Минно-геоложки университет "Св. Иван Рилски"
Годишник, том 47, свитък III, Механизация, електрификация и автоматизация на мините, София 2004, стр. 125-130

Съвременни методи за обмен на данни в минната промишленост

Кънчо Иванов, Динко Белчевски

Минно-геоложки университет „Св. Иван Рилски", 1700 София

РЕЗЮМЕ. В минният бранш в много от предприятията данните се пазят локално в собствен формат използван от специализирано софтуерно приложение, като използването им в друго приложение е трудно и често е свързано със загуба на информация. На настоящия етап е възможно да се поддържат по-сложна и прозрачна размяна на техническа и изследователска информация.
Настоящият доклад прави обзор на развитието на стандартите и третира методите за селекция на базисно множество от компоненти на мрежовите протоколи и инструменти на комуникацията, чието приложение осигурява:

- наборите от данни да са описани чрез добре формулирани общи модели
- файловите формати да са съвместими с протоколите за електронен бизнес
- прозрачност на интерфейса към общите бази данни
- удобното им разширяване до файлове за многопотребителска работа във WWW
- непосредствено прехвърляне на данни между различни софтуерни приложения.

CONTEMPORARY METHODS FOR DATA EXCHANGE IN MINING INDUSTRY
ABSTRACT. In mining branch most enterprises data is held locally in proprietary formats for use within given specialised software application, and re-use in another can be difficult and is often accompanied by net information loss. Now maturing to a stage, where is possible to support more sophisticated and transparent exchange of technical exploration information.
The report summarizes the developments in the standards and examine selection methods on basis multitude of components on net protocols and communication tools, whose applying ensures:

- data sets were described with well-understood common models
- file formats are compatible with e-business protocol
- database interface transparency
- database conveniently extends into multiclient file data on the WWW
- immediate import and transfer between a variety of applications software

Въведение

Широкото навлизане на WWW постави въпроса за свободната и унифицирана обмяна на данни между различни софтуерни приложения. Използването на „частни" двоични файлови формати не отговаря на някои от съвременните изисквания за обмен на данни. Причината за това е различното представяне на метаданните, описващи самите данни, специфични за различните фирми разработващи софтуер. Това затруднява обмена на данни между различни софтуерни приложения, тъй като се налага преобразуване на двоичните данни от един формат в друг. Това преобразуване не винаги е успешно, било поради недостатъчната документация за дадения файлов формат, било поради невъзможността на разработчиците да се справят с тази задача.

В противовес с тях са текстовите файлове. Един текстов файл може да бъде отворен с произволен текстов редактор независимо с каква програма е бил създаден. Това ги прави подходящи за пренос на данни между различни платформи и приложения. Основен недостатък на чистите текстови формати е липсата на структурираност на информацията. Този проблем се решава с използването на езиците за форматиране. Типичен представител на езиците за форматиране е HTML (Hypertext Markup Language). Той е най-известното приложение на езика SGML (Standart Generalized Markup Language), който е текстово-базиран език използван за форматиране (mark-up) на данни – т.е. за добавяне на метаданни по един самоописателен начин. HTML е универсален език за форматиране, който служи за представянето на информация и за свързването на различни парчета информация.

Въпреки успеха на HTML, той също има своите ограничения – предназначен е само за показване на документи в браузър. От него не може да се разбере кое парче информация за какво отнася, тъй като HTML няма средства за описание на специализирана информация.

Точно поради тази причина е създаден XML (Extensible Markup Language). Той представлява подмножество на SGML и има същите цели – форматиране на произволен тип данни. Но XML е значително по-прост от SGML, като при това опростяване е постигната съвместимост между XML и SGML (но не и обратно).

Език за форматиране XML

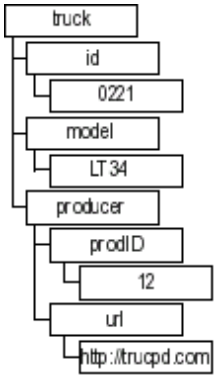
Основа на езика

XML е език, който позволява структуриране на информация от произволен тип. Самият XML е метаезик, позволяващ дефинирането и създаването на различни подезици на негова база. Информацията се пази в прости текстови файлове, като структурата й се дефинира чрез допълнителна (мета) информация наречена тагове. За разлика от HTML таговете в XML не са предефинирани, като това е

