eCISE-OWL : Représentation OWL du Schéma de Données de l'Environnement Commun d'Échange d'Information pour le Domaine Maritime

Nathalie Aussenac-Gilles, Catherine Comparot, Antoine Dupuy, Nabil El Malki, Ronan Tournier, Ba-Huy Tran, Cassia Trojahn dos Santos

eCISE-OWL : Représentation OWL du Schéma de Données de l'Environnement Commun d'Échange d'Information pour le Domaine Maritime

Nathalie Aussenac-Gilles, Catherine Comparot, Antoine Dupuy, Nabil El Malki, Ronan Tournier, Ba-Huy Tran, Cassia Trojahn

[1] IRIT, Université de Toulouse, CNRS, UT1, UT2, Toulouse, France

prenom.nom ou prenom.nom-composé@irit.fr

Résumé

L'Environnement Commun de Partage d'Information (CISE) est un modèle de donnée qui résulte d'une initiative collaborative pour promouvoir le partage automatisé d'informations entre les autorités de surveillance maritime. L'exploitation de CISE est cependant limitée par sa sérialisation via uniquement un schéma XML, peu adapté pour fournir une sémantique plus riche, une interopérabilité sémantique et des capacités de raisonnement. La transformation de la version étendue de CISE (eCISE) en ontologie est indispensable pour faciliter l'accès aux données d'échanges maritimes selon les principes de l'Ontology Based Data Access, et le raisonnement sur ces données. Cet article présente eCISE-OWL, une représentation OWL de eCISE, désormais disponible en accès ouvert. Pour obtenir cette ontologie, nous proposons un processus de transformation de schémas XML (en XSD) vers OWL qui améliore l'état de l'art, dont le code est rendu public et qui comporte une étape de validation par des experts du domaine.

Mots-clés

CISE, ontologie, domaine maritime, interopérabilité

Abstract

The Common Information Sharing Environment (CISE) is the result of a collaborative initiative to promote automated information sharing among maritime surveillance authorities. L'exploitation de CISE is however limited by its serialization of CISE via only an XML schema, insufficient to provide richer semantics, semantic interoperability and reasoning capabilities. This paper presents eCISE-OWL, an OWL representation of the extended version of CISE (eCISE). eCISE-OWL is the result of an improved process of transforming XML schemas (XSD) into OWL and of validation and correction efforts by domain experts. We discuss the use of eCISE-OWL in maritime exchange data access (OBDA) and reasoning tasks.

Keywords

CISE, ontology, maritime domain, interoperability

1 Introduction

Rendre les systèmes de surveillance maritime interopérables est crucial pour la coopération entre les pays, en particulier en cas de crises maritimes dans des zones frontalières entre pays. Dans cet objectif, l'hétérogénéité entre les système nationaux et les structures de données des différents acteurs soulèvent de nombreux problèmes. Afin de permettre aux autorités maritimes d'échanger des informations de manière automatique et sécurisée, elles ont adopté un environnement commun de partage d'informations (CISE) [1]. Il fournit un cadre décentralisé et un modèle de données pour l'échange d'informations point à point entre les secteurs et les frontières. Il implique plus de 300 autorités européennes et nationales ayant des responsabilités en matière de surveillance maritime, et effectuant de nombreuses tâches de surveillance opérationnelle. Ces autorités bénéficient directement d'être connectées au réseau CISE, pour des objectifs aussi divers que la sûreté et la sécurité du transport maritime, le contrôle de la pêche, la pollution, et la défense. Depuis 2014, CISE est retenu pour soutenir la mise en œuvre de la stratégie de sécurité maritime de l'Union Européenne (EUMSS).

L'adoption du modèle de données CISE [2] et de ses différentes versions – en particulier, *Extended-CISE* (eCISE) [1] – est cependant limitée par sa sérialisation via un schéma XML uniquement, dont la sémantique n'est pas assez riche pour garantir une interopérabilité sémantique des données ou pour servir de support à un raisonnement. Or ces fonctionnalités s'avèrent très utiles pour associer des données venant de différentes sources de données CISE, les vérifier ou encore en déduire de nouveaux éléments. Un premier effort dans cette direction a été proposé par le projet européen ROBORDER [14], qui a généré une représentation ontologique du modèle de données CISE tel qu'il a été défini dans le projet EUCISE2020 [5], en convertissant le modèle UML en OWL. Même si l'ontologie et son processus de construction sont décrits (incomplètement) dans l'article cité, aucun

1. http://www.emsa.europa.eu/cise.html
2. http://emsa.europa.eu/cise-documentation/cise-data-model-1.5.3/

des deux n'est disponible publiquement.
Bien que le passage de données XML ou de schémas XSD à une représentation sémantique (RDF, RDFS ou OWL) soit une question étudiée de longue date dans le domaine du Web sémantique, une transformation automatique simple est rarement correcte. Ce processus se heurte à la difficulté de gérer des noeuds anonymes, de traiter la représentation de noeuds complexes, de capturer la sémantique des balises purement structurelles, ou encore de produire des constructeurs liés à la structuration [12, 10, 4, 22].
Ce travail est réalisé dans le cadre du projet H2020 EFFECTOR [3], qui veut proposer un cadre d'interopérabilité et des services de fusion et d'analyse de données associés pour la surveillance maritime et la sécurité des frontières. Ainsi, EFFECTOR vise à améliorer le processus d'aide à la décision et à favoriser la collaboration des acteurs maritimes au niveau local, régional et transnational. Le modèle de données CISE joue un rôle essentiel dans le projet car les messages échangés par les différents acteurs sont basés sur les diverses versions de ce modèle. Cependant, ces messages sont difficilement accessibles car stockés dans plusieurs bases de données internes. Un système d'accès aux données via une ontologie (OBDA) va donc être mis en place afin de contribuer à l'interoperabilité des systèmes et de faciliter les échanges de données entre partenaires. De plus, afin d'aider l'opérateur chargé de la surveillance maritime, l'ontologie permettra de produire des inférences et de générer de nouveaux faits signalant de potentielles anomalies.
Cet article présente eCISE-OWL, une représentation OWL du modèle eCISE. eCISE étend le modèle CISE en améliorant le vocabulaire maritime de CISE et en élargissant sa portée à la surveillance terrestre et à l'échange d'informations opérationnelles. Outre l'ontologie elle-même, notre contribution réside aussi dans le processus original de transformation d'un schéma XML (XSD) en OWL. Ce processus tient compte des particularités du modèle de données CISE pour générer un premier modèle OWL, que des experts du domaine ensuite valident et corrigent. Il intègre et étend des travaux existants, décrits de manière formelle dans plusieurs articles mais qui n'avaient jamais été regroupés dans une unique chaîne de traitement et qui n'étaient pas réutilisables. Nous avons amélioré significativement la gestion des collections, représentées jusque là comme des collections de `owl:oneOf`, ce qui ralentit les performances au moment de requêter ces données. De plus, nous avons simplifié la représentation des classes d'association, qui sont très nombreuses dans eCISE. Enfin, aussi bien le code de génération de l'ontologie que l'ontologie [4] elle-même sont disponibles publiquement.
Nous poursuivons cet article en présentant un état de l'art sur les ontologies maritimes en lien avec le modèle de données CISE et ses variantes, ainsi que sur les solutions existantes pour générer des modèles OWL à partir de schémas XSD (section 2. La section 3 introduit le modèle CISE et son extension eCISE. Les cas d'usages de l'ontologie sont ensuite présentés (Section 4). Nous détaillons la méthodologie de construction en Section 5. L'ontologie ainsi que son évaluation sont présentés dans la Section 6. Finalement, la Section 7 conclut l'article et dessine les perspectives pour les travaux futurs.

