Version	Date	Remark
1.0	01.06.2020	First version of OSP-MDO

Contents

1 INTRODUCTION

The intention with the OSP Maritime Domain Ontology[1] (MDO) is to provide a framework for assigning properties to a model, such as the *type of physical system* or *the phenomena the model represents*, and *the type of capabilities of the model.* Examples of model types range over hull, electric motor etc. and environment types of models like a wave-wind-and-current model.

A framework such as this can be used for communicating model properties across organizations, in the context of model sharing. Maritime domain model *developers* can communicate model *capabilities* in a standardized way by tagging models with MDO terms. Analogously, a company *requesting* models can communicate model *requirements* in a standardized way. In a future centralized *Maritime Domain Simulation Model Library*, populated with a variety of pre-made models, a user querying the library with key search criteria, will subsequently receive a list of models that match the criteria.

For physical entities like an electric motor, a propeller or a rudder, a rich and well-used component vocabulary already exists, namely the Vessel Information Structures (VIS), developed, maintained and used by DNV GL. VIS is also used in ISO 19848 [1]. For models representing vessel equipment, the OSP group believes that MDO should leverage this existing work, making use of the VIS vocabulary for adding equipment tags to a model. VIS is openly available through DNV GL's website [2].

Besides the above-mentioned equipment tags there is a need for attributes describing other simulation capabilities, in addition to more general high-level information such as information on model vendor, maker, model make, etc.

Within the scope of the OSP project, we have developed a first version of MDO. It should be noted that this version does not attempt to encompass all possible maritime equipment, along with all meaningful simulation properties. This version of MDO should rather be considered as a first attempt to map out a structure that can be used as a foundation, and which with time will allow MDO to grow to include more and more content, to the point where it includes a large portion of the maritime domain.

2 USING VIS TERMINOLOGY FOR MODEL ANNOTATIONS

DNV GL maintains VIS, a data model for describing offshore vessels. VIS is used to create an information model of a vessel which describes and holds data about the vessel throughout its lifetime. Since 2005, DNV GL has operated VIS in the Nauticus Production System (NPS) which to date holds over 20,000 information models of vessels constituting up to 30% of the entire offshore fleet worldwide.

The VIS specification provides a uniform framework that is used in DNV GL's classification work. The framework is used for structuring of governing documentation, and storage, retrieval and analysis of generic and vessel specific information.

The Generic product model (Gmod) is an essential part of VIS. It provides a generic description of a vessel that is used as a blueprint for creating more detailed product models. The Gmod consists of a hierarchy of functions, a library of equipment, and rules for assigning equipment items to functions.

[1] The term ontology is frequently used within this document; see [7] for an introduction to the concept of ontologies, and [6] for the current state and use of ontologies, as well as future directions.

All Gmod items have a unique code and name. The coding is based on the Universal Decimal Classification (UDC) numbering system giving rise to an inherent hierarchy. A Gmod provides the basis for a vessel specific Product model (Pmod) with more detailed information about a particular vessel.

3 MDO CURRENT STAT AND CONNECTION WITH VIS

3.1 CURRENT STATE - XML

The intention with MDO is to provide a framework for assigning properties to a model. The properties should describe both what type of equipment the model represents and what type of simulation capabilities it has.

The current specification of MDO is implemented as an XML schema – MDO.xsd. It presents users with a simple syntax that enables us to extend on the signal interface specification defined in OSP Interface Specification [3], adding meta-data tags to a model. The user can generate an MDO model instance by creating an XML file that matches the simulation model's properties, according to the schema defined MDO.xsd. See Appendix A for examples.

With regards to VIS, the intention is that users should associate equipment names and accompanying VIS codes with the simulation model described by the MDO instance. This can be achieved by adding <Subsystem> elements as described in Chapter 4. However, the user will have to manually lookup the appropriate VIS name/code in [2], as there is no validation of these fields within the schema.

The current format of VIS makes it non-trivial to use for direct validation in an XML context. To illustrate an alternative approach, parts of VIS have been experimentally transformed into an ontology to showcase its potential. This is further explained in Chapter 3.2.

Compared with a proper ontology language, XML can only specify rather simple relationships between terms. One can certainly think of relationships that would be difficult or even impossible to model using XML, due to its limitations, in which an ontology approach would be better suited. In the context of a facilitating platform - like the future "Maritime Domain Simulation Model Library" mentioned in Chapter 1 - the ontology approach would be a powerful fit. The idea of transforming MDO into a proper ontology has been briefly explored. The results seem promising and should be revisited in future work.

