Министерство науки и высшего образования Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение науки и высшего образования**

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

**(ВлГУ)**

**Кафедра информационных систем и программной инженерии**

Лабораторная работа №1

по дисциплине

" Интеграция кроссплатформенных программных систем"

Тема: " Определение структуры XML-документа средствами DTD и XML-схем"

Выполнил:

ст. гр. ИСТм-121

В.Г. Хлызова

Принял:

Спирин И.В.

Владимир, 2022 г.

ЦЕЛЬ РАБОТЫ

Данная работа предусматривает разработку формального описания структуры XML-документов, представляющих информацию из какой-то предметной области. Модель предметной области необходимо представить в виде диаграммы классов UML. Описание необходимо выполнить средствами DTD и в виде XML-схемы, при этом привести пример XML-документов, соответствующих DTD-описанию и схеме.

ХОД РАБОТЫ

Предметная область: проектирование информационных систем.

[Ключевые элементы BPMN моделей.](https://www.comindware.ru/blog/%d0%bd%d0%be%d1%82%d0%b0%d1%86%d0%b8%d1%8f-bpmn-2-0-%d1%8d%d0%bb%d0%b5%d0%bc%d0%b5%d0%bd%d1%82%d1%8b-%d0%b8-%d0%be%d0%bf%d0%b8%d1%81%d0%b0%d0%bd%d0%b8%d0%b5/)

Модель данных

Требования к модели данных (разработанная модель данных будет единой для всех лабораторных работ этого семестра):

1) модель данных должна содержать не менее 3-х взаимосвязанных сущностей (документов); кроме сущностей должны присутствовать вспомогательные классы (не менее 4-х), описывающие комплексные типы данных.

2) между классами должны быть определены отношения типа «один-ко-многим», могут быть использованы другие виды отношений; должны присутствовать как однонаправленные, так и двунаправленные связи.

3) крайне желательно выбрать предметную область, совпадающую с темой выпускной квалификационной работой.

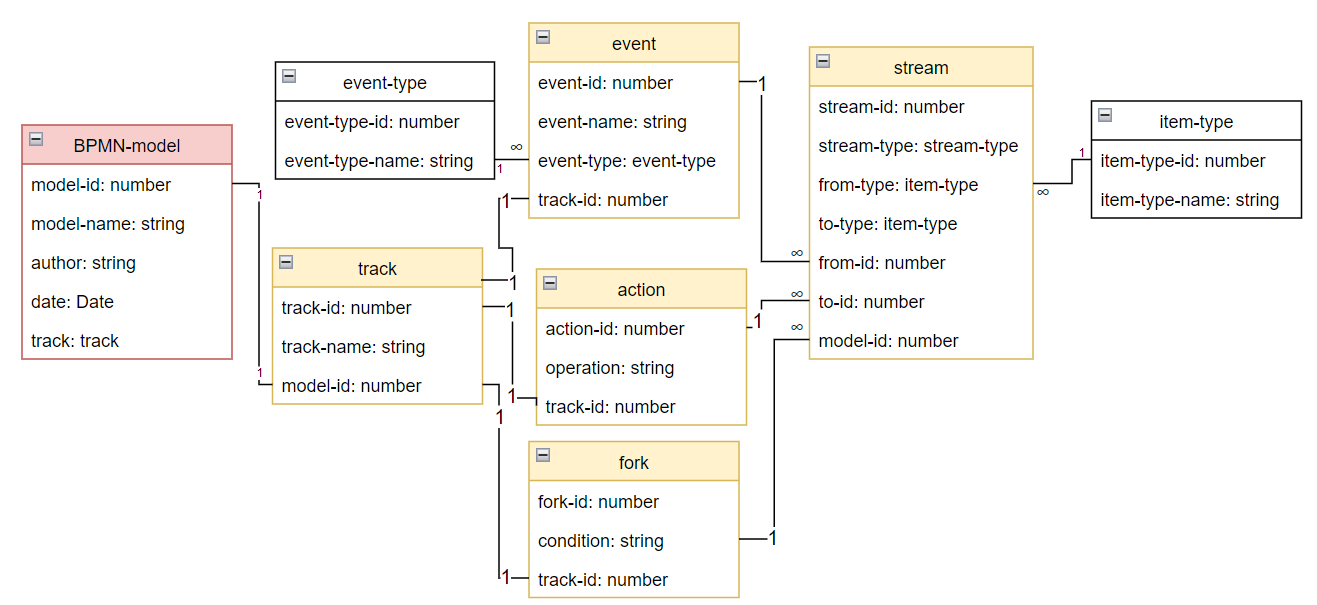


Рисунок 1 - Модель данных

Технические требования для DTD-описания

1) должно быть выполнено в отдельном внешнем файле, а не непосредственно в рамках документа-примера.

2) документ-пример должен содержать ссылку на DTD-описание.

3) DTD-описание должно демонстрировать использование конструкций описания элементов с вложенными элементами и применением следования, альтернативы, необязательности и повторяемости этих вложенных элементов.

