Simple Semantic Data Modeling in XML (SeMoX)

Renzo Kottmann
Coordination Office for IT-Standards
<renzo.kottmann@finanzen.bremen.de>

Cedric Pauken
Research Group E-Government, University Koblenz
<cpauken@uni-koblenz.de>

Andreas Schmitz
Research Group E-Government, University Koblenz
<andreasschmitz@uni-koblenz.de>

Abstract

The aim of Simple Semantic Data Modeling in XML (SeMoX) is to provide non-technical domain experts a simple model and additional tooling for capturing semantics of data with a technology-neutral approach. It is foremost designed for modeling data exchange standards between heterogeneous systems. SeMoX is simple because all it needs are five basic concepts: Terms, Semantic Datatypes, Rules, Structures and Syntax Bindings. The core artifact of SeMoX is `semo.xsd`[1]*. This XML Schema defines a concise and linear unfolding XML structure for users to create own SeMoX based semantic data modeling projects. In contrast to many UML based model-driven approaches in standardization, SeMoX is able to leverage the entirety of the fully interoperable XML technology stack. SeMoX is set as the modeling approach of the whole German XEinkauf and already proved to be valuable in production for the development and maintenance of procurement standards such as eForms-DE and XRechnung. SeMoX is open source under a permissive MIT Licence and invites usage and participation.*

For further details see homepagehttps://semo-xml.org [2] *and project repository https://projekte.kosit.org/semox/semox-model.*

Keywords: XML, schema, semantic, modeling, standardization

[1] https://projekte.kosit.org/semox/semox-model/-/blob/master/src/main/xsd/semo.xsd
[2] https://semo-xml.org

1. Introduction

SeMoX provides domain experts a simple model and additional tooling for capturing the semantics of data with a technology-neutral approach. It is foremost specific to data standardization within certain knowledge domains and mainly focuses on the interoperable aspects of data exchange between heterogeneous systems. Hence, SeMoX defines a simple Domain-Specific Language in concise XML to structure semantics of data from and for domain experts. It is designed to separate domain knowledge and intent from technical implementations while keeping clear and close connections.

Therefore, SeMoX is not an ontology that models concepts of knowledge, but rather a semantic framework that supports the capturing and structuring of data semantics using a simple domain-specific language defined in an XML-Schema.

SeMoX can be seen as an amalgamation of years of experience in data standardization. The initial idea behind SeMoX came up based on the necessity to integrate existing international standards for public procurement such as the European Committee for Standardization (CEN) [11] or Peppol [9] into the German XSE-Standards (XStandards Einkauf). The declared goal of XSE is to summarize, harmonize and make interoperable all IT-Standards in German public procurement including the domains of ordering, delivery and payment [8]. As such, XSE needs to adopt and maintain several international standards from different standardization bodies in its own format with certain specific national features. SeMoX was designed to support this adoption and maintenance. It is very much informed by the approach of CEN EN16931 norm [5] while avoiding common challenges present in standardization approaches.

It aims to fulfill the following general goals:

1. Create a shareable and validatable model for the definition of specification and standards
2. Facilitate accessibility for domain experts without XML knowledge
3. Bridge the gap between individuals with differing business, technical and legal expertises
4. Promote and facilitate collaboration and consistency between different standards and standardization organizations

Furthermore, SeMoX must accommodate various simple and complex standardization usage scenarios:

- Writing simple specifications (see SeMoX itself model)
- Composing top-down standards (see XRechnung)
- Composing downstream standards as differential expression to upstream standards (see eForms-DE)

2. Basic concepts of SeMoX

SeMoX leverages several basic concepts for creating simple semantic data models in XML. In order to describe these basic concepts, the following continuous example is the XML representation of SeMoX concepts by SeMoX-itself. Utilizing the dogfooding principle allows fully focusing on SeMoX and avoids discussing SeMoX concepts in parallel to other concepts of another data model.

For clarity, we denote a concept by its name written in capitalization, e.g., Rule, XML representation by its tag names, e.g., `rule` and other names or values are in quotes if it is necessary to differentiate them from concept and tag names, e.g., an `id` with the value "rule".

2.1. Term

At the heart of SeMoX is the Term concept (a.k.a. Business Term (BT)). A Term is a word or a phrase that describes a concept that is used in a particular knowledge or business domain. Whatever concept a Term expresses, domain experts have to name and describe it. Hence, the basic XML representation of a Term is the following:

```
<term id="term">
  <name>Term</name>
  <description>A Term to be defined by domain experts</description>
</term>
```

This XML instance of a Term is used to define the semantics of a Term.