3.2 EXPERIMENTAL VIS ONTOLOGY

Though MDO currently is XML-based and not implemented as an ontology, the future ambition is to update MDO and implement it in OWL 2. We believe that VIS will be an important part of MDO in the future, therefore, to explore how the codes from the VIS Gmod equipment library could be made available as metadata, the equipment library has been replicated in the OWL 2 Web Ontology Language [4]. The VIS Gmod equipment library contains 14 disciplines that are further subdivided into equipment groups. For each equipment group, the Gmod provides a list of components together with a functional breakdown structure for the component. The equipment library is given in the form of complex Excel spreadsheets, one spreadsheet for each discipline [2]. The spreadsheets contain further information related to specific applications of VIS. A Python script has been created to convert the MDO-relevant part of the library into an OWL 2 ontology. This experimental ontology of the VIS Gmod equipment library (which in short could be called the *VIS ontology*) can in the future be imported into MDO, when MDO is updated and implemented in OWL 2.

3.3 FUTURE WORK

Going forward, a potential hybrid solution could be to allow users to create MDO instances using XML, and then implement a tool for validating the content of these instances against a corresponding ontology, with reasoning capabilities, as demonstrated in OSP Interface Specification [3].

The long-term ambition should nevertheless be to develop a proper Maritime Domain Ontology, with a rich vocabulary and an extensive range of defined concepts that will capture the semantics needed to properly describe the maritime domain in a simulation environment context. As a minimum, more model types, with associated features, should be established.

3.3.1 Example ontology queries

To illustrate why a proper ontology could be useful, consider the following example queries which could potentially be performed against some model library:

1. *"List all models of type <A> that I can connect to model <B>"*
2. *"List all models of type <A> which has parameter <P> within range <a,b>"*
3. *"List all models of type <A> from supplier <S>"*
4. *"List all models of type <A> with property or capability <X>"*
5. *"List all models of type Propulsor with capability to calculate ventilation losses"*
6. *"List all controllers of type <C> which can connect to model <A>"*

Some of these queries would be possible to execute against a relational database, but others would not, such as 1, as it would require reasoning. However, a system based on ontology principles would be able to achieve this if designed and implemented appropriately.

4 IMPLEMENTATION DETAILS

MDO.xsd defines the current metadata schema. The MDO-Model element is the intended root node to be used for model developers who are using MDO. Reference implementations, illustrating usage of MDO-Model, can be found in Appendix A. Figure 1 illustrates the structure of MDO-Model. It should be noted that the figure does not include all possible elements. There are currently three defined model types in MDO: <Electric Motor>, <Electric Power Plant> and <Hull>. Each model type has its own set of <Model Features> and <Model Principles>. Some of these are common across the model types, and some are specific for a given model type. One example of a common model feature is <Model Subsystems>, with child node(s) <Subsystem>. Another example of a common model feature is <Rated>. Definitions for the complete list of items follow below. Elements that are common across several model types, or similar, are documented only on first encounter. However, the list of child elements for any given item should be complete.

