Applying the Akoma Ntoso XML Schema to Japanese Legislation

KAWACHI GEN[*]**, NAGAI AKIRA**[†]**, NAKAMURA MAKOTO**[‡]**, OGAWA YASUHIRO**[§]**, OHNO TOMOHIRO**[**]**, TOYAMA KATSUHIKO**[††]

Abstract

This paper presents a case study for avoiding ambiguous annotation when applying Akoma Ntoso, an XML schema for parliamentary documents, to Japanese statutes, which follow strict drafting rules. The Japanese statutory schema is designed to reflect the rules, while the Akoma Ntoso schema has underlying structural ambiguity due to its flexibility. We propose a method to convert from the former to the latter schema and provide a subset of Akoma Ntoso that retains a strict annotation approach.

1 Introduction

Japanese statutes have maintained a common descriptive style following a Japanese legislation drafting manual for more than one hundred years. This manual includes strict rules about the hierarchical structure of statutory documents, the typefaces of sequential numbers, grammatical expressions peculiar to statutory documents, the notation of characters (letters), and so on. Noteworthy is that these rules have been maintained for all statutes, which facilitates their understanding.

Some of the statutes are provided in extensive markup language (XML) following a schema designed exclusively for the structure of Japanese

[*] KAWACHI Gen, final year Master's student at Graduate School of Information Science, Nagoya University.

[†] NAGAI Akira, final year student at School of Engineering, Nagoya University.

[‡] NAKAMURA Makoto, Doctor of Information Science, Designated Assistant Professor at Japan Legal Information Institute, Graduate School of Law, Nagoya University.

[§] OGAWA Yasuhiro, Doctor of Engineering, Associate Professor at Information Technology Center and Graduate School of Information Science, Nagoya University.

[**] OHNO Tomohiro, Doctor of Information Science, Assistant Professor at Information Technology Center and Graduate School of Information Science, Nagoya University.

[††] TOYAMA Katsuhiko, Doctor of Engineering, Professor at Information Technology Center and Graduate School of Information Science, Nagoya University.

statutes.[1] This schema is so strict that it basically does not allow a document to have ambiguous structures. In other words, this schema can be regarded as a digitised version of the Japanese legislation drafting manual. In this paper, we consider how to disseminate Japanese legislation while maintaining this strict annotation approach. The key issue is how to adapt an international standard for XML schemata to Japanese legislation.

Akoma Ntoso (Architecture for Knowledge-Oriented Management of Any Normative Texts using Open Standards and Ontologies),[2] is a jurisdiction-independent XML standard that can be used for interchange between public organisations or business enterprises and as a platform for generic legal software.[3] Although Akoma Ntoso was originally created to share legislative documents among African countries, it is now widely used as the OASIS (Organization for the Advancement of Structured Information Standards) *LegalDocumentML* [4] which was developed based on Akoma Ntoso's specifications. In fact, a number of government bodies in several countries, such as EUCases,[5] LexML Brazil,[6] and the Serbian judiciary,[7] apply this schema to legal documents. Flexibility is a desirable trait of Akoma Ntoso, allowing many jurisdictions to employ this schema. However, this *flexibility* can cause ambiguous annotation. For example, there are no restrictions among the elements for hierarchical structures because the Akoma Ntoso

---

[1]   Katsuhiko Toyama, Daichi Saito, Yasuhiro Sekine, Yasuhiro Ogawa, Tokuyasu Kakuta, Tariho Kimura, and Yoshiharu Matsuura, 'Design and Development of Japanese Law Translation Database System' (Paper presented at the Law via the Internet Conference 2011, University of Hong Kong, 9 June 2011).

[2]   Monica Palmirani and Fabio Vitali, 'Akoma-Ntoso for Legal Documents' in Giovanni Sartor, Monica Palmirani, Enrico Francesconi, and Maria Angela Biasiotti (eds), Legislative XML for the Semantic Web (Springer Netherlands, 2011) 75; Akoma Ntoso, Akoma Ntoso <http://www.akomantoso.org/>.

[3]   Alexander Boer, Radboud Winkels, and Fabio Vitali, 'Proposed XML Standard for Law: MetaLex and LKIF' in Arno R Lodder and Laurens Mommers (eds), *Legal Knowledge and Information Systems: JURIX 2007: The Twentieth Annual Conference* (IOS Press, 2007) 19.

[4]   OASIS, *OASIS LegalDocumentML (LegalDocML) TC* <https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=legaldocml>.

[5]   Guido Boella, Loredana Cupi, Luigi di Caro, Monica Palmirani, Livio Robaldo, and Andrea Violato, D2. 2 Legal XML-schema (XSD) (Integration, 2014).

[6]   Palmirani and Vitali, above n 2, 75.

[7]   Marko Marković, Stevan Gostojić, Zora Konjović, and Mart Laanpere, 'Machine-readable Identification and Representation of Judgments in Serbian Judiciary' (2014) 44(1) Novi Sad Journal of Mathematics 165.

schema validates many kinds of structures by itself to support various legal document formats around the world.

Our purpose in this paper is to create a seamless connection from Japanese statutes following strict drafting rules to Akoma Ntoso as an international standard for jurisdiction-independent XML schema. We present a case study for avoiding ambiguous annotation when applying the Akoma Ntoso XML schema to Japanese legislation and propose a method to convert statutory documents structured by our XML schema to that of Akoma Ntoso, enabling us to show other jurisdictions how to overcome this problem.

Our paper is organised as follows: in Part 2, we introduce the characteristics of Japanese statutes and a sophisticated proofreading process for the Japanese legislative system. We then provide examples of ambiguity in Akoma Ntoso annotation in Part 3. In Part 4, we explain the XML schema for Japanese statutes and our approach to convert the schema's annotation to that of Akoma Ntoso. We summarise our findings in Part 5.

2   *Japanese Legislation*

In this part, we briefly explain the process and characteristics of Japanese legislation and its strict rules for the structure, format, and expression of all Japanese statutes.

2.1   Basic Organisation of Japanese Laws

Laws are generally divided into written and unwritten categories. Although unwritten laws include local customs and judicial precedents, we address only written laws in this paper. Written laws are also called statutes, which are further divided into acts and bylaws. In Japan, while acts are enacted by the Diet (Parliament), bylaws consist of orders enacted by the Cabinet and ordinances and regulations enacted by various governmental organisations, such as ministries and administrative commissions.

In this paper, we focus on statutory texts. A statute consists of a number of articles, each of which may be further subdivided into paragraphs or items. Articles are integrated into a chapter and chapters are integrated into a part based on their content. Articles, paragraphs, and items have sequential numbers with different typefaces. A provision denotes an individual article or a paragraph.

2.2   Characteristics of Japanese Legislation

The logic of the legislative system in Japan maintains the notation of the Japanese statute expressions. *Figure 1* outlines the legislative process in the national government from drafting to promulgation. Although both Diet members and the Cabinet can submit a bill to the Diet, most are introduced

by the Cabinet. For the 10,164 acts enacted from 1947 to 2014, about 80% were presented by the Cabinet.

Before the Cabinet submits a bill to the Diet, it sends a draft to the Cabinet Legislation Bureau, which scrutinises it for consistency with other statutes, expressions, formats, etc. The Bureau's mission is described on its website as follows:[8]

> During the examination by the Bureau, the bill is examined from all angles, legally and technically. The points examined include the following:
>
> - The relationship between the proposed bill on one hand and the Constitution and other existing laws on the other, as well as the legal appropriateness of the contents of the bill;
>
> - Whether or not the intentions of the proposed bill are accurately expressed in the text;
>
> - Whether or not the structure of the bill (for example, the order of articles) is appropriate;
>
> - Whether the usage of letters or words is correct.

