Московский государственный университет имени М.В.Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра Системного Программирования

Богатенкова Анастасия Олеговна

Извлечение иерархической логической структуры из текстовых документов в формате docx

Выпускная квалификационная работа

Научный руководитель:

Козлов Илья Сергеевич
Научный консультант:
Гомзин Андрей Геннадьевич

Москва, 2020

Аннотация

Извлечение иерархической логической структуры из текстовых документов в формате docx

Богатенкова Анастасия Олеговна

Многие документы имеют логическую структуру, выделение которой может помочь при решении задач автоматизированного анализа документов. В настоящее время большое количество документов создаётся и хранится в формате docx. Однако данный формат позволяет описывать только физическую структуру документа, то есть описывается то, как выглядит документ. В данной работе представлен один из способов, которым можно выделять из подобных документов логическую структуру.

Abstract

Hierarchical logical structure extraction from text documents in docx format

Bogatenkova Anastasiia Olegovna

Many documents have a logical structure, the extraction of which can help in solving problems of automated document analysis. Currently, a large number of documents are created and stored in docx format. However, this format allows you to describe only the physical structure of the document, that is, it describes how the document looks. This work presents one of the ways in which a logical structure can be extracted from such documents.

Содержание

1 Введение

Документы имеют определённую логическую структуру. Например, законы делятся на главы, статьи, разделы. Научные статьи состоят из аннотации, введения, обзора существующих работ и других секций. Информация о логической структуре полезна для автоматического анализа документа.

В настоящее время большое количество документов создается, редактируется и хранится в формате docx. Однако существует мало библиотек для работы с этим форматом. Существующие библиотеки могут игнорировать стили документа и символы нумерации в списках, однако эта информация крайне важна с точки зрения извлечения метаданных, необходимых для автоматического извлечения структуры.

Таким образом, задача обработки документов в формате docx и последующего извлечения структуры актуальна и может быть полезна во многих сферах деятельности, связанных с работой с подобными документами.

2 Постановка задачи

Задача состоит в извлечении логической структуры из документов в docx формате. Существуют различные способы представления структуры документа. Мы будем извлекать иерархическую структуру в виде дерева, так как многие документы состоят из последовательности вложенных друг в друга частей.

В рамках поставленной задачи необходимо выполнить следующее:

1. Провести обзор некоторых форматов документов;
2. Провести обзор способов представления логической структуры документа;
3. Описать особенности формата docx;
4. Описать структуру, которую необходимо извлечь;
5. Создать обучающий набор документов и осуществить его разметку;
6. Реализовать извлечение текста и необходимых метаданных из документов в формате docx;
7. Провести тестирование реализованного метода обработки docx документов;
8. Реализовать метод извлечения структуры и провести экспериментальную проверку реализованного метода;
9. Провести оценку качества.

3 Обзорная часть

Разные форматы для хранения текстовых документов разрабатывались для разных целей. В первой подсекции рассмотрим некоторые из форматов, которые использовались и используются для хранения и представления логической и физической структуры документа. Во второй подсекции опишем различные точки зрения на то, в каком виде можно представлять документ. И, наконец, в третьей подсекции более подробно изучим устройство docx формата.

3.1 Обзор по форматам документов

SGML и основанные на нем форматы

- SGML - язык, использующий разметку (дополнительные аннотации в содержимом документа). Стандартизован ISO в 1986 году.

  SGML документ состоит из трех файлов:

  1. DTD (определение типа документа),
  2. SGML декларация (описание символов, используемых в DTD и тексте документа),
  3. экземпляр документа (текст документа + ссылка на DTD).

  SGML предназначался для описания только логической структуры документа.

- DAFS предназначен для представления изображений документов и результатов распознавания таких документов. DAFS за основу берет SGML, однако расширяет его, позволяя хранить не только логическую структуру. Документ представляет собой иерархию вложенных друг в друга сущностей (документ, глава, блок), при этом есть два дерева - для логической и физической структуры, листья деревьев с содержимым документа общие. Сущность может иметь несколько предков, поэтому структура документа может иметь несколько вариантов иерархии. Это может использоваться для представления логической и физической структуры, а также могут храниться альтернативные структуры (если структура извлекалась не со 100% точностью).