An excerpt from the SeMoX-itself model illustrates the definition of a list of terms:

```
<definitions>
    <term id="model" datatype="group">
        <name>Model</name>
        <description>The model describing and capturing the semantics
 of data using Semantic Datatypes, Terms, Codelists etc..
        </description>
    </term>
    <term id="term" datatype="string">
        <name>Term</name>
        <description>A Term to be defined by
                    domain experts</description>
    </term>
    <term id="identifier" datatype="internal-id">
        <name>Identifier</name>
        <description>An identifier</description>
    </term>
    <term id="name" datatype="string">
```

```
        <name>A Name of something</name>
        <description>Term for giving something a name
                    (sometimes in the sense of a label)</description>
    </term>
    <term id="description" datatype="string">
        <name>Description</name>
        <description>Human readable explanation</description>
    </term>
    <term id="definitions" datatype="group">
        <name>Definitions</name>
        <description>A list of <term>term</term>s.
        </description>
    </term>
</definitions>
```

This list of `terms` is flat and each `term` has to have a unique identifier with the XML representation of an `id` attribute.

A SeMoX model with only a list of Terms can be regarded as a glossary.

2.2. Semantic Datatype

Because datatypes are fundamental to software and programming languages in general, SeMoX establishes the Semantic Datatype concept.

Whatever the semantic or intent of a `datatype` is, domain experts have to name and describe it.

Hence, the basic XML representation of a Semantic Datatype is as follows:

```
<datatype id="string">
    <name>String</name>
    <description>Sequence of characters</description>
</datatype>
```

This XML instance of a Semantic Datatype is used to define the semantics of the String concept.

Overall, Semantic Datatypes are nothing more than high-level declarations of computer representations of data. But already, as such, they give valuable information for technical implementations.

Therefore, each definition of a `term` includes a declaration of a `datatype`.

```
<term id="description" datatype="string">
    <name>Description</name>
    <description>Human readable explanation.</description>
</term>
```

In this example, the `term` with `id` "description" expresses the Description concept and is declared to have the Semantic Datatype with name "String" herewith referred to by `id` of value "string".

Some more Semantic Datatypes are defined in the SeMoX-itself model:

```
<semantic-datatypes>
    <datatype id="any" minOccurs="1" maxOccurs="1">
        <name>Any</name>
        <description>The ur-type, basically no restriction
                      on data values.</description>
    </datatype>
    <datatype id="group" minOccurs="1" maxOccurs="1">
        <name>Group</name>
        <description>Group of <term>term</term>s.
        </description>
    </datatype>
    <datatype id="string" minOccurs="1" maxOccurs="1">
        <name>String</name>
        <description>Sequence of characters</description>
    </datatype>
    <datatype id="internal-id" minOccurs="1" maxOccurs="1"
              restricts="string">
        <name>Internal Identifier</name>
        <description>An id which is unique within a
                     single structure</description>
    </datatype>
</semantic-datatypes>
```

Which and how many semantic datatypes are to be defined in a SeMoX model is a design choice.

The SeMoX-itself model defines the semantic datatype "Any" which can be used as a placeholder in `term` definitions where name and definition are already defined, but a decision on the `datatype` is missing.

The sole purpose of the Semantic Datatype `group` is to indicate that a `term` of this datatype is a composition of other terms.

For example, the `term`"model" includes the terms `term`, `semantic datatype` and several other terms.

```
<term id="model" datatype="group">
  <name>Model</name>
  <description>The model describing and capturing the semantics
               of data using a composition of Semantic Datatypes,
                 Terms, Codelists etc..</description>
</term>
```

2.3. Structure

Terms can be aggregated to form a Semantic Structure in order to express which Term belongs to which group of Terms. Such Structures can be defined according to the intents and needs of the domain as defined by experts. This is independent

of concrete implementations. A Structure also defines the cardinalities for all Terms and groups of Terms. The default cardinality is being optional, the two other cardinalities are mandatory or forbidden. Additionally, a Term or group of Terms can occur repeatably. While one Semantic Structure may specify the cardinality of a certain Term as mandatory, another structure may choose to define it as optional or forbidden - depending on the business context.