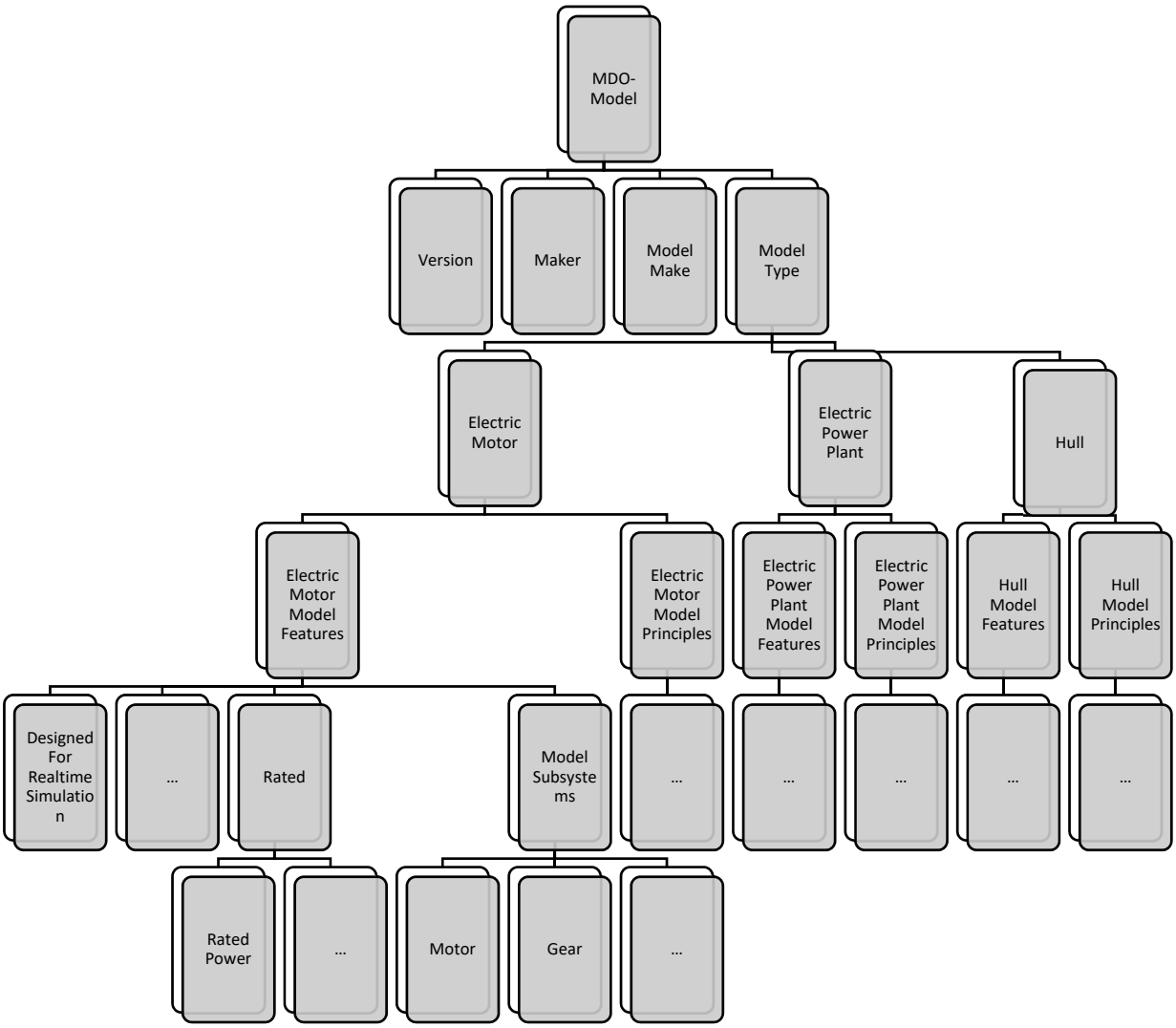

Figure 1 MDO structure

<MDO-Model>

This is the root element of the `MDO.xml` file that should be used for model implementations.

Children	Min	Max
<Version>	1	1
<Vendor>	0	1
<Maker>	0	1
<ModelMake>	0	1
<ModelType>	1	1

<Version>

This is the schema version. The element value is of type xsd:string.

<Vendor>

The model vendor. The element value is of type xsd:string.

<Maker>

The model maker. The element value is of type xsd:string.

<ModelMake>

The model and/or make that the simulation model represents. The element value is of type xsd:string.

<ModelType>

The type of equipment that the model represents. Currently supported types are ElectricMotor, ElectricPowerPlant and Hull.

Children	Min	Max
<ElectricMotor>	0	1
<ElectricPowerPlant>	0	1
<Hull>	0	1

<ElectricMotor>

Model of type Electric Motor.

Children	Min	Max
<ModelFeaturesForElectricMotor>	0	1
<ModelPrinciplesForElectricMotor>	0	1

<ModelFeaturesForElectricMotor>

The features included in the model.

Children	Min	Max

<DesignedForRealtimeSimulation>	0	1
<DesignedForControlSystemInterfacing>	0	1
<DesignedForTesting>	0	1
<FailureModesAvailable>	0	1
<Inertia>	0	1
<AC>	0	1
<DC>	0	1
<ReactivePower>	0	1
<InductionMotor>	0	1
<PermanentMagnetMotor>	0	1
<Rated>	0	unbounded
<ModelSubsystems>	0	1

<DesignedForRealtimeSimulation>

Model is designed for real time simulations. The element value is of type xsd:boolean.

