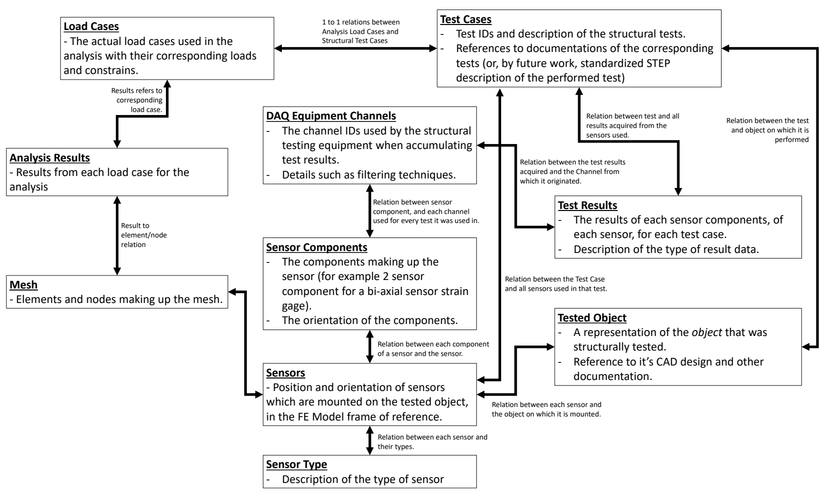

Figure 2: Overview of main data represented in the model and how they relate.

In the next sections we cover a little about the background of the STEP ISO-10303 standard, followed by Section 3 where we present the outline of the proposed model, while Section 4 to 6 goes into specific details of the model.

2. STEP ISO 10303

2.1. Background

Started in 1994 the goal of this standard was to standardize the representation of a product throughout its whole life-cycle across all relevant domains. The whole standard is represented in EXPRESS [4], a lexical language which is both human and computer readable, in a form of objects, inheritance, attributes and rules.

Part 21 of the STEP standard [5] describes the ASCII representation of STEP, which is commonly known as the *STEP file format*. In addition, STEP defines different schemes that can be used to create database repositories for data sharing. This is standardized in yet another part, Part 22 *SDAI, Standard Data Access Specification Interface* [6]. Programming language interfaces for STEP data is also specified in for example Part 23 [7] for C++. Having all these standardized methods for accessing STEP data, simplifies the creation of STEP based tools and software, and allowing these to have a unified understanding of the data.

The standard is composed of a collection of parts, some of which covers the use of the standard, such as the parts mentioned above, while most parts covers the different areas supported by the standard, i.e. geometric representations, FEA, mathematical descriptions, product structures etc. Each of these are holding the definition of entities with their attributes and inheritance, which in an Object-Oriented Programming (OOP) view are essentially objects or classes.

2.2. Application Protocols

Application Protocols (AP's) collects different parts with the intention that each AP represent a domain or phase in the product life-cycle. The AP's also holds information on how to combine the different parts, which in a sense, specializes the parts for that particular APs purpose. A certain application or software supporting STEP, defines which AP it covers and in this way each AP could be considered different formats. While in reality, this is just the STEP format applied to different domains. This means that all STEP files are based on the same type of data structure, and have the same high level definitions, allowing SDM and PDM tools to easily process files from different domains (i.e. CAD, FEA, manufacturing). STEP has also several managements concepts (such as requirements, assignments, classifications, roles, activities. . .) embedded within certain parts, which can be directly integrated within a Data Management tool.

Since the release of STEP, AP203 [8] and AP214 [9] have been the most successful, and are now widely used as an exchange formats between CAD and PLM software.

In Figure 3 we see how entities with inheritance and attributes are part of a *Part* which again is related to (*used by*) an Application Protocol. The example shows two high level entities, **representation_item** and **representation** which belongs to (*are defined by*) Part 43 [10]. This part has multiple very generic entities and is used by all APs. Each entity may be a parent (*supertype*) of multiple entities which are defined in other Parts that further specializes them. For simplicity the figure shows a single inheritance branch **(representation_item** and **representation** actually have many child (*subtypes*) entities defined in other parts). AP209, which covers the domain