The following XML representation shows a partial semantic Structure of the SeMoX-itself model.

```
<structures>
  <structure id="semox">
    <name>Logic Structure</name>
    <type/>
    <description/>
    <message>
      <group ref="model">
        <group ref="metadata">
          <term ref="shortname"/>
        </group>
        <group ref="definitions">
          <group ref="term">
            <term ref="identifier"/>
            <term ref="name"/>
            <term ref="description"/>
            <term ref="name"/>
          </group>
        </group>
        <group ref="rules">
          <group ref="rule" maxOccurs="unbounded">
            <term ref="identifier"/>
            <term ref="name" minOccurs="0" maxOccurs="unbounded"/>
            <term ref="description" maxOccurs="unbounded"/>
            <group ref="rule-implementations" minOccurs="0">
              <term ref="schematron" minOccurs="0"/>
            </group>
          </group>
        </group>
      </group>
    </message>
  </structure>
</structures>
```

2.4. Syntax Binding

Parsing of data is a necessary operation for data exchange. Therefore, SeMoX has the concept of Syntax Binding. The explanation follows the example:

```
<syntax-bindings>
  <binding>
    <structure>semox</structure>
    <syntax id="model-syntax">
      <name>SeMoX Model</name>
      <type>xml</type>
      <query-language>xpath</query-language>
    </syntax>
    <term ref="model" path="/model"/>
  </binding>
</syntax-bindings>
```

This binds the "semox" Structure as previously defined (see Section Structure) to XML instances named "SeMoX Model". Therefore, the Term "model" can be found i.e. queried with the XPath expression `/model` in all such instances. In other words, the actual data with the semantics as defined by the `term` with the referenced `id` "model" are to be found in XML instances with the root element also named `model`.

2.5. Rule

Data Rules are means of further imposing restrictions, assertions and constraints on the use of terms and structures. On principle, these can be of any kind. Although Rules can be defined in a variety of ways, they are most useful when they can be tested via a simple yes or no answer. In SeMoX, `rules` should therefore be developed from a business perspective and be validatable, as business rules set the expectation levels against which the technical implementation must be measured against.

Additionally, Rule-Groups can be defined. A set of Rules may belong to several Rule-Groups. Each Rule-Group needs an `id`, a `name` and a `description`, whereas each rule only needs an `id` and a `description`. Ideally, Rules should specify a list of terms on which they apply to and provide references regarding implementations. The implementation and validation of rules works via schematron.

Provided below is another simple XML example from the SeMoX itself model. As discussed earlier, each structure contains terms and groups. A Rule may now enforce that, e.g., each structure must begin with a root group. To do this, a `rule-group` that contains rules about the use of datatypes must be defined first. Subsequently, a `rule` that mandates structures to begin with a root-group can now be specified. In practice, when building the model, this rule can easily be tested via a simple yes or no answer.

```
<rules>
  <rule-group id="datatype-rules">
    <name>Datatype Rules</name>
```

```
    <description>Rules about datatype use</description>
  </rule-group>
  <rule id="root-group" groups="datatype-rules">
     <description>Any message in a structure must begin with a
                  term of the datatype "group"</description>
  </rule>
</rules>
```

3. Schema design of SeMoX

The previous section followed the dogfooding principle and explained the basic concepts for semantic modeling of data in XML by using XML for modeling the data and capturing the semantics of SeMoX-itself. Accordingly, an XML Schema (XSD) [6] was developed in order to describe and validate all kinds of SeMoX based data models (`semo.xsd`).

This XSD makes some more notable features of SeMoX models possible:

1. *Model Metadata*: Each model can have metadata for author attribution, lifecycle management (`version`, `status`) etc.

Example 1. Metadata section of the current eForms-DE model.

```
<m:meta>
  <m:shortName>eForms-DE</m:shortName>
  <m:name>eForms-DE Standard</m:name>
  <m:id>eforms-de</m:id>
  <m:extends version="1.10.1" id="eforms-eu">
    <m:name>eforms-eu</m:name>
  </m:extends>
  <m:version>1.2.0</m:version>
  <m:status>active</m:status>
  <m:date>2024-02-02</m:date>
  <m:abstract>Deutsche Anpassung und Erweiterung der
 "DURCHFÜHRUNGSVERORDNUNG (EU) 2022/2303  DER KOMMISSION
  vom 24. November 2022 […].
</m:abstract>
</m:meta>
```

2. *Multilingual support*: Every basic concept has a `name` and a `description`, each of which can be repeated in different languages denoted by the standard `xml:lang` attribute

```
<m:term id="t1" datatype="sdt-1">
  <m:name>Nonsense term 1</m:name>
  <m:name xml:lang="de">Nonsense in Deutsch</m:name>
  <m:description>A term using a semantic datatype.</m:description>
  <m:description xml:lang="de">Eine deutsche Beschreibung.</
```

```
m:description>
</m:term>
```

3. *Extensible*: Each element allows attributes from different namespaces. Almost all elements with a complex content model may have additional sub-elements from different namespaces.

