POLSKIE
TOWARZYSTWO
INFORMACJI
PRZESTRZENNEJ

ROCZNIKI GEOMATYKI 2012

Modele danych przestrzennych w UML i ich transformacja do schematów GML i struktur baz danych

Tom X
Zeszyt 1(51)
Warszawa

POLSKIE TOWARZYSTWO INFORMACJI PRZESTRZENNEJ
ROCZNIKI GEOMATYKI 2012 ❍ Tom X ❍ Zeszyt 1(51)

Agnieszka Chojka

6. Budowa schematu aplikacyjnego GML – reguły budowy, narzędzia i przykłady

W rozdziale opisano reguły budowy schematu aplikacyjnego GML, w tym reguły przekształcania schematu aplikacyjnego UML na schemat aplikacyjny GML. Umieszczono w nim również przykłady przekształcania poszczególnych elementów UML na stosowne konstrukcje w GML oraz przykłady dostępnych narzędzi programowych wspomagające takie odwzorowania.

6.1. Reguły budowy schematów aplikacyjnych GML

Norma ISO 19136

Norma ISO 19136 to specyfikacja implementacyjna języka GML (ang. *Geography Markup Language*) – języka znaczników geograficznych, który jest otwartym formatem wymiany danych przestrzennych pomiędzy różnymi systemami geoinformacyjnymi. GML jest aplikacją XML, określa regułę kodowania dla schematów aplikacyjnych zgodnych z normami ISO serii 19100, opartą na XML zgodnie z ISO 19118. Definiuje sposób zapisu w XML Schema określonych właściwości przestrzennych i nieprzestrzennych (zdefiniowanych w normach ISO serii 19100) obiektów geograficznych, np. jednostki miary, geometria i topologia, systemy odniesienia.

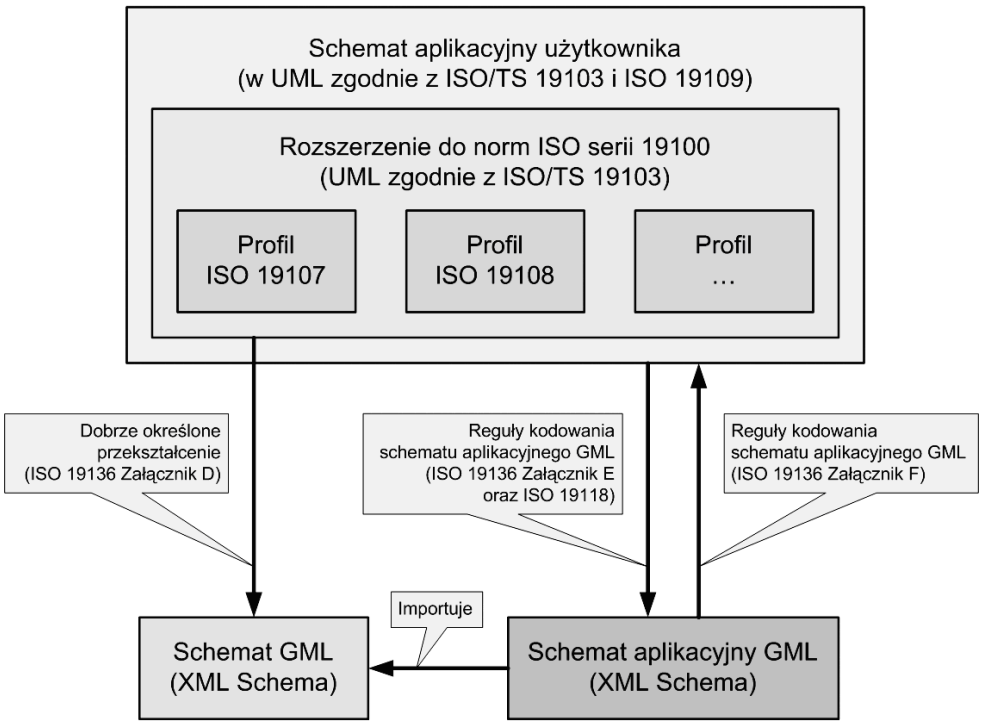
Norma ISO 19136 określa zasady przekształcania schematów aplikacyjnych zapisanych w UML zgodnie z normą ISO 19109 na schematy aplikacyjne GML zapisane w XML Schema.

Schemat GML i schemat aplikacyjny GML

Schemat GML składa się z komponentów w przestrzeni nazw XML, której pełna nazwa podawana jest w postaci adresu URL (szczególny przypadek URI, czyli jednoznacznego identyfikatora zasobu w sieci Internet): „http://www.opengis.net/gml/3.2". Komponenty te przeznaczone są do zapisu określonych właściwości przestrzennych i nieprzestrzennych obiektów geograficznych.

Schemat aplikacyjny GML jest schematem aplikacyjnym zapisanym w języku XML Schema zgodnie z regułami określonymi w normie ISO 19136.

Budowa schematu aplikacyjnego GML dla określonej dziedziny zastosowań może wymagać rozszerzenia lub ograniczenia typów zdefiniowanych przez schemat GML. Schemat aplikacyjny GML musi być określony w XML Schema i musi importować schematy GML (rys. 6.1).



Rys. 6.1. Zależności między normą ISO 19136 a innymi normami ISO z serii 19100 oraz między schematem aplikacyjnym GML a schematem GML (ISO/TC 211, 2007a lub Portele, 2007).