Even the usage of such punctuation as commas and periods is maintained. When a Diet member submits a bill, it is reviewed by the Legislation Bureau of the House of Representatives or Councillors.

The Cabinet Legislation Bureau has a manual for drafting Japanese legislation. Although the original version has never been published, many derivations are available for drafters of local governments and other organisations.

---

[8]   Cabinet Legislation Bureau, Until the Law Can Be Done
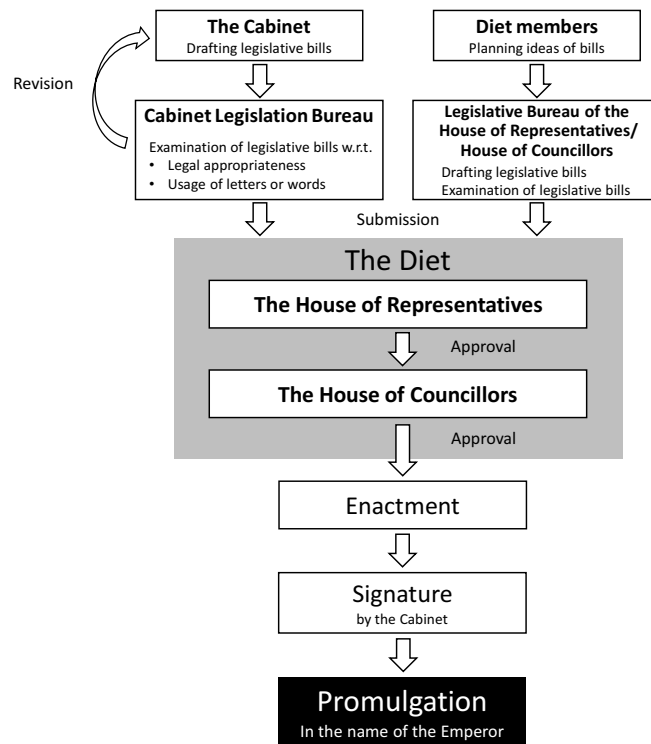   <http://www.clb.go.jp/english/process.html>.

Figure 1: Outline of the Japanese legislative process.[9]

Not every country's legislative system resembles Japan's. In the United Kingdom, the verification of legislation is less strict, since in most cases the bill is drafted outside of the ministry. In the United States, no organisation or system verifies legislation. In Asian countries, other than Japan and Korea, often each ministry independently prepares a draft of a bill without coordinating with other ministries. As a result, the notation of bills differs among ministries. In some countries, bills are often modified during deliberation in the national assembly, which may result in inconsistency of descriptive styles, while bills generally pass the Diet in Japan as drafted.[10]

---

[9] In some cases, bills are sent up for consideration to the House of Councillors first, rather than the House of Representatives.

[10] This paragraph is based on a discussion with Prof. Yoshiharu Matsuura at Graduate School of Law, Nagoya University. Ministry of Justice, Japan, Japanese Law Translation Database System (2016) <http://www.japaneselawtranslation.go.jp/>.

2.3   Dissemination of Japanese Legal Information

The Japanese government publishes its statutes online in the Japanese Law Translation Database System (JLT).[11] JLT has provided an English version of important statutes since 2009. These statutes are digitised in XML to simplify their management and to convert their format. Since the main purpose of JLT is to offer English translation of Japanese statutes, we can only access statutory documents that have been translated into English. Since JLT's release, the small number of statutes that have actually been translated has been problematic.[12] As of July 2015, only 524 statutes were included in JLT, which is merely 6.5% of all statutes currently in effect.

The Japanese government has been preparing a more sophisticated scheme, called the e-Legislative Activity and Work Support System (*e-LAWS*), to support drafting bills in government and to disseminate all Japanese statutes currently in effect. It will follow the XML schema based on Document Type Definition (DTD) in JLT. We refer to this new schema as the Japanese Statutory Schema (JSS). It is designed exclusively for the structure of Japanese statutes and does not allow a document to have ambiguous structures. Thus far, Akoma Ntoso has never been considered in the development of JSS. This study is the first attempt to apply Akoma Ntoso to Japanese statutes.

3   *Ambiguity in Akoma Ntoso Annotation*

In this part, we highlight problematic annotations for two types of structures that are often seen in legal documents—hierarchical structures and provisions in amendment acts.

---

[11]   Toyama et al, above n 1. Note that, despite an official website by the Ministry of Justice, all of the translations contained in JLT are unofficial. Since Japanese laws and regulations are written in Japanese, only their original Japanese texts have legal effect; translations are to be used solely as reference materials to aid in the understanding of Japanese laws and regulations. The government of Japan is not responsible for the accuracy, reliability, or currency of the legislative material provided in the website, or for any consequence resulting from use of the information on the website.

[12]   Yasuhiro Sekine, Yasuhiro Ogawa, Katsuhiko Toyama, and Yoshiharu Matsuura, 'The Development of Translation Memory Database System for Law Translation' (Paper presented at the Law via the Internet Conference 2012, Cornell University, 9 October 2012).

3.1   Ambiguity in Annotating Hierarchical Structures

The Akoma Ntoso XML Schema validates many kinds of elements and various structures to deal with the legal documents of different nations that follow different formats.[13] In other words, this flexibility allows the schema to validate unintended structuralisation to the same document. We regard flexibility in annotation as *ambiguity* in formal language theory[14].

For example, Akoma Ntoso has difficulty annotating hierarchical structures. We use the article in *Figure 2* to discuss how to annotate an article of a statute.[15] This article about term definitions has two paragraphs, one of which consists of two items. We need to determine an appropriate hierarchical element for the article's children. Akoma Ntoso defines 27 elements, such as 'article' and 'paragraph', to describe the hierarchical structure of legal documents. The problem is that there are no restrictions on the hierarchical relationships among the elements. Therefore, this schema validates both structures   shown   in *Figure 3*.[16] The children of 'article' can be 'paragraph,' as shown in Candidate 1, while 'rule' can be used for the same role, as shown in Candidate 2.

The annotation for the content in each article is also ambiguous. There are multiple possible annotations to the ordered items with parenthesised Roman numerals, (i) and (ii) in Figure 3. In Candidate 1, these items are regarded as a part of the hierarchical structure and are expressed with the tag 'point.' On the other hand, in Candidate 2, the tag 'tblock' is used. The schema validates both candidates, which we consider ambiguous.

---

[13]   Fabio Vitali, Monica Palmirani, Roger Sperberg, and Véronique Parisse, Akoma Ntoso Version 1.0. Part 2: Specifications (14 January 2015) <http://docs.oasis-open.org/legaldocml/akn-core/v1.0/akn-core-v1.0-part2-specs.html>.

[14]   John E Hopcroft and Jeffrey D Ullman, *Introduction to Automata Theory, Languages, and Computation* (Addison-Wesley, 1979).

[15]   Japanese statutes are originally written in Japanese. The article in *Figure 2* is the English translation.

[16]   In *Figure 3*, attributes are omitted for simplification.

(Definitions, etc.)

Article 2 (1) The term "Bank" as used in this Act means a person who engages in Banking under the license from the Prime Minister prescribed in Article 4, paragraph (1).

(2) The term "Banking" as used in this Act means commercial pursuits carried out through any of the following acts:

    (i)   Acceptance of deposits or Installment Savings, in addition to loans of funds, or the discounting of bills and notes; or

    (ii) Carrying out exchange transactions.