<DesignedForControlSystemInterfacing>

Model is designed for interfacing with control systems. The element value is of type xsd:boolean.

<DesignedForTesting>

Model is designed for testing. For instance, the model includes failure modes. The element value is of type xsd:boolean.

<FailureModesAvailable>

The failure modes made available in the model.

Children	Min	Max
<FailureMode>	0	unbounded

<FailureMode>

Failure mode types.

Children	Min	Max
<Blackout>	0	unbounded
<FailToMax>	0	unbounded
<FailToMin>	0	unbounded
<ShortCircuit>	0	unbounded

<FailureModeDescription>

Optional description for element of type failure mode. The element value is of type xsd:string.

<Blackout>

Failure mode of type blackout.

Children	Min	Max
<FailureModeDescription>	0	unbounded

<FailToMin>
Failure mode of type fail-to-min.

Children	Min	Max
<FailureModeDescription>	0	unbounded

<FailToMax>
Failure mode of type fail-to-max.

Children	Min	Max
<FailureModeDescription>	0	unbounded

<ShortCircuit>
Failure mode of type short circuit.

Children	Min	Max
<FailureModeDescription>	0	unbounded

<Inertia>
Model simulates inertia. The element value is of type xsd:boolean.

<AC>
Model simulates AC. The element value is of type xsd:boolean.

<DC>
Model simulates DC. The element value is of type xsd:boolean.

<ReactivePower>
Model simulates Reactive Power. The element value is of type xsd:boolean.

<InductionMotor>
Motor model is of type induction motor. The element value is of type xsd:boolean.

<PermanentMagnetMotor>
Motor model is of type permanent magnet motor. The element value is of type xsd:boolean.

<Rated>
Used for specifying some rated quantity by either a single value or a range.

Children	Min	Max
<Quantity>	1	1
<Unit>	1	1
<PrefixMultiplier>	1	1

<ValueOrRange>	1	1

<Quantity>
The type of quantity. Select one of the specified enumerations.

Value	Type	Description
Power	xsd:enumeration	
Voltage	xsd:enumeration	
Current	xsd:enumeration	
Speed	xsd:enumeration	
Torque	xsd:enumeration	
Pressure	xsd:enumeration	
Temperature	xsd:enumeration	

<Unit>
The type of unit. Select one of the specified enumerations.

Value	Type	Description
Watt	xsd:enumeration	
Volt	xsd:enumeration	
Ampere	xsd:enumeration	
MetrePerSecond	xsd:enumeration	
NewtonMetre	xsd:enumeration	
Bar	xsd:enumeration	
Kelvin	xsd:enumeration	

<PrefixMultiplier>
The type of unit. Select one of the specified enumerations.

Value	Type	Description
yotta	xsd:enumeration	Denotes a factor of one septillion (short scale).
zetta	xsd:enumeration	Denotes a factor of one sextillion (short scale).
exa	xsd:enumeration	Denotes a factor of one quintillion (short scale).
peta	xsd:enumeration	Denotes a factor of one quadrillion (short scale).
tera	xsd:enumeration	Denotes a factor of one trillion (short scale).
giga	xsd:enumeration	Denotes a factor of one billion (short scale).
mega	xsd:enumeration	Denotes a factor of one million.
kilo	xsd:enumeration	Denotes a factor of one thousand.
one	xsd:enumeration	Denotes a factor of one.
milli	xsd:enumeration	Denotes a factor of one thousandth.
micro	xsd:enumeration	Denotes a factor of one millionth.

nano	xsd:enumeration	Denotes a factor of one billionth (short scale).
pico	xsd:enumeration	Denotes a factor of one trillionth (short scale).
femto	xsd:enumeration	Denotes a factor of one quadrillionth (short scale).
atto	xsd:enumeration	Denotes a factor of one quintillionth (short scale).
zepto	xsd:enumeration	Denotes a factor of one sextillionth (short scale).
yocto	xsd:enumeration	Denotes a factor of one septillionth (short scale).

<ValueOrRange>
Select single value or range depending on the need.

Children	Min	Max
<SingleValue>	0	1
<Range>	0	1

<SingleValue>
A single, rated value. The element value is of type xsd:double.

<Range>
A range of values defined by upper and lower limits.