Schemat aplikacyjny GML może zostać skonstruowany na dwa sposoby przez:

- zastosowanie reguł dla schematów aplikacyjnych GML określonych w normie ISO 19136 dla tworzenia schematów aplikacyjnych bezpośrednio w XML Schema;
- opracowanie schematu aplikacyjnego w UML z zastosowaniem reguł określonych w normie ISO 19109, a następnie przekształcenie go na odpowiadający mu schemat aplikacyjny w GML, zapisany w XML Schema zgodnie z regułami kodowania określonymi w normach ISO 19136 i ISO 19118.

Reguły przekształcenia modelu UML do schematu GML

Przekształcenie schematu aplikacyjnego UML, zgodnego z normą ISO 19109, na odpowiadający mu schemat aplikacyjny GML, oparte jest na zbiorze reguł kodowania określonych w normie ISO 19136. Zasady te podano w Załączniku E (ISO/TC 211, 2007a lub Portele, 2007) i oparto na ogólnym założeniu, że definicja klasy w schemacie aplikacyjnym UML jest przekształcana na deklarację typu i elementu w schemacie GML (XML Schema) według zależności podanych w poniższej tabeli (tab. 6.1).

Dodatkowo dla różnych elementów modelu UML można określić metki (tab. 6.2), które pozwalają kontrolować przekształcanie schematu aplikacyjnego UML na XML Schema (schemat aplikacyjny GML), szczególnie podczas stosowania narzędzi umożliwiających automatyczne generowanie plików XSD, np. systemy przekształcania modeli *ShapeChange* lub *FullMoon*.

Tabela 6.1. Reguły kodowania UML do GML (ISO/TC 211, 2007a lub Portele, 2007)

Schemat aplikacyjny UML	Schemat aplikacyjny GML
Pakiet	Jeden dokument XML Schema na pakiet (przekształcenie domyślne)
>>application Schema<<	Dokument XML Schema
<<dataType>>	Element globalny, którego modelem zawartości jest element complexTyp w XML Schema o zakresie globalnym, typ właściwości
<<enumeration>>	Ograniczenie xsd:string z wartościami wyliczenia
<<codeList>>	Unia wyliczenia i wzorca (przekształcenie domyślne, przekształceniem alternatywnym jest odwołanie do słownika)
<<union>>	Grupa wyboru, której składnikami są obiekty GML lub obiekty odpowiadające DataType
<<featureType>>	Element globalny, którego modelem zawartości jest typ XML Schema o zakresie globalnym, pochodzący z bezpośredniego/pośredniego rozszerzenia gml:AbstractFeatureType, typ właściwości
Brak stereotypu lub <<type>>	Element globalny, którego modelem zawartości jest typ XML Schema o zakresie globalnym, pochodzący z bezpośredniego/pośredniego rozszerzenia gml:AbstractGMLType, typ właściwości
Operacje	Nie kodowane
Atrybut	Lokalny xsd:element, typ jest również typem właściwości (jeśli typ jest typem złożonym) lub typem prostym
Rola powiązania (asocjacji)	Lokalny xsd:element, typ jest zawsze typem właściwości (tylko role nazwane i nawigowalne)
Ograniczenia OCL	Nie kodowane** Jednak w XML Schema istnieje możliwość zapisania ograniczenia za pomocą elementu restriction

Tabela 6.2. Podstawowe metki *(Tagged Values)* stosowane dla poszczególnych elementów modelu UML (ISO/TC 211, 2007a lub Portele, 2007)

Element modelu UML	Stosowana metka
Pakiet	– documentation – xsdDocument – targetNamespace (tylko dla <<aplication Schema>> – xmlns (tylko dla <<application Schema>>) – version (tylko dla <<application Schema>>) – gmlProfileSchema (tylko dla <<application Schema>>)
Klasa	– documentation – noPropertyType – byValuePropertyType – isCollection – asDictionary (tylko dla <<codeList>>) – xmlSchemaType (tylko dla <<type>>)
Atrybut lub zakończenie powiązania	– documentation – sequenceNumber – inlineOrByReference – isMetadata

Element Pakiet

Pakiet w UML przekształcany jest na jeden dokument XML Schema z metką `xsdDocument` określającą nazwę pliku XSD. Jeśli metka `xsdDocument` została ustawiona dla pakietu, wtedy dokument schematu zawiera wszystkie składniki XML Schema wynikające z klas UML zawartych bezpośrednio w pakiecie. W przeciwnym wypadku, wszystkie składniki schematu deklarowane są w dokumencie schematu pakietu, w którym zawarty jest ten pakiet. Ustawienie metki `xsdDocument` jest obowiązkowe dla wszystkich pakietów ze stereotypem «`applicationSchema`».

Dla każdego dokumentu schematu należy ustawić wartości atrybutów `targetNamespace` i `version` elementu głównego za pomocą odpowiednich metek w pakiecie reprezentującym schemat aplikacyjny UML. Ponadto powinien zostać określony atrybut `xmlns` dla docelowej przestrzeni nazw (metka `xmlns`).

Należy również określić wzajemne zależności między poszczególnymi pakietami, aby ustalić, czy wymagane będzie zaimportowanie (element `import`), czy włączenie (element `include`) schematów aplikacyjnych z innych pakietów. Element `import` pozwala na dodanie do dokumentu XML Schema schematów z różnymi docelowymi przestrzeniami nazw, a element `include` pozwala na dodanie schematów z tą samą docelową przestrzenią nazw.

Element klasa

Na schemat aplikacyjny GML przekształcane są jedynie klasy UML bez stereotypu lub ze stereotypem «`featureType`», «`type`», «`dataType`», «`union`», «`codeList`» i «`enumeration`». Wszystkie klasy UML powinny mieć „zero lub jeden nadtyp", to znaczy że klasa może dziedziczyć jedynie od jednej klasy. Każda klasa UML przekształcana jest na typ nazwany i otrzymuje przyrostek `Type`.