XML и основанные на нем форматы

- XML - расширяемый язык разметки, являющийся подмножеством SGML. Расширяемый, так как не фиксируется конкретная разметка. XML-процессор (парсер) — программа, анализирующая разметку и передающая информацию о структуре документа другой программе — приложению. XML унаследовал от SGML описание типов с помощью DTD (но это не обязательно). Существуют парсеры, проверяющие соответствие документа типу, описанному в DTD.

  Основное различие между SGML и XML версиями состоит в следующем:

  - XML элементы должны быть всегда закрыты.
  - XML элементы должны иметь правильную вложенность.
  - значения атрибутов должны быть обязательно в кавычках.

- DocBook - стандарт, разработанный в основном для технической документации. Не предназначен для описания визуального представления документа, этим занимаются специальные утилиты. DocBook может быть основан на SGML или на XML.

  Элементы Docbook можно разделить на

  - структурные - книги, разделы, секции и т. п.
  - блочные - различные списки и параграфы.
  - строчные - форматирование внутристрочных элементов.
  - метаэлементы - вспомогательные метаэлементы, «атрибутирующие» другие элементы.

- DITA – специфический стандарт для работы с контентом.

  - Основной единицей контента в DITA является топик – «озаглавленный блок информации, который может быть понят в отдельности от других блоков и используется в различных контекстах».
  - DITA-карта представляет собой список указателей на набор конкретных топиков, определяющий, какие именно топики должны быть включены в поставку. Также карта определяет порядок и иерархию топиков и обеспечивает навигацию.

  - Выходные форматы DITA позволяют создавать поставки, включающие контент, организованный с помощью DITA-карт, в различных формах, предназначенных для различных целей.

- HTML, XHTML, HTML5 - языки для представления содержимого всемирной паутины. HTML — теговый язык разметки документов. Набор тегов и атрибутов указан в DTD - описании типа документа, общем для всех документов для конкретной версии HTML. Язык HTML до пятой версии соответстовал SGML стандарту, XHTML соответствует стандарту XML (более строгие требования к правильности документов), HTML5 является расширением SGML, от предыдущих версий отличается набором элементов и атрибутов, синтаксисом и поддержкой специальных данных (математических формул векторной графики). HTML5 был создан как единый язык разметки, который мог бы сочетать синтаксические нормы HTML и XHTML.

- ALTO - спецификация XML, предназначенная для хранения документов, обработанных с помощью систем оптического распознавания символов. Зачастую используется вместе с METS (является его частью). METS - спецификация XML, предназначенная для описания метаданных документа.

- TEI - формат, предназначенный для кодирования литературных и лингвистических текстов. Делается упор на семантику и логическую структуру документов, а не на визуальное представление. Основа документа - рекурсивно вложенные друг в друга элементы div, семантический смысл которых указан в атрибутах. TEI используется во многих сферах, поэтому спецификация очень обширна. Однако предоставляется возможность приспособить стандарт для конкретной предметной области.

- Office Open XML (OOXML, DOCX, XLSX, PPTX) - серия форматов файлов для хранения электронных документов пакетов офисных приложений — в частности, Microsoft Office. Формат представляет собой zip-архив, содержащий текст в виде XML, графику и другие данные. Форматы предназначены прежде всего для отображения физической структуры документа. Более подробная информация о docx находится в третьей подсекции.

ODA (Open Document Architecture)

Язык описания структурированных данных объектного типа. Язык более сложный, чем SGML, так как помимо логической структуры он описывает также физическую структуру.

Основные концепции ODA:

1. Логическая и физическая структуры документа (содержимое из логической структуры связывается с объектами физической структуры);

2. Специфическая (конкретная реализация) и общая (шаблонная) структуры документа;

3. Класс документа - набор характеристик, присущих отпределенной категории документов.

ODA-документ состоит из 6 частей: Logical View (логическая структура), Layout View (геометрическая структура), Logical and Layout Generic Structures (описывают класс документа), Content Information (текст документа + изображения), Styles (стили), Document Profile (имя автора, дата создания и т.д.).