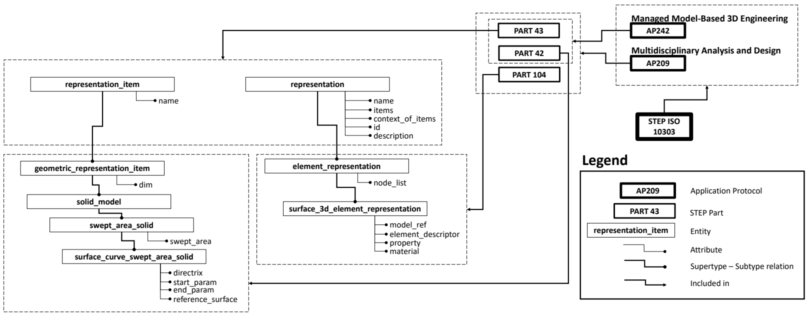

Figure 3: Example of how entities are included in Parts which again are included in Application Protocols.

Analysis and Design, includes Part 42 [11], Part 43 and Part 104 [12], while AP242 [13], intended as a CAD format, includes only Part 42 and 43. (Both AP209 and AP209 include several other Parts which are not shown in the figure.) Figure 4 shows how **representation, element_representation** and **surface_3d_element_representation** are defined in the standard AP documents in the EXPRESS language.

```
ENTITY representation;
  name : label;
  items : SET[1:?] OF representation_item;
  context_of_items : representation_context;
DERIVE
  id : identifier := get_id_value (SELF);
  description : text := get_description_value (SELF);
WHERE
  WR1: SIZEOF (USEDIN (SELF, 'BASIC_ATTRIBUTE_SCHEMA.' + 'ID_ATTRIBUTE.IDENTIFIED_ITEM')) <= 1;
  WR2: SIZEOF (USEDIN (SELF, 'BASIC_ATTRIBUTE_SCHEMA.' + 'DESCRIPTION_ATTRIBUTE.DESCRIBED_ITEM')) <= 1;
END_ENTITY;


ENTITY surface_3d_element_representation
    SUBTYPE OF ( element_representation );
      model_ref            : fea_model_3d;
      element_descriptor   : surface_3d_element_descriptor;
      property             : surface_element_property;
      material             : element_material;
    UNIQUE
      url : model_ref, SELF\representation.name;
    WHERE
      wr1:
      wr2:
      wr3:
      fu1:
END_ENTITY;
```

```
ENTITY element_representation
    SUPERTYPE OF (
        ONEOF (
            volume_3d_element_representation,
            axisymmetric_volume_2d_element_representation,
            plane_volume_2d_element_representation,
            surface_3d_element_representation,
            axisymmetric_surface_2d_element_representation,
            plane_surface_2d_element_representation,
            curve_3d_element_representation,
            axisymmetric_curve_2d_element_representation,
            plane_curve_2d_element_representation,
            point_element_representation,
            directionally_explicit_element_representation,
            explicit_element_representation,
            substructure_element_representation ) )
    SUBTYPE OF ( representation );
      node_list  : LIST [1 : ?] OF node_representation;
    WHERE
      wr1:
    END_ENTITY;
```

Figure 4: Extract of the content in the AP209 document. (Some fields are left out for simplicity.)

A STEP file or database holds a population of such entities, and can be interpreted by an application if the AP schema is implemented (an extract of such a STEP file is included in Section 3.1).

2.3. Application Protocol 209

AP209 (edition 2) is called *Multidisciplinary analysis and design*, and is primarily meant as an exchange format between simulation solvers. An overview of the data that it can represent is shown in Figure 5.

As seen, it covers the representation of composites, analysis definition and analysis results (FEM and CFD), design (CAD) and more. Note that an important aspect of AP209 is the capability of not only representing Analysis and Design *separately*, but also allowing the interconnection between both domains (such as relations between mesh and loads, and the design geometry).