Figure 2: Example of a Japanese statute (Article 2 of Banking Act (Act No. 59 of 1981))

Candidate 1

```
<article>
 <heading>(Definitions, etc.)</heading>
 <subheading>Article 2</subheading>
 <paragraph>
   <num>(1)</num>
   <content>
     <p>The term "Bank" as used in this Act means a person who engages
in Banking under the license from the Prime Minister prescribed in Article 4,
paragraph (1).</p>
   </content>
 </paragraph>
 <paragraph>
   <num>(2)</num>
   <list>
     <intro>
       <p> The term "Banking" as used in this Act means commercial
pursuits carried out through any of the following acts:</p>
     </intro>
     <point>
       <num>(i)</num>
       <content>
         <p>Both acceptance of deposits or Installment Savings, and loans
of funds or discounting of bills and notes; or </p>
       </content>
     </point>
     <point>
       <num>(ii)</num>
       <content>
         <p> Carrying out exchange transactions.</p>
       </content>
```

```
          </point>
        </list>
      </paragraph>
    </article>
```

Candidate 2

```
  <article>
    <heading>(Definitions, etc.)</heading>
    <subheading>Article 2</subheading>
    <rule>
      <num>(1)</num>
      <content>
        <p>The term "Bank" as used in this Act means a person who
engages in Banking under the license from the Prime Minister prescribed in
Article 4, paragraph (1).</p>
      </content>
    </rule>
    <rule>
      <num>(2)</num>
      <content>
        <p>The term "Banking" as used in this Act means commercial
pursuits carried out through any of the following acts:</p>
        <tblock>
          <num>(i)</num>
          <p>Acceptance of deposits or Installment Savings, in addition to
loans of funds, or the discounting of bills and notes; or</p>
        </tblock>
        <tblock>
          <num>(ii)</num>
          <p>Carrying out exchange transactions.</p>
        </tblock>
      </content>
    </rule>
  </article>
```

Figure 3: Possible annotation of Figure 2 validated by Akoma Ntoso

3.2   Inconsistent Annotation of Amendment Acts

The issue of inconsistency in Akoma Ntoso's annotation is also seen in the provisions in amendment acts. In this part, we demonstrate that there are at least two ways to use the element 'mod.' Figures 4 and 5 show a comparison of the element between amendment acts that were enacted in (A) Kenya and (B) the United Kingdom (UK), respectively. *Table 1* details these acts. Both provisions include the new provisions to be inserted into the target act. The provision in Figure 4 explains how s 50 of the existing act is to be amended,

while the provision in Figure 5 explains where to insert sub-s (8A) into s (2) of the existing act.

Note that statutes are written in two types of languages: an object language for new enactments and metalanguage, which re-writes the description in object language, for amendments, rearrangements, and to repeal acts. In Figure 4, the first three lines are written in the metalanguage, while the rest is in the object language. In Figure 5, the first two lines are written in the metalanguage, while the rest is in the object language.

---

7. Section 50 of the principal act is amended -

(a) by deleting subsection (2) and **inserting** the following -

   (2) Where anything other than a trophy is seized and detained under section 49 of this Act it shall be forfeited to the government after a period of three months unless within that period some person is charged with a forfeiture offence under the Act and it is alleged that -

      (a) the offence was committed in relation to or in connection with that thing; or

      (b) the thing was used in, or for the purpose of, the commission

---

Figure 4: Amending provision in an act enacted in Kenya

In section 2 of the Regulation of Investigatory Powers Act 2000 (meaning of "interception" etc), after subsection (8) **insert —**
      "(8A) For the purposes of the definition of "telecommunications service" in subsection (1), the cases in which a service is to be taken to consist in the provision of access to, and of facilities for making use of, a telecommunication system include any case where a service consists in or includes facilitating the creation, management or storage of communications transmitted, or that may be transmitted, by means of such a system."

Figure 5: Amending provision in an act enacted in the UK

Table 1: Information on the acts shown in Figures 4 and 5

Label	Corresponding Figure	Nation	Name	Excerpted position	Name of the act to be amended
(A)	Figure 4	Kenya	The Wildlife (Conservation and Management) (Amendment) Act, 1989 (No. 16 of 1989)	s 7	The Wildlife (Conservation and Management) Act (Cap. 376)

(B)	Figure 5		the UK	Data Retention and Investigatory Powers Act 2014 (2014 Chapter 27)	ss 2, s 4	Regulation of Investigatory Powers Act 2000 (2000 Chapter 23)

Figures 6 and *7* show examples of the Akoma Ntoso annotation of the provisions in (A) and (B), respectively. Both are downloadable.[17]

Here, we focus on the element 'mod,' which is an XML tag defined for modification. This element is used for separating the metalanguage and the object language. However, 'mod' is used in a different way in each of these two examples. For (A), the instruction '... and inserting the following—...' is included within 'mod.' On the other hand, the instruction '... after subsection (8) insert' in (B) is not included within 'mod,' even though both elements play the same role. In other words, the region of the object language includes a part of the metalanguage in (A).

The gap between the above two examples implies an underlying problem of the Akoma Ntoso XML schema, under which annotations can differ depending on the annotator because details in the usage are not published. Furthermore, such an inconsistency may occur not only among different articles in a single act, but also different acts in a single nation and different nations in a single XML standard, which violates the intended interchangeability of Akoma Ntoso. Considering that JSS is designed exclusively for the structure of Japanese statutes and does not allow a document to have ambiguous structures, we need to carefully determine how to implement Akoma Ntoso to Japanese legislation.

```
 <section>
  <num>7.</num>
  <content>
   <blockList>
    <listIntroduction>Section 50 of the principal act is amended -
</listIntroduction>
   <item>
    <num>(a)</num>
    <p>by deleting
     <mod>
      <ref     href="/ke/act/1980-01-01/1/main#sec50-sub2">subsection
```

```
(2)</ref> and inserting the following –
      <quotedStructure>
       <subsection>
        <num>(2)</num>
        <content>
         <blockList>
          <listIntroduction>Where anything other than a trophy is seized
and detained under section 49 of this Act it shall be forfeited to the
government after a period of three months unless within that period some
person is charged with a forfeiture offence under the Act and it is alleged
that -</listIntroduction>
           <item>
            <num>(a)</num>
            <p> the offence was committed in relation to or in connection
with that thing; or</p>
           </item>
           <item>
            <num>(b)</num>
            <p>the thing was used in, or for the purpose of, the commission
of such an offence.</p>
           </item>
         </blockList>
        </content>
       </subsection>
      </quotedStructure>
     </mod>
    </p>
   </item>
   <item>
    <num>(b)</num>
    <p>by deleting
     <mod>
      <ref  href="/ke/act/1980-01-01/1/main#sec50-sub3-itmc">paragraph
(c) of subsection (3)</ref>
    </mod>.
    </p>
   </item>
  </blockList>
 </content>
</section>
```

Figure 6: Annotation for the provision in Figure 4

```
 <section>
  <num>5</num>
  <heading>Meaning of "telecommunications service"</heading>
  <content>
```

```
    <p>In                <ref              href="/akn/uk/act/2000-07-
28/chapter23/main#sec_2__subsec_8"> section 2 of the Regulation of
Investigatory Powers Act 2000 (meaning of "interception" <abbr title="Et
cetera" xml:lang="la">etc</abbr>), after subsection (8)</ref>insert—</p>
    <p class="BlockAmendment">
    <mod for="#ref_12">
      <quotedStructure class="primary main" startQuote=""" endQuote="""">
        <subsection>
          <num>8A</num>
          <content>
            <p>For the purposes of the definition of "telecommunications
service" in subsection (1), the cases in which a service is to be taken to consist
in the provision of access to, and of facilities for making use of, a
telecommunication system include any case where a service consists in or
includes facilitating the creation, management or storage of communications
transmitted, or that may be transmitted, by means of such a system.</p>
          </content>
        </subsection>
      </quotedStructure>
    </mod>
    </p>
  </content>
 </section>
```

Figure 7: Annotation for the provision in Figure 5

4   *Akoma Ntoso Annotation for Japanese Statutes*

In this Part, we consider how to apply Akoma Ntoso to Japanese statutes.
We first explain the Japanese Statutory Schema in Part 4.1. We then propose
a method to convert JSS to Akoma Ntoso in Part 4.2 and report the
implementation of an automatic XML converter from JSS to Akoma Ntoso in
Part 4.3. In Part 4.4, we address the problem of unnatural annotation
resulting from Akoma Ntoso's restrictions.