4. Usage scenarios in practice

In order to showcase some aspects of SeMoX's versatility and utility, some real world usage scenarios of SeMoX are discussed. Common to all scenarios is the requirement to have a book publication made available in PDF format which serves as the standard publication. Each standard has varying requirements on content, style and layout in terms of cover page design, additional chapters, and other aspects. Therefore, a custom publishing pipeline was developed. It is based on Docbook and the Ant build tool. The primary source of content (in publishing lingo) is the SeMoX model, whereas Docbook fragments and markdown files serve as secondary input.

The results are publicly available open standards like eForms-DE (specification[3] and model[4]) and XRechnung (specification[5]).

4.1. SeMoX-itself

The actual design of the SeMoX-itself model instance serves two purposes. First, during development, it illustrates how to best represent the SeMoX concepts in XML. The XSD is manually developed in parallel to the development of the XML representation of SeMoX concepts. This separate design process forces clear separation of what is semantically important and needs description in the model as well as what are technical necessities, trivialities and specifics of XSD. This parallel process as exercised with SeMoX-itself includes iterations and considerations going back and forth between a semantic description and design of actual XML representation. This is and will be common to all data standard developments from scratch. Second, it also demonstrates the two common standard artifacts of SeMoX as a result of writing a specification a validation artifact: Here, having a validatable specification in a SeMoX model instance and an XSD for validation.

[3] https://xeinkauf.de/app/uploads/2024/02/specification-eforms-de-v1.2.0.pdf
[4] https://projekte.kosit.org/eforms/eforms-de-specification/-/blob/master/src/main/model/eforms-de-model.xml
[5] https://xeinkauf.de/app/uploads/2023/09/301-XRechnung-2023-09-22.pdf

4.2. XRechnung

The key characteristics of the development of the XRechnung standard are that XRechnung is a top-down standard and has one semantic data model which has to be bound to two vastly different XML syntaxes.

Here, top-down standard means practically that there is the European Norm CEN 16931 for B2G (Business-to-Government) electronic invoicing published by CEN which includes several specifications in PDF format of which the CEN 16931-1 document is normative. This norm is on top because all European member states are required to adhere to the norm by law. However, because there are too many nationally specific requirements on electronic invoices, the norm allows "down"stream standards to deviate from the norm by certain normative rules. Hence, XRechnung is the German downstream standard with some additional requirements on electronic invoices in the German market compliant to the CEN Norm. At the heart of the CEN Norm is a single and explicit semantic model of a core invoice. This core model is then bound to the XML invoice representation as defined by the Universal Business Language (UBL) and also Cross Industrie Invoice (CII) by UN/CEFACT. Therefore, the major achievement of CEN EN-16931 is to semantically integrate UBL and CII in a way that both syntaxes can practically be used for invoicing while ensuring semantic interoperability.

In the following the Term designated "BG-30" is used to show how this semantic integration is declared in the XRechnung SeMoX model:

```
<m:group ref="BG-30">
  <m:term ref="BT-151" />
  <m:term ref="BT-152" minOccurs="0" />
</m:group>
```

then the Syntax Binding with UBL for the above Terms is as follows:

```
<m:term ref="BG-30"
         path="/Invoice/cac:InvoiceLine/cac:Item/
cac:ClassifiedTaxCategory"/>
<m:term ref="BT-151"
        path="/Invoice/cac:InvoiceLine/cac:Item/
cac:ClassifiedTaxCategory/cbc:ID"/>
<m:term ref="BT-152"
        path="/Invoice/cac:InvoiceLine/cac:Item/
cac:ClassifiedTaxCategory/cbc:Percent"/>
```