Typy danych zdefiniowane przez normy ISO serii 19100 przekształcane są na odpowiadające im typy danych w XML Schema (tab. 6.3).

W tabeli 6.3 klasa UML to klasa pochodząca z norm ISO serii 19100, zaimplementowana w schemacie GML. Typ GML określa typ obiektu zapisanego w XML Schema, a typ składnika (właściwości) GML jest typem XML Schema używanym jako wartość określonej właściwości w schemacie aplikacyjnym GML (np. atrybut, rola).

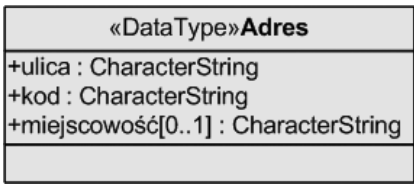
Tabela 6.3. Przykłady implementacji typów danych z norm ISO serii 19100 w schemacie aplikacyjnym GML (ISO/TC 211, 2007a lub Portele, 2007)

Klasa UML	Element obiektu GML	Typ GML	Typ składnika (właściwości) GML
GM_Object	gml:AbstractGeometry	gml:AbstractGeometryType	gml:GeometryPropertyType
GM_Point	gml:Point	gml:PointType	gml:PointPropertyType
TP_Edge	gml:Edge	gml:EdgeType	gml:DirectedEdgePropertyType
SC_CRS	gml:AbstractCRS	gml:AbstractCRSType	gml:CRSPropertyType
CharacterString	–	–	xsd:string
Integer	–	–	xsd:integer, xsd:nonPositiveInteger, xsd:negativeInteger, xsd:nonNegativeInteger, xsd:positiveInteger
Length, Distance	–	–	gml:LengthType

Klasa ze stereotypem «dataType»

- Klasa ze stereotypem «dataType» jest przekształcana na typ złożony (element complexType) w XML Schema.
- Jeśli klasa nie ma nadtypu (tzn. nie jest utworzona poprzez dziedziczenie od innej klasy), nie jest typem pochodnym w XML Schema (nie stanowi rozszerzenia extension żadnego elementu XML Schema). W przeciwnym przypadku stanowi rozszerzenie swojego nadtypu, który nie może być typem pochodnym od gml:AbstractGMLType, tzn. nie może bezpośrednio lub pośrednio dziedziczyć z gml:AbstractGMLType (stanowić rozszerzenia tego elementu w XML Schema). Nadtypy abstrakcyjne bez atrybutów i nawigacyjnych ról asocjacji są pomijane.
- Dla klas ze stereotypem «dataType» należy zdefiniować elementy globalne XML z odpowiednimi ustawieniami dla nazwy (nazwa klasy UML), typu (nazwa klasy z przyrostkiem Type), abstrakcyjności (jeśli klasa jest abstrakcyjna) i grupą zastępowania (nazwa określająca element nadtypu lub gml:AbstractObject, jeśli klasa nie ma nadtypu).
- Należy utworzyć nazwany typ złożony, którego nazwa zawiera nazwę klasy UML z przyrostkiem PropertyType, jeśli dla klasy nie ustawiono metki noPropertyType z wartością true.

Przykład (rys. 6.2) przekształcenia klasy *Adres* na odpowiednią strukturę w GML:

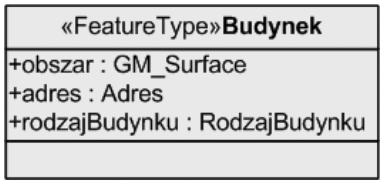


Rys. 6.2. Przykład klasy ze stereotypem «dataType».

```
<complexType name="AdresType">
  <sequence>
    <element name="ulica" type="string"/>
    <element name="kod" type="string"/>
    <element name="miejscowość" type="string" minOccurs="0"/>
  </sequence>
</complexType>

<element name="Adres" type="np:AdresType" substitutionGroup="gml:AbstractObjec

<complexType name="AdresPropertyType">
  <sequence>
    <element ref="np:Adres"/>
  </sequence>
  <attributeGroup ref="gml:OwnershipAttributeGroup"/>
</complexType>
```

Klasa ze stereotypem «`featureType`»

- Klasy ze stereotypem «`featureType`» pochodzą bezpośrednio lub pośrednio od `gml:AbstractFeatureType`. Jeśli klasa nie ma nadtypu to stanowi bezpośrednie rozszerzenie `gml:AbstractFeatureType`, w przeciwnym razie stanowi rozszerzenie nadtypu, który powinien pochodzić od `gml:AbstractFeatureType` (bezpośrednio lub pośrednio).
- Dla klas ze stereotypem «`featureType`» należy zdefiniować elementy globalne XML z odpowiednimi ustawieniami dla nazwy (nazwa klasy UML), typu (nazwa klasy z przyrostkiem `Type`), abstrakcyjności (jeśli klasa jest abstrakcyjna) i grupą zastępowania (nazwa nadtypu lub `gml:AbstractFeature`).
- Jeśli klasa ma pojedyncze powiązanie, które jest agregacją zwykłą lub całkowitą (kompozycją), rola powiązania jest przekształcana na element właściwości, a klasa przenosi metkę `isCollection` z wartością `true` oraz do typu złożonego dodawana jest grupa atrybutów `gml:AggregationAttributeGroup`.
- Dla klasy należy utworzyć nazwany typ złożony, którego nazwa zawiera nazwę klasy UML z przyrostkiem `PropertyType`, jeśli dla klasy nie ustawiono metki `noPropertyType` z wartością `true`.
- Dla klasy należy utworzyć nazwany typ złożony, którego nazwa zawiera nazwę klasy UML z przyrostkiem `PropertyByValueType`, jeśli dla klasy ustawiono metkę `byValuePropertyType` z wartością `true`.