Note that the Akoma Ntoso Schema requires a 'meta' section, which
indicates what the resource is about and how it can be accessed.[18] Since the
content of the 'meta' section in JSS is the same as that of Akoma Ntoso, we
omit further discussion of it in our paper.

---

[18]   Fabio Vitali and Flavio Zeni, 'Towards a Country-independent Data Format: The
       Akoma Ntoso Experience' in Giovanni Sartor, Carlo Biagioli, and Enrico Francesconi
       (eds), *Proceedings of the V Legislative XML Workshop* (European Press Academic
       Publishing, 2007) 67.

4.1   The Japanese Statutory Schema

In the Japanese Statutory Schema (JSS), which is based on DTD in JLT,[19] many elements and structures are defined to fit Japanese statutes. This schema is so strict that the documents validated by it are guaranteed to conform to the descriptive Japanese statute styles in document structure. Since JSS is designed based on a thorough study of the styles of Japanese statutes,[20] we can annotate all statutes in accordance with the schema without ambiguity.

In this paper, we discuss the DTD used in JLT, which is basically the same as the current version of JSS, while JSS conforms to W3C (World Wide Web Consortium) XML schema. JSS will be updated for the launch of e-LAWS.

Under JSS, the structure shown in *Figure 8* is the only valid annotation for the article in *Figure 2*. Based on the hierarchical structure of actual Japanese statutes, this schema does not allow elements other than 'Paragraph' to be children of 'Article,' or those other than 'Item' or 'Class' to be children of 'Paragraph.'

In JSS, the 12 hierarchical elements shown in *Table 2* are defined. This hierarchy is strictly ordered except for the specified skipped levels. Therefore, 'Chapter' cannot be a child of 'Section.'

JSS concretely defines not only the hierarchical structures, but also the type and order of their content. For example, no element can be used as the children of 'Chapter' other than 'ChapterTitle' and 'Section,' and only 'Sentence' and 'Table' can be contents of 'Item.' Since Japanese statutes are described by strict rules, their hierarchical structures are achieved by a strict schema.

```
<Article Num="2">
  <ArticleCaption>(Definitions, etc.)</ArticleCaption>
  <ArticleTitle>Article 2</ArticleTitle>
  <Paragraph Num="1">
   <ParagraphNum>(1)</ParagraphNum>
   <ParagraphSentence>
    <Sentence>The term "Bank" as used in this Act means a person who
engages in Banking under the license from the Prime Minister prescribed in
Article 4, paragraph (1).</Sentence>
   </ParagraphSentence>
```

---

[19]   Japanese Law Translation, Related Information
       <http://www.japaneselawtranslation.go.jp/rel_info/rel_info_dtd?re=02>.

[20]   Toyama et al, above n 1.

```
   </Paragraph>
  <Paragraph Num="2">
   <ParagraphNum>(2)</ParagraphNum>
   <ParagraphSentence>
    <Sentence>The term "Banking" as used in this Act means commercial
pursuits carried out through any of the following acts:</Sentence>
   </ParagraphSentence>
   <Item Num="1">
    <ItemTitle>(i)</ItemTitle>
    <ItemSentence>
     <Sentence>Acceptance of deposits or Installment Savings, in addition
to loans of funds, or the discounting of bills and notes; or</Sentence>
    </ItemSentence>
   </Item>
   <Item Num="2">
    <ItemTitle>(ii)</ItemTitle>
    <ItemSentence>
     <Sentence>Carrying out exchange transactions.</Sentence>
    </ItemSentence>
   </Item>
  </Paragraph>
 </Article>
```

Figure 8: The only valid JSS annotation for the article in Figure 2

Table 2: Hierarchical elements defined in JSS

Level	Elements in JSS
1	Part
2	Chapter
3	Section
4	Subsection
5	Division
6	Article
7	Paragraph
8	Class
9	Item
10	Subitem1
11	Subitem2
12	Subitem3

4.2   Converting JSS to Akoma Ntoso

Our aim is to create Japanese statute documents structured by Akoma Ntoso. Our approach is to convert the documents structured in JSS into those in Akoma Ntoso through the following two steps:

1. Create rules to replace the JSS tags with those of Akoma Ntoso.

2. Design a custom schema of Akoma Ntoso to validate only the structures that are suitable for Japanese statutes.

We determined 131 rules for 103 elements defined in JSS. Some JSS elements have several replacement rules. These rules are applied properly according to the structure around the element. Appendix A lists the rules to replace the JSS tags with those of Akoma Ntoso. The replacement rules follow some basic policies. First, for easy access to the semantic information of each statute, we basically replace a JSS tag with an Akoma Ntoso tag whose meaning corresponds to that of the JSS tag. For example, we replace the tag 'LawTitle' in JSS with 'docTitle,' as shown in *Figure 9*, because the meanings of these tags are similar. Table 3 shows some of the tags replaced in accordance with this policy.

Annotation in JSS

<LawTitle>Act on Special Measures for Enhancement of the Conservation and Management of Tuna Resources</LawTitle>

Akoma Ntoso conversion

<docTitle>Act on Special Measures for Enhancement of the Conservation and Management of Tuna Resources</docTitle>

Figure 9: Replacement of JSS tag with corresponding Akoma Ntoso tag

Table 3: Elements in JSS and Akoma Ntoso with similar meanings

JSS	Akoma Ntoso
Part	part
Chapter	chapter
Section	section
Article	article
Paragraph	paragraph
Signature	signature
LawNum	docNumber
LawTitle	docTitle
ChapterTitle	heading

Annotation in JSS
 <ParagraphSentence>
   <Sentence>The term "Bank" as used in this Act means a person who engages in Banking under the license from the Prime Minister prescribed in Article 4, paragraph (1).</Sentence>

```
 </ParagraphSentence>
Akoma Ntoso conversion
 <blockContainer class="ParagraphSentence">
  <p>
   <inline name="Sentence">The term "Bank" as used in this Act means a
person who engages in Banking under the license from the Prime Minister
prescribed in Article 4, paragraph (1).</inline>
  </p>
 </blockContainer>
```

Figure 10: Replacement of tags unique to JSS

The second policy is that, when replacing tags unique to JSS, we give an attribute with the name of the element to the replaced elements in Akoma Ntoso. For example, as shown in *Figure 10*, the tags 'ParagraphSentence' and 'Sentence' are defined in JSS. The former is to contain character-based content, such as sentences and tables, and the latter is to contain just one sentence. Since Akoma Ntoso does not have any corresponding tags, we replace them with 'blockContainer' and 'inline' with the attributes 'class="ParagraphSentence"' and 'name="Sentence".'

We can replace the structured document shown in *Figure 8* with that in *Figure 11*. The tables of contents, the supplementary provisions, and the appendixes in the Japanese statutes structured in JSS can also be converted into Akoma Ntoso based on our rules.