and the Syntax Binding with CII for the above Terms is as follows:

```
<m:term ref="BG-30"
         path="/rsm:CrossIndustryInvoice/rsm:SupplyChainTradeTransaction/
ram:IncludedSupplyChainTradeLineItem/ram:SpecifiedLineTradeSettlement/
ram:ApplicableTradeTax">
```

```
<m:term ref="BT-151"
        path="/rsm:CrossIndustryInvoice/rsm:SupplyChainTradeTransaction/
ram:IncludedSupplyChainTradeLineItem/ram:SpecifiedLineTradeSettlement/
ram:ApplicableTradeTax/ram:CategoryCode"/>

<m:term ref="BT-151"
        path="/rsm:CrossIndustryInvoice/rsm:SupplyChainTradeTransaction/
ram:IncludedSupplyChainTradeLineItem/ram:SpecifiedLineTradeSettlement/
ram:ApplicableTradeTax/ram:TypeCode"/>

<m:term ref="BT-152"
path="/rsm:CrossIndustryInvoice/rsm:SupplyChainTradeTransaction/
ram:IncludedSupplyChainTradeLineItem/ram:SpecifiedLineTradeSettlement/
ram:ApplicableTradeTax/ram:RateApplicablePercent"/>
```

The Syntax Binding of CII includes to different XPath for "BT-151" which is ambiguous and might be of disadvantage in certain situations. But, here it shows that SeMoX allows to declaratively make explicit the ambiguity of certain bindings. Moreover, it is conceivably simple to use a SeMoX for semantic integration of different data format standards such as UBL and CII.

4.3. eForms-DE

Currently, the Coordination Office for IT-Standards in Germany (CoSIT) develops and maintains the German downstream standard eForms-DE of the upstream EU eForms regulation (“Commission Implementing Regulation (EU) establishing standard forms for the publication of notices in the field of public procurement and repealing Implementing Regulation (EU) 2015/1986 (“eForms”)”[6]) . Similar to the above XRechnung setting, this regulation also allows national deviations from the EU regulation in order to allow member states to implement national specific requirements. The EU requirements and interoperability with national requirements on notices in the domain of public procurement are rather complex. Therefore, the regulation defines several hundreds of Terms for more than 40 different notices (Structures in SeMoX) and 3 different syntaxes where each Term has different cardinality requirements and sometimes slightly different semantics. Currently, the upstream definition and technical implementations change on a bi-monthly basis.

Hence, the CoSIT implemented a "tailoring process" based on SeMoX. The idea is to upstream create a SeMoX model of each new version of "eForms-EU" and then to downstream always re-create a national SeMoX eForms-DE model. For this, SeMoX was extended to declare only the semantically specified deviations from - or in other words, tailoring of the eForms-EU model. The creation of

[6] https://ec.europa.eu/docsroom/documents/43488

the eForms-EU SeMoX model is a complex integration of several sources of information and hence an own converter was developed and is available at https://projekte.kosit.org/eforms/eforms-converter.

In order to explain the actual tailoring step in the process an excerpt of the Model Tailoring is shown here:

```
<m:model-tailoring version="0.4.0" xml:lang="de" xmlns:m="semantic-model"
  xmlns:e="eforms">
  <m:meta>
    <m:shortName>eForms-DE</m:shortName>
    <m:name>eForms-DE Standard</m:name>
    <m:id>eforms-de</m:id>
    <m:extends version="1.10.1" id="eforms-eu">
      <m:name>eforms-eu</m:name>
    </m:extends>
    <m:specification />
    <m:version>1.2.0</m:version>
       <!-- … for brevity -->
  </m:meta>

  <m:change-datatypes>
    <m:datatype tailor="code">
      <m:name mode="add" xml:lang="de">Code</m:name>
      <m:description mode="add" xml:lang="de"
        >Werte aus einer definierten Codeliste.</m:description>
    </m:datatype>
    <!-- … for brevity -->
  </m:change-datatypes>
<!-- … for brevity -->
  <m:change-terms>
    <m:term tailor="BG-7141">
      <m:description xml:lang="de" e:eg="3"
        >Einzelheiten zu den Fahrzeugen im
Anwendungsbereich des Saubere-Fahrzeuge-Beschaffungs-
Gesetzes zur Umsetzung der Richtlinie 2009/33/EG.
      </m:description>
    </m:term>
    <!-- … for brevity -->
  </m:change-terms>
  <m:rules>
    <m:rule id="BR-DE-20" on-terms="BT-06 BT-775 BT-776" e:eg="prea-6">
      <m:description>Wenn BT-06 (Strategische Beschaffung) einen Wert
 ungleich "entfällt" enthält, ist mindestens eines der Felder BT-774
 (Grüne Beschaffung BT-775 (Soziale Beschaffung) oder BT-776
(Beschaffung von Innovation) zu füllen.</m:description>
    </m:rule>
```

```
    <!-- … for brevity -->
  </m:rules>
<m:model-tailoring>
```