Что еще можно использовать для структурирования информации

- JSON (JavaScript Object Notation) - простой формат обмена данными. Он основан на подмножестве языка программирования JavaScript. JSON основан на двух структурах данных:
  - Коллекция пар ключ/значение;
  - Упорядоченный список значений.
- YAML - формат, удобный для сериализации данных для всех языков программирования. YAML можно рассматривать как "расширенный JSON". Существуют некоторые отличия между данными форматами, например, возможность комментирования внутри файла, поддержка расширяемых типов данных, поддержка блочного синтаксиса с отступами, механизм для создания переменных, на которые можно ссылаться.

- TeX - Система компьютерной вёрстки для создания компьютерной типографии. В основном используется в научных документах, однако может подойти практически для любого домена. Основной замысел системы верстки заключается в том, что пользователь имеет возможность сосредоточиться на содержимом документа, практически не отвлекаясь на детали визуального представления. Автор документа может описывать различные логические структуры: глава, секция, таблица, рисунок, список и т. д., в то же время имеется возможность описания физической структуры документа вручную.

- PDF (Portable Document Format) - это формат файла, созданный Adobe Systems для обмена документами. PDF используется для представления страниц документов способом, независимым от прикладного программного обеспечения, оборудования и операционной системы. В настоящее время формат PDF широко используется для длительного хранения и архивирования документов в электронной форме. PDF можно сгенерировать в любом ПО для обработки документов, чтобы получить точное и фиксированное визуальное представление исходного документа. Однако PDF ориентирован на сохранение внешнего вида документа и не гарантирует сохранения его физической и логической структуры.

3.2 Обзор по представлениям структуры документов

Здесь описаны различные точки зрения на то, в каком виде можно представлять документ.

Представление структуры документа в виде дерева

Дерево помогает получить представление документа в виде иерархической структуры, то есть документ разбивается на последовательность вложенных друг в друга элементов (рис. 1).

Представление структуры документа в виде графа

Дерево - частный случай графа. Произвольный граф позволяет представить разбиение документа на части (каждая часть является вершиной графа), а также описать порядок чтения частей (ребра графа могут быть помечены и описывать тип взаимоотношений между частями документа). Структура докумета при этом может получиться не обязательно иерархической (рис. 2).