We also designed a custom schema that is a subset of Akoma Ntoso for the elements and structure of Japanese statutes. The custom schema validates only the annotation shown in *Figure 11*, but does not validate the annotations shown in Figure 3. Appendix B shows the definitions related to 'chapter' in Akoma Ntoso and our custom schema to validate the Japanese statutes, and Appendix C illustrates these definitions in terms of their children. The definition in the custom schema is simplified and only two hierarchical elements are allowed to be children of 'chapter.'

```
 <article eId="chp1-art2">
  <num>(Definitions, etc.)</num>
  <heading>Article 2</heading>
  <paragraph eId="chp1-art2-par1">
   <heading>(1)</heading>
   <content>
    <blockContainer class="ParagraphSentence">
     <p>
      <inline name="Sentence">The term "Bank" as used in this Act means a
person who engages in Banking under the license from the Prime Minister
prescribed in Article 4, paragraph (1).</inline>
     </p>
```

```
    </blockContainer>
   </content>
  </paragraph>
  <paragraph eId="chp1-art2-par2">
   <heading>(2)</heading>
   <intro>
    <blockContainer class="ParagraphSentence">
     <p>
      <inline name="Sentence">The term "Banking" as used in this Act
means commercial pursuits carried out through any of the following
acts:</inline>
     </p>
    </blockContainer>
   </intro>
   <subparagraph eId="chp1-art2-par2-itm1">
    <heading>(i)</heading>
    <content>
     <blockContainer class="ItemSentence">
      <p>
       <inline name="Sentence">Acceptance of deposits or Installment
Savings, in addition to loans of funds, or the discounting of bills and notes;
or</inline>
      </p>
     </blockContainer>
    </content>
   </subparagraph>
   <subparagraph eId="chp1-art2-par2-itm2">
    <heading>(ii)</heading>
    <content>
     <blockContainer class="ItemSentence">
      <p>
       <inline        name="Sentence">Carrying         out         exchange
transactions.</inline>
      </p>
     </blockContainer>
    </content>
   </subparagraph>
  </paragraph>
 </article>
```

Figure 11: The only valid annotation for the Article 2 outcome of the replacement of the structure in Figure 2 validated by our custom schema

4.3 Automatic Converter

We also implemented an automatic converter for Japanese statutes from JSS to Akoma Ntoso.

The converting process consists of the following two steps:

1. Analyse the structure of JSS documents; and

2. Re-write the name of the elements and attributes and reorganize the document structure to adapt the description rules of Akoma Ntoso.

These operations are based on the replacement rules determined in Part 4.2. We employed *Racc*, a Ruby-implemented Look-ahead LR (LALR) parser generator, for (1) and *REXML*, an XML parser, for (2).

We examined the converter with an input set consisting of 1,276 documents in JSS format that are Japanese and English versions of 638 Japanese statutes.[21] We obtained the same number of outputs in Akoma Ntoso format, which were thrown into the XML processor *xmllint* (a component of XML toolkit, LibXML2) for the validation check.[22]

As a result, we found that 2.7% of the input in JSS format, which is automatically annotated, includes incorrect annotations, resulting in conversion failure. In other words, we confirmed that all of the outputs were validated, as long as the inputs were correctly annotated.

Figure 12 shows an example of a JSS document including incorrect annotations and its Akoma Ntoso conversion. The attribute 'Num' of the elements 'Article' for Articles 1 and 3 is given an identical value '1,' although the correct values for the former and latter are '1' and '3,' respectively. JSS validates this annotation despite a semantically incorrect attribute for Article 3.

This kind of incorrect annotation creates incorrect values of attribute 'eId' in the Akoma Ntoso documents, which is generated from the attribute 'Num' in the JSS documents. In the Akoma Ntoso document in Figure 12, the values of attribute 'eId' overlap between two 'article' elements, which are converted from 'Article' in the JSS document. The Akoma Ntoso XML schema does not validate this structure because the schema requires that attribute 'eId' of every element in a single document have a unique value.

After modification of the incorrect annotations of inputs, they were properly converted and validated in the Akoma Ntoso XML schema.

---

[21] All of the statutes used in this paper are downloadable at: Ministry of Justice, Japan, *Japanese Law Translation* (2016) <http://www.japaneselawtranslation.go.jp>.

[22] Daniel Veillard, *The XML C Parser and Toolkit of Gnome* <http://xmlsoft.org/>.

```
Annotation in JSS
 <SupplProvision>
  <SupplProvisionLabel>Supplementary Provisions (Act No. 92 of June 20,
2007) (Excerpts)</SupplProvisionLabel>
  <Article Num="1" >
   <ArticleCaption>(Effective Date)</ArticleCaption>
   <ArticleTitle>Article 1</ArticleTitle>
   (snip)
  </Article>
  (snip)
  <Article Num="1" >
   <ArticleTitle>Article 3</ArticleTitle>
   (snip)
  </Article>
  (snip)
 </SupplProvision>
Akoma Ntoso conversion
 <hcontainer eId="sup6" name="SupplProvision">
  <heading>Supplementary Provisions (Act No. 92 of June 20, 2007)
(Excerpts)</heading>
  <article eId="sup6-art1">
   <num>(Effective Date)</num>
   <heading>Article 1</heading>
   (snip)
  </article>
  (snip)
  <article eId="sup6-art1">
   <num/>
   <heading>Article 3</heading>
   (snip)
  </article>
 </hcontainer>
```

Figure 12 JSS document including incorrect annotations and its Akoma Ntoso conversion (Supplementary Provisions in Act for Implementation of the Mutual Recognition between Japan and Foreign States in Relation to Results of Conformity Assessment Procedures of Specified Equipment (Act No. 111 of 2001))

```
(a) Annotation in JSS
<ArticleCaption>(Definitions, etc.)</ArticleCaption>
<ArticleTitle>Article 2</ArticleTitle>
(b) Akoma Ntoso conversion
<num>(Definitions, etc.)</num>
<heading>Article 2</heading>
(c) More desirable conversion (invalid in Akoma Ntoso)
<heading>(Definitions, etc.)</heading>
```

```
<num>Article 2</num>
```

Figure 13: Possible and desirable replacements of JSS elements into those of Akoma Ntoso

4.4   Unnatural Annotations Due to Restrictions of Akoma Ntoso

As mentioned in Part 4.2, Akoma Ntoso seems applicable to Japanese statutes. Since JSS is stricter than Akoma Ntoso, it is not difficult to convert from the former to the latter. However, some constraints on Akoma Ntoso prevent us from converting the tags in JSS into conventional ones in Akoma Ntoso.

As shown in *Figure 2*, articles in Japanese statutes have a caption above their titles that describes their content. It is impossible to annotate the caption and the title of articles in Japanese statutes with the relevant tags provided by Akoma Ntoso because Akoma Ntoso does not allow any elements other than 'num' before 'heading' in a hierarchical structure, as shown in the definition of 'xsd:complexType name="basehierarchy"' in Appendix B. Therefore, 'ArticleCaption'   and   'ArticleTitle'   in *Figure 13*(a)   are   replaced   with   'num'   and   'heading'   in *Figure 13*(b). If Akoma Ntoso allowed the order to be shifted between 'num' and 'heading,' we could replace 'ArticleCaption' and 'ArticleTitle' with 'heading'   and   'num,'   as   shown   in *Figure 13*(c).