3. `https://www.effector-project.eu/`
4. L'ontologie est disponible `https://www.irit.fr/recherches/MELODI/ontologies/ecise/index-en.html`

2 Travaux liées

Ontologies maritimes La surveillance maritime automatisée reçoit un intérêt particulier depuis quelques années, ce qui est attesté par un grand nombre de projets abordant les différents défis du domaine (ROBORDER, EUCISE2020, ANDROMEDA, MARISA, datAcron, et CoopP, pour en citer quelques uns). Les technologies sémantiques ont prouvé leur pertinence en facilitant le partage automatisé d'informations entre les systèmes de surveillance maritime. En particulier, dans le cadre du projet datAcron, une ontologie a été proposée pour représenter des trajectoires d'objets en mouvement[15]. Plus proche de nos travaux, dans le cadre du projet ROBORDER, l'ontologie EUCISE-OWL résulte [14] d'une conversion UML en OWL du modèle de données EUCISE2020 (une extension du modèle CISE Data Model v1.0 [5] pour couvrir des sources de données supplémentaires). EUCISE-OWL a été la première tentative d'exploiter entièrement le schéma CISE pour développer une ontologie qui facilite l'échange d'informations dans le domaine maritime. Cependant, la ressource et son processus de construction ne sont pas publiquement disponibles.
D'autres ontologies maritimes se trouvent dans la littérature. Les travaux de [8] proposent une représentation ontologique des différents types de navires et des paramètres pertinents, en fonction de l'AIS (Automatic Identification System), dans le cadre de la tâche d'analyse de trafic maritime. Dans [6], l'analyse de trajectoires sémantiques et les localisations géographiques des objets maritimes est réalisée grâce à des ontologies de domaine. L'approche combine le traitement de données statiques et en temps réel provenant de différentes sources à l'aide de techniques d'accès aux données basées sur l'ontologie (OBDA). Un autre aspect du domaine concernant la détection ou la prédiction de comportements anormaux des navires, plusieurs travaux ont utilisé des ontologies dédiées à cette tâche. Dans [19, 20], les ontologies spatiales et la représentation sémantique des trajectoires servent à caractériser le comportement anormal des navires, sur la base de propriétés sémantiques formelles servant à raisonner sur les données. Alors que ces méthodes reposent principalement sur des ontologies créées manuellement ou dérivées de ressources non ontologiques comme des schémas XML ou des diagrammes UML, d'autres ontologies maritimes ont été construites à partir de textes [24]. Finalement, sous un autre angle, une ontologie de la réglementation maritime a été définie décrivant par exemple les procédures portuaires et la maintenance de navires, afin d'évaluer l'impact des nouveaux règlements et de retracer leur origine législative [11].

Passage XML/XSD vers OWL La représentation sémantique (RDF ou OWL) de données XML ou de schémas XSD

est une question étudiée de longue date dans le domaine du Web sémantique. Cependant, les balises ne se situant pas toutes au même niveau d'abstraction, une transformation automatique simple est rarement efficace et correcte. Lorsque les éléments définis entre balises sont eux mêmes complexes, ils relèvent de plusieurs types en lien représentés par de multiples propriétés. Ils peuvent même contribuer à la fois à enrichir l'ontologie et à la peupler. C'est le cas de l'exploitation de fiches XML dans le projet MOANO par exemple [3]. Dans tous les cas, le passage du XSD à des classes OWL ou des types RDF se heurte à la difficulté de gérer des noeuds anonymes, de traiter la représentation de noeuds complexes, des énumérations, de gérer les types XSD, ou encore les balises liées à la structuration et non sémantiques.

Plusieurs approches sont mises en oeuvre dans les logiciels de l'état de l'art. Un premier ensemble d'outils, dits de "lifting", convertissent un schéma XML (XSD) en un schéma RDF, tel que RDFS ou OWL. C'est le cas de XML2OWL (qui part d'un document XML ou d'un XSD) ou de XSD2OWL [5] (à partir d'un XSD). L'approche la plus classique consiste à utiliser XSLT, le langage de transformation de schémas XSD, en considérant RDF-XML comme un schéma XML particulier [6]. XSLT est d'ailleurs à la base de GRDDL [7], un mécanisme qui permet de rajouter des balises dans un document XML pour indiquer que les données décrites peuvent être traduites vers RDF à l'aide de XSLT. Cependant, un wiki du W3C [8] souligne que si XSLT est adapté pour convertir une majorité de modèles XML en RDF, il se heurte à plusieurs limites : il génère des modèles pas optimisés et illisibles par un humain ; dans le cas de modèles complexes, XSLT ne sait pas traiter toutes les situations, par exemple des structures imbriquées ou des textes longs entre balises. C'est en effet le cas de l'outil Ontmalizer [9] [23] qui fournit l'ensemble de règles de transformations adoptées dans cet article, mais qui génère un résultat de conversion complexe. Le site MIT Simile fournissait une liste de quelques autres "RDF-izers" [10] dont la plupart ne sont plus accessibles ou ne gèrent pas le format XML. Le W3C dresse une autre liste [11] d'outils de conversion de XML vers RDF, dont TopBraid Composer (un logiciel commercial) avec un plugin qui gère le passage de XSD vers OWL ; KREXTOR, une plateforme qui sait traiter plusieurs variantes de XML à l'aide de transformations XSLT ; Rhizomik, qui fait appel à XML2RDF et XSD2RDF ; GRDDL ; XHTML, etc.