With the current state of AP209, only linear static and frequency analyses are completely supported. But as noted by [14], the standard was designed to

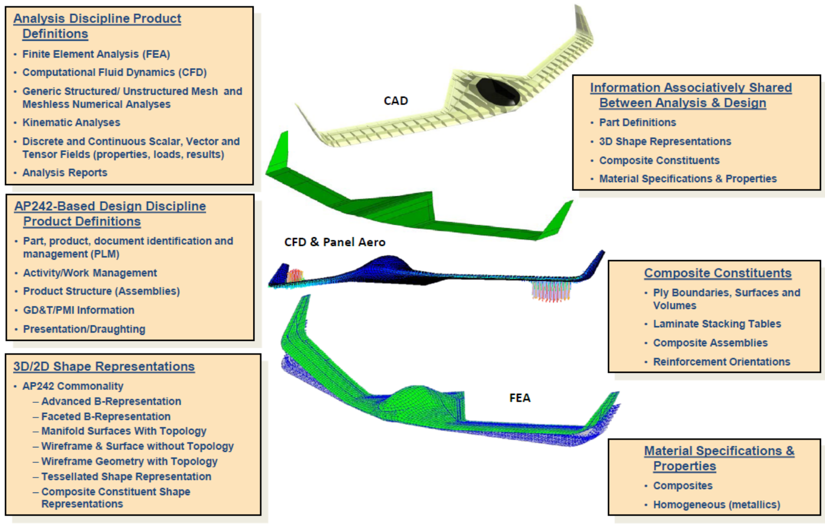

Figure 5: Data which is supported by AP209 [15]

easily be updated to support non-linear analysis, as it already covers roughly 90% of this problem.

Multiple implementations of the standard have been performed, specially regarding the exchange of composite data for design, analysis and manufacturing purposes. Some of these are summarized in [16].

Ongoing work and implementations of the standard are led by the LOTAR EAS: Engineering Analysis & Simulation Workgroup [17] which is co-led by Airbus and Boeing.

As described earlier, an AP is composed of parts, which are principally a set of entities (objects). The parts can be used in different APs, therefore many entities are very general in nature. They can be viewed as *building-blocks* for representing certain items or concepts. As we will see in the next section, the *building-blocks* or entities, can be used, not only to represent FEA and CAD, but also information concerning structural testing. This holds as long as an agreed upon structure is defined. The next sections describes the main outlines of a proposed structure for using AP209 to represent the additional data. No extensions of the AP209 standard are suggested, but as will be discussed, future work may include it.

3. The Higher Structure of a Combined Structural & FEA STEP Model

3.1. Overview

This subsection is dedicated to explain some general concepts to better understand the subsequent sections.

In STEP every represented high level item is represented as a **product**. By high level item we mean, *an Analysis, a CAD assembly, a CAD part, a manufactured part* etc. Items that wouldn't be considered a **product** could be a FEM element, a color definition, a property, a geometric shape etc.

The **product** entity has certain mandatory attributes and related entities. For example, a **product** entity must have a version, a context and a category classifying the **product**. The definition of the **product**, i.e. the data making up the analysis or CAD model, relates to a **product_definition** entity, which again relates to a **product_definition_formation** where the version is defined. The version links everything back to the specific **product**.

By enforcing these rules on the data, it ensures that an application (such as an SDM tool) can understand what is being imported before handling the complete model. The high level entities also acts as a way of organizing multiple STEP populations within the same system. Multiple STEP data-sets can reside within a database without being constrained as *files*. The constituents of each model or data set, are then identified by their high level entities.

An extract of an STEP P21 file (ASCII) showing some of these high level entities can be seen in Figure 6. As shown, each instance of an entity has an identifier followed by the entity name. Enclosed by parentheses, and comma separated, are the attributes. When an entity is an attribute of another entity, it is referenced by this identifier. Throughout the paper, graphical instantiation of this structure will be used (not to be confused with EXPRESS-G which is the standardized graphical representation of the EXPRESS language defined in Part 11). Instances are represented by boxes with the entity name in capital. Arrows show the referencing of an instance from another instance. A string beside an arrow specifies the name of the attribute. In some cases STEP entity structures can be quite complex. If an entity box has its text in italic it represents a simplification of a more complex structure, or a shortening of the entity name. Bold text beside an entity box is an additional description for the reader to better relate the graphics to the context.

```
(...)
#42= PRODUCT('1234','winglet analysis','',(#44));
#53= PRODUCT_DEFINITION_FORMATION('v.2','',#42);
#59= PRODUCT_DEFINITION('winglet analysis fine mesh',$,#53,#60);
#70= PRODUCT_RELATED_PRODUCT_CATEGORY('linear_static_analysis',$,(#42));
#44= PRODUCT_CONTEXT('design_context',#1,'design_context');
(...)
```

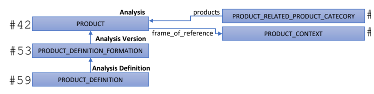

Figure 6: Left: Extract of a STEP P21 file. Right: Graphical representation used in this paper.