The XML representation of the Model Tailoring follows the structure of a normal `model`but named `model-tailoring`to distinguish the purpose. The tags `change-datatypes` and `change-terms`to declare the changes to be made w.r.t. the upstream model. The nested elements are of same structure as in a normal `model` except that each element has a `tailor` attribute instead of an own `id` attribute which identifies the respective element in the upstream model:

```
<m:datatype tailor="code">
   <m:name mode="add" xml:lang="de">Code</m:name>
   <m:description mode="add" xml:lang="de"
        >Werte aus einer definierten Codeliste.</m:description>
</m:datatype>
```

here `tailor="code"` states that the Semantic Dataype "code" is to be changed with the nested information. The attribute `mode="add"` of the `name` element tells a processor that first the element `name` is not to be expected to exist in the upstream model and second to add it in the downstream model. Currently, a simple XSLT script exists which has an upstream model and a model-tailoring instance as input and generates the "tailored" model.

Overall this allows maintaining a stable semantic definition of specific German changes to eForms-EU with an own independent life cycle of the upstream changes and rate of changes. This approach to semantic data modeling in a complex and highly changing setting proves to be very useful in many ways. First, by creating an upstream SeMoX model from different sources and then validating this model made possible to find several mistakes and inconsistencies in the upstream standard. Second, being able to define German deviations and re-apply them allowed a German expert group - involving dozens of people from many different institutions and ministries - to work on its own pace with its own quality control, and procedures independent from the rate of changes. Practically, Germany does skip several upstream versions while maintaining own "model-tailoring" versions and only re-apply model-tailoring at own defined time points. Extensive automatic validating and testing throughout this process also ensures full interoperability to upstream eForms-EU.

The current status of eForms-DE is available: eForms Model[7]

5. Conclusion and future work

The way SeMoX is conceptualized and technically implemented proved to be viable in different real world usage scenarios.

[7] https://projekte.kosit.org/eforms/eforms-ts/-/blob/master/eforms-model.xml

The SeMoX-itself usage scenario exemplifies writing specifications in XML for XML instance validation.

The domain knowledge is captured in a model instance separated from the technical implementation. The model instance is shareable (see `semox-itself-model.xml`[8]). The model is validatable using the `semo.xsd` implementation of SeMoX.

The XRechnung usage scenario demonstrates that the separation allows for the binding of common and harmonized semantics to two different XML Syntaxes which already existed and have completely different XML Schemas.

Last, the eForms scenario demonstrates that the extensible design of the SeMoX implementation makes it possible to only capture the semantic difference of a downstream standard and create a full SeMoX instance as a tailored version of the upstream standard. This makes it possible to accommodate any rate of change of an upstream standard.

Overall, this demonstrates the versatility and utility of SeMoX in even rather complex standardization scenarios which are modeling data exchange standards between heterogeneous systems. By no means it attempts to provide a general approach to semantic data modeling. Capturing and making explicit semantics of data is a long-term research topic. A plethora of discussions of what semantics actually is, including capturing and modeling approaches and solutions, have been held, published and deployed. Among which are Resource Description Framework ([15]), Simple Knowledge Organisation System ([16]) and Topic Maps ([17]). Whereas RDF has a very broad scope and a generalized "subject-predicate-object" graph model for describing resources in general, the scope of Topic Maps and SKOS is on modeling and capturing real world knowledge. A full and detailed discussion of commonalities and differences between the aforementioned approaches and SeMoX is beyond the limits of this paper. However, the differentiation is mainly driven by the narrow scope of modeling data for interoperable data exchange scenarios between heterogeneous systems. SeMoX´ scope is neither on general descriptions of resources nor on knowledge and concepts and relations between these. Hence, the concept Term only includes a name and description. And consequently, the concept of Structure is not meant to express semantic relations between knowledge concepts, it just only groups Terms in order to describe that a group of data elements together represent a more complex data description such as an address that consists of a street name and a city name. Therefore, SeMoX is dubbed simple.