Children	Min	Max
<LowerValue>	0	1
<UpperValue>	0	1

<LowerValue>
Lower limit rated value. The element value is of type xsd:double.

<UpperValue>
Upper limit rated value. The element value is of type xsd:double.

<ModelSubsystems>
The subsystems (lower level components) that the model includes. For instance, an ElectricMotor model could include subsystems that represent Motor and Shaft.

Children	Min	Max
<Subsystem>	0	unbounded

<Subsystem>
An individual subsystem that represent a model subsystem/ component. VISCode refers to the component's corresponding code, as defined in VIS. Note that there is currently no validation of this element against official VIS codes. It is therefore the user's responsibility to ensure that the code applied is valid. Subsystem is a recursive element, meaning that it can contain child elements that are also of type Subsystem.

Children	Min	Max

<Name>	1	1
<VISCode>	0	1
<Min>	0	1
<Max>	0	1
<Subsystem>	0	unbounded

<Name>
The name of the subsystem. The element value is of type xsd:string.

<VISCode>
Corresponding VIS Code for the subsystem/ component. This element value is of type xsd:string.

<Min>
The minimum allowed number of this subsystem type supported by the model. The element value is of type xsd:unsignedInt.

<Max>
The maximum allowed number of this subsystem type supported by the model. The element value is of type xsd:unsignedInt.

<ModelPrinciplesForElectricMotor>
Fundamental principles that the model is using.

Children	Min	Max
<ElectricalModellingPrinciple>	0	1

<ElectricalModellingPrinciple>

The model's underlying electrical modelling principle.

Children	Min	Max
<Phasor>	0	1
<Sinusoidal>	0	1

<Phasor>
Electrical modelling principle of type phasor/ quasi-static. The element value is of type xsd:boolean.

<Sinusoidal>
Electrical modelling principle of type sinusoidal. The element value is of type xsd:boolean.

<ElectricPowerPlant>
Model of type Electric Power Plant.

Children	Min	Max
<ModelFeaturesForElectricPowerPlant>	0	1
<ModelPrinciplesForElectricPowerPlant>	0	1

<ModelFeaturesForElectricPowerPlant>
The features included in the model.

Children	Min	Max
<DesignedForRealtimeSimulation>	0	1
<DesignedForControlSystemInterfacing>	0	1
<DesignedForTesting>	0	1
<Inertia>	0	1
<AC>	0	1
<DC>	0	1
<ReactivePower>	0	1
<Rated>	0	unbounded
<ModelSubsystems>	0	1

<ModelPrinciplesForElectricPowerPlant>
Fundamental principles that the model is using.

Children	Min	Max
<ElectricalModellingPrinciple>	0	1

<Hull>
Model of type Hull.

Children	Min	Max
<ModelFeaturesForHull>	0	1
<ModelPrinciplesForHull>	0	1

<ModelFeaturesForHull>
The features included in the model.

Children	Min	Max
<DesignedForRealtimeSimulation>	0	1
<DesignedForTesting>	0	1
<ExternalLoads>	0	1
<Bulb>	0	1
<ActiveAntirollTank>	0	1
<PassiveAntirollTank>	0	1
<BilgeKeel>	0	1
<Skeg>	0	1
<Moonpool>	0	1
<DOF>	0	1
<Loa>	0	1
<Lpp>	0	1
<B>	0	1
<T>	0	1
<Displacement>	0	1
<Gmt>	0	1
<HullType>	0	1
<CurrentLoadsDrag>	0	1
<WaveLoad>	0	1
<WindLoad>	0	1
<ModelSpeedRangeValidity>	0	1
<CG>	0	1
<Rated>	0	unbounded
<ModelSubsystems>	0	1

<ExternalLoads>
Model can interact with external loads. The element value is of type xsd:boolean.

<Bulb>
Hull has bulb. The element value is of type xsd:boolean.

<ActiveAntirollTank>
Hull has active anti-roll tank. The element value is of type xsd:boolean.

<PassiveAntirollTank>
Hull has passive anti-rolll tank. The element value is of type xsd:boolean.

<BilgeKeel>
Hull has bilge keel. The element value is of type xsd:boolean.

<Skeg>
Hull has skeg. The element value is of type xsd:boolean.

<Moonpool>
Hull has moonpol. The element value is of type xsd:boolean.

<DOF>
Degrees of freedom. Typically, 3-6. The element value is of type xsd:unsignedInt.