оставено на разработчика. Информацията може да се представи чрез затварянето й между два тага (наречени отварящ и затварящ) или като атрибут на даден таг.

```
<truck>
        <id>0221</id>
        <model>LT34</model>
        <producer>
                <prodID>12</prodID>
                <url>http://trucpd.com</url>
        </producer>
</truck>
```

Всеки един XML документ организира информацията в дървовидна йерархия. Всяка част от дървото която има наследници се нарича клон, докато частите които нямат – листа. Йерархията на горния документ има следния вид:



Фиг. 1. Йерархия на XML документ

Тук елемента truck се нарича корен (root) на дървото, елемента producer е клон, а елементите id, model, prodID и url – листа.

Document Type Definition (DTD), XML Shema

Добре конструираните XML данни гарантирано използват правилен XML синтаксис и имат подходящо свързана (йерархична) структура, която е основна за всички XML данни. Това е достатъчно са статични вътрешни приложения, особено ако кода е програмно генериран, но води до трудности при определянето и промяната на структурата на данните [1].

Отделянето на описанието на XML данните от индивидуални приложения позволява всички кооперищи се приложения да споделят едно единствено описание на данните, наречено XML речник. Когато група от XML документи споделят общ XML речник, то те се наричат документен тип.

Спецификацията XML 1.0 описва стандартизиран метод за описание на XML документни типове наречена дефиниция на документен тип (DTD)[1].

Друг метод за описание на данните и създаване на валиден код е използването на схеми. Под схема се разбира организацията или структурата на база данни, която обикновено се поражда от моделирането на данни. В контекста на XML под схема се обикновено се има предвид предстоящото приложение на W3C за стандарт XML Shema, която използва XML за всички свои описания на данни. Най-общо XML схемите трябва[1]:

- Да осигуряват богато множество от типове данни от примитивни типове данни.
- Да осигуряват система от типове данни, позволяваща създаването на вградени и дефинирани от потребителя типове данни.
- Да отличават представянето на лексикалните данни от залегналото информационно множество.

Представяне на XML данни (CSS, XSLT)

Тъй като XML акцентира върху структурата на данните, а на върху тяхното представяне, то за тази цел са разработени няколко технологии позволяващи визуализирането на XML данните или тяхното конвертиране в подходящ формат.

Таблиците със стилове (style sheets) служат за подобряване на представянето на XML данни, а таблиците с каскадни стилове (CSS – Cascading Style Sheets) може да се използват за форматиране на данните. Използването на CSS позволява директното показване на XML документ в браузър, който поддържа XML. Такива са повечето съвременни браузъри като Internet Explorer и Mozilla. Когато някои от тези браузъри отвори XML документ, той проверява за асоциирана таблица с каскадни стилове и ако намери такава, то той визуализира документа спрямо дефинициите й. Ако към документа няма асоциирани каскадни стилове, браузъра визуализира дървовидната структура на документа.

Въпреки, че е възможен този метод на визуализиране има някои съществени недостатъци. Например не всички браузъри поддържат директно XML. Освен това е възможно XML документа да е структуриран по начин, който да не позволява директното му визуализиране или пък да се налага трансформиране на документа в друг формат.

За тази цел е разработен Extensible Style sheet for Transformation (XSLT). XSLT е език който позволява трансформиране на XML документа в произволен текстово базиран формат (фиг. 2).

Фиг. 2. XSLT трансформация

XSLT е подкомпонент на един по-голям език наречен Extensible Style sheet Language (XSL), който е XML базиран език и служи за създаване на таблици със стилове, които дефинират изобразяването на изходния документ (резултатно дърво – result tree), както и от къде да се вземат данните от входния документ (изходно дърво – source tree). След това XML машината използва таблиците със стилове за преобразуване на XML документи в други документни типове, както и за форматиране на изхода.

XSLT таблиците със стилове са изградени посредством структури, наречени шаблони (templates), които определят какво да се търси в изходното дърво и какво да се постави в резултатното дърво.

Благодарение на XSLT е възможно преобразуването на един тип XML документ в друг, в HTML документ или в документ с обикновен текст.

XML в минната индустрия

На базата на XML са разработени множество подезици,

които имат за цел да решат различни задачи. Почти всяка сфера от науката или индустрията използват XML в дадена степен. Това важи и за минната индустрия, в която се използват както общи спецификации, така и спецификации разработени специално за нея. Статията разглежда само някои от тях, а именно GML (Geography Markup Language), XMML (Exploration and Mining Markup Language), IREDES (International Rock Excavation Data Exchange Standart) и SVG (Scalable Vector Graphics).