Une alternative performante à XSLT repose sur des règles RML, dont le pouvoir d'expression est plus puissant à traiter des schémas complexes, et dont le format se prête bien à une reformulation lisible par un humain. Le logiciel SDM-RDFIZER [12] s'appuie sur RML pour traiter une grande diversité de formats. Son avantage est le passage à l'échelle, les règles étant optimisées pour traiter de grands volumes de données.

Malheureusement, les outils présentés dans les articles de recherche (comme les deux que nous venons de citer) sont rarement accessibles.

3 Schémas CISE et eCISE

3.1 De CISE à eCISE

Le modèle de données CISE a pour ambition de servir de format pour le partage d'information de surveillance maritime entre secteurs et pays. Dans cette optique, il décrit sept entités principales (`Agent`, `Object`, `Location`, `Document`, `Event`, `Risk`, `Period`) et onze entités secondaires (`Vessel`, `Cargo`, `Operational Asset`, `Person`, `Organization`, `Movement`, `Incident`, `Anomaly`, `Action`, `Unique Identifier`, `Metadata`). Ce modèle permet aux différentes autorités de bénéficier d'un vocabulaire commun pour décrire notamment les événements observés. Il est décliné dans les formalismes RDF et XSD. Sur la Figure 1, les entités du modèle CISE correspondent aux hexagones non colorés.

Le modèle de données eCISE enrichit le vocabulaire de CISE pour les domaines maritime et terrestre.Il fournit un ensemble plus riche de types de navires, d'informations fournies par le système d'identification de navires AIS (Automatic Identification System) et de capteurs radar ; il liste également un ensemble plus complet d'anomalies et de règles maritimes, mais aussi terrestres, avec un nombre important de types pour chacune de ses entités. Ce modèle est construit sur la dernière version du modèle de données CISE utilisé dans le projet EUCISE2020 [9]. Sur la Figure 1, les entités centrales de eCISE qui complètent celles du modèle CISE, correspondent aux hexagones de couleur.

3.2 Le modèle XML d'eCISE

Les modèles de données CISE et eCISE sont décrits dans un document de spécification (diagrammes de classes UML) et implémentés en XSD (schémas XML). Les fichiers XSD de CISE ont été produits à partir de transformations, i.e. un ensemble de règles de correspondance indiquant comment générer des éléments XSD à partir des éléments UML. Les choix effectués lors de ce processus impactent la structure XSD obtenue et doivent être pris en compte lors de la génération de sa représentation OWL. En ce qui concerne CISE, chaque fichier .xsd représente une ou plusieurs entités du modèle, où chaque entité est représentée via une balise `xs:complexType`. La notion de hiérarchie de spécialisation entre entités est représentée par la balise `xs:extension`. Les éléments (au sens XML) qui composent plus spécifiquement une entité sont alors décrits en XSD au sein de la balise

5. https://gist.github.com/pebbie/5704765
6. https://rml.io/docs/rml/tutorials/xml/
7. Gleaning Resource Descriptions from Dialects of Languages https://www.w3.org/TR/grddl/
8. https://www.w3.org/community/rax/wiki/XML_to_RDF_Transformation_processes_using_XSLT
9. https://www.w3.org/wiki/HCLSIG/Tools\#Ontmalizer
10. https://en.wikipedia.org/wiki/List_of_SIMILE_projects#RDFizer
11. https://www.w3.org/wiki/ConverterToRdf#XML
12. https://pypi.org/project/rdfizer/

FIGURE 1 – Aperçu des vocabulaires du modèle de base CISE (entités en blanc) et du modèle eCISE (entités CISE enrichies par les entités en couleur).

`xs:complexContent`. Ils sont listés dans l'ordre où ils doivent apparaître dans une balise `xs:sequence`. Chacun de ces éléments correspond soit à une propriété propre à l'entité, soit à une association avec une autre entité. L'extrait ci-dessous décrit l'entité `Vehicle` comme une sous-classe de `Object` liée à l'entité Cargo par une association représentée en XML par la propriété `CargoRel` ; la valeur explicite 0 de `xs:minOccurs` et la valeur implicite 1 de `xs:maxOccurs` indiquent une cardinalité de 0,1 ; la propriété est donc facultative et ne peut apparaître qu'une fois. Il est à noter que `Cargo` désigne une cargaison (i.e., un ensemble de marchandises transporté par un véhicule entre deux ports), et non un type de bateau.

```
<xs:complexType name="Vehicle" abstract="true">
 <xs:annotation>
  <xs:documentation>
   The Vehicle is a sub-class of Object [...]
  </xs:documentation>
 </xs:annotation>
 <xs:complexContent>
  <xs:extension base="object:Object">
   <xs:sequence>
    <xs:element name="CargoRel" minOccurs="0">
     <xs:complexType>
      <xs:complexContent>
       <xs:extension base="rel:Relationship">
        <xs:sequence>
         <xs:element name="Cargo" type="cargo:Cargo"
                     minOccurs="0"/>
        </xs:sequence>
       </xs:extension>
      </xs:complexContent>
     </xs:complexType>
    </xs:element>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
```

4 Exigences en matière d'ontologie

Chaque système informatique employé dans les centres de coordination et de sauvetage maritime ayant son propre vocabulaire et son propre schémas de données, une ontologie commune offre un modèle pivot de haut niveau qui, d'un côté, facilite l'interopérabilité et, de l'autre, permet d'envisager un raisonnement sur les données de surveillance. Le modèle CISE représentant le vocabulaire de la surveillance maritime, dans le cadre d'EFFECTOR, nous avons retenu ce modèle comme base pour construire cette ontologie de domaine. Nous décrivons dans ce qui suit les deux services retenus dans EFFECTOR qui justifient l'utilisation d'une ontologie pivot.

