IMF Model Builder Concept Paper

Background

Resulting from the READI JIP, an asset information modelling framework has been developed (IMF) which provides a way to create, verify, share, and use, a standardised virtual representation of a facility or asset. This promises to solve a number of challenges related to data-centric sharing of information, as well as building data-rich models that can be a foundation for Digital Twin applications.

Product Vision

The vision for the IMF Model Builder (MB) is to be an efficient **means for paving the way for industrial adoption** of the IMF, by providing a user-friendly authoring tool that demonstrates how asset information models can be built and shared. To achieve this, involvement from industry stakeholders as part of the development shall be ensured:

* When developing and demonstrating *how to build a model*, industry end user feedback shall be incorporated into the functionality and user interface.
* When developing and testing *the sharing of asset information models* between different IT-systems, alignment with users of applicable industry standards shall be a prime objective.

Product Scope

The Model Builder shall be developed towards two scope milestones: 1) a Model Builder Prototype, and 2) a Model Builder MVP. Prior to a working Model Builder Prototype a Demo shall be provided. The Demo shall visualise the features of the Prototype but need not be fully functional.  
As per version 5 of this Concept Paper the product scope has been expanded to include a Basic Type Editor (BTE).

Prototype:

* The Model Builder Prototype shall incorporate end user functionality and interface to support the authoring of a high level (low granularity) asset model as visualised by creating Process Block Diagrams of:
  + Level 0: the main facility with its connections
  + Level 1: the process functions (‘boxes’) of the facility, including ‘Main Process’, and their connections,
  + Level 2 partial: the sub-functions (‘boxes’) that are part of ‘Main Process’, and their connections.
* Modelling shall be possible by configuring **Function System Blocks** (functions) and **Transports** (streams) to connect between them.
* It shall be possible to pivot between – or view simultaneously - Block Diagram view and Tree Structure view while editing the model.
* Only the Process discipline perspective shall be implemented, and for Transports only Fluid transports shall be included, meaning that the Electricity, Information, and Force transports shall be excluded.
* Only the **Functional** **Aspect** and **Location Aspect** shall be implemented, meaning that other aspects such as and **Product,** and **Installed** shall not be included.
* The Model Builder Prototype shall produce a set of data that represents the asset information model, in an open format suitable for exchange across industry.
* The model data exchange format may also contain the *essential* data about visual layout that is needed for the receiving end to re-establish relative location of graphical elements, and the references to type of elements such that visualisation will be similar to that at the source.
* It is acceptable for the prototype to be single-user.
* It is acceptable for the prototype to have limited functionality for roll-back of edits
* It is acceptable for the prototype to have no version control of the model data.

BTE:

* The Basic Type Editor shall incorporate end user functionality and interface to support a continuous expansion and enhancement of the library of **Types** that the Model Builder user depends on for populating models with **Function System Blocks** (functions), **Transports** (streams), and **Interfaces**
* The Basic Type Editor shall provide a workspace where a Type can be built in any chosen **Aspect** (i.e. Function, Product or Location), further by selecting Function System Block, Transport or Interfaces, and then proceed to further detailing the Type by choosing from libraries or lists of predefined categories and elements:
  + A list of **RDS Categories** shall be implemented. These are selected from to create the basis for assigning Attributes and populating the Type with Terminals. The source of RDS Categories shall be the 81346 O& G standard.
    - Example: ‘=KE Pressure Increase’
  + A library of **Terminal Types** shall be implemented. These are used when populating connection points, aka Terminals. The source of Terminal Types shall be the IMF document R1, para 5.4.2.
    - Example: Material/Fluid  
      (later version may expand details to e.g. Material/Fluid/Gas/Export Gas)
  + Also, a library of **Attributes** shall be implemented. These are used to specify the attributes that shall be part of the type definition. The source of these attributes shall be relevant existing Equinor library.
    - Example: Nominal Flow Rate [Sm3/h]
* How a user can create and configure a new Type shall be governed by underlying rules. The BTE shall only implement basic rules, but this shall be scalable:
  + A **Transport** can only have 1 input and 1 output **Terminal**, and these must be of same **Type** (e.g. Material/Fluid==>Material/Fluid)
  + An **Interface** can only have 1 input and 1 output **Terminal**, and these must be of same **Type** (e.g. Energy/Mechanical==>Energy/Mechanical)
  + Any **Type** must have a minimum of 2 terminals
  + Any **Type** must have a minimum of **Attributes**, including Name and ID, and there shall be a mechanism for enforcing required attributes based on what Type it is
* Thus, a **Type** shall be created or edited by making the above selections, assigning a name and ID, setting appropriate work status, and then storing as a new Type in the Type Library.