It is also vital to understand that these high level entities can hold names and descriptions. In fact, many of the low level entities such as nodes, elements and loads have ways of holding additional meta-data. This is often disregarded, but depending on how the post- and pre-processors of the STEP model implement these, there are good possibilities of holding information describing intentions and comments regarding the creation of the model.

3.2. The Analysis Model

The data structure of an Analysis STEP AP209 data set is well described in the Recommended Practices for AP209 [18].

Only few details of the data structure will be discussed here, but with focus on the parts that will eventually get related with the structural testing data.

In AP209 the analysis is represented by the entity **product**, which as described in the previous section, has a version and a definition. The entity **shape_definition_representation** represents the shape which relates to the data making up the CAD model on which the analysis based on. More importantly, the shape can be defined by a **fea_model_definition**, which is the link to the nodes and elements making up the mesh **shape**. The whole analysis definition is then built up of entities linked to each other giving meaning to the data.

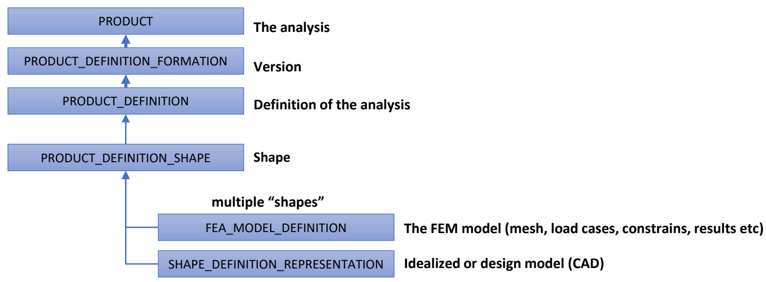

Figure 7: High level entities in the Analysis Model

Analysis load cases in AP209 are represented by **control_linear_static_analysis_step** entities that relates different **states**. Each **state** are collectors of loads, constrains or other nested **states**. In section 6.3 we will see how the model relates these to the actual test cases.

3.3. The Structural Test Model

With Figure 7 showing the high level structure of the FEA model, we introduce a similar structure representing the object that is being tested. The **product** in that case is the tested part which also has a version, definition and shape. The two versions are linked via relationship entities and the shape might be linked to the same Nominal Design data set related to the Analysis, unless the part being tested has its own Design version.

Another **product** represents all the result data from every tests that relate to load cases in the Analysis Model. This product has a version as well, and multiple definitions which each represents the results from individual structural tests.

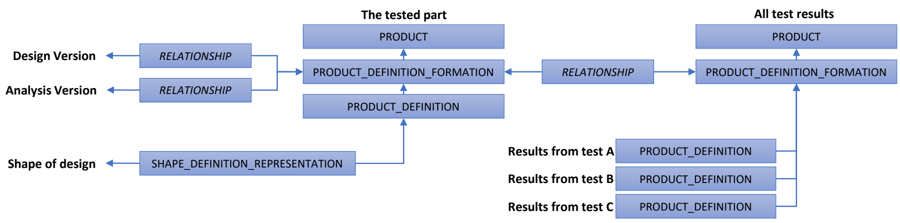

Figure 8: Relations between FEA, Design, Tested Part and Test Results. In this case, there are three individual tests.

The sensors and the tests are also represented with entities. The sensors are related to the tested part **product**, while the tests relate the sensors and the test results. Sensors and test representations are further discussed in section 4 and 5 respectively.

4. Sensors

There exists a wide variety of sensors such as strain gages, accelerometers, vibration sensors, displacements sensors and more. Many of these are assemblies of multiple sensors, for example a triaxial gage is just three sensors assembled together with specified angles between each.

To generically cover all types of sensors we represent each sensor as an assembly of multiple sensor components. Each sensor assembly and component have their own **product** with a definition holding properties.

To not have repetitive information, we introduce a **product** representing the type of sensors used. As an example, the specification of a tri-axial strain gage of a specific type, brand and model would be represented by one sensor type **product**. For each sensor of this type, mounted on the tested part, there exists a sensor assembly **product** having three individual sensor component **product**s.