4.1 Ontology Based Data Access

L'*Ontology Based Data Access* (OBDA) [7] est un paradigme d'accès aux données à travers une couche conceptuelle. Habituellement, cette couche est spécifiée via un schéma RDF ou une ontologie OWL et les données sont stockées dans des bases relationnelles. Les termes du niveau conceptuel sont associés à la couche de données via un mapping associant individuellement chaque élément de la couche conceptuelle à une requête SQL (potentiellement complexe) sur les sources de données relationnelles. Le graphe virtuel ainsi obtenu peut être requêté via un langage de requête sur des données RDF, tel que SPARQL.

Les acteurs de la surveillance et du sauvetage maritime reçoivent les message CISE via le réseau de noeuds CISE des états membres. Les informations échangées sont stockées dans les bases de données locales à chaque système national, qui gèrent également d'autres données en provenance d'autres sources (AIS, Sat-AIS, radar ...). Comme le projet EFFECTOR vise à améliorer le processus de prise de décision et à favoriser la collaboration entre les acteurs, échanger des données les plates-formes est un enjeu clé. Dans cet objectif, une solution adaptée repose sur l'approche OBDA où le niveau conceptuel fait appel à une ontologie construite sur un vocabulaire commun trans-frontalier et multi-plateformes.

4.2 Raisonnement sur des messages CISE

Une autre utilisation de l'ontologie concerne la capacité de raisonnement afin d'inférer de nouveaux éléments à partir de l'instanciation de parties spécifiques de l'ontologie, selon les différents scénarios définis dans le projet. Il s'agit notamment d'offrir un support à l'opérateur humain dans le processus de détection des anomalies. Par souci de confidentialité, nous ne pouvons communiquer les détails des scénarios et les exemples suivants sont des cas classiques du domaine [18, 20].

Alerte vitesse Si un bateau se déplace à une vitesse supérieure à la vitesse autorisée pour le type de bateau en question, une alerte vitesse doit être générée. En s'inspirant de [20], cette alerte peut être produite via une règle SWRL adaptée au vocabulaire eCISE, comme suit :

```
ObjectLocation(?objectLocation),
hasInvolvedLocation(?objectLocation,
```

```
?location), hasInvolvedObject(?objectLocation,
?vehicle), Vessel(?vehicle),
speed(?objectLocation,?speed),
maximumspeed(?vehicule,?maxSpeed),
greaterThan(?speed,?maxSpeed)
→ MaritimeAnomaly(?anomaly),
hasMaritimeAnomalyType(?anomaly,:HighSpeed)
```

Risque de pollution S'il existe une risque de collision imminente entre deux bateaux et qu'au moins l'un des navires impliqués a une cargaison dangereuse, alors il y a un risque de pollution :

```
ObjectEvent(?objectEvent),
hasInvolvedEvent(?objectEvent,
:VesselImminentCollision),
hasInvolvedObject(?objectEvent,?vehicle),
Vessel(?vehicle), hasCargo(?vehicle,?cargo),
hasContainedCargoUnit(?cargo,?containmentUnit),
hasPollutionCode(?containmentUnit,?pollution-
CodeType) → Risk(?risk),
hasRiskType(?risk,:Pollution)
```

5 Réingénierie de ressources non ontologiques

La méthodologie de construction de l'ontologie eCISE-OWL est conforme au Scénario 2 *Réutilisation et réingénierie des ressources non ontologiques (RNO)* de la méthodologie NeOn [16, 21]. Le processus de réingénierie des RNO a été défini pour transformer les RNO en ontologies. Dans notre cas, les RNO correspondent aux schémas XSD (décrits en XML) : ils sont homogènes dans leur modèle de données. En [21], les auteurs suggèrent de se placer au niveau conceptuel pour étudier les correspondances entre le modèle source et sa conversion RDF, en explicitant comment chaque élément du schéma XML doit être traduit en RDF ou OWL. La mise en forme (manuelle) de ces règles pour former un patron de conversion est une étapes clé de la traduction, qui permet d'automatiser ensuite la traduction de données décrites selon ce modèle. Dans notre cas, le problème est de transformer un schéma, et de traiter de la même manière chaque structure similaire dans ce schéma, selon les mêmes règles.

5.1 De XML à OWL

Faute de disposer de logiciel opérationnel et libre pour transformer les modèles XML en RDF qui sachent traiter la représentation des relations n-aires entre classes, nous avons implémenté des règles de conversion de XML vers RDF qui répondent aux exigences du modèle CISE. Il nous a paru souhaitable que ce processus produise aussi automatiquement des descriptions des éléments (via `rdfs:comment` par exemple) à partir de la documentation. Afin de traiter le modèle CISE, les difficultés à dépasser sont notamment liées au nombre important d'énumérations (types possibles d'entités spécifiques) à convertir, au manque d'information quant au nommage des classes d'association et à la conversion des contraintes de cardinalité.

Pour répondre à ces besoins, nous avons développé un processus de transformation. Il s'inspire du travail de [2] et utilise un ensemble de règles de transformation. Ce processus, implémenté en Python, utilise `rdflib`. Il parcourt des sources XSD et des sources externes afin d'extraire les éléments nécessaires à la construction de l'ontologie. Pour chaque type d'élément XSD, une règle de transformation correspondante est appliquée vers le formalisme OWL. Le script récupère la documentation du modèle de données eCISE pour en extraire les commentaires qui serviront à documenter les entités de l'ontologie (`rdfs:label` et `rdfs:comment`). Pour cette étape d'enrichissement terminologique de l'ontologie, nous utilisons comme source externe le livrable D3.1 d'Andromeda [1] au format PDF. Notre outil utilise la librairie PDFReader et effectue une lecture page à page de ce PDF. Cette étape est à généraliser afin de rentre le processus capable de traiter d'autres sources de données.