<Loa>
Length over all [m]. The element value is of type xsd:double.

<Lpp>
Length between perpendiculars [m]. The element value is of type xsd:double.

<B>
Breadth [m]. The element value is of type xsd:double.

<T>
Draught [m]. The element value is of type xsd:double.

<Displacement>
Weight at the specified draught [kg]. The element value is of type xsd:double.

<Gmt>
Tranverse metacentric height [m]. The element value is of type xsd:double.

<HullType>
The type or shape of the hull.

Value	Type	Description
Mono-hull	xsd:enumeration	Hull of type Mono-hull
Semi-submersible	xsd:enumeration	Hull of type Semi-submersible
Catamaran	xsd:enumeration	Hull of type Catamaran
Cylindrical	xsd:enumeration	Hull of type Cylindrical

<CurrentLoadsDrag>
The type of current load/ drag included in the model.

<Static>	0	1
<Dynamic>	0	1

<ShallowWater>	0	1

<Static>
Current load/ drag of type static. The element value is of type xsd:boolean.

<Dynamic>
Current load/ drag of type dynamic. The element value is of type xsd:boolean.

<ShallowWater>
Current load/ drag of type shallow-water. The element value is of type xsd:boolean.

<WindLoad>
The type of wind load included in the model.

Children	Min	Max
<Static>	0	1
<Dynamic>	0	1

<Static>
Wind load of type static. The element value is of type xsd:boolean.

<Dynamic>
Wind load of type dynamic. The element value is of type xsd:boolean.

<CG>
Centre of gravity (x,y,z) [m]. Defined relative according to origin placed at: mid ship, keel

Children	Min	Max
<x>	1	1
<y>	1	1
<z>	1	1

<x>
x-position [m]. The element value is of type xsd:double.

<y>
y-position [m]. The element value is of type xsd:double.

<z>
z-position [m]. The element value is of type xsd:double.

<ModelSpeedRangeValidity>
Valid speed range for the model [m/s].

Children	Min	Max
<RangeOrValue>	1	1

<ModelPrinciplesForHull>
Fundamental principles that the model is using.

Children	Min	Max
<Seakeeping>	0	1
<Maneuvering>	0	1
<HullDynamics>	0	1
<Other>	0	1

<Seakeeping>
Model is intended for real time simulations. The element value is of type xsd:boolean.

<Maneuvering>
Model is intended for maneuvering simulations. The element value is of type xsd:boolean.

<HullDynamics>

Children	Min	Max
<Unified>	0	1
<LowFrequencyWaveFrequency>	0	1

<Unified>
Hull dynamics based on unified theory. The element value is of type xsd:boolean.

<LowFrequencyWaveFrequency>
Hull dynamics based on low-frequency/ wave-frequency theory. The element value is of type xsd:boolean.

5 REFERENCES

[1] International Organization for Standardization [ISO], “Ships and marine technology — Standard data for shipboard machinery and equipment (ISO 19848),” 2018.

[2] “DNVGL-VIS Naming rules,” DNV GL, [Online]. Available: https://data.dnvgl.com/dnvgl-vis/. [Accessed 7 May 2020].

[3] The Open Simulation Platform Consortium, “Open Simulation Platform Interface Specification 1.0,” 2020.

[4] “OWL 2 Web Ontology Language Document Overview (Second Edition),” W3C, [Online]. Available: https://www.w3.org/TR/owl2-overview/. [Accessed 16 May 2020].

[5] “Reasonable Ontology Templates,” [Online]. Available: https://ottr.xyz/. [Accessed 20 May 2020].

[6] T. Tudorache, “Ontology Engineering: Current State, Challenges, and Future Directions,” *Semantic Web – Interoperability, Usability, Applicability,* p. 14.

[7] P. Hitzler, M. Krötzsch and S. Rudolph, Foundations of Semantic Web Technologies, Chapman & Hall/CRC, 2009.

Appendix A XML File Examples

The appendix contains example implementations which serve as a guide for how to use MDO.