Each of the representations, sensor, sensor component and sensor type, are able to hold properties.

Properties that are related to the sensor assembly:

1. **Position:** the position based on the coordinate system of the FE model
2. **Orientation:** the orientation of the sensor in the FE model
3. **Reference Element:** the element (or a set of elements) in the analysis model on which the sensor is placed
4. **Element Face ID:** an ID (or a set of IDs) representing the face of the elements on which the sensor is placed

Properties that are related to the sensor components:

1. **Direction:** the direction of the sensor component in the FE model
2. **ID:** An ID to number the sensor component

The complete set of properties to relate to the sensor type is still under ongoing work. However suggested properties are:

1. **Sensor Type:** Strain gage / Accelerometer / Displacement Sensor
2. **Sensor Description:** Further description of the sensor type
3. **Manufacturer:** The name of the manufacturer
4. **Model name:** The model name of the sensor type
5. **Number of sensor components:** a number specifying the number of sensor components
6. **Angles:** For strain gages, a set of angles defining the angles between each sensor components

All these properties originates from different input sources, but are now contained within the same model and therefor facilitates the storing, organizing and sharing of the complete data set. Additional properties are also planned to be added in future work to hold a complete description of the sensors.

Properties that relates to the sensors, but are test case dependent are defined differently. For example filtering techniques performed on the data by the DAQ System (Data Acquisition system) are not necessarily the same for every usage of the sensor. These properties are related directly to the result data which we cover in the next section.

An example of how the sensor data structure can look in a STEP model is shown in Figure 9. Note that the *reference element* property of the sensor assembly is a direct link to the actual element in the FE model, and in such, holds the traceability between the two in the same model.

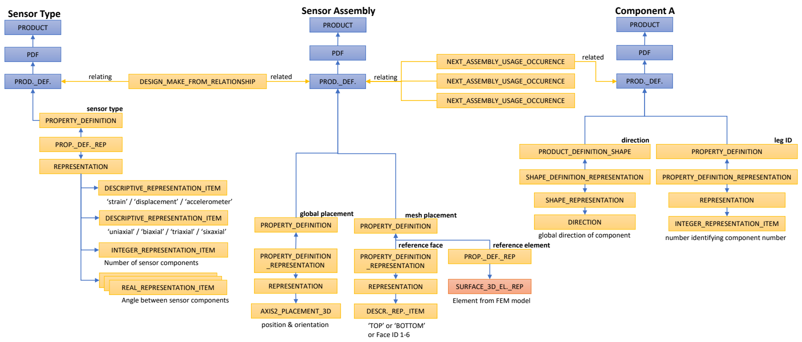

Figure 9: Example of data structure for sensor with three sensor components (only one is shown).

5. Structural Tests

In STEP the generic entity **action** will be used to represent *the action of performing a structural test*. The items used in the test are assigned to this entity by an **applied_action_assignment**, which in turn assigns each item a role of *input* or *output* to the **action**. The *input* items to the test are the sensors and the tested part, while the *output* is the sensor result data for that particular test.

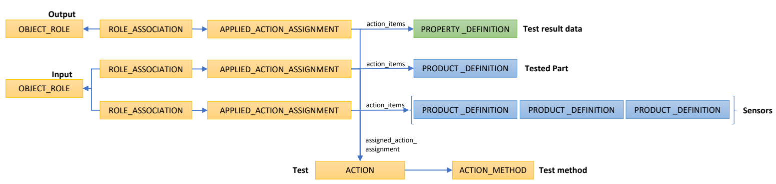

Figure 10: Data structure for a test performed on a part with three sensors, resulting in a certain test result.

The **action_method** is the link to the description of how the test was performed. This could be in the form of a reference to a certain external document, or in a more structured form with STEP entities. The work related to this is ongoing and is not presented in this paper.

6. Structural Test Data

6.1. Structural Test Results

The original test result data coming from the test equipment software will generally be in the form of Excel files or other proprietary format. The data can be extremely large, and it is generally expected that it has been filtrated before converted or imported to this STEP model.

The storing of test data in STEP is based on Part 50 name [19]. Essentially the entity **listed_real_data** holds the values, but the complexity of the STEP standard requires multiple other entities to define what sort of data is held within it. The details around this is outside the scope of this paper, and we define for simplicity the entity ***data_array*** to represent an array of values. The information within this *entity* is an array of result data corresponding to the data output from *one* sensor component for *one* test case, the type of data (i.e. strain or displacement) and the size of the data. The **data_array** relates to a **property_definition** allowing us to use the result data as a property to other entities.