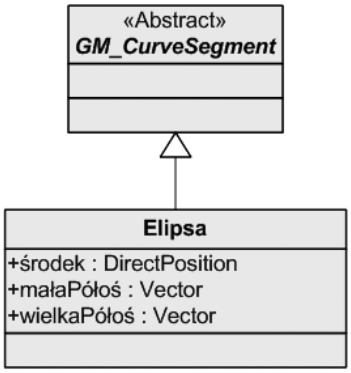
Rys. 6.3. Przykład klasy ze stereotypem «`featureType`».

Przykład (rys. 6.3) przekształcenia klasy *Budynek* na odpowiednią strukturę w GML:

```
<complexType name="BudynekType">
 <complexContent>
  <extension base="gml:AbstractFeatureType">
   <sequence>
    <element name="obszar" type="gml:SurfacePropertyType"/>
    <element name="adres" type="np:AdresPropertyType"/>
    <element name="rodzajBudynku" type="np:RodzajBudynkuType"/>
   </sequence>
  </extension>
 </complexContent>
</complexType>

<complexType name="BudynekPropertyType">
 <sequence minOccurs="0">
  <element ref="np:Budynek"/>
 </sequence>
 <attributeGroup ref="gml:AssociationAttributeGroup"/>
 <attributeGroup ref="gml:OwnershipAttributeGroup" />
</complexType>

<complexType name="BudynekPropertyByValueType">
 <sequence>
  <element ref="np:Budynek"/>
 </sequence>
 <attributeGroup ref="gml:OwnershipAttributeGroup" />
</complexType>

<element name="Budynek" type="np:BudynekType" substitutionGroup="gml:AbstractFeature"/>
```

Klasa ze stereotypem «type» lub bez stereotypu

- Klasa bez stereotypu lub ze stereotypem «type» pochodzi bezpośrednio lub pośrednio z gml:AbstractGMLType. Jeśli klasa nie ma nadtypu stanowi bezpośrednie rozszerzenie gml:AbstractGMLType, w przeciwnym razie stanowi rozszerzenie nadtypu, który powinien pochodzić od gml:AbstractGMLType (bezpośrednio lub pośrednio), ale nie z gml:AbstractFeatureType (bezpośrednio lub pośrednio).
- Dla klas ze stereotypem «type» należy zdefiniować elementy globalne XML z odpowiednimi ustawieniami dla nazwy (nazwa klasy UML), typu (nazwa klasy z przyrostkiem Type), abstrakcyjności (jeśli klasa jest abstrakcyjna) i grupą zastępowania (nazwa nadtypu lub AbstractGML).
- Jeśli klasa ma pojedyncze powiązanie, które jest agregacją zwykłą lub całkowitą, rola powiązania jest przekształcana na element właściwości, a klasa przenosi metkę isCollection z wartością true oraz do typu złożonego dodawana jest grupa atrybutów gml:AggregationAttributeGroup.
- Dla klasy należy utworzyć nazwany typ złożony, którego nazwa zawiera nazwę klasy UML z przyrostkiem PropertyType, jeśli dla klasy nie ustawiono metki noPropertyType z wartością true.
- Dla klasy należy utworzyć nazwany typ złożony, którego nazwa zawiera nazwę klasy UML z przyrostkiem PropertyByValueType, jeśli dla klasy ustawiono metkę byValuePropertyType z wartością true.



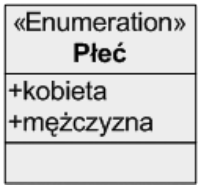
Rys. 6.4. Przykład klasy bez stereotypu.

Przykład (rys. 6.4) przekształcenia klasy Elipsa na odpowiednią strukturę w GML. Klasa abstrakcyjna GM_CurveSegment pochodzi z normy ISO 19107 i reprezentuje prosty element geometryczny – segment krzywej:

```
<element name="Elipsa" type="np:ElipsaType" substitutionGroup="gml:AbstractCurveSegment"/>
<complexType name="ElipsaType">
 <complexContent>
  <extension base="gml:AbstractCurveSegmentType">
   <sequence>
    <element name="środek" type="gml:DirectPositionType"/>
    <element name="małaPółoś" type="gml:VectorType"/>
    <element name="wielkaPółoś" type="gml:VectorType"/>
   </sequence>
  </extension>
 </complexContent>
</complexType>
```

Klasa ze stereotypem «enumeration»

Klasa ze stereotypem «enumeration» jest przekształcana na typ prosty (simpleType) w XML Schema. Typem podstawowym jest string, a dziedzina wartości została ograniczona do zbioru wartości określonych przez nazwy atrybutów klasy UML.

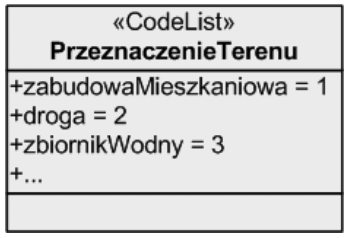


Rys. 6.5. Przykład klasy ze stereotypem «enumeration».

Przykład (rys. 6.5) przekształcenia klasy *Płeć* na odpowiednią strukturę w GML:

```
<simpleType name="Płeć">
 <restriction base="string">
  <enumeration value="kobieta"/>
  <enumeration value="mężczyzna"/>
 </restriction>
</simpleType>
```

Klasa ze stereotypem «codeList»



Rys. 6.6. Przykład klasy ze stereotypem «codeList».