5.2 Règles de transformation

Les règles de transformations adoptées ici reprennent la plupart des règles de transformation de l'outil Ontmalizer [13] [23]. Concernant les classes d'association UML et les énumérations, nous avons adapté ces règles en suivant la proposition de [14]. La table 1 liste les correspondances entre les éléments XSD et les définitions OWL ainsi que des éléments définis pour le traitement de certains types spécifiques tels que les énumérations et les classes d'association. Des exemples de transformation sont introduits par la suite, pour les principaux type de constructeurs OWL.

Espaces de noms Les fichiers sources XSD indiquent des espaces de nom par groupe d'entités. Ces espaces de nom ont été retranscrits tels quels dans l'ontologie pour respecter les conventions du modèle de données eCISE. La version actuelle de l'ontologie eCISE-OWL comporte ainsi des espaces de nom tels que `event`, `vessel`, `object` et `location`.

Classes Toute entité ou classe décrite dans le modèle de données eCISE est une sous-classe de `:Entity` (sous-classe de `:owl:Thing`). A titre d'exemple, nous présentons ci-dessous les représentations XML et OWL de l'entité `:Vessel` :

```
<xs:complexType name="Vessel">
 <xs:complexContent>
  <xs:extension base="object:Vehicle">
   <xs:sequence>
    [...]
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
```

```
[:Vessel
    rdf:type owl:Class ;
    rdfs:subClassOf :Vehicle ;
    rdfs:comment "The class Vessel is a
     sub-class of the class Vehicle. A vessel refers
     to a ship or a boat. [...]" ;
    rdfs:label "Vessel" .
]
```

13. https://www.w3.org/wiki/HCLSIG/Tools\#Ontmalizer

Élément XSD	*Définitions OWL*
xs :simpleType	rdfs :Datatype
xs :simpleType	rdfs :Datatype
xs :enumeration	owl :Class et owl :Individual
xs :complexType over xs :complexContent	owl :Class
xs :complexType over xs :simpleContent	owl :Class
xs :element (global) with complex type	owl :Class and rdfs :subclassOf
xs :element (global) with simple type	owl :Datatype
xs :element (local to a type)	owl :DatatypeProperty or owl :ObjectProperty
xs :group	owl :Class
xs :attributeGroup	owl :Class

TABLE 1 – Règles de conversion XSD en OWL.

Propriétés Les classes sont liées à d'autres types de données (soit des classes, soit des valeurs). Pour ce type d'élément, nous avons suivi les bonnes pratiques de nommage introduites en [17] (à noter les noms de propriétés de classes préfixés par un 'has'). Dans l'exemple, la propriété de classe `:hasCargo` associe `:Vehicle` et `:Cargo`, selon l'extrait du schéma XSD reproduit en Section 3.2.

```
<xs:complexType name="Vehicle" abstract="true">
 <xs:complexContent>
  <xs:extension base="object:Object">
   <xs:sequence>
   [...]
   <xs:element name="CargoRel"  minOccurs="0">
    <xs:complexType>
     <xs:complexContent>
    <xs:extension base="rel:Relationship">
   <xs:sequence>
    <xs:element name="Cargo" type="cargo:Cargo"/>
</xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
  </xs:element>
 </xs:sequence>
</xs:extension>
</xs:complexContent>

[:hasCargo
     rdf:type owl:ObjectProperty ;
     rdfs:domain :Vehicle ;
     rdfs:range :Cargo .
]
```

Le renommage de la propriété CargoRel suit la règle de nommage que nous avons définie pour les entités `ObjectProperty` par similarité avec des pratiques en usage dans la communauté [14] : retrait du suffixe Rel et ajout de "has" en début de mot.

Classes d'association Une classe d'association (UML) est un type de classe spécifique qui définit la connexion entre les entités principales du modèle, en utilisant des attributs spécifiques appelés "rôles d'association". Les classes d'association du modèle eCISE héritent systématiquement de la classe `Relationship` que nous avons choisi, inspirés de [14], de renommer `:AssociationClass` dans l'ontologie.

14. https://github.com/G-Node/python-odml/issues/112

```
[
:AssociationClass rdf:type owl:Class ;
  rdfs:comment "Abtract class representing a
    relationship of the CISE data model." ;
  rdfs:label "AssociationClass" .
]
```

Les classes d'association peuvent avoir des propriétés et des types de données supplémentaires qui leur sont propres. Inspirés de [14], nous représentons les classes d'association comme des éléments du type `owl:Class` (et non simplement comme des propriétés d'objet), tandis que les rôles d'association sont représentés par des propriétés de type `owl:ObjectProperty` dont le domaine est la classe d'association.
Dans le fragment XSD ci-dessous, l'entité `Object` possède un élément renvoyant à la classe `Event` et inversement. Les éléments `InvolvedObjectRel` (élément de Event) et `InvolvedEventRel` (élément de Object) possèdent les mêmes attributs, à l'exception de la référence à la classe associée (Object pour Event et inversement). On considère ces deux éléments pour créer une classe d'association regroupant les attributs communs aux deux éléments XSD et les deux références aux classes associées. Dans le cas où il n'y a pas d'attribut en plus des références aux classes associées, la création d'une entité de type `ObjectProperty` est préférée.

```
<xs:complexType name="Object" abstract="true">
 <xs:complexContent>
  <xs:extension base="entity:Entity">
   <xs:sequence>
    <xs:element name="InvolvedEventRel">
     <xs:complexType>
      <xs:complexContent>
       <xs:extension base="rel:Relationship">
        <xs:sequence>
         <xs:element name="Event"
              type="event:Event"/>
         <xs:element name="ObjectRole"
              type="event:ObjectRoleInEventType"/>
        </xs:sequence>
       </xs:extension>
      </xs:complexContent>
     </xs:complexType>
    </xs:element>
    [...]
   </xs:sequence>
  </xs:extension>
```

```
 </xs:complexContent>
</xs:complexType>

<xs:complexType name="Event" abstract="true">
 <xs:complexContent>
  <xs:extension base="entity:Entity">
   <xs:sequence>
    [...]
    <xs:element name="InvolvedObjectRel">
     <xs:complexType>
      <xs:complexContent>
       <xs:extension base="rel:Relationship">
        <xs:sequence>
         <xs:element name="Object"
              type="object:Object"/>
         <xs:element name="ObjectRole"
              type="event:ObjectRoleInEventType"/>
         <xs:element name="InvolvementPeriod"
              type="period:Period"/>
        </xs:sequence>
       </xs:extension>
      </xs:complexContent>
     </xs:complexType>
    </xs:element>
    [...]
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
```

```
[
:ObjectEvent rdf:type owl:Class ;
   rdfs:subClassOf :AssociationClass ;
   rdfs:label "ObjectEvent" .

:hasObject rdf:type owl:ObjectProperty ;
    rdfs:domain event:ObjectEvent ;
    rdfs:range object:Object ;

:hasEvent rdf:type owl:ObjectProperty ;
    rdfs:domain object:ObjectEvent ;
    rdfs:range event:Event ;

:hasInvolvementPeriod rdf:type owl:ObjectProperty ;
    rdfs:domain object:ObjectEvent ;
    rdfs:range period:Period ;
]
```