4) DTD-описание должно демонстрировать использование конструкций описания атрибутов, как обязательных, так и необязательных, в том числе со значением по умолчанию для атрибутов следующих типов: ID, IDREF или IDREFS, CDATA, а также одного из перечисляемых типов

5) DTD-описание должно демонстрировать использование параметризованных сущностей и ссылок на них.

6) DTD-описание должно демонстрировать использование общих сущностей и ссылок на них в документе-примере.

7) DTD-описание должно демонстрировать использование внешних не анализируемых сущностей в сочетании с описанием атрибута.

DTD-описание:

<!DOCTYPE INFORMATION\_SYSTEM\_DESIGN [

<!ELEMENT BPMN-model (model-name, author, date, tracks, streams)>

<!ELEMENT model-name (PCDATA)>

<!ELEMENT author (PCDATA | EMPTY)>

<!ELEMENT date (PCDATA | EMPTY)>

<!ELEMENT tracks (track+)>

<!ELEMENT streams (stream\*)>

<!ELEMENT track (track-name, actions, events, forks)>

<!ATTLIST track id ID #REQUIRED>

<!ELEMENT track-name (PCDATA)>

<!ELEMENT actions (action\*)>

<!ELEMENT forks (fork\*)>

<!ELEMENT events (event\*)>

<!ELEMENT action (operation)>

<!ATTLIST action id ID #REQUIRED>

<!ELEMENT operation (PCDATA)>

<!ELEMENT fork (condition)>

<!ATTLIST fork id ID #REQUIRED>

<!ELEMENT condition (PCDATA)>

<!ELEMENT event (event-name)>

<!ATTLIST event id ID #REQUIRED>

<!ATTLIST event type (start | intermediate | end)>

<!ELEMENT event-name (PCDATA | EMPTY)>

<!ENTITY %item>

<!ATTLIST %item id ID #REQUIRED>

<!ATTLIST %item type (action | fork | event)>

<!ATTLIST %item item-id IDREF>

<!ELEMENT stream (from, to, stream-name)>

<!ATTLIST stream id ID #REQUIRED>

<!ELEMENT from (%item)>

<!ELEMENT to (%item)>

<!ELEMENT stream-name (PCDATA | EMPTY)>

]>

Технические требования для XML-схемы

1) схеме должно быть сопоставлено определенное пространство имен, при этом должна быть предусмотрена квалификация элементов, определенных схемой, в документе, построенном на ее основе

2) документ-пример должен содержать ссылку на XML-схему

3) схема должна демонстрировать использование конструкций для описания сложного типа, образованного вложенными элементами и атрибутами; при этом должен быть хотя бы один сложный тип, хотя бы один вложенный элемент которого был бы также сложного типа; при описании сложных типов необходимо продемонстрировать использование не менее двух видов модельных групп (sequence, choice и all)

4) схема должна содержать описание абстрактного сложного типа и элемента этого типа, а пример документа - демонстрировать использование последнего

5) схема должна демонстрировать возможность описания нового сложного типа на базе как простого типа, так и сложного как путем расширения, так и путем ограничения

6) схема должна демонстрировать возможности описания новых простых типов на базе существующих (в том числе описание перечислимого типа) с использованием не менее трех видов фасетов

7) схема должна содержать описание ограничения уникальности, ключа и ссылки на ключ, применение чего должно иметь смысл в предметной области.

XML-схема:

<?xml version="1.0" encoding="UTF-8" ?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <xs:element name="BPMN-model">