Klasa ze stereotypem «codeList» i bez ustawionej metki asDictionary na true powinna zostać przekształcona tak jak klasa ze stereotypem «enumeration», ale z tą różnicą, że:

- należy dodać wzorzec <pattern value='other: \w{2,}'/>, który dopuszcza inne wartości tekstowe poza zdefiniowanymi (wartości te są poprzedzone przedrostkiem other);
- jeśli określony jest kod dla wartości listy kodowej, tylko kod powinien być reprezentowany jako wzorzec wyliczenia;
- wartość kodu powinna być określona za pomocą elementów annotation oraz appinfo z elementem gml:description określającym wartość tekstową wartości wyliczanej.

Przykład (rys. 6.6) przekształcenia klasy *PrzeznaczenieTerenu* na odpowiednią strukturę w GML:

```
<simpleType name="PrzeznaczenieTerenuType">
 <union memberTypes="np:PrzeznaczenieTerenuEnumerationType np:PrzeznaczenieTerenuOther1
</simpleType>

<simpleType name="PrzeznaczenieTerenuEnumerationType">
 <restriction base="string">
  <enumeration value="1">
   <annotation>
    <appinfo><gml:description>zabudowaMieszkaniowa</gml:description></appinfo>
   </annotation>
  </enumeration>
  <enumeration value="2">
   <annotation>
    <appinfo><gml:description>droga</gml:description></appinfo>
   </annotation>
  </enumeration>
  <enumeration value="3">
   <annotation>
    <appinfo><gml:description>zbiornikWodny</gml:description></appinfo>
   </annotation>
  </enumeration>
 </restriction>
</simpleType>

<simpleType name="PrzeznaczenieTerenuOtherType">
 <restriction base="string">
  <pattern value="other: \w{2,}"/>
 </restriction>
</simpleType>
```

Klasa ze stereotypem «union»

Klasa ze stereotypem «union» jest przekształcana na typ złożony (complexType) w XML Schema, podobnie jak klasa ze stereotypem «dataType», przy czym zamiast elementu sequence pojawia się element choice, który oznacza, że tylko jedna z właściwości może pojawić się w instancji unii.

Przykład przekształcenia klasy ze stereotypem «union» na odpowiednią strukturę w GML:

```
<complexType name="ArtystaType">
  <choice>
    <element name="nazwisko" type="string"/>
    <element name="pseudonim" type="string"/>
  </choice>
</complexType>
```

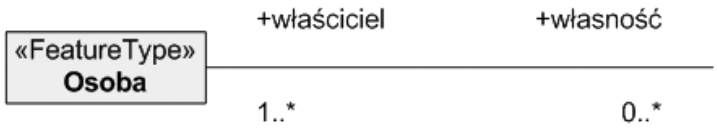
Atrybuty i nazwy ról

Atrybut lub rola powiązania obiektu w UML są przekształcane na elementy lokalne z tą samą nazwą typu złożonego (complexType), który definiuje model zawartości typu obiektu. Ograniczenia atrybutów minOccurs i maxOccurs są ustawiane zgodnie z definicjami ich krotności (liczności) w modelu UML. Typ zależy od typu wartości właściwości w UML. Jeżeli typ wartości właściwości jest:

- zawartością prostą, wtedy taki typ stosowany jest bezpośrednio (np. integer);
- zawartością złożoną, wtedy zostanie użyta właściwość. Domyślna transformacja (bez ustawiania metek) właściwości typu pozwala zarówno na reprezentację wbudowaną (inline), jak i przez referencję (by-reference) dla typów obiektów oraz reprezentację wbudowaną (inline) dla typów danych i unii. Dla typów obiektów reprezentacja może zostać ograniczona do wbudowanej lub przez referencję przez zastosowanie metki inlineOrByReference odpowiednio z wartością inline lub byReference . Należy zastosować transformację domyślną jeśli metka nie zostanie ustawiona lub zostanie ustawiona na inlineOrByReference.

Jeśli typ właściwości został już określony w schemacie aplikacyjnym, jako typ nazwany (należy sprawdzić metki noPropertyType i byValuePropertyType), należy odwołać się do tego składnika schematu, w przeciwnym razie lokalnie zostanie zdefiniowany anonimowy typ właściwości w elemencie właściwości.

Jeżeli kodowana właściwość jest końcem powiązania i drugi koniec powiązania jest również kodowany na schemat aplikacyjny GML, nazwa właściwości drugiego końca powiązania powinna zostać przekształcona na element gml:reversePropertyName w elementach annotation i appinfo elementu właściwości.



Rys. 6.7. Przykład ról w powiązaniu.

Przykład (rys. 6.7) przekształcenia właściwości typu na reprezentację wbudowaną `(inline)` i przez referencję `(by-reference)` w GML:

```
<element name="właściciel" type="np:OsobaPropertyType" maxOccurs="unbounded">
 <annotation>
  <appinfo>
   <gml:reversePropertyName>np:własność</gml:reversePropertyName>
  </appinfo>
 </annotation>
</element>
...
<complexType name="OsobaPropertyType">
 <sequence minOccurs="0">
  <element ref="np:Osoba"/>
 </sequence>
 <attributeGroup ref="gml:AssociationAttributeGroup"/>
 <attributeGroup ref="gml:OwnershipAttributeGroup" />
</complexType>
```

lub:

```
<element name="właściciel" maxOccurs="unbounded">
 <annotation>
  <appinfo>
   <gml:reversePropertyName>np:własność</gml:reversePropertyName>
  </appinfo>
 </annotation>
 <complexType>
  <sequence minOccurs="0">
   <element ref="np:Osoba"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
 </complexType>
</element>
```

Alternatywnie typ właściwości może posiadać tylko jedną z reprezentacji, wbudowaną lub przez referencję, w zależności od wartości metki `inlineOrByReference`.