A.1 Electric Motor

```
<?xml version="1.0" encoding="utf-8"?>

<MDO-Model xmlns="https://opensimulationplatform.com/mdo">
  <Version>1.0</Version>
  <Vendor>Motor Model Vendor</Vendor>
  <Maker>Motor Model Maker</Maker>
  <MakeModel>SuperMotor ACDelux</MakeModel>
  <ModelType>
    <ElectricMotor>
      <ModelFeaturesForElectricMotor>
        <DesignedForRealtimeSimulation>true</DesignedForRealtimeSimulation>
<DesignedForControlSystemInterfacing>true</DesignedForControlSystemInterfacing>
        <DesignedForTesting>true</DesignedForTesting>
        <FailureModesAvailable>
          <FailureMode>
            <FailToMin>
              <FailureModeDescription>Torque output from motor is
zero</FailureModeDescription>
            </FailToMin>
          </FailureMode>
        </FailureModesAvailable>
        <Inertia>true</Inertia>
        <AC>true</AC>
        <DC>false</DC>
        <ReactivePower>true</ReactivePower>
        <InductionMotor>true</InductionMotor>
        <PermanentMagnetMotor>false</PermanentMagnetMotor>
        <Rated>
          <Quantity>Power</Quantity>
          <Unit>Watt</Unit>
          <PrefixMultiplier>mega</PrefixMultiplier>
          <ValueOrRange>
            <Value>2</Value>
          </ValueOrRange>
        </Rated>
        <Rated>
          <Quantity>Torque</Quantity>
          <Unit>NewtonMetre</Unit>
          <PrefixMultiplier>kilo</PrefixMultiplier>
          <ValueOrRange>
            <Value>900</Value>
          </ValueOrRange>
        </Rated>
        <ModelSubsystems>
          <Subsystem>
            <Name>Electric motor</Name>
            <VISCode>E31</VISCode>
            <Min>1</Min>
            <Max>4</Max>
          </Subsystem>
        </ModelSubsystems>
      </ModelFeaturesForElectricMotor>
      <ModelPrinciplesForElectricMotor>
```

```
        <ElectricalModellingPrinciple>
          <Sinusoidal>true</Sinusoidal>
        </ElectricalModellingPrinciple>
      </ModelPrinciplesForElectricMotor>
    </ElectricMotor>
  </ModelType>
</MDO-Model>
```

A.2 ELECTRIC POWER PLANT

```
<?xml version="1.0" encoding="utf-8"?>

<MDO-Model xmlns="https://opensimulationplatform.com/mdo">
  <Version>1.0</Version>
  <Vendor>Power Plant model vendor</Vendor>
  <Maker>Power Plant model maker</Maker>
  <MakeModel>Powerful Power Plant</MakeModel>
  <ModelType>
    <ElectricPowerPlant>
      <ModelFeaturesForElectricPowerPlant>
        <DesignedForRealtimeSimulation>true</DesignedForRealtimeSimulation>
<DesignedForControlSystemInterfacing>true</DesignedForControlSystemInterfacing>
        <DesignedForTesting>true</DesignedForTesting>
        <FailureModesAvailable>
          <FailureMode>
            <Blackout>
              <FailureModeDescription>Total blackout of entire power
system</FailureModeDescription>
            </Blackout>
          </FailureMode>
          <FailureMode>
            <ShortCircuit>
              <FailureModeDescription>Short circuit for a given
switchboard</FailureModeDescription>
            </ShortCircuit>
          </FailureMode>
        </FailureModesAvailable>
        <Inertia>true</Inertia>
        <AC>true</AC>
        <DC>true</DC>
        <ReactivePower>true</ReactivePower>
        <ModelSubsystems>
          <Subsystem>
            <Name>Generator</Name>
            <VISCode>E32</VISCode>
            <Min>1</Min>
            <Max>10</Max>
          </Subsystem>
          <Subsystem>
            <Name>Switchboard</Name>
            <VISCode>E11</VISCode>
            <Min>1</Min>
            <Max>20</Max>
          </Subsystem>
          <Subsystem>
            <Name>Transformer</Name>
            <VISCode>E41</VISCode>
            <Min>1</Min>
            <Max>12</Max>
          </Subsystem>
        </ModelSubsystems>
      </ModelFeaturesForElectricPowerPlant>
      <ModelPrinciplesForElectricPowerPlant>
        <ElectricalModellingPrinciple>
          <Phasor>true</Phasor>
        </ElectricalModellingPrinciple>
      </ModelPrinciplesForElectricPowerPlant>
    </ElectricPowerPlant>
  </ModelType>
```

```
</MDO-Model>
```