As demonstrated SeMoX is capable of achieving its outlined goals. It is in productive use and a core component of XEinkauf (XSE). However, the further facilitation of data standardization and interoperability between different standards

[8] https://projekte.kosit.org/semox/semox-model/-/blob/master/src/main/model/semox-itself-model.xml

will be the focus of SeMoX's ongoing development and is discussed in the following.

5.1. Facilitate more effective data standardization

Many XML based data standards already exist today. Most of which have specification documents and validation artifacts of various sorts. The content and comprehensibility vary for non-technical domain experts, whereas SeMoX based standards have a clear focus on only including semantics and related business intent in specifications. Thus, technical implications which can be solved and implemented in validation artifacts and related tools are left out. This makes SeMoX based standard developments more amenable to non-technical domain experts and therefore facilitates more effective data standardization w.r.t. correctness and accuracy of a knowledge domain's semantics. To illustrate this: In a group of experts, it only needs one person who can encode the results of expert discussions in a SeMoX model and generate a specification document. All others only need to know the knowledge domain and can verify the correctness based on the specification. Or, the other way around: SeMoX allows limited options to "dilute" the model with technicalities and gives full freedom to technical implementations.

Moreover, every data standard based on SeMoX can validate its SeMoX model instance with `semo.xsd`. Consequently, many aspects of semantic consistency (e.g. does every term use a defined datatype? Are all terms used in a certain structure?) can be validated. For example, to ensure consistency in eForms-DE, the project includes dozens of additional Schematron rules which are specific to the context of this domain and standard.

Although, SeMoX models are not validation artifacts by themselves, the models can be used as input data for tests which check consistency between semantic model and validation artifacts during standard development. A very simple consistency check can be implemented if a SeMoX `rule` has the same `id` as a Schematron `rule` or `assertion` which implements the corresponding SeMoX rule.

A SeMoX model instance with `syntax-bindings` and hence having information on an `xpath` for any given `term` can be used to generate simple occurrence/existence constraints. It is also conceivable to generate simple XSDs from models based on the information of semantic datatypes and structures including occurrences.

Taken together, SeMoX already facilitates more quality controlled and effective data standardization. Further design and tooling will seek to increase effectiveness.

5.2. Enhancing interoperability between different standards and standard organisations

Other standardization approaches such as Peppol and CEN encounter common challenges that hinder their effectiveness and interoperability. One such challenge is their tendency to prioritize either technical or business perspectives, creating difficulties for editors from the opposing viewpoint to participate effectively. This imbalance can lead to misunderstandings, inefficiencies, and barriers to collaboration, especially when onboarding new people. Additionally, these approaches necessitate vastly different production environments with external dependencies. For instance, Peppol relies on a rigidly defined project and file structure based on different XML and YAML technologies. While such requirements may ensure consistency within each standard, they pose significant hurdles for individuals with varying backgrounds and skill sets. Furthermore, changes in these dependencies can lead to delays and complications, disrupting implementation efforts and hindering progress. Moreover, the lack of interoperability between standards further amplifies these challenges. Without seamless integration and compatibility between different standardization approaches, organizations face difficulties in mapping between data and standards across diverse standardization environments. This limitation stifles innovation and hampers the potential benefits of standardization efforts. Addressing these challenges requires a concerted effort to bridge the gap between technical and business perspectives, harmonize production environments, and enhance interoperability between standards.

SeMoX is addressing these gaps by providing an interoperable methodology for standard definition, acting as a bridge between various different actors, organizations or processes that need to collaborate. First, SeMoX endeavors to bridge the gap between individuals with technical and business/legal expertise that often have challenges in communicating on the same level. By accommodating both perspectives, it promotes understanding and cooperation among diverse stakeholder groups. Second, SeMoX seeks to bridge the gap between maintenance systems associated with different standards. By adopting a technology-neutral approach, it enables integration and collaboration across various systems, ensuring adaptability and compatibility with evolving external technologies. Finally, SeMoX aims to bridge the gap between the technical formats of different standards, enhancing interoperability between different standardization approaches. By offering a common foundation for standard definition, it promotes consistency and facilitates collaboration between different standardization organizations.