Przykład przekształcenia właściwości typu tylko na reprezentację wbudowaną `(inline)` w GML:

```
<element name="właściciel" type="np:OsobaPropertyByValueType" maxOccurs="unbounded"/>
...
<complexType name="OsobaPropertyByValueType">
 <sequence>
  <element ref="np:Osoba"/>
 </sequence>
</complexType>
```

lub:

```
<element name="właściciel" maxOccurs="unbounded">
 <complexType>
  <sequence>
   <element ref="np:Osoba"/>
  </sequence>
 </complexType>
</element>
```

Jeśli dopuszczalna jest tylko reprezentacja przez referencję, wtedy element właściwości należy ograniczyć za pomocą znaczników `annotation` i `appinfo` elementem `gml:targetElement`, który określa nazwę elementu typu docelowego:

```
<element name="elementDocelowy" type="string"/>
```

Jeśli kodowana właściwość jest końcem powiązania i drugi koniec powiązania jest również kodowany na schemat aplikacyjny GML, wtedy nazwę właściwości drugiego końca powiązania należy zapisać za pomocą znaczników `annotation` i `appinfo` oraz elementu `gml:reversePropertyName`.

Przykład przekształcenia właściwości typu tylko na reprezentację przez referencję `(by-reference)` w GML:

```
<element name="właściciel" type="gml:ReferenceType" maxOccurs="unbounded">
  <annotation>
    <appinfo>
      <gml:targetElement>np:Osoba</gml:targetElement>
      <gml:reversePropertyName>np:własność</gml:reversePropertyName>
    </appinfo>
  </annotation>
</element>
```

Dziedziczenie

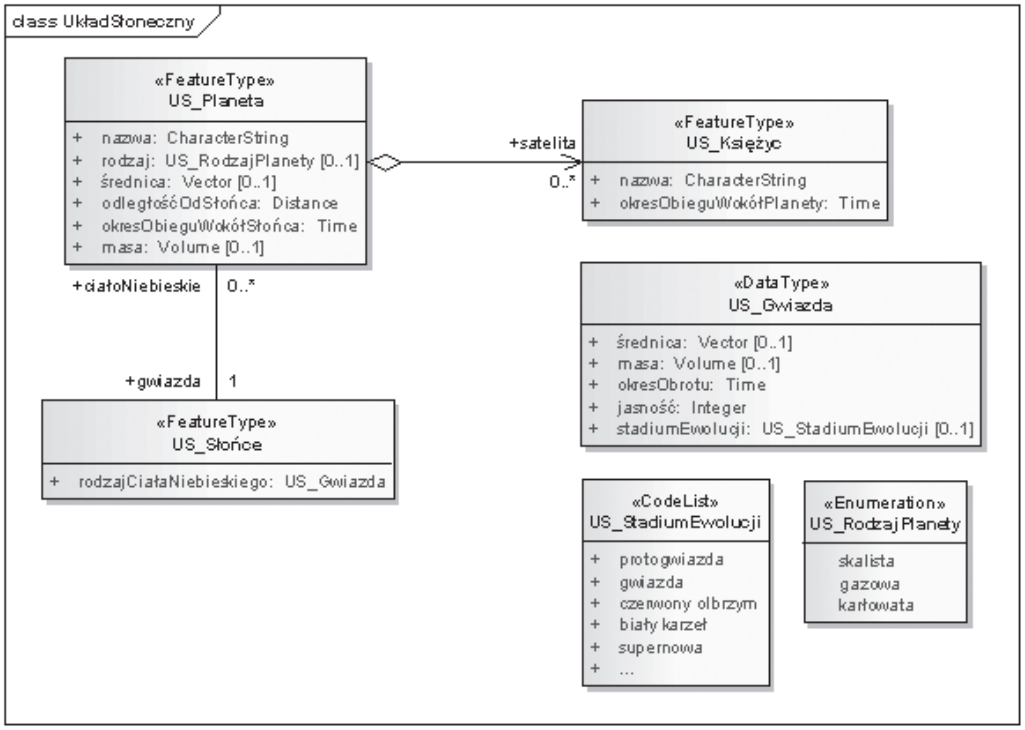
Do XML Schema może być transformowane tylko dziedziczenie pojedyncze, które jest realizowane poprzez mechanizm rozszerzenia `(extension)` lub ograniczenia `(restriction)` XML Schema oraz wykorzystanie elementu zastępowania `(substitutionGroup)`.

Przykład (rys. 6.4, str. 61) przekształcenia klasy `Elipsa`, która dziedziczy z klasy abstrakcyjnej `GM_CurveSegment`, na odpowiednią strukturę w GML:

```
<element name="Elipsa" type="np:ElipsaType" substitutionGroup="gml:AbstractCurveSegment"/>
<complexType name="ElipsaType">
  <complexContent>
    <extension base="gml:AbstractCurveSegmentType">
      <sequence>
        <element name="środek" type="gml:DirectPositionType"/>
        <element name="małaPółoś" type="gml:VectorType"/>
        <element name="wielkaPółoś" type="gml:VectorType"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

6.2. Przykład przekształcenia schematu aplikacyjnego UML na GML

W podrozdziale tym przedstawiono przykład przekształcenia schematu aplikacyjnego UML, zapisanego w postaci prostego diagramu klas, na odpowiadający mu schemat aplikacyjny GML.



Rys. 6.8. Przykład schematu aplikacyjnego w UML

Przykład (rys. 6.8) przekształcenia schematu aplikacyjnego w UML na odpowiadający mu schemat aplikacyjny w GML:

```
<?xml version="1.0" encoding="ISO-8859-2"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" xmlns:us="http://www.astronomia.pl/uklad_sloneczny"
xmlns:gml="http://www.opengis.net/gml/3.2" targetNamespace="http://www.astronomia.pl/uklad_sloneczny"
elementFormDefault="qualified" version="1.0">
  <!-- ================================== -->
  <import namespace="http://www.opengis.net/gml/3.2"
schemaLocation="http://schemas.opengis.net/gml/3.2.1/gml.xsd"/>
  <!-- ================================== -->
  <element name="US_Planeta" substitutionGroup="gml:AbstractFeature">
    <complexType>
      <complexContent>
        <extension base="gml:AbstractFeatureType">
          <sequence>
            <element name="nazwa" type="string"/>
            <element name="rodzaj" type="us:US_RodzajPlanetyType" minOccurs="0"/>
```

```
          <element name="średnica" type="gml:VectorType" minOccurs="0" maxOccurs="unbounded"/>
          <element name="odległośćOdSłońca" type="gml:LengthType"/>
          <element name="okresObieguWokółSłońca" type="gml:TimeType"/>
          <element name="masa" type="gml:VolumeType" minOccurs="0"/>
          <element name="gwiazda" type="us:US_SłońcePropertyType">
            <annotation>
              <appinfo>
                <gml:reverseProperty>us:ciałoNiebieskie</gml:reverseProperty>
              </appinfo>
            </annotation>
          </element>
          <element name="satelita" type="us:US_KsiężycPropertyType" minOccurs="0"
maxOccurs="unbounded"/>
        </sequence>
      </extension>
    </complexContent>
  </complexType>
 </element>
 <complexType name="US_PlanetaPropertyType">
  <sequence minOccurs="0">
   <element ref="us:US_Planeta"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
  <attributeGroup ref="gml:OwnershipAttributeGroup"/>
 </complexType>
 <!-- ================================== -->
  <element name="US_Księżyc" substitutionGroup="gml:AbstractFeature">
  <complexType>
   <complexContent>
    <extension base="gml:AbstractFeatureType">
     <sequence>
      <element name="nazwa" type="string"/>
      <element name="okresObieguWokółPlanety" type="gml:TimeType"/>
     </sequence>
    </extension>
   </complexContent>
  </complexType>
 </element>
 <complexType name="US_KsiężycPropertyType">
  <sequence minOccurs="0">
   <element ref="us:US_Księżyc"/>
  </sequence>
  <attributeGroup ref="gml:AssociationAttributeGroup"/>
  <attributeGroup ref="gml:OwnershipAttributeGroup"/>
 </complexType>
 <!-- ================================== -->
 <element name="US_Słońce" substitutionGroup="gml:AbstractFeature">
  <complexType>
   <complexContent>
    <extension base="gml:AbstractFeatureType">
     <sequence>
      <element name="rodzajCiałaNiebieskiego">
       <complexType>
        <sequence>
         <element name="US_Gwiazda" type="us:US_GwiazdaPropertyType"/>
        </sequence>
       </complexType>
      </element>
      <element name="ciałoNiebieskie" type="us:US_PlanetaPropertyType" minOccurs="0"
maxOccurs="unbounded">
       <annotation>
        <appinfo>
```