A.3 HULL

```
<?xml version="1.0" encoding="utf-8"?>

<MDO-Model xmlns="https://opensimulationplatform.com/mdo">
  <Version>1.0</Version>
  <Vendor>Hull Model Vendor</Vendor>
  <Maker>Hull Model Maker</Maker>
  <MakeModel>HullSuberb</MakeModel>
  <ModelType>
    <Hull>
      <ModelFeaturesForHull>
        <DesignedForRealtimeSimulation>true</DesignedForRealtimeSimulation>
        <ExternalLoads>true</ExternalLoads>
        <DOF>6</DOF>
        <CurrentLoadsDrag>
          <Static>false</Static>
          <Dynamic>true</Dynamic>
          <ShallowWater>true</ShallowWater>
        </CurrentLoadsDrag>
        <WaveLoad>
          <SlowlyVaryingMeanWaveDrift>true</SlowlyVaryingMeanWaveDrift>
          <WaveFrequencyLoad>true</WaveFrequencyLoad>
          <ShallowWater>true</ShallowWater>
          <InteractionWithCurrent>true</InteractionWithCurrent>
        </WaveLoad>
        <Windload>
          <Static>true</Static>
          <Dynamic>false</Dynamic>
        </Windload>
        <ModelSpeedRangeValidity>
          <Rated>
            <Quantity>Speed</Quantity>
            <Unit>MetrePerSecond</Unit>
            <PrefixMultiplier>one</PrefixMultiplier>
            <ValueOrRange>
              <Range>
                <LowerValue>0.05</LowerValue>
                <UpperValue>1.25</UpperValue>
              </Range>
            </ValueOrRange>
          </Rated>
        </ModelSpeedRangeValidity>
        <HullType>Semi-submersible</HullType>
        <Bulb>false</Bulb>
        <ActiveAntirollTank>false</ActiveAntirollTank>
        <PassiveAntirollTank>false</PassiveAntirollTank>
        <BilgeKeel>false</BilgeKeel>
        <Skeg>false</Skeg>
        <Moonpool>true</Moonpool>
        <Loa>100.0</Loa>
        <Lpp>90.5</Lpp>
        <B>70.1</B>
        <T>20.4</T>
        <Displacement>60000000</Displacement>
        <CG>
          <X>0.0</X>
          <Y>0.0</Y>
          <Z>30.1</Z>
        </CG>
        <Gmt>2.3</Gmt>
```

```
        <ModelSubsystems>
           <!--Often times, a Hull model will be more or less free-standing, i.e.
without subsystems like this.
        A thruster subsystem is included only to illustrate the principle of
subsystems and how they can be linked to VIS-->
          <Subsystem>
            <Name>Thruster, azimuth</Name>
            <VISCode>C322</VISCode>
            <Min>1</Min>
            <Max>12</Max>
            <Subsystem>
              <Name>Steering gear actuator</Name>
              <VISCode>C322.3</VISCode>
              <Min>1</Min>
              <Max>1</Max>
            </Subsystem>
            <Subsystem>
              <Name>Input shafting</Name>
              <VISCode>C322.4</VISCode>
              <Min>1</Min>
              <Max>1</Max>
            </Subsystem>
            <Subsystem>
              <Name>Underwater unit</Name>
              <VISCode>C322.6</VISCode>
              <Min>1</Min>
              <Max>1</Max>
              <Subsystem>
                <Name>Lower gear</Name>
                <VISCode>C323.52</VISCode>
                <Min>1</Min>
                <Max>1</Max>
              </Subsystem>
              <Subsystem>
                <Name>Propeller shafting</Name>
                <VISCode>C322.63</VISCode>
                <Min>1</Min>
                <Max>1</Max>
              </Subsystem>
            </Subsystem>
            <Subsystem>
              <Name>Propeller</Name>
              <VISCode>C322.7</VISCode>
              <Min>1</Min>
              <Max>1</Max>
            </Subsystem>
          </Subsystem>
        </ModelSubsystems>
      </ModelFeaturesForHull>
      <ModelPrinciplesForHull>
        <HullDynamics>
          <Unified>true</Unified>
          <LowFrequencyWaveFrequency>false</LowFrequencyWaveFrequency>
        </HullDynamics>
        <Seakeeping>true</Seakeeping>
        <Manoeuvring>true</Manoeuvring>
      </ModelPrinciplesForHull>
    </Hull>
  </ModelType>
</MDO-Model>
```