One way to further foster collaboration would be to integrate the concept of transactions into SeMoX. European Peppol [10] and CEN [12] standards commonly use transactions to standardize the XML-message-flow between a contracting body and an economic operator in different phases of the procurement process. Most notably, transactions are used in the pre-award phase of the pro-

curement process to express the concept of a request and a response between a contracting body and an economic operator. Within a pre-award process, one actor might need to request information from another actor via a standardized request-transaction. The other actor can then fill out the required information from the request-transaction and transmit a standardized response-transaction back to the requester. Integrating the concept of transactions into SeMoX would lead to full interoperability between SeMoX-standards and European Peppol/CEN standards in the domain of public procurement. Therefore, SeMoX aims to develop fully EU-interoperable procurement standards in the future by adjusting the current framework to include transactions. In practice, this may work by allowing for the definition of several transactions within a single SeMoX-instance (similar to `structures`).

Additionally, more practical research focused on the interoperability between SeMoX and other standardization methodologies is planned.

First, conceptional mappings between SeMoX and Peppol / CEN should be created. Based on these mappings, transformations from Peppol / CEN to SeMoX can be created and tested. A first step may be an XSL transformation from Peppol to SeMoX. More such technical studies can provide further insights into how SeMoX can act as a bridge between standardization methodologies.

5.3. Further research

Overall, this paper demonstrates how SeMoX can be utilized to provide a semantic standardization model. Many current issues in the domain of standardization can be specifically addressed by SeMoX. This research provides an introduction to the concepts and benefits of SeMoX and is supposed to be the foundation for further research. Literature shows a clear need for methodologies and frameworks in the domains of standardization and public procurement in particular, that SeMoX can address [14]. Further SeMoX should also be considered as a part for more holistic frameworks regarding interoperability, for example as an extension of the Framework for Interoperable Service Architecture Development [13]. Besides the technical additions to SeMoX discussed above in detail, further generic research needs on SeMoX include:

1. Comprehensive classification of SeMoX in the status quo of scientific literature to ensure rigor

2. Empirical verification of SeMoX's applicability in practice by conducting a structured evaluation of its effectiveness across a wide range of different use-cases

3. Development of methodological frameworks for best-practice applications of SeMoX

Bibliography

[1] Kay, Michael: *XSLT 2.0 and XPath 2.0*. Wiley Publishing, 2008.

[2] Gao, Shudi– Sperberg-McQueen, C.M. – Thompson, Henry S.: *W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures*. W3C Recommendation, 5 April 2012. https://www.w3.org/TR/xmlschema11-1/

[3] Jelliffe, Rick. *Schematron*, 1999. Retrieved from http://xml.ascc.net/schematron

[4] *XRechnung*. Retrieved from https://www.xoev.de/de/xrechnung

[5] *Electronic invoicing - Part 1: Semantic data model of the core elements of an electronic invoice; German version EN 16931-1:2017*. Retrieved from https://www.din.de/de/mitwirken/normenausschuesse/nia/normen/wdc-beuth:din21:274990963

[6] *Validation artefacts for the European eInvoicing standard EN 16931*. Retrieved from https://github.com/ConnectingEurope/eInvoicing-EN16931

[7] *Universal Business Language Version 2.1.*. 04 November 2013. OASIS Standard. http://docs.oasis-open.org/ubl/os-UBL-2.1/UBL-2.1.html

[8] *XStandards Einkauf - Die Standardfamilie des öffentlichen Einkaufs*. https://xeinkauf.de/

[9] *Peppol (Pan-European Public Procurement OnLine)*. https://peppol.org/

[10] *Peppol Pre-Award Profiles Overview*. Retrieved from https://docs.peppol.eu/pracc/

[11] *About the European Committee for Standardization (CEN)*. https://www.cencenelec.eu/about-cen/

[12] *CEN/TC 440 'Electronic Public Procurement'*. https://www.cencenelec.eu/areas-of-work/cen-cenelec-topics/public-procurement/cen-tc-440-electronic-public-procurement/

[13] *Framework for interoperable service architecture development*. https://www.sciencedirect.com/science/article/abs/pii/S0740624X23000692

[14] *Closing the gap: Leveraging data for seamless integration between pre-award and post-award in public procurement*. https://scholarspace.manoa.hawaii.edu/items/be322fe7-6932-4132-baea-af702e6081a5

[15] *RDF 1.1 Concepts and Abstract Syntax*. https://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/

[16] *SKOS Simple Knowledge Organization System Reference*. https://www.w3.org/TR/skos-reference/

[17] *The TAO of Topic Maps*. https://www.ontopia.net/topicmaps/materials/tao.html