Enumérations Les énumérations en XML définissent les types possibles d'entités spécifiques. Contrairement aux propositions de conversion des outils existants, qui utilisent le constructeur `owl:oneOf` sur les valeurs possibles, une solution plus élégante consiste à définir une classe `:EnumerationType` dont les valeurs possibles énumérées sont des instances. Inspirés de [14], nous choisissons de représenter les énumérations comme des classes (`owl:Class`) qui possèdent en outre une liste prédéfinie d'instances. La plupart des axiomes de l'ontologie proviennent alors des énumérations du modèle de données.

```
<xs:simpleType name="VesselType">
 <xs:restriction base="xs:string">
  [...]
  <xs:enumeration value="Ferry">
  </xs:enumeration>
  <xs:enumeration value="Fishing">
  </xs:enumeration>
  [...]
 </xs:restriction>
</xs:simpleType
```

```
:VesselType rdf:type owl:Class ;
            rdfs:subClassOf :EnumerationType ;
            rdfs:label "VesselType" .

:Ferry rdf:type owl:NamedIndividual, :VesselType ;
       rdfs:label "Ferry" .

:Fishing rdf:type owl:NamedIndividual, :VesselType ;
         rdfs:label "Fishing" .
```

Les énumérations représentent la grande majorité des conversions à traiter. Les requêtes effectuées depuis le point d'accès SPARQL de GraphDB sur l'ontologie eCISE permettent de chiffrer la proportion de classes d'énumération et d'individus les peuplant. Les classes d'association représentent un total de 141 classes sur les 269 de l'ontologie. Ces classes sont peuplées de 16 648 individus.

Contraintes La conversion des contraintes de cardinalité des schémas de données a soulevé des questions sur la pertinence de leur retranscription dans un modèle sémantique géré par l'hypothèse du monde ouvert. En effet, à cause de cette hypothèse, les contraintes de cardinalité ne peuvent indiquer que des maximums ou minimums possibles, mais ne peuvent contraindre une cardinalité multiple. Une autre difficulté vient des valeurs par défaut de ces cardinalités (minOccurs et maxOccurs) en XSD, qui est 1, alors qu'elle doit être explicitée en OWL. Suivant l'utilisation prévue de l'ontologie, les contraintes sont gérées par des systèmes annexes à l'ontologie (à la réception d'un message CISE ou eCISE par exemple) et avant l'utilisation de cette ontologie. Nous proposons d'ajouter une option permettant de choisir si les contraintes de cardinalité doivent être retranscrites ou non lors de la génération des ontologies. Ces contraintes sont représentées par des restrictions OWL sur des éléments de type `owl:ObjectProperty` et `owl:DataProperty`.

```
<xs:complexType name="Vessel">
 <xs:annotation>
  <xs:documentation>The class [...].
  </xs:documentation>
 </xs:annotation>
 <xs:complexContent>
  <xs:extension base="object:Vehicle">
   <xs:sequence>
    [...]
    <xs:element name="ConditionOfTheCargoAndBallast"
    type="vessel:ConditionOfTheCargoAndBallastType" />
    [...]
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
```

```
[ a owl:Restriction ;
  owl:onProperty
vessel:hasConditionOfTheCargoAndBallastMinCardinality ;
  owl:minCardinality 0 .
]
```

5.3 Validation manuelle

La génération de l'ontologie eCISE-OWL à partir du modèle eCISE exige que l'ontologie possède les caractéristiques inscrites dans le modèle de base. Dans cette optique,

il a été nécessaire de vérifier la transformation des propriétés du modèle (entités générées vs. spécifications du modèle du livrable du projet ANDROMEDA [1]). Cette opération a été effectuée manuellement par 3 experts, avec l'aide du logiciel Protege. Pour chaque type d'entité, son étiquette, sa classe et le commentaire associé ont été vérifiés Ce travail de vérification et de correction a nécessité plusieurs ateliers de travail et s'est déroulé sur une vingtaine d'heures.
Au cours de la vérification, les incohérences identifiées impliquent exclusivement les noms des classes d'association. En effet, les étiquettes de ces classes ne sont pas spécifiées dans les fichiers sources .xsd. Deux solutions sont alors possibles pour obtenir un résultat de conversion en accord avec les spécifications : reconstruire le nommage pour qu'il corresponde aux spécifications du modèle de données, en spécifiant dans un fichier le nom de la classe d'association pour deux classes données ; ou alors effectuer la conversion en s'appuyant uniquement sur le nommage spécifié dans les fichiers .xsd. Le premier cas implique une programmation spécifique aux modèles de données CISE et eCISE. Dans le second cas, il est possible de proposer une solution plus générique qui serait utilisable à partir d'autres modèles de données.
Afin de ne pas altérer le modèle de données source l'ontologie doit être conforme à la description du modèle UML des documents PDF), la modification des sources XSD a été écartée des solutions de correction de l'ontologie. Ces corrections ont été apportées par un script Python de renommage ayant pour paramètres les noms de classes incohérents et les noms de classes du livrable ANDROMEDA. Le renommage est effectué de façon itérative.