GML

Geography Markup Language (GML)[4] е XML базиран стандарт използван за предаване и съхраняване на географска информация, който поддържа както пространствени, така и други характеристики (примитиви) на релефа. Тази спецификация дефинира XML Shema синтаксис, чиито механизми и конвенции:

- Осигуряват отворена и независима рамка за дефиниране на географски схеми и обекти.
- Поддържат описание на географските схеми за специализирани области.
- Позволяват създаване и управление на свързани географски схеми и набори от данни.
- Предават и съхраняват географските схеми и набори от данни.
- Увеличават способността на организациите да споделят географска информация.

В GML реалния свят се представя цифрово, като набор от характеристики на релефа. Всяка характеристика е дефинирана чрез набор от свойства, като всяко свойство има име, стойност и тип. В GML свойствата са два типа – основни свойства и геометрични свойства. Всяка характеристика може да има както множество основни свойства, така и множество геометрични свойства.

Важна особеност е факта, че една характеристика може да бъде съставена от няколко други характеристики. Такава характеристика е наречена колекция от характеристики. Колекцията от характеристики притежава свой собствен тип както и собствени свойства различни от тези на съставляващите я характеристики. Например един град може да се представи като колекция от характеристики, които описват отделните пътища, сгради, реки и др.

GML спецификацията се базира на т.нар. „проста“ характеристика (примитив). Тя е опростена версия на по-общия модел описан от OGC (Open GIS Consortium) Abstract Specification. Направени са основно две опростявания, като те водят до дефинирането на „простата“ характеристика като:

- Характеристиката има или основни свойства (цели числа, числа с плаваща запетая, низове) или геометрични свойства.
- Характеристиките, които геометрични свойства се описват в двуизмерна координатна система, използват линейна интерполация между координатите.

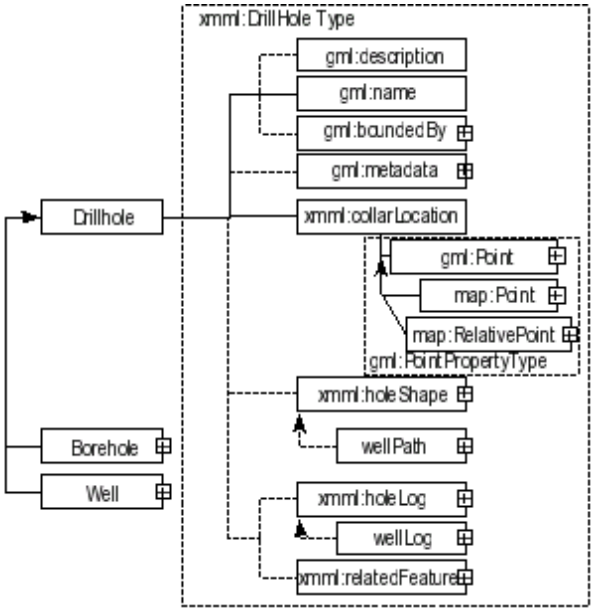
Чрез комбинирането на „простите“ характеристики се постига пълно описание на релефа на местността [2].

XMML

Въпреки, че GML е разработен за работа с географски данни, то той не е особено подходящ за представяне на геоложки данни, тъй като не поддържа от специфичните за сондирането характеристики.

XMML[5] е приложна схема на GML. XMML използва основните характеристики на GML като добавя допълнителни, които са специфични за геологията. На практика това се постига чрез добавяне на подходящи геометрични и не геометрични свойства към GML схемата.

Например ако се разглежда един сондажен отвор (borehole) като XMML характеристика, освен типичните GML свойства като име (name), описание (description) и обкръжение (boundedBy), той може да има и типично геоложки свойства като местоположение на сондажа (collarLocation) и вид на сондажа (holeShape). Тази структурата е показана на фиг. 3:



Фиг. 3. XMML „Borehole“ модел

Част от документа описващ „Borehole“ схемата има следния вид:

```
<xsd:element name="Drillhole" type="xmml:DrillholeType"
substitutionGroup="gml:_Feature"/>
<xsd:element name="Borehole" type="xmml:DrillholeType"
substitutionGroup="xmml:Drillhole"/>
<xsd:element name="Well" type="xmml:DrillholeType"
substitutionGroup="xmml:Drillhole"/>

<xsd:complexType name="DrillholeType">
  <xsd:annotation>
    <xsd:documentation>A drillhole is a Feature with a collarLocation, zero or
more  drillHoleShape's, zero or more drillHoleLog's, and zero or more related
features.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexContent>
    <xsd:extension base="gml:AbstractFeatureType">
      <xsd:sequence>
        <xsd:element ref="md:metadata" minOccurs="0"/>
        <xsd:element ref="xmml:collarLocation"/>
        <xsd:element ref="xmml:holeShape" minOccurs="0"
maxOccurs="unbounded"/>
        <xsd:choice minOccurs="0" maxOccurs="unbounded">
          <xsd:element ref="xmml:holeLog"/>
          <xsd:element ref="xmml:relatedFeature"/>
        </xsd:choice>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

В добавка към схемите, XMML се разпространява и с XSLT стилове, чрез които XMML документите може да се

преобразуват в различни текстови формати като SVG и X3D (Extensible 3D) графика, HTML и чист текст. По този начин е възможно тяхното директно показване във WEB чрез браузър.

IREDES

International Rock Excavation Data Exchange Standart (IREDES) е стандарт за обмяна на информация между изкопните машини и централен компютър. За първи път е представен на петата конференция ISMMA and Telemin през 1999, като в момента се поддържа официално от един от най-големите производители на изкопно оборудване Atlas Copco.

IREDES дефинира общ електронен език за комуникация между изкопните машини и централен компютър или друго оборудване. Той се базира на XML и използва DTD за описанието на данните.

Информацията обменяна с IREDES се структурира в „набор от данни“ (Data Sets), като един набор от данни представлява цялата информация предадена наведнъж. Всеки един Data set съдържа следната информация:

- Главна заглавна част (General Header Object – GenHead) – съдържа типа и идентификацията на машината. Този блок се предава винаги първи и съдържа основна информация като тип на машината, производителя й, сериен номер и др. Освен това може да съдържа информация и за версията на протокола с цел избягване на несъвместимости.
- Местоположение (Site Header Object) – съдържа информация за местоположението или мината, в която се намира машината. Това заглавно поле не е задължително.
- Приложни данни (Application Object) – може да съдържа основно три типа информация. Профила за изпълнение на производството (Production Performance Profile) съдържа цялата информация свързана с това доколко машината с завършила зададената й работа. Информацията в този блок се предава на определен интервал – ден, смяна и т.н. Профила за качеството на продукцията (Production Quality Profile) се предава отделно, тъй като обикновено в минните предприятия качеството и изпълнението на продукцията се следят от различни отдели. Плановата информация (Planing Data) се използва за предаване на цялата информация необходима на дадена машина, така че тя да може да извърши дадена операция.
- Завършващия блок (Trailer Block) съдържа единствено един параметър, който служи за проверка по четност на останалата информация.

В един набор от данни информацията е структурирана в блокове от данни (Data Blocks), като всеки блок може да има подблокове. Всеки блок също съдържа и елементи, които имат определен фиксиран тип. Тук се проявява и недостатъка на DTD, тъй като DTD не може да се опише тези данни (предвижда се преминаване към XML Shema на по-късен етап).

Анализа и подготовката на един IREDES документ може да стане по два начина:

- Подготовка и анализ с помощта на база данни с възможности за експорт и импорт на XML документи.
- Подготовка и анализ с помощта на външно приложение.

Първият начин изисква наличие на филтър за работа с XML на използваната в мината база данни. В зависимост от възможностите на този филтър изхода може да се нуждае и от допълнителна обработка, така че да стане съвместим с IREDES спецификацията. Освен това в този случай се препоръчва и валидация на DTD на документа от страна на оборудването към което той се изпраща с цел избягване на грешна интерпретация.

Втория начин използва външна програма за извличането на данните от базата данни и подготвянето им в подходящ IREDES документ.

Данните може да се предават основно по два начина – първия е без наличието на връзка между централния компютър и минното оборудване (offline) и втория е чрез директна връзка по изградената мрежа (online). Независимо от носителя предаваната информация е идентична. Въпреки това предаването на данни за моментно състояние на оборудването и съобщения за грешки не може да се осъществява по първия метод. От друга страна данни за планиране и работните дневници на машината се пренасят предимно чрез носители, въпреки че предаването им по мрежова инфраструктура не е изрично забранено.

IREDES не позволява отдалечено управление на машините, което обаче не означава че не може да се изпращат задачи към автономните машини.

Следващият код показва един примерен IREDES документ [6]:

```
<?xml version="1.0" encoding="UTF-8">
<!DOCTYPE IREDES SYSTEM "IREDES.dtd">
<IREDES>
  <GenHead>
    <StdVersion val="0.5"/>
    <EquProfType val="DRig"/>
    <EquManuf val="Someone"/>
    <SerialNo val="08/15"/>
  </GenHead>
  <ApplicationObject>
    <PPGen>
      <PPRepPerStart val="20000905 10:13"/>
      <PPRepPerEnd val="20000907 17:34"/>
      <PPGPwrSrc>
        <PPRun val="2445"/>
        <PPStopTime val="2342"/>
        <PPRepair val="23345"/>
        <PPPrevMaint val="243"/>
      </PPGPwrSrc>
      <PPDRig>
        <DrilledLenght unit="mm" val="3143"/>
        <NumHFAuto val="3"/>
        <NumHFErr val="4"/>
        <NumSAuto val="2"/>
        <NumSErr val="1"/>
        <NumHMan val="3"/>
        <RDOpTime val="554"/>
      </PPDRig>
    </PPGen>
  </ApplicationObject>
  <GenTrailer>
    <ChkSum val="FE5A"/>
  </GenTrailer>
</IREDES>
```

SVG

Scalable Vector Graphics (SVG) е XML базиран език за описание на двуизмерни графични данни във векторен формат, които са готови за публикуване в Интернет или други приложения. SVG използва няколко прости форми, при комбинирането на които се получават сложни фигури.

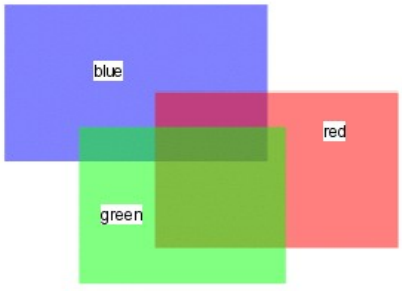
Тези форми са:

- Правоъгълник
- Кръг
- Елипса
- Линия
- Начупена линия
- Многоъгълник

Тези прости форми се отнасят към графичните обекти в един SVG файл. Освен графични обекти той може да съдържа още и изображения и текст. Тези обекти се рендерират на базата на наслагването, като първия елемент се рендерира пръв, втория върху него и т.н. Когато даден обект не е напълно непрозрачен, то смесването между него и обектите под него се изчислява на чрез математически алгоритми (например чрез алфа смесване – alpha blending) [8].

Едно SVG изображение има следния примерен вид:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.0//EN"
"http://www.w3.org/TR/2001/REC-SVG-20010904/DTD/svg10.dtd">
<svg xmlns="http://www.w3.org/2000/svg">
 <rect
        style="font-size:12.000;fill:#0000ff;fill-rule:evenodd;stroke-width:1;fill-
opacity:0.4870;"
   width="188.885529"
   height="121.426453"
   x="37.1025124"
   y="145.037048" />
 <rect
        style="font-size:12.000;fill:#ff0000;fill-rule:evenodd;stroke-width:1;fill-
opacity:0.5043;"
   width="175.393692"
   height="121.426453"
   x="145.037094"
   y="212.496155" />
 <rect
        style="font-size:12.000;fill:#00ff00;fill-rule:evenodd;stroke-width:1;fill-
opacity:0.4870;"
   width="148.410065"
   height="121.426453"
   x="91.0698013"
   y="239.479797" />
</svg>
```



Фиг. 4. SVG пример

Освен показване на изображения, чрез SVG може да се правят анимации и интерактивни изображения, които се променят в зависимост от действията на потребителя. Последното се постига благодарение на факта, че SVG използва DOM (Document Object Model) за достъп до отделните елементи. Това позволява интерактивното им управление чрез езици като Java и JavaScript например. Използването на JavaSript позволява не само контролиране на елементите в SVG изображението, но и контрол върху елементи на заобикалящия го HTML документ.

Тъй като SVG е прост текстов документ, то е възможно генерирането на SVG изображение директно от даден GML/XMML документ чрез проста XSLT трансформация. След това генерираното SVG изображение може да бъде включвано в HTML отчет и да бъде показано директно във WEB или да бъде предадено към определено графично приложение.