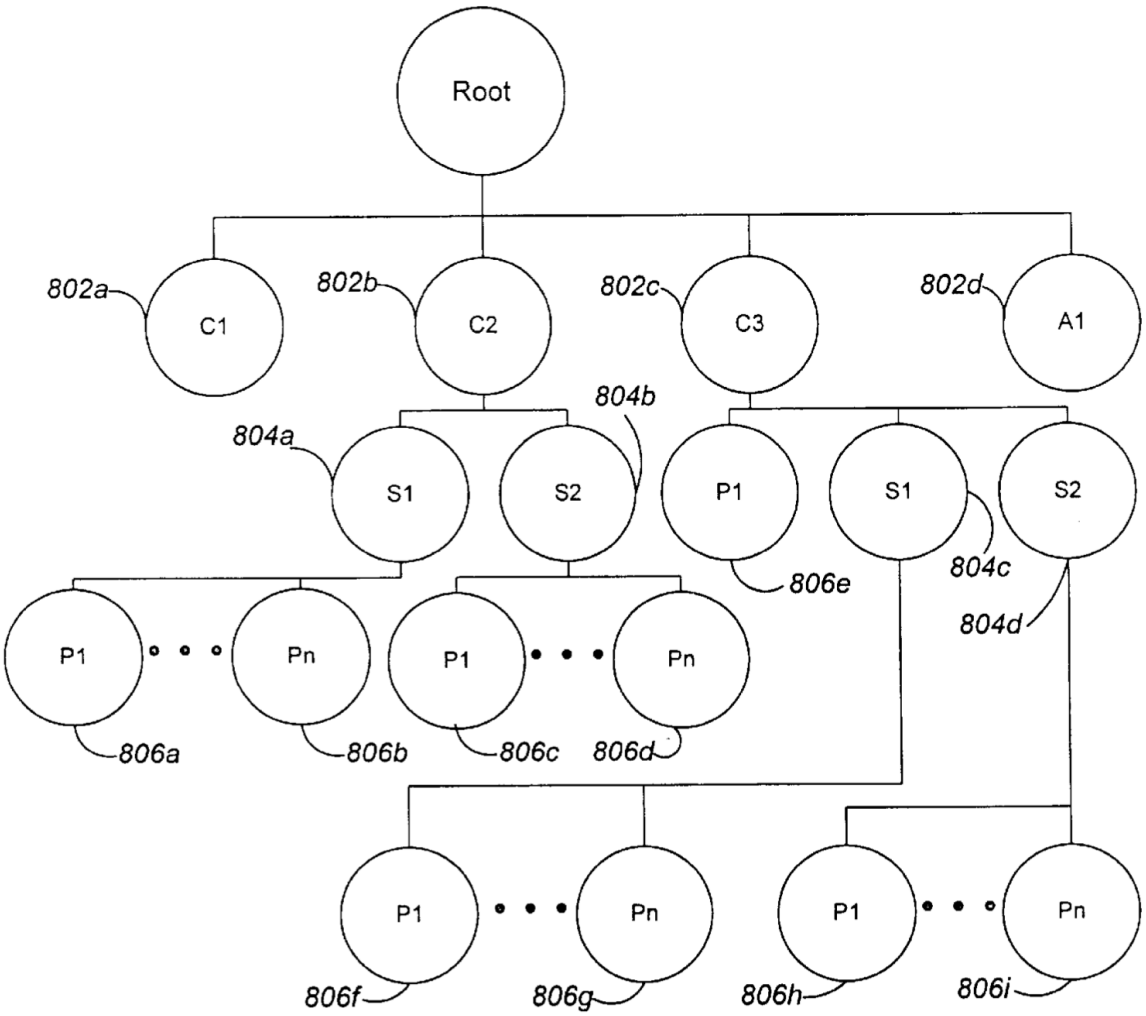

Рис. 1: Пример структуры документа в виде дерева

Представление структуры документа с использованием формальных грамматик

Документ может быть представлен последовательностью правил, которые необходимо обработать с помощью специального парсера. В результате такой обработки получается исходный документ (рис. 3).

Представление структуры документа в виде зон и логических меток

Документ может быть представлен как плоская структура: последовательность частей какого-либо типа. Такими частями могут быть страницы, текстовые блоки, строки, слова, символы и т. д. Эти части могут описывать документ в физическом смысле