6 eCISE-OWL

L'ontologie eCISE-OWL contient au total 268 classes, 297 propriétés d'objets et 332 propriétés de données. Les principales métriques de l'ontologie, comparées à celles de l'ontologie EUCISE-OWL, sont résumées dans le Tableau 2. Elles confirment la plus grande exhaustivité de notre ontologie par rapport à la première initiative de création de EUCISE-OWL. La Figure 2 présente la hiérarchie des concepts principaux de l'ontologie.
L'ontologie générée a été évaluée avec différentes métriques. Dans une première évaluation, nous avons utilisé OOPS ! (OntOlogy Pitfall Scanner !)[13] [15], permettant d'évaluer la qualité de modélisation de l'ontologie. Cet outil identifie les erreurs de modélisation selon les dimensions structurelles, fonctionnelles et de profilage de l'utilisabilité. Il fournit également un indicateur (critique, important, mineur) pour chaque écueil identifié, en fonction de l'indice respectif. Dans le cas d'eCISE-OWL, aucun écueil n'a été détecté concernant les dimensions structurelle, fonctionnelle, cohérence, exhaustivité, et concision.

7 Conclusions et travaux futurs

Cet article a présenté l'ontologie maritime eCISE-OWL ainsi que le processus qui a permis de l'obtenir à partir

15. http://oops.linkeddata.es/catalogue.jsp

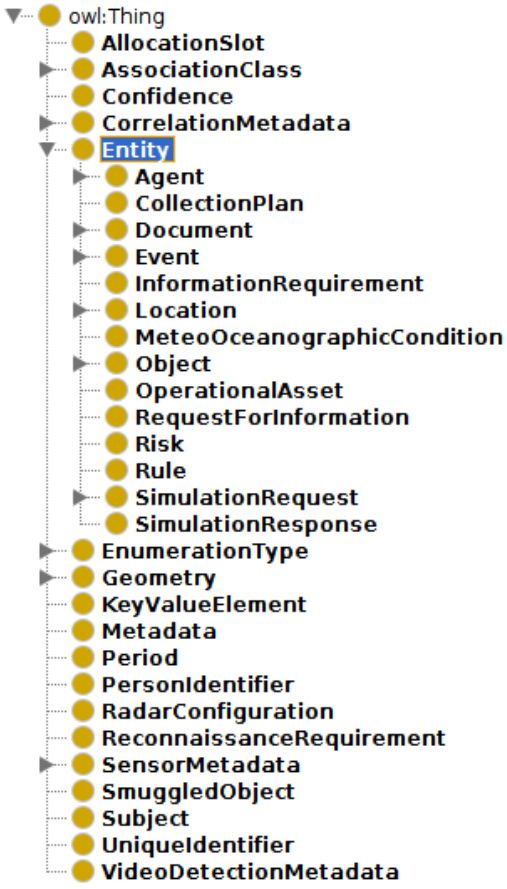

FIGURE 2 – Hiérarchie des concepts principaux d'eCISE-OWL.

du modèle de données eCISE. Ce processus comporte deux étapes : i) la conversion automatique de fichiers XSD en langage OWL selon une approche qui améliore l'état de l'art, et ii) des efforts de validation manuelle par des experts du domaine. Cette ontologie est la première tentative de valorisation du modèle de données CISE suivant une approche sémantique de telle sorte que le résultat (l'ontologie et le code de transformation) soit rendu public et disponible pour tous.

A l'avenir, nous envisageons de poursuivre ce travail suivant plusieurs pistes. Un premier ensemble de perspectives vise à améliorer le processus, décrit dans cet article, de génération d'ontologie à partie d'une source XSD : améliorer l'extraction des terminologies des sources externes car pour certaines classes d'association et pour certaines valeurs d'énumérations, l'extraction de termes des tableaux présents dans les fichiers sources exige l'utilisation d'outils d'extraction d'information plus sophistiqués que ceux que nous avons retenus ; revoir les divers espaces de nom actuellement présents au sein de l'ontologie, pour n'en définir qu'un seul associé à l'ensemble de l'ontologie, afin que toutes les entités aient des URI déréférençables dans cet espace (cela suppose de gérer des relations dont les identifiants sont identiques dans chacun des espaces actuels).

Un deuxième ensemble d'objectifs vise à rendre l'ontologie

Métrique	*eCISE-OWL*	*EUCISE-OWL*
Nombre de classes	268	153
Nombre de propriétés d'objets	297	127
Propriété d'objet - nombre d'axiomes de domaine	336	116
Propriété d'objet - nombre d'axiomes de range	310	116
Nombre de propriétés de données	332	135
Propriété des données - nombre d'axiomes de domaine	350	132
Propriété des données - nombre d'axiomes de range	332	132
Nombre d'individus	16,423	869
Expressivité DL	OWL 2	SHIF(D)
Nombre de triplets	55,322	6,209
Nombre de classes d'association	29	10
Nombre de classes d'énumération	141	

TABLE 2 – Métriques des ontologies eCISE-OWL et EUCISE-OWL [14]

conforme à des principes de partage et de réutilisation. Pour cela, nous allons : générer des alignements entre eCISE-OWL et des ontologies existantes (GeoSPARQL, FOAF, SOSA, etc.) ; rendre l'ontologie entièrement conforme aux principes du FAIR ; gérer les différentes versions des ontologies, issues des différents schémas XSD, y compris la gestion des metadonnées décrivant les ressources non-ontolologiques utilisées comme sources.

Enfin, l'ontologie sera utilisée au sein du système d'information en cours de réalisation dans le projet EFFECTOR, ce qui nécessitera de mettre en place un processus de conversion de messages CISE en messages eCISE en passant par leur représentation en RDF.

Remerciements

Ces travaux sont financés par le programme Horizon 2020 d'innovation et de recherche de l'Union Européenne avec l'accord de financement No. 883374. Ce document ne reflète que la vision des auteurs et la REA (Research Executive Agency) ainsi que la Commission Européenne ne peuvent être tenue responsables pour tout usage de l'information qu'il contient.

Références

[1] Spyros Antonopoulos, Manolis Tsogas, Marios Moutzouris, Antonis Kostaridis, Aggelos Aggelis, and Leonidas Perlepes. Andromeda D.3.1 e-CISE Data Model description. Andromeda project (n. 833881) deliverable, EU, April 2020.

[2] Peb Ruswono Aryan. Converting xml schema to owl in python, 2013.