```
            <gml:reverseProperty>us:gwiazda</gml:reverseProperty>
           </appinfo>
          </annotation>
         </element>
        </sequence>
       </extension>
     </complexContent>
   </complexType>
  </element>
  <complexType name="US_SłońcePropertyType">
   <sequence minOccurs="0">
    <element ref="us:US_Słońce"/>
   </sequence>
   <attributeGroup ref="gml:AssociationAttributeGroup"/>
   <attributeGroup ref="gml:OwnershipAttributeGroup"/>
  </complexType>
  <!-- ================================== -->
  <element name="US_Gwiazda" type="us:US_GwiazdaType"/>
  <complexType name="US_GwiazdaType">
   <sequence>
    <element name="średnica" type="gml:VectorType" minOccurs="0"/>
    <element name="masa" type="gml:VolumeType" minOccurs="0"/>
    <element name="okresObiegu" type="gml:TimeType"/>
    <element name="jasność" type="integer"/>
    <element name="stadiumEwolucji" type="us:US_StadiumEwolucjiType" default="gwiazda" minOccurs="0"/>
   </sequence>
  </complexType>
  <complexType name="US_GwiazdaPropertyType">
   <sequence minOccurs="0">
    <element ref="us:US_Gwiazda"/>
   </sequence>
   <attributeGroup ref="gml:OwnershipAttributeGroup"/>
  </complexType>
  <!-- ================================== -->
  <simpleType name="US_RodzajPlanetyType">
   <restriction base="string">
    <enumeration value="skalista"/>
    <enumeration value="gazowa"/>
    <enumeration value="karłowata"/>
   </restriction>
  </simpleType>
  <!-- ================================== -->
  <simpleType name="US_StadiumEwolucjiType">
   <union memberTypes="us:US_StadiumEwolucjiEnumerationType us:US_StadiumEwolucjiOtherType"/>
  </simpleType>
  <simpleType name="US_StadiumEwolucjiEnumerationType">
   <restriction base="string">
    <enumeration value="protogwiazda"/>
    <enumeration value="gwiazda"/>
    <enumeration value="czerwony olbrzym"/>
    <enumeration value="biały karzeł"/>
    <enumeration value="supernowa"/>
    <enumeration value="..."/>
   </restriction>
  </simpleType>
  <simpleType name="US_StadiumEwolucjiOtherType">
   <restriction base="string">
    <pattern value="other: \w{2,}"/>
   </restriction>
  </simpleType>
</schema>
```

Problematyka niniejszej monografii stanowi przedmiot szerokiego zainteresowania środowisk współtworzących i współużytkujących infrastrukturę informacji przestrzennej budowaną w Polsce zgodnie z przepisami krajowymi i unijnymi. Zainteresowanie to znalazło swój wyraz w warsztatach „Modele danych przestrzennych w UML i ich transformacja do schematów GML i struktur baz danych", które odbyły się w ramach konferencji Polskiego Towarzystwa Informacji Przestrzennej na temat „Informacja przestrzenna dla Polski i Europy", Warszawa, 7–9 listopada 2011 roku. Odpowiadając na ujawnione wówczas zapotrzebowanie, zespół wykładowców podjął trud zawarcia zaprezentowanych przez siebie treści w opracowaniu o charakterze monograficznym. W rezultacie powstała publikacja, która przedstawia w sposób uporządkowany bogaty zasób wiadomości określonych tytułem warsztatów i dotyczących wybranych metod i technologii geoprzestrzennych.

Godne uznania jest, że zespół autorski w składzie: dr inż. A. Chojka, dr inż. A. Zwirowicz-Rutkowska, dr inż. Z. Parzyński i dr hab. J. Michalak, pełniący rolę redaktora naukowego, zrealizował podjęte przedsięwzięcie w stosunkowo krótkim terminie z niewątpliwą korzyścią dla potencjalnych Czytelników.

Jerzy Gaździcki

Warszawa, maj 2012 r.

Autorzy

dr hab. Janusz Michalak
Wydział Geologii, Uniwersytet Warszawski
J.Michalak@uw.edu.pl
Redakcja naukowa i rozdziały:
1. Wstęp
2. Różnice pomiędzy językiem zapisu danych i jego dziedzinową aplikacją
9. Najczęściej popełniane błędy w modelach UML dla schematów aplikacyjnych GML
11. Schematy aplikacyjne tematów aneksów II i III Dyrektywy INSPIRE
12. Podsumowanie

Słownik podstawowych terminów stosowanych w tekście

dr inż. Agnieszka Chojka
Wydział Geodezji i Gospodarki Przestrzennej, Uniwersytet Warmińsko-Mazurski
agnieszka.chojka@uwm.edu.pl
Rozdziały:
3. Wprowadzenie do modelowania informacji przestrzennej – metodyka MDA i diagramy klas UML
6. Budowa schematu aplikacyjnego GML – reguły budowy, narzędzia i przykłady
7. Transformacja schematu aplikacyjnego UML do schematu aplikacyjnego GML – wymagania, ograniczenia i wybrane narzędzia

dr inż. Agnieszka Zwirowicz-Rutkowska
Wydział Geodezji i Gospodarki Przestrzennej, Uniwersytet Warmińsko-Mazurski
agnieszka.zwirowicz@uwm.edu.pl
Rozdziały:
4. Przegląd standardów i narzędzi wykorzystywanych do modelowania informacji geograficznej
5. Schematy aplikacyjne UML – reguły budowy i przykłady
10. Zastosowanie metodyki MDA – wybrane zagadnienia transformacji schematów aplikacyjnych UML do struktur relacyjnych baz danych

dr inż. Zenon Parzyński
Wydział Geodezji i Kartografii, Politechnika Warszawska
z.parzynski@gik.pw.edu.pl
8. Przykład zastosowania metod modelowania danych z zakresu Służby Geodezyjno-Kartograficznej

POLSKIE TOWARZYSTWO INFORMACJI PRZESTRZENNEJ
ROCZNIKI GEOMATYKI 2012 ❍ Tom X ❍ Zeszyt 1(51)

MODELE DANYCH PRZESTRZENNYCH W UML I ICH TRANSFORMACJA DO SCHEMATÓW GML I STRUKTUR BAZ DANYCH

Słowa kluczowe: geoinformacja, informacja geograficzna, model pojęciowy, UML, schemat aplikacyjny, GML, model relacyjny, transformacja

Streszczenie

Celem monografii jest przedstawienie czytelnikom podstawowych metodyk, technik i narzędzi przeznaczonych do budowy modeli pojęciowych danych przestrzennych na poziomie pojęciowym i implementacyjnym, a następnie do transformacji tych modeli do schematów XSD bazujących na języku GML i do zapisów struktur baz danych w języku DDL. Całość składa się z dwunastu rozdziałów dotyczących poszczególnych aspektów budowy modeli i ich transformacji. Wstęp wprowadza czytelników w całą przedstawianą problematykę i naświetla szerszy teoretyczny kontekst z zakresu modelowania i wykorzystania modeli w zastosowaniach praktycznych. Rozdział drugi poświęcony jest nowym metodom zapisu danych przestrzennych opartego na językach znacznikowych, a w szczególności na języku GML, objaśnia zasady takiego zapisu, zawiera krótką historię języka GML i przedstawia inne języki znacznikowe z nim powiązane. Rozdziały trzeci i czwarty stanowią wprowadzenie do modelowania informacji przestrzennej opartego o metodykę MDA z wykorzystaniem języka UML i zawierają przegląd standardów i narzędzi służących temu modelowaniu. W rozdziałach piątym i szóstym przedstawione są podstawowe zasady budowy tematycznych schematów aplikacyjnych w metodyce języka UML i języka GML zilustrowane przykładami. Rozdział siódmy poświęcony jest zagadnieniom transformacji schematów aplikacyjnych z UML do GML, a w szczególności wymaganiom i ograniczeniom, jakie muszą być spełnione, a także dostępnym narzędziom. Kolejny ósmy rozdział dotyczy modeli UML dedykowanych komponentowi infrastruktury krajowej, przeznaczonym dla