The Akoma Ntoso XML schema has great flexibility in annotating hierarchical structure. The same could be true for captions and titles. Enhancing the flexibility would enable Japanese legislation to adapt Akoma Ntoso to enhance the schema as a platform for the international exchange of legal information.[23]

5   Conclusion

We introduced an approach to applying Akoma Ntoso to Japanese statutes. We identified an underlying problem of the structural ambiguity of Akoma Ntoso, that is, that the flexibility of annotations allows the schema to validate

---

[23]   This issue on the restrictions was modified in the latest version of Akoma Ntoso (WD17 AN 3.0) in response to our previous paper presented at the Law via the Internet Conference 2015: see Gen Kawachi, Makoto Nakamura, Yasuhiro Ogawa, Tomohiro Ohno, and Katsuhiko Toyama, 'Applying the Akoma Ntoso XML Schema to Japanese Legislation' (Paper presented at the Law via the Internet Conference 2015, The University of New South Wales, Australia, 11 November 2015).

unintended structuralisation to the same document. On the other hand, we can annotate all the statutes in accordance with JSS without ambiguity. Therefore, we proposed a method to convert JSS to Akoma Ntoso. We determined the rules to replace JSS tags with Akoma Ntoso and designed a custom schema for Japanese statutes. Our approach made it possible to cope with consistency of the strict rules in the descriptive styles of Japanese statutes and achieve the advantage of Akoma Ntoso documents in exchanging legal information.

The Japanese legislation system applies strict rules for the styles of statutes, as reflected in JSS. Namely, we confirmed that the Akoma Ntoso XML schema does not fit Japanese statutes at the native level, taking processing overhead into account. On the other hand, as we revealed in this paper, it is not difficult to convert JSS documents into Akoma Ntoso documents, which has an advantage in terms of sharing legal information internationally. Therefore, we can easily obtain structured Japanese legal documents in Akoma Ntoso format whenever needed.

For these reasons, e-LAWS, which is the new legal information system in Japan mentioned in Part 2, is expected to employ a JSS-based XML schema, instead of the direct use of Akoma Ntoso for structured documents of Japanese statutes. Since we have continuously supported the development of Japanese legal schemata from JLT,[24] it is not an exaggeration to say that our work greatly contributed to this decision.

6   *Acknowledgements*

7   *Appendixes*

The appendixes to this work can be found at the end of this issue at pages 90 (Appendix A), 100 (Appendix A2), 101 (Appendix B) and 103 (Appendix C).

---

[24]   Toyama et al, above n 1.

Appendix A : List of Rules for Replacing JSS Elements to Akoma Ntoso Elements

JSS	Akoma Ntoso	eId	level	class	name	Other attributes
Law	akomaNtoso/act					
PromulgateStatement	act/preface/container				[Name of the statute] @act PromulgateStatement @container	
LawNum	act/preface/p/docNumber			LawNum@p		
LawBody	act/					
Signature	act/conclusions/ p/signature			Signature@p		
Signature	p/signature			Signature@p		
MinisterialTitle	signature/role					refersTo="#somtething"
Name	signature/person					refersTo="#someone"
PromulgateStatement	preface/container				PromulgateStatement @container	
PromulgateBody	container/p			PromulgateBody		
ImperialSignature	container/p/signature			ImperialSignature@p		
PromulgateDate	container/p/docDate					
Signature	container/p/signature			Signature@p		
LawBody	/					
LawTitle	preface/p/docTitle			LawTitle@p		
EnactStatement	preface/p			EnactStatement@p		
TOC	preface/toc					
Preamble	preamble					
Paragraph (Children of "Preamble") MainProvision	preamble/blockContainer			Paragraph		
	body/					
AppdxTable	body/hcontainer	atb			AppdxTable	
SupplProvision	body/hcontainer	sup			SupplProvision	style= "amendLawNum:[amendLawNum];"
AppdxNote	body/hcontainer	ant			AppdxNote	
AppdxStyle	body/hcontainer	ast			AppdxStyle	

JSS	Akoma Ntoso	eId level	class	name	Other attributes
Appdx	body/hcontainer	adx		Appdx	
AppdxFig	body/hcontainer	afg		AppdxFig	
AppdxFormat	body/hcontainer	afm		AppdxFormat	
TOC	toc				
TOCLabel	toc/tocItem/inline	0		TOCLabel@inline	href=[URL of the statute]@tocItem
TOCPreambleLabel	toc/tocItem/inline	0		TOCPreambleLabel@inline	href=[corresponding eId]
TOCPart	toc/tocItem	1	TOCPart		href=[corresponding eId]
TOCChapter	toc/tocItem	2	TOCChapter		href=[corresponding eId]
TOCArticle	toc/tocItem	6	TOCArticle		href=[corresponding eId]
TOCSupplProvision	toc/tocItem	1	TOCSupplProvision		href=[corresponding eId]
TOCAppdxTableLabel	toc/tocItem/inline	1		TOCAppdxTableLabel@inline	href=[corresponding eId]
TOCPart	tocItem	1	TOCPart		href=[corresponding eId]
PartTitle (Child of "TOCPart")	tocItem/inline			PartTitle	
TOCChapter	tocItem	2	TOCChapter		href=[corresponding eId]
ArticleRange	tocItem/inline			ArticleRange	
TOCChapter	tocItem	2	TOCChapter		href=[corresponding eId]
ChapterTitle (Child of "TOCChapter")	tocItem/inline			ChapterTitle	
TOCSection	tocItem	3	TOCSection		href=[corresponding eId]
ArticleRange	tocItem/inline			ArticleRange	
TOCSection	tocItem	3	TOCSection		href=[corresponding eId]
SectionTitle (Child of "TOCSection")	tocItem/inline			SectionTitle	
TOCSubsection	tocItem	4	TOCSubsection		href=[corresponding eId]
ArticleRange	tocItem/inline			ArticleRange	
TOCSubsection	tocItem	4	TOCSubsection		href=[corresponding eId]
SubsectionTitle (Child of "TOCSubsection")	tocItem/inline			SubsectionTitle	
TOCDivision	tocItem	5	TOCDivision		href=[corresponding eId]
ArticleRange	tocItem/inline			ArticleRange	

JSS	Akoma Ntoso	eId level	class	name	Other attributes
TOCDivision	tocItem	5	TOCDivision		href=[corresponding eId]
DivisionTitle (Child of "TOCDivision")	tocItem/inline			DivisionTitle	
ArticleRange	tocItem/inline			ArticleRange	
TOCArticle	tocItem	6	TOCArticle		href=[corresponding eId]
ArticleTitle (Child of "TOCArticle")	tocItem/inline			ArticleTitle	
ArticleCaption (Child of "TOCArticle")	tocItem/inline			ArticleCaption	
TOCSupplProvision	tocItem	1	TPCSupplProvision		href=[corresponding eId]
SupplProvisionLabel (Child of "TOCSupplprovision")	tocItem/inline			SupplProvisionLabel	
TOCArticle	tocItem	6	TOCArticle		href=[corresponding eId]
MainProvision	body				
Part	body/part	prt			
Chapter	body/chapter	chp			
Article	body/article	art			
Paragraph	body/paragraph	par			
Part	part	prt			
PartTitle (Child of "Part")	part/heading				
Chapter	part/chapter	chp			
Article	part/article	art			
Chapter	chapter	chp			
ChapterTitle (Child of "Chapter")	chapter/heading				
Section	chapter/section	sec			
Article	chapter/article	art			
Section	section	sec			
SectionTitle (Child of "Section")	section/heading				
Subsection	section/subsection	sbs			
Article	section/article	art			

JSS	Akoma Ntoso	eId level	class	name	Other attributes
Subsection	subsection	sbs			
SubsectionTitle (Child of "Subsection")	subsection/heading				
Division	subsection/division	div			
Article	subsection/article	art			
Division	division	div			
DivisionTitle (Child of "Division")	division/heading				
Article	division/article	art			
Article	article	art			
ArticleCaption (Otherwise)	article/num				style= "CommonCaption:[Yes|No];"
ArticleCaption (If appearing after "ArticleTitle")	article/subheading				
ArticleTitle (Child of "Article")	article/heading				
Paragraph	article/paragraph	par			
SupplNote	article/wrapUp/p		SupplNote@p		
Paragraph	paragraph	par			
ParagraphCaption	paragraph/num				
ParagraphNum	paragraph/heading				
ParagraphSentence (If the parent "Paragraph" has "Class" or "Item")	paragraph/intro/ blockContainer		ParagraphSentence @blockContainer		
ParagraphSentence (Otherwise)	paragraph/content/ blockContainer		ParagraphSentence @blockContainer		
Sentence	paragraph/[intro|content]/ blockContainer/[intro|content]/p/inline content/blockList/item/p			Sentence	
AmendProvision		amp@item	AmendProvision@p		
Class	paragraph/subparagraph	cls			
Item	paragraph/subparagraph	itm			
TableStruct	paragraph/wrapUp/ blockContainer		TableStruct @blockContainer		style= "CommonCaption:[Yes|No];"