The Basic Type Editor shall be executable from within the Model Builder or as standalone.

MVP:

* The Model Builder MVP shall incorporate end user functionality and interface to support the authoring of asset model as visualised by creating diagrams of:
  + Level 0: the main facility with its connections
  + Level 1: the main functions (‘boxes’) of the facility and their connections,
  + Level 2: the sub-functions (‘boxes’) that are part of main functions, and their connections,
  + Level n: the sub-functions of sub-functions, until the hierarchy depth where inter-aspect relations can be established, e.g. motor **Function** fulfilled by motor **Product.**
* Modelling shall be possible by configuring **Function System Blocks** (functions) and **Transports** (streams) or **Interfaces** to connect between them.
* Graphical symbols shall represent various Types of **Function System Blocks**, such as pump symbol to represent pump function or product, etc.
* It shall be possible to pivot between – or view simultaneously - Block Diagram view and Tree Structure view while editing the model.
* The Process discipline and the Electrical discipline perspective shall be implemented – including their area layout - meaning that Fluid and Electricity **Transports** and **Interfaces** shall be included, whereas Information, and Force **Transports** and **Interfaces** shall be excluded.
* The **Functional, Location,** and **Product** aspects shall be implemented, and the MVP Model Builder shall be built to be extended with further aspects.
* The Model Builder Prototype shall produce a set of data that represents the model, in an open format suitable for exchange across industry, including exchange by means of I4.0 AAS.
* The model data exchange format shall contain the *essential* data about visual layout that is needed for the receiving end to re-establish relative location of graphical elements, and the references to type of elements such that visualisation will be similar to that at the source.
* The model data exchange format may also contain the *enhanced* data about visual layout, such as information about colour, lines, fills, forms, and other visual characteristics, enabling a more accurate reproduction of the user experience at the receiving end.
* Multi-user functionality shall be supported.
* Full functionality for roll-back of edits shall be supported.
* Version control of the model data shall be supported.

Product Targets

The main target of the product development is industrial adoption of the IMF to make it possible to begin realising value, both in Krafla project and across the Oil & Gas industry. To achieve this, the development of the product shall result in a proven model building functionality, built in a modular fashion and available as Open Source such that it can be freely incorporated in other engineering tools, as well as be further developed with enhanced functionality such as augmented engineering functions. The following are the main achievements towards the main target:

1. Aibel and equinor begin trials of exchanging structured design information for the Krafla project, using the Model Builder prototype.
2. The NOAKA collaboration forms a Model Builder User Group that provides user needs into the further development of the Model Builder towards an MVP.
3. Aibel and equinor formally exchange structured design information for the Krafla project, using the Model Builder MVP.

Furthermore, the product development shall serve to drive the definition and alignment on an open asset information model data format, fully supporting interoperability across the industry. A clear preference is that this is based on the i4.0 Asset Administration Shell format.

Project Timeline and Milestones

Demo: April 7th

Prototype: May 26th

MVP: June 23rd

Methods and Techniques

The detailed specification of the Model Builder is not known in advance but shall be developed as part of the project execution. Frequent iterations between users, designers, front-end, and back-end developers shall be employed to navigate towards a successful end result, whilst avoiding scope creep. The range of users who are involved shall begin small and shall expand as the product matures. Testing shall be the main means of approval. The product and supporting documentation shall be Open Source.

Roles and Responsibilities

Developer team

* Shall develop and test the product, including design, front-end and back-end.
* Shall deploy the product in Krafla project
* Shall support deployment in the NOAKA cooperation

Product Owner or PO proxy

* Define product vision and goals
* Prioritize scope, cost, time, and resources for a successful delivery
* Communicate with stakeholders and represent their needs