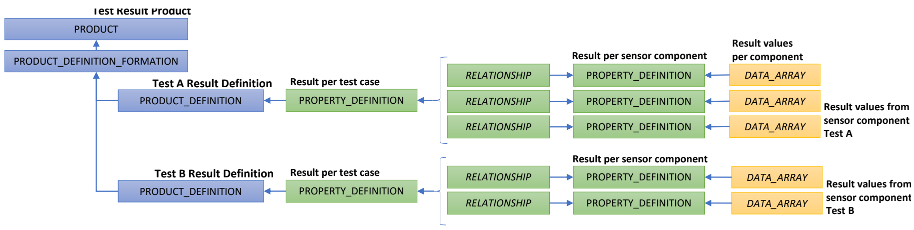

Figure 11: Example of data structure showing the relation between the sensor result values and the output result data **product**

As seen in Figure 11, relationships are used to group the results from each sensor component to **property_definition**s corresponding to the whole test case. These are pointing to the corresponding **product_definition** of the output data. They are the same **property_definition**s set as output of the **action** in Figure 10 in the previous section.

The sensor component's **product_definitions** are attributes of the **property_definition**s related to the ***data_array***s holding their result data as seen in Figure 12.

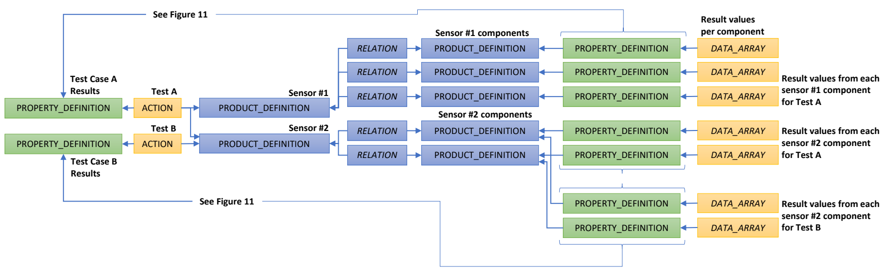

Figure 12: Data structure showing relation between sensor results, sensors and tests. Here we have two tests and 2 sensors. One of the sensors (sensor 2) is used in both tests.

6.2. Structural Test Result Properties

In section 5 sensor properties were presented. We will now look at properties that are related to sensors, but that may vary for each test case. They are essentially properties originating from the DAQ equipment and software used for retrieving test data.

This applies to properties such as:

1. The channel ID from the test Equipment
2. Filtering Techniques
3. Sample Rate
4. Scaling
5. Gage Factor

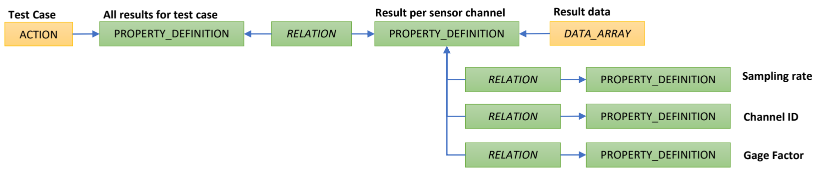

Figure 13: The Data values references the result property of sensor component, which is also referenced by properties that are unique for this sensor channel and test case.

6.3. Structural Test Relation to Analysis Load Case

In section 4.2 we shortly presented how load cases are defined in an Analysis Model. The test cases that are representing the load cases are related via the **action_view_relationship** entity. It is meant as an entity relating a *discretized model* (the load case) and an *idealized action* (the test **action** that is being idealized).

Figure 14: Load cases from the Analysis data set are related to the Test Case of the Structural Test data set via the action_view_relationship.

7. Method

As AP209 is primarily meant to store and share simulation data, structural testing was not part of its original intention. The first question when investigating the use of this standard for another domain, was if the standard itself required an extension? (Does additional entities and types need to be added to the AP209 schema?).

To answer the above, a careful examination of the AP209 schema was performed to get a detailed overview of which data the data model *can* represent.

A good understanding of the whole schema was acquired after a converter to translate FEM analyses in Nastran format to AP209 was developed.