[3] Nathalie Aussenac-Gilles, Mouna Kamel, Davide Buscaldi, and Catherine Comparot. Construction semi-automatique d'ontologies à partir d'une collection de pages web structurées. In Raphaël Troncy, editor, *IC 2013 : 24es Journées francophones d'Ingénierie des Connaissances (Proceedings of the 24th French Knowledge Engineering Conference), Lille, France, July 1-5, 2013*, pages 165–180, 2013.

[4] I. Bedini, C. Matheus, P. Patel-Schneider, A. Boran, and B. Nguyen. Transforming xml schema to owl using patterns. In *2011 IEEE Fifth International Conference on Semantic Computing*, pages 102–109, 2011.

[5] D. Berger, J. Hermida, F. Oliveri, and G. Pace. The entity service model for cise–service model specifications. Technical report, Joint Research Centre of the European Commission, 2017.

[6] Stefan Brüggemann, Konstantina Bereta, Guohui Xiao, and Manolis Koubarakis. Ontology-based data access for maritime security. In *Proceedings of the 13th International Conference on The Semantic Web. Latest Advances and New Domains - Volume 9678*, page 741–757, Berlin, Heidelberg, 2016. Springer-Verlag.

[7] Diego Calvanese, Benjamin Cogrel, Sarah Komla-Ebri, Roman Kontchakov, Davide Lanti, Martin Rezk, Mariano Rodriguez-Muro, and Guohui Xiao. Ontop : Answering SPARQL queries over relational databases. *Semantic Web*, 8(3) :471–487, 2017.

[8] G.K.D. de Vries, V. Malaisé, M van Someren, P. Adriaans, and A.T. Schreiber. Semi-automatic ontology extension in the maritime domain. In *In Proceedings of the 20th Belgian-Netherlands Conference on Artificial Intelligence (BNAIC 2008)*, 2008. De-Vries :BNAIC2008 University of Twente, Enschede, the Netherlands.

[9] EUCISE2020. Eucise2020 : Technical specifications. deliverable 4.3, revision 1, annex b. Technical report, EUCISE Data Model, 2020.

[10] Mokhtaria Hacherouf, S. N. Bahloul, and C. Cruz. Transforming `XML` schemas into `OWL` ontologies using formal concept analysis. *Software & Systems Modeling*, 18 :2093–2110, 2017.

[11] M. Hagaseth, L. Lohrmann, A. Ruiz, F. Oikonomou, D. Roythorne, and S. Rayot. An ontology for digital maritime regulations. *Journal of Maritime Research*, XIII(II) :7–18, 2016.

[12] Aniello Minutolo, Angelo Esposito, Mario Ciampi, Massimo Esposito, and G. Cassetti. An automatic method for deriving OWL ontologies from XML documents. In *2014 Ninth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, Guangdong, China, November 8-10, 2014*, pages 426–431. IEEE Computer Society, 2014.

[13] María Poveda-Villalón, Asunción Gómez-Pérez, and Mari Carmen Suárez-Figueroa. OOPS ! (OntOlogy Pitfall Scanner !) : An On-line Tool for Ontology Evaluation. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 10(2) :7–34, 2014.

[14] Marina Riga, Efstratios Kontopoulos, Konstantinos Ioannidis, Spyridon Kintzios, Stefanos Vrochidis, and Ioannis Kompatsiaris. EUCISE-OWL : an ontology-based representation of the common information sharing environment (CISE) for the maritime domain. *Semantic Web*, 12(4) :603–615, 2021.

[15] Georgios M. Santipantakis, George A. Vouros, Apostolos Glenis, Christos Doulkeridis, and Akrivi Vlachou. The datacron ontology for semantic trajectories. In Eva Blomqvist, Katja Hose, Heiko Paulheim, Agnieszka Ławrynowicz, Fabio Ciravegna, and Olaf Hartig, editors, *The Semantic Web : ESWC 2017 Satellite Events*, pages 26–30, Cham, 2017. Springer International Publishing.

[16] Mari Carmen Suárez-Figueroa, Asunción Gómez-Pérez, and Mariano Fernández-López. The neon methodology framework : A scenario-based methodology for ontology development. *Appl. Ontology*, 10(2) :107–145, 2015.

[17] Vojtěch Svátek and Ondrej Sváb-Zamazal. Entity naming in semantic web ontologies : Design patterns and empirical observations. In *9th Czecho-Slovak Knowledge Engineering Conference*, 2009.

[18] Arnaud Vandecasteele, Rodolphe Devillers, and Aldo Napoli. A semi-supervised learning framework based on spatio-temporal semantic events for maritime anomaly detection and behavior analysis. In *Coast-GIS 2013 - The 11th International Symposium for GIS and Computer Cartography for Coastal Zone Management*, page 4 pages, Victoria, Canada, June 2013.

[19] Arnaud Vandecasteele and Aldo Napoli. An enhanced spatial reasoning ontology for maritime anomaly detection. In *2012 7th International Conference on System of Systems Engineering (SoSE)*, pages 1–6, 2012.

[20] Arnaud Vandecasteele and Aldo Napoli. Spatial ontologies for detecting abnormal maritime behaviour. In *2012 Oceans - Yeosu*, pages 1–7, 2012.

[21] Boris Villazón-Terrazas and Asunción Gómez-Pérez. Reusing and re-engineering non-ontological resources for building ontologies. In Mari Carmen Suárez-Figueroa, Asunción Gómez-Pérez, Enrico Motta, and Aldo Gangemi, editors, *Ontology Engineering in a Networked World*, pages 107–145. Springer, 2012.

[22] Diego Vinasco-Alvarez, John Samuel Samuel, Sylvie Servigne, and Gilles Gesquière. From CityGML to OWL. Technical report, LIRIS UMR 5205, September 2020.

[23] Mustafa Yüksel. A semantic interoperability framework for reinforcing post market safety studies. Technical report, Middle East Technical University, 2013.

[24] Dong Xia Zheng and Xue Da Sun. A knowledge acquisition model in maritime domain based on ontology. In *Proceedings IEEE International Conference on Security, Pattern Analysis, and Cybernetics, SPAC 2014, Wuhan, China, October 18-19, 2014*, pages 372–375. IEEE, 2014.