Предимствата на използването на SVG изображения вместо JPEG или PNG за изобразяването на сложни обекти (каквито са геоложките данни в рудниците) са:

- Увеличаване и намаляване на изображението без загуба на качеството, което е характерно за растерните изображения като JPEG и PNG. Това гарантира достигането на детайлност недостижима чрез растерен формат без допълнителни заявки към сървъра.
- По-малък размер на файла, особенно за компресираните SVG изображения.
- Лесна преносимост, което е характерно за XML документите.
- Възможност за интерактивност и анимации.

Тези предимства се отнасят и за Flash анимациите (www.macromedia.com), но Flash има един основен недостатък, който се избягва при SVG: той е затворен файлов формат собственост на една компания за разлика от SVG, който е отворен стандарт и се контролира от много организации и компании (включително и от World Wide Web Consortium).

Визуализирането на SVG изображенията може да стане както в специализиран софтуер, така и директно в WEB браузър. Adobe разпространява безплатно своя SVG плъгин, който позволява визуализиране в почти всички по известни браузъри. Освен това в последните версии на Mozilla визуализирането на SVG изображения е вградено.

Уеб услуги (Web Services)

Уеб услугите представляват набор от инструменти, които позволяват изграждането на разпределени приложения и споделяне на ресурси на базата на съществуващата WEB инфраструктура.

Идеята на Уеб услугите е да се разработят по такъв начин, че да може да работят еднакво добре на различни платформи независимо от разстоянието.

За да стане това възможно Уеб услугите използват общ език, а именно XML.

Основните стандарти на които се опират Уеб услугите са SOAP (Simple Object Application Protocol)[9] и WSDL (Web Services Description Language)[10]. Тук SOAP служи за пренос на съобщенията през Интернет по HTTP протокол [11], а WSDL описва технологичните особености на услугата – метод за връзка, входни и изходни данни и т.н.

След като бъде изградена, за да стане публично достъпна, една Уеб услуга, тя трябва да се опише в UDDI (Universal Description, Discovery Language) регистрите. Тези регистри съдържат информация за Уеб услугите, компаниите, които ги поддържат, описание на възможностите и изискванията на отделните услуги [12].

В минната промишленост Уеб услугите може да се използват както за комуникация между минни в рамките на едно предприятия и изпълнение на отдалечени процедури между тях, така и за предоставяне на услуги и информация през Уеб.

Заключение

Както бе отбелязано, използването на затворени бинарни формати за предаване на данни между различни устройства в повечето случаи е трудно и понякога свързано със загуба на данни. Въпреки предимствата на тези формати (бързина на обработката и намален обем на файла), то вероятността от грешна интерпретация на данните е сравнително голяма, което не ги прави предпочитани за предаване на разнородни данни между различни приложение, където е необходимо често преминаване от един формат към друг.

Използването на XML позволява много по-лесен трансфер на данни между различните приложения и оборудване, като освен това позволяват и използването на Уеб услуги за тези данни.

От една страна Уеб услугите ще позволят лесен трансфер на данни между различни мини, даже те да използват различна база данни и софтуер, като освен това осигуряват възможност за изпълнение на отдалечени процедури между тях.

От друга страна Уеб услугите осигуряват лесно изграждане на b2b (business-to-business) приложения и публикуването им в Интернет, което ще направи търсената информация по лесно достъпна за клиенти и персонал.

Литература

Кейгъл, К., Гибънс, Д., Хънтър, Д. 2001 *Програмиране с XML*. С. СофтПрес.

Shrestha, B. B., *XML Database Technology and its use for GML*, Enschede 18-28.

Cox, S., *Geologic data transfer using XML*, Nedlands 6-14

http://www.opengis.net/gml/

http://xmml.arrc.csiro.au/

Kauppi, I., *Use of XML Based Communication in Mining Industry (IREDES Initiative) and Mining Machines*. 2-8

http://www.w3c.de/Events/2002/svgtutorial.html

Chengyuan, P., *Scalable Vector Graphics (SVG), Helsinki*. 2-4, 6-7

http://www.w3.org/TR/soap/

http://www.w3.org/TR/wsdl

http://www.w3.org/Protocols/

Стойчев, П., *Уеб услуги*, BestOfWebDevMagazine. 54-58