JSS	Akoma Ntoso	eId level class	name	Other attributes
FigStruct (Children of "Paragraph")	paragraph/wrapUp/ blockContainer	FigStruct @blockContainer		
AmendProvision	p/mod	AmendProvision@p		
AmendProvisionSentence	mod/inline		AmendProvisionSentence	
Sentence	inline/inline		Sentence	
NewProvision	mod/ [quotedStructure|quotedText]			
NewProvision	See Appendix A2			
Class	subparagraph	cls		
ClassTitle	subparagraph/heading			
ClassSentence (If the parent "Class" has "Item")	subparagraph/intro/ blockContainer	ClassSentence @blockContainer		
ClassSentence (Otherwise)	subparagraph/content/ blockContainer	ClassSentence @blockContainer		
Column	subparagraph/[intro|content]/ blockContainer/p	Column@p		
Sentence	subparagraph/[intro|content]/ blockContainer/p/inline		Sentence	
Table	subparagraph/[intro|content]/ blockContainer/table			
Item	subparagraph/subparagraph			
Item	subparagraph	itm		
ItemTitle	subparagraph/heading			
ItemSentence (If the parent "Item" has "Subitem1")	subparagraph/intro/ blockContainer	ItemSentence @blockContainer		
ItemSentence (Otherwise)	subparagraph/content/ blockContainer	ItemSentence @blockContainer		
Column	subparagraph/[intro|content]/ blockContainer/p	Column@p		
Sentence	subparagraph/[intro|content]/ blockContainer/p/inline		Sentence	
Table	subparagraph/[intro|content]/ blockContainer/table			
Subitem1	subparagraph/subparagraph			
Subitem1	subparagraph	lsi		

JSS	Akoma Ntoso	eId level	class	name	Other attributes
Subitem1Title	subparagraph/heading				
Subitem1Sentence (If the parent "Subitem1" has "Subitem2")	subparagraph/intro/ blockContainer		Subitem1Sentence @blockContainer		
Subitem1Sentence (Otherwise)	subparagraph/content/ blockContainer		Subitem1Sentence @blockContainer		
Column	subparagraph/[intro|content]/ blockContainer/p		Column@p		
Sentence	subparagraph/[intro|content]/ blockContainer/p/inline			Sentence	
Table	subparagraph/[intro|content]/ blockContainer/table				
Subitem2	subparagraph/subparagraph	2si			
Subitem2	subparagraph	2si			
Subitem2Title	subparagraph/heading				
Subitem2Sentence (If the parent "Subitem2" has "Subitem3")	subparagraph/intro/ blockContainer		Subitem2Sentence @blockContainer		
Subitem2Sentence (Otherwise)	subparagraph/content/ blockContainer		Subitem2Sentence @blockContainer		
Column	subparagraph/[intro|content]/ blockContainer/p		Column@p		
Sentence	subparagraph/[intro|content]/ blockContainer/p/inline			Sentence	
Table	subparagraph/[intro|content]/ blockContainer/table				
Subitem3	subparagraph/subparagraph	3si			
Subitem3	subparagraph	3si			
Subitem3Title	subparagraph/heading				
Subitem3Sentence	subparagraph/content/ blockContainer		Subitem3Sentence @blockContainer		
Column	subparagraph/content/ blockContainer/p		Column@p		
Sentence	subparagraph/content/ blockContainer/p/inline			Sentence	
Table	subparagraph/content/ blockContainer/table				
SupplProvision	hcontainer	sup		SupplProvision	style= "amendLawNum:[amendLawNum];"

JSS	Akoma Ntoso	eId	level	class	name	Other attributes
SupplProvisionLabel (Child of "SupplProvision")	hcontainer/heading					
Article	hcontainer/article	art				
Paragraph	hcontainer/paragraph	par				
SupplProvisionAppdxTable	hcontainer/hcontainer	sat			SupplProvisionAppdxTable	
SupplProvisionAppdxTable	hcontainer	sat			SupplProvisionAppdxTable	
SupplProvisionAppdxTableTitle	hcontainer/heading					
RelatedArticleNum	hcontainer/subheading					
TableStruct	hcontainer/content/blockContainer			TableStruct		
AppdxTable	hcontainer	atb			AppdxTable	
AppdxTableTitle	hcontainer/heading					
RelatedArticleNum	hcontainer/subheading					
TableStruct	hcontainer/content/blockContainer			TableStruct		
Item	hcontainer/subparagraph	itm				
TableStruct	blockContainer			TableStruct		
TableStructTitle	blockContainer/heading					
Remarks	blockContainer/blockContainer			Remarks		
Table	blockContainer/table					
Table	table					
TableHeaderRow	table/tr			TableHeaderRow		
TableHeaderColumn	table/tr/th			TableHeaderColumn		
TableRow	table/tr			TableRow		
TableColumn	table/tr/td			TableColumn		
Paragraph (Children of "TableColumn")	table/tr/td/blockContainer			Paragraph		
Item (Children of "TableColumn")	table/tr/td/blockContainer			Item		
Sentence	table/td/p/inline				Sentence@inline	

JSS	Akoma Ntoso	eId	level class	name	Other attributes
Column	table/td/p		Column@p		
Paragraph (Children of "TableColumn" or "Preamble")	blockContainer		Paragraph		
ParagraphCaption	blockContainer/num				
ParagraphNum	blockContainer/heading				
ParagraphSentence	blockContainer/blockContainer/p		ParagraphSentence@blockContainer		
Sentence	blockContainer/blockContainer/p/inline			Sentence@inline	
Item	blockContainer		Item		
ItemTitle	blockContainer/heading				
ItemSentence	blockContainer/blockContainer		ItemSentence		
Sentence	blockContainer/blockContainer/p/inline			Sentence@inline	
Column	blockContainer/blockContainer/p		Column		
Table	blockContainer/blockContainer/table				
AppdxNote	hcontainer	ant		AppdxNote	
AppdxNoteTitle	hcontainer/heading				
RelatedArticleNum	hcontainer/subheading				
NoteStruct	hcontainer/content/blockContainer		NoteStruct@blockContainer		
FigStruct	hcontainer/content/blockContainer		FigStruct@blockContainer		
TableStruct	hcontainer/content/blockContainer		TableStruct@blockContainer		
Remarks	hcontainer/content/blockContainer		Remarks@blockContainer		
NoteStruct	blockContainer		NoteStruct		
NoteStructTitle	blockContainer/heading				
Remarks	blockContainer/blockContainer		Remarks		
Note (If "Note" does not have any children)	blockContainer/p		Note		