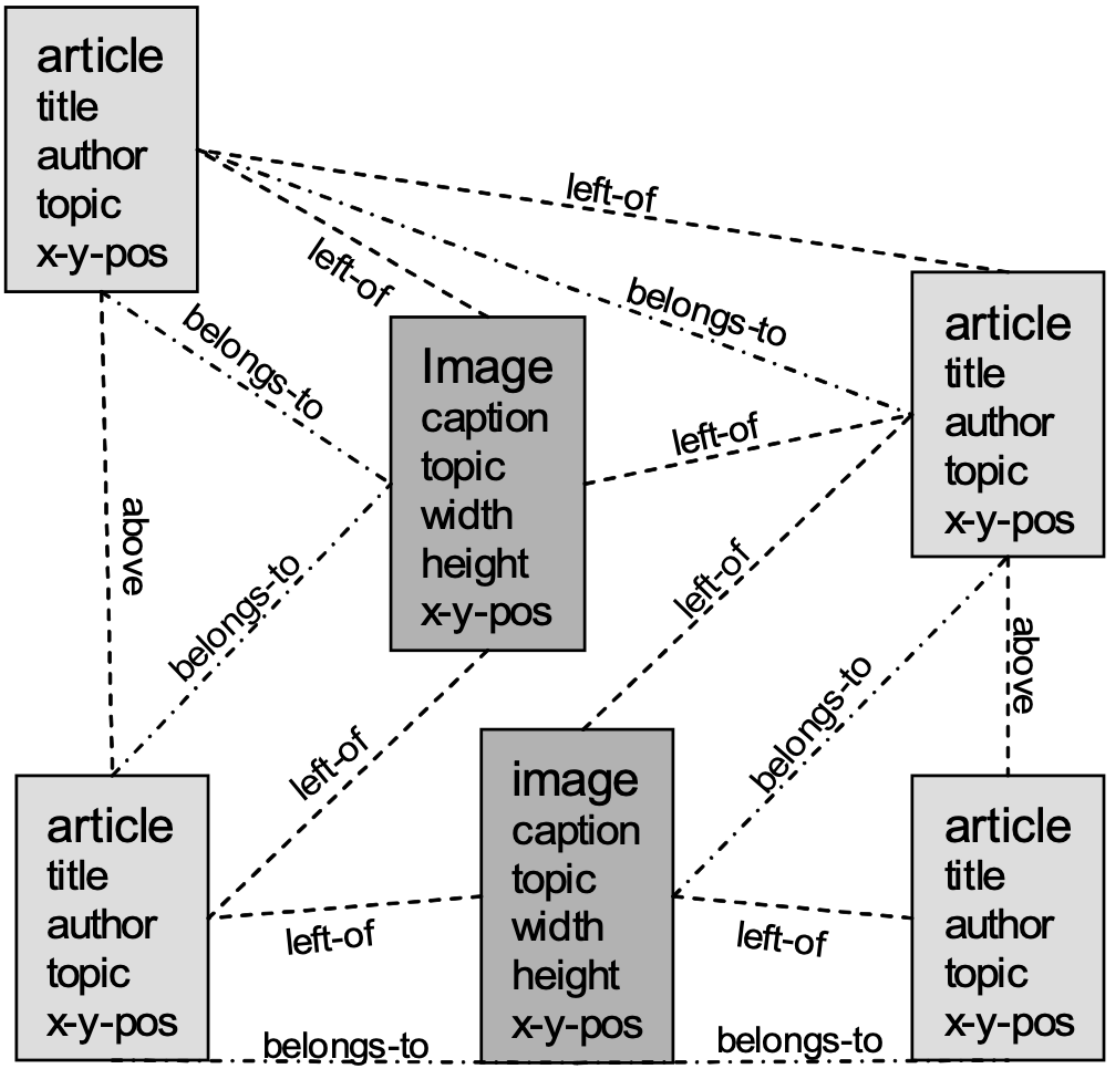

Рис. 2: Пример структуры документа в виде графа

или в логическом. Для представления всей структуры может быть установлена взаимосвязь логических блоков документа с геометрическими. Другой подход: документ можно разбить на физические блоки, каждому блоку можно назначить семантическую метку (таким образом получим логическую структуру).

3.3 Структура docx формата

Docx-файл —- это zip-архив который физически содержит 2 типа файлов:

$[0.5] < START > \rightarrow < TITLE > < COLUMN > < COLUMN >$
$[0.5] < START > \rightarrow < TITLE > < COLUMN >$
$[1.0] < TITLE > \rightarrow < text\_line >$
$[1.0] < COLUMN > \rightarrow < TEXT\_BLOCKS >$
$[0.8] < TEXT\_BLOCKS > \rightarrow < TEXT\_BLOCK > < space > < TEXT\_BLOCKS >$
$[0.2] < TEXT\_BLOCKS > \rightarrow < TEXT\_BLOCK >$
$[1.0] < TEXT\_BLOCK > \rightarrow < TEXT\_LINES >$
$[0.9] < TEXT\_LINES > \rightarrow < text\_line > < newline > < TEXT\_LINES >$
$[0.1] < TEXT\_LINES > \rightarrow < text\_line >$

(a)

$< START > \rightarrow < TITLE > < COLUMN >$
$\rightarrow < text\_line > < COLUMN >$
$\rightarrow < text\_line > < TEXT\_BLOCKS >$
$\rightarrow < text\_line > < TEXT\_BLOCK > < space > < TEXT\_BLOCKS >$
$\rightarrow < text\_line > < text\_line > < newline > < TEXT\_BLOCK > < space > < TEXT\_BLOCKS >$
$\rightarrow < text\_line > < text\_line > < newline > < text\_line > < space > < TEXT\_BLOCKS >$
$\rightarrow < text\_line > < text\_line > < newline > < text\_line > < space > < TEXT\_BLOCK >$
$\rightarrow < text\_line > < text\_line > < newline > < text\_line > < space > < text\_line >$