The next step was to define which type of data from the structural testing domain that needed to be included, which was then mapped to AP209 elements (entities, attributes, data types etc.). Careful attention was given to how to relate this domain to the analysis elements.

As noted in the previous chapters, many of the STEP elements are very generic, and can be used to represent a wide variety of data. Of course, the pre- and post-processors need to know how to interpret it. An example is the entity **action**, a very generic item, but with certain attributes to specify what the *action* represents (here, used to define the test case). This is where documents such as Recommended Practices are needed. The standard itself contains the formal description of every STEP elements, while the Recommended Practices describe *how* it is intended to be used and implemented in applications. Such a document is currently being developed to formally describe all the details behind this paper. We concluded with that no extension is needed at the moment. Possible extensions of the standard, by introducing new subtypes of entities, as well as type enumeration specifically for the domain of structural testing, could be proposed at a later stage.

After the mapping was defined, another converter was created. The input to it being the results from structural tests in .csv format as well as certain input files defining the sensors and test cases. The converter directly creates STEP data into an AP209 database (using Jotne's tools EDMS [21] and Open-SimDM [22]). The analysis related to the test case is already residing within that database, allowing the converter to read from it and creating direct links between the new structural test data population and the analysis.

As discussed in the introduction, a use case for this would be Simulation Data Management. An example case is being performed to validate the usage of the model. An airplane winglet has been designed, simulated, manufactured and tested to imitate the different phases of product development. The data of each phase have then been either exported or converted to STEP AP209 and imported to the OpenSimDM application having AP209 as its database schema. This tool is now being further developed to let the user access and manage the data.

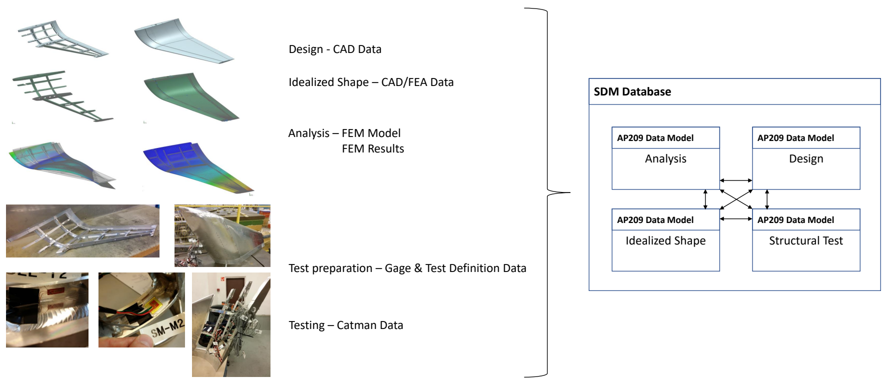

Figure 15: Different AP209 data sets imported to the SDM tool.

8. Conclusion and Future Work

We have now shown how the structural test related data can be represented in a data model, and how the relevant pieces of data can be connected to an Analysis and its results.

The purpose of a SDM software is to manage and have an overview over all the simulation related information and have quick access to specific data. Having all the different aspects of the product in the same format in a database enables exactly this. If implemented correctly, it enables the user to do this without having to open files in their original software.

Accessing information can easily be done by executing simple query function on the data sets. Examples of queries could be, retrieving the type of sensor used, the location of it on the mesh, getting the result data from a particular sensor for a particular test, and comparing it to the corresponding analysis results. Different views on the AP209 population can be implemented, such as an overview over all sensors that were used in a specific test, and their maximum result in both analysis and testing.

The whole management of data can also be specified via AP209 data. This includes defining who created a model, who accepted it, deadlines, tasks to be performed etc. These specifications can be directly linked to specific entities within the data sets of the analysis and structural test.

The complete data set can also be exported to ASCII or binary STEP files that are compliant with the LOTAR (Long Term Archiving and Retrieval) specifications [20]. This enables to share the files to other systems conforming to the STEP standard, and archiving the project with all data still being traceable.

---

[1] ISO 10303-1:1994 Industrial automation systems and integration – Product data representation and exchange – Part 1: Overview and fundamental principles. Geneva (Switzerland): International Organization for Standardization (ISO), 1994.

[2] ISO 10303-209:2014 Industrial automation systems and integration – Product data representation and exchange – Part 209: Application protocol: Multidisciplinary analysis and design. Geneva (Switzerland): International Organization for Standardization (ISO), 2014.

[3] FAA-21-48 - Using Electronic Modeling Systems as Primary Type Design Data, U.S. Department of Transportation, Federal Aviation Administration, 2010.

[4] ISO 10303-11:2004 Industrial automation systems and integration – Product data representation and exchange – Part 11: Description methods: The EXPRESS language reference manual. Geneva (Switzerland): International Organization for Standardization (ISO), 2004.

[5] ISO 10303-21:2016 Industrial automation systems and integration – Product data representation and exchange – Part 21: Implementation methods: Clear text encoding of the exchange structure. Geneva (Switzerland): International Organization for Standardization (ISO), 2016.

[6] ISO 10303-22:1998 Industrial automation systems and integration – Product data representation and exchange – Part 22: Implementation methods: Standard data access interface. Geneva (Switzerland): International Organization for Standardization (ISO), 1998.

[7] ISO 10303-23:2000 Industrial automation systems and integration – Product data representation and exchange – Part 23: Implementation methods: C++ language binding to the standard data access interface. Geneva (Switzerland): International Organization for Standardization (ISO), 2000.

[8] ISO 10303-203:1994 Industrial automation systems and integration – Product data representation and exchange – Part 203: Application protocol: Configuration controlled 3D designs of mechanical parts and assemblies. Geneva (Switzerland): International Organization for Standardization (ISO), 1994.

[9] ISO 10303-214:2001 Industrial automation systems and integration – Product data representation and exchange – Part 214: Application protocol: Core data for automotive mechanical design processes. Geneva (Switzerland): International Organization for Standardization (ISO), 2001.

[10] ISO 10303-43:2011 Industrial automation systems and integration – Product data representation and exchange – Part 43: Integrated generic resource: Representation structures. Geneva (Switzerland): International Organization for Standardization (ISO), 2011.

[11] ISO 10303-42:2014 Industrial automation systems and integration – Product data representation and exchange – Part 42: Integrated generic resource: Geometric and topological representation. Geneva (Switzerland): International Organization for Standardization (ISO), 2014.

[12] ISO 10303-104:2000 Industrial automation systems and integration – Product data representation and exchange – Part 104: Integrated application resource: Finite element analysis. Geneva (Switzerland): International Organization for Standardization (ISO), 2000.

[13] ISO 10303-242:2014 Industrial automation systems and integration – Product data representation and exchange – Part 242: Application protocol: Managed model-based 3D engineering. Geneva (Switzerland): International Organization for Standardization (ISO), 2014.

[14] Keith A. Hunten. CAD/FEA Integration With STEP AP209 Technology and Implementation. MSC Aerospace Users Conference Proceedings, September 1997, http://web.mscsoftware.com/support/library/conf/auc97/p01297.pdf [accessed 18 january 2018].

[15] PDES Inc., ISO 10303-209 "Multidisciplinary Analysis and Design", http://www.ap209.org/introduction (accessed on 16 January 2018).

[16] Keith A. Hunten, Allison Barnard Feeney, Vijay Srinivasan. Recent advances in sharing standardized STEP composite structure design and manufacturing information. Computer-Aided Design, 2013;45: 1215-1221.

[17] LOTAR (Long Term Archiving and Retrieval). LOTAR EAS: Engingeering Analysis & Simulation Workgroup, http://www.lotar-international.org/lotar-workgroups/engineering-analysis-simulation/scope-activities.html (accessed on 19 January 2018).

[18] Keith A. Hunten, Recommended Practices for AP209ed2 10303-209:2014, CAx Implementor Forum (2016).

[19] ISO 10303-50:2002 Industrial automation systems and integration – Product data representation and exchange – Part 50: Integrated generic resource: Mathematical constructs. Geneva (Switzerland): International Organization for Standardization (ISO), 2002.

[20] NAS9300-001, Long Term Archiving and Retrieval of digital technical product documentation such as 3D, CAD and PDM data : part 101: Structure, Aerospace Industries Association of America Inc., 2017.

[21] EDMS [STEP Data Manager software], Jotne EPM Technology AS

[22] EDMopenSimDM [Simulation Data Manager software], Jotne EPM Technology AS