    <xs:complexType>

    <xs:sequence>

        <xs:element name="model-name" type="xs:string"/>

        <xs:element name="author" type="xs:string"/>

        <xs:element name="date" type="xs:date"/>

        <xs:element name="tracks">

        <xs:complexType>

        <xs:sequence>

            <xs:element name="track">

            <xs:complexType>

                <xs:sequence>

                <xs:element name="track-name" type="xs:string"/>

                <xs:element name="events">

                <xs:complexType>

                <xs:sequence>

                <xs:element name="event">

                <xs:complexType>

                <xs:sequence>

                <xs:element name="event-name" type="xs:string"/>

                </xs:sequence>

                <xs:attribute name="id" type="xs:string" use="required"/>

                <xs:attribute name="type" type="start | intermediate | end" />

            </xs:complexType>

            </xs:element>

        </xs:sequence>

        </xs:complexType>

        </xs:element>

        <xs:element name="actions">

        <xs:complexType>

        <xs:sequence>

            <xs:element name="action">

            <xs:complexType>

                <xs:sequence>

                <xs:element name="operation" type="xs:string"/>

                </xs:sequence>

                <xs:attribute name="id" type="xs:string" use="required"/>

            </xs:complexType>

            </xs:element>

        </xs:sequence>

        </xs:complexType>

        </xs:element>

        <xs:element name="forks">

        <xs:complexType>

        <xs:sequence>

            <xs:element name="fork">

            <xs:complexType>

            <xs:sequence>

            <xs:element name="condition" type="xs:string"/>

            </xs:sequence>

            <xs:attribute name="id" type="xs:string" use="required"/>

            </xs:complexType>

            </xs:element>

            </xs:sequence>

            </xs:complexType>

            </xs:element>

        </xs:sequence>

        </xs:complexType>

        </xs:element>

        </xs:sequence>

        </xs:complexType>

        </xs:element>

        <xs:element name="streams">

        <xs:complexType>

        <xs:sequence>

            <xs:element name="from" type="Item" />

            <xs:complexType name="Item">

            <xs:attribute name="id" type="xs:string" use="required"/>

            <xs:attribute ref="event11"/>

            <xs:attribute name="type" type="action | fork | event" />

            </xs:complexType>

            <xs:attribute name="to" type="Item" />

            <xs:complexType name="Item">

            <xs:attribute name="id" type="xs:string" use="required"/>

            <xs:attribute ref="event11"/>

            <xs:attribute name="type" type="action | fork | event" />

            </xs:complexType>

            <xs:element name="stream-name" type="xs:string"/>

        </xs:sequence>

        </xs:complexType>

        </xs:element>

    </xs:sequence>

    </xs:complexType>

    </xs:element>

</xs:schema>

Документ пример:

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE system SYSTEM "system.dtd">

<BPMN-model xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="system.xsd">

    <model-name>Example</model-name>

    <author>Valeria khlyzova</author>

    <date>2022-06-06</date>

    <tracks>

        <track id="track1">

            <track-name>Customer</track-name>

            <events>

                <event id="event11" type="start">

                    <event-name>Start</event-name>

                </event>

            </event>

            <actions>

                <action id="action11">

                    <operation>Order and pay for the product</operation>

                </action>

            </actions>

        </track>

        <track id="track2">

            <track-name>Seller</track-name>

            <events>

                <event id="event21" type="end">

                    <event-name>End</event-name>

                </event>

            </events>

            <actions>

                <action id="action21">Check the product in storehouse</action>

                <action id="action22">Send the ordered product</action>

                <action id="action23">Return money</action>

            </actions>

            <forks>

                <fork id="fork21">

                    <condition>is the product in storehouse?</condition>

                </fork>

            </forks>

        </track>

    </tracks>

    <streams>

        <stream id="stream1">

            <from id="from1" type="event" item-id="event11" />

            <to id="to1" type="action" item-id="action11" />

        </stream>

        <stream id="stream2">

            <from id="from2" type="action" item-id="action11" />

            <to id="to2" type="action" item-id="action21" />

        </stream>

        <stream id="stream3">

            <from id="from3" type="action" item-id="action21" />

            <to id="to3" type="fork" item-id="fork21" />

        </stream>

        <stream id="stream4">

            <from id="from4" type="fork" item-id="fork21" />

            <to id="to4" type="action" item-id="action23" />

            <stream-name>no</stream-name>

        </stream>

        <stream id="stream5">

            <from id="from5" type="fork" item-id="fork21" />

            <to id="to5" type="action" item-id="action22" />

            <stream-name>yes</stream-name>

        </stream>

        <stream id="stream6">

            <from id="from6" type="action" item-id="action22" />

            <to id="to6" type="event" item-id="event21" />

        </stream>

        <stream id="stream7">

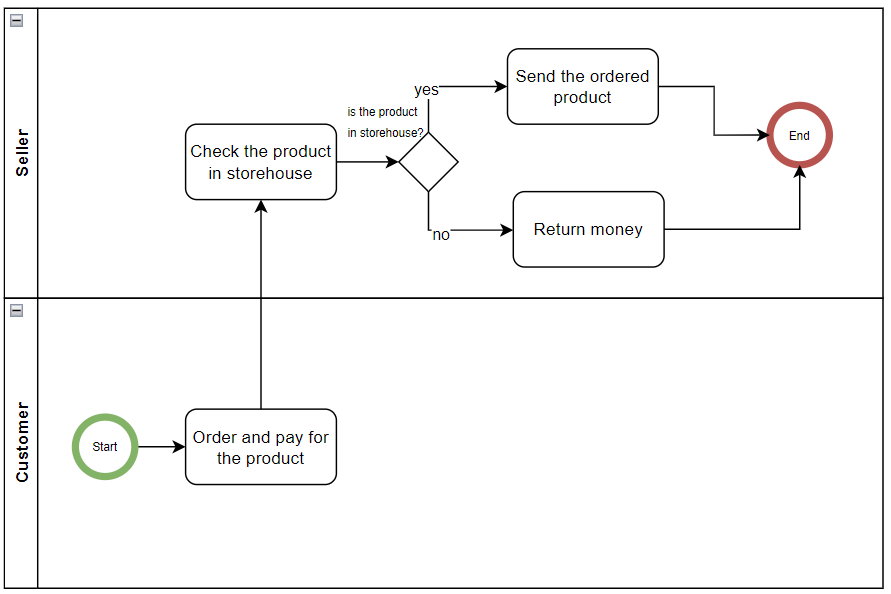
            <from id="from7" type="action" item-id="action23" />

            <to id="to7" type="event" item-id="event21" />

        </stream>

    </streams>

</BPMN-model>



Ссылка на репозиторий: <https://github.com/arranay/Integration_of_cross_platform_software_systems>

ВЫВОД

В ходе выполнения лабораторной работы были изучены структура XML-документов и DTD.