(b)

Fig. 6. Representing a document in terms of formal grammars: (a) example of a stochastic context free grammar that derives a text document with a title. Upper case symbols refer to non-terminal symbols, while lower case symbols show terminal symbols. (b) a sample document with a title and three lines, derived using the grammar in (a)

Рис. 3: Пример структуры документа в виде формальной грамматики

- xml файлы с расширениями xml и rels;
- медиа файлы (изображения и т.п.).

Логически — 3 вида элементов:

- Типы (Content Types) —- список типов медиа файлов (например png) встречающихся в документе и типов частей документов (например документ, верхний колонтитул);
- Части (Parts) — отдельные части документа (например document.xml, footer1.xml, header1.xml, comments.xml, endnotes.xml), сюда входят как xml документы, так и медиа файлы;

- Связи (Relationships) идентифицируют части документа для ссылок (например связь между разделом документа и колонтитулом), а также тут определены внешние части (например гиперссылки).

Документация для word/document.xml

Выделяются следующие основные теги:

- document – соответствует содержимому документа;
- body – содержит весь текст документа, разделенный на параграфы;
- p (paragrath) – основная форма хранения текста, параграфы разделяются между собой символом переноса строки; pPr – свойства параграфа (например, выравнивание и отступ);
- r (raw) – элемент, составляющий текстовый регион с набором общих свойств (работает на уровне символов); rPr – свойства элемента;
- sectPr (section properties) – в документах могут встречаться секции – наборы параграфов с общими свойствами.

Свойства параграфа – основные свойства, которые описывают отдельный блок текста. Перечислим свойства, которые могут пригодиться в рамках выделения логической структуры:

- ind – отступ от края страницы;
- js – выравнивание;
- numPr – индикатор того, что параграф является элементом нумерованного или маркированного списка;
- pStyle – стиль параграфа;

Свойства элемента – основные свойства, которые описывают отдельный набор подряд идущих символов. Перечислим свойства, которые могут пригодиться в рамках выделения логической структуры:

- b (bold) – жирный шрифт;
- i (italic) – курсив;
- u (underline) – подчеркивание;
- rStyle – стиль элемента;
- sz – размер шрифта;
- t – текст элемента, отображающийся в документе;
- tab, br, cr, sym – специальные символы, которые могут встретиться в тексте элемента.

Документация для word/numberings.xml

В файле word/numberings.xml содержится информация обо всех типах списков, используемых в документе и их настройках (рис. 4). Выделяются следующие теги:

- abstractNum – описание свойств конкретного типа списка, этому типу присваивается номер (abstractNumId), который используется в word/document.xml (свойства наследуются);
- abstractNumId (val="...") – уникальный идентификатор типа списка;
- ilvl (val="...") – уровень вложенности;
- lvl – описывает свойства списка определенного уровня внутри lvlOverride или внутри abstractNum;
- lvlOverride (Numbering Level Definition Override) – ипользуется внутри num, перегружает свойства конкретного уровня, отнаследованные от abstractNum и заменяет их;
- num – тег для word/numberings.xml;
- numId – номер экземпляра типа списка.

```
<w:abstractNum w:abstractNumId="0">
 <w:multiLevelType w:val="hybridMultilevel"/>
 <w:numStyleLink w:val="Imported Style 2"/>
</w:abstractNum>
```

Рис. 4: Пример файла word/numberings.xml

Документация для word/styles.xml

В данном файле описывается набор стилей, примененных к документу и отношения наследования между ними. Если один из стилей является наследником другого, то все не перекрытые свойства стиля-предка становятся свойствами стиля-наследника. Свойства текста для документа определяются в соотвествии с диаграммой, показанной на рис. 5.