JSS	Akoma Ntoso	eId level	class	name	Other attributes
Note	blockContainer/		Note@blockContainer		
(Otherwise)	blockContainer/foreign				Add prefix "jp" to the children of element "foreign"
Remarks	blockContainer		Remarks		
RemarksLabel	blockContainer/heading				
Item	blockContainer/		Item		
(Children of "Remarks")	blockContainer				
Sentence	blockContainer/p/inline			Sentence@inline	
AppdxStyle	hcontainer	ast		AppdxStyle	
AppdxStyleTitle	hcontainer/heading				
RelatedArticleNum	hcontainer/subheading				
StyleStruct	hcontainer/content/ blockContainer		StyleStruct		
Remarks	hcontainer/content/ blockContainer		Remarks		
StyleStruct	blockContainer		StyleStruct		
StyleStructTitle	blockContainer/heading				
Remarks	blockContainer/ blockContainer		Remarks		
Style (If "Style" does not have any children)	blockContainer/p		Style		
Style (Otherwise)	blockContainer/ blockContainer/foreign		Style		Add prefix "jp" to the children of element "foreign"
AppdxFormat	hcontainer	afm		AppdxFormat	
AppdxFormatTitle	hcontainer/heading				
RelatedArticleNum	hcontainer/subheading				
FormatStruct	hcontainer/content/ blockContainer		FormatStruct		
Remarks	hcontainer/content/ blockContainer		Remarks		
FormatStruct	blockContainer		FormatStruct		
FormatStructTitle	blockContainer/heading				
Remarks	blockContainer/blockContainer		Remarks		
Format (If "Format" does not have any children)	blockContainer/p		Format		

JSS	Akoma Ntoso	eId level	class	name	Other attributes
Format	blockContainer/		Format		Add prefix "jp" to the children of element "foreign"
(Otherwise)	blockContainer/foreign				
Appdx	hcontainer	adx		Appdx	
ArithFormulaNum	hcontainer/heading				
ArithFormula	hcontainer/content/p		ArithFormula		
AppdxFig	hcontainer	afg		AppdxFig	
AppdxFigTitle	hcontainer/heading				
RelatedArticleNum	hcontainer/subheading				
FigStruct	hcontainer/content/ blockContainer		FigStruct		
FigStruct	blockContainer		FigStruct		
FigStructTitle	blockContainer/heading				
Remarks	blockContainer/blockContainer		Remarks		
Fig (If "Fig" does not have any children)	blockContainer/p		Fig		
Fig (Otherwise)	blockContainer/ blockContainer/foreign		Fig		Add prefix "jp" to the children of element "foreign"

Appendix A2: Replacement for the Children of "`newProvision`"

JSS

NewProvision

- PartTitle
- ChapterTitle
- SectionTitle
- SubsectionTitle
- DivisionTitle
- Sentence
- LawTitle
- Preamble
- TOC
- Part
- Chapter
- Section
- Subsection
- Division
- Article
- SupplNote
- Paragraph
- Item
- Subitem1
- Subitem2
- Subitem3
- AmendProvision
- AppdxTable
- AppdxFig
- AppdxNote
- SupplProvisionAppdxTable
- TableStruct
- TableRow
- TableColumn
- FigStruct
- NoteStruct
- StyleStruct
- FormatStruct
- Remarks

Akoma Ntoso

quotedText

- inline name="PartTitle"
- inline name="ChapterTitle"
- inline name="SectionTitle"
- inline name="SubsectionTitle"
- inline name="DivisionTitle"
- inline name="Sentence"

quotedStructure

- p class="LawTitle"
- blockContainer class="Paragraph"
- toc
- part eId="prt"
- chapter eId="chp"
- section eId="sec"
- subsection eId="sbs"
- division eId="div"
- article eId="art"
- p class="SupplNote"
- paragraph eId="par"
- subparagraph eId="itm"
- subparagraph eId="1sl"
- subparagraph eId="2sl"
- subparagraph eId="3sl"
- docTitle
- p class="AmendProvision" — mod
- hcontainer name="AppdxTable" eId="atb"
- hcontainer name="AppdxFig" eId="afg"
- hcontainer name="AppdxNote" eId="ant"
- hcontainer name="SupplProvisionAppdxTable" eId="sat"
- blockContainer class="TableStruct"
- tr
- td
- blockContainer class="FigStruct"
- blockContainer class="NoteStruct"
- blockContainer class="StyleStruct"
- blockContainer class="FormatStruct"
- blockContainer class="Remarks"

Appendix B : Definitions of Element "chapter" in Akoma Ntoso and our Custom Schema

Akoma Ntoso

```
<xsd:element name="chapter" type="hierarchy"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema" />
<xsd:complexType name="hierarchy"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:complexContent>
    <xsd:extension base="basehierarchy">
      <xsd:choice>
        <xsd:sequence>
          <xsd:element ref="intro" minOccurs="0" maxOccurs="1" />
          <xsd:choice minOccurs="0" maxOccurs="unbounded">
            <xsd:element ref="componentRef" />
            <xsd:group ref="hierElements" />
          </xsd:choice>
          <xsd:element ref="wrapUp" minOccurs="0" maxOccurs="1" />
        </xsd:sequence>
        <xsd:element ref="content" />
      </xsd:choice>
      <xsd:attributeGroup ref="corereq" />
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="basehierarchy"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:sequence>
    <xsd:element ref="num" minOccurs="0" maxOccurs="1" />
    <xsd:element ref="heading" minOccurs="0" maxOccurs="1" />
    <xsd:element ref="subheading" minOccurs="0" maxOccurs="1" />
  </xsd:sequence>
</xsd:complexType>
<xsd:group name="hierElements" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:choice>
    <xsd:group ref="ANhier" />
    <xsd:element ref="hcontainer" />
  </xsd:choice>
</xsd:group>
<xsd:group name="ANhier" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:choice>
    <xsd:element ref="clause" />
    <xsd:element ref="section" />
    <xsd:element ref="part" />
    <xsd:element ref="paragraph" />
    <xsd:element ref="chapter" />
    <xsd:element ref="title" />
    <xsd:element ref="article" />
    <xsd:element ref="book" />
    <xsd:element ref="tome" />
    <xsd:element ref="division" />
    <xsd:element ref="list" />
    <xsd:element ref="point" />
    <xsd:element ref="indent" />
    <xsd:element ref="alinea" />
    <xsd:element ref="rule" />
    <xsd:element ref="subrule" />
    <xsd:element ref="proviso" />
    <xsd:element ref="subsection" />
    <xsd:element ref="subpart" />
    <xsd:element ref="subparagraph" />
    <xsd:element ref="subchapter" />
    <xsd:element ref="subtitle" />
    <xsd:element ref="subdivision" />
    <xsd:element ref="subclause" />
    <xsd:element ref="sublist" />
    <xsd:element ref="transitional" />
  </xsd:choice>
</xsd:group>
```

Our custom schema

```
<xsd:element name="chapter">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="heading" />
      <xsd:choice>
        <xsd:sequence>
          <xsd:element ref="article" minOccurs="1" maxOccurs="unbounded" />
          <xsd:element ref="section" minOccurs="0" maxOccurs="unbounded" />
        </xsd:sequence>
        <xsd:element ref="section" minOccurs="1" maxOccurs="unbounded" />
      </xsd:choice>
    </xsd:sequence>
    <xsd:attributeGroup ref="idreq" />
  </xsd:complexType>
</xsd:element>
```

Appendix C : Illustrations of Definitions in Appendix B

Global schema

- chapter
  - num (?)
  - heading (?)
  - subheading (?)
  - intro (?)
  - content (*)
    - componentRef
    - clause
    - section
    - part
    - paragraph
    - chapter
    - title
    - article
    - book
    - tome
    - division
    - list
    - point
    - indent
    - alinea
    - rule
    - subrule
    - proviso
    - subsection
    - subpart
    - subparagraph
    - subchapter
    - subtitle
    - subdivision
    - subclause
    - sublist
    - transitional
    - hcontainer
  - wrapUp (?)

Custom schema

- chapter
  - heading
  - article (+)
  - section (*)
  - section (+)