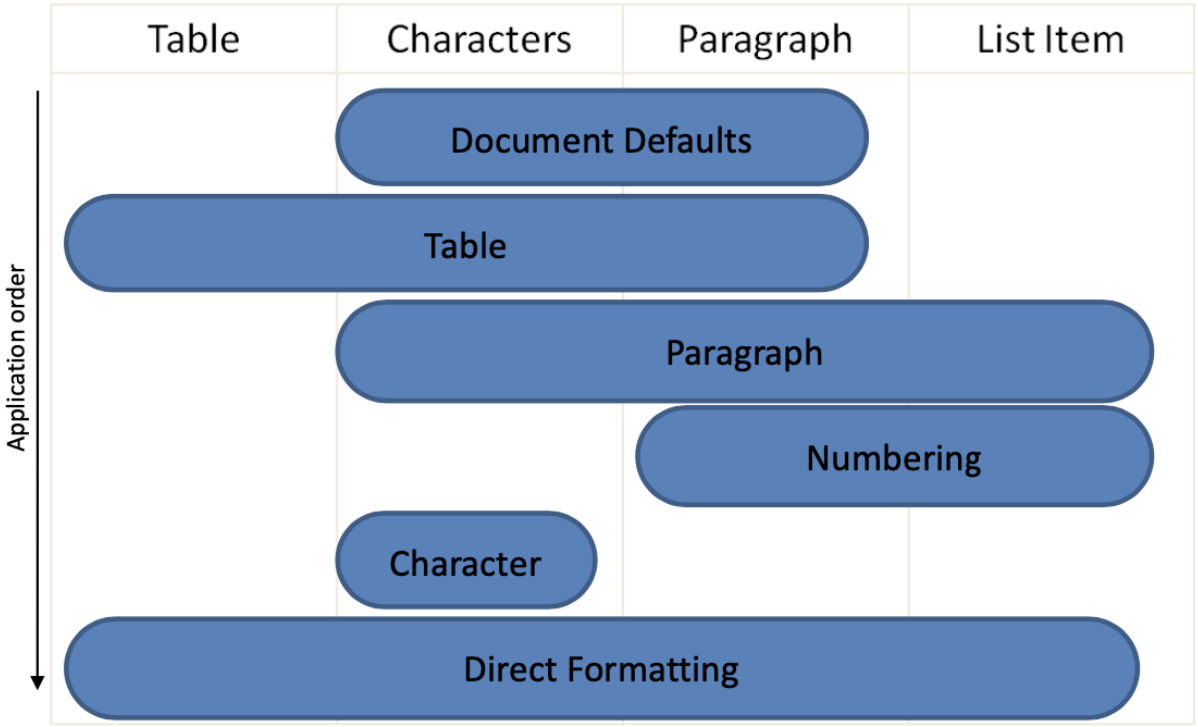

Рис. 5: Иерархия стилей

Выделяются следующие теги:

- w:styles (w:style) – содержат информацию о стилях;

- basedOn – наследование стилей (параграфы и символы наследуют свойства параграфов и символов соответственно, нумерация не наследуется);
- docDefaults – настройки стиля по умолчанию.

Документация для word/header1.xml и word/footer1.xml

Помимо основного текста в документе могут содержаться области, находящиеся выше или ниже основных областей с текстом (хедеры, футеры), а также различные заметки и сноски. Для этого в docx-документе предусмотрены отдельные файлы word/header1.xml, word/footer1.xml (до четырех экземпляров), word/footnotes.xml, word/endnotes.xml.

Основные теги:

- hdr – содержимое header (аналогично содержимому body для документа);
- ftr – содержимое footer;

В файле word/document.xml находятся ссылки в свойствах секции sectPr:

- headerReference;
- footerReference.

Внутри секции хедеры и футеры могут быть трёх типов - для четных, нечетных страниц и первой страницы (для каждого типа отдельный файл).

Аналогичным образом определяются теги для заметок и сносок (footnote и footnoteReference, endnote и endnoteReference).

4 Исследование и построение решения задачи

5 Описание практической части

6 Заключение

Список литературы

[1] Github repository with code. `https://github.com/NastyBoget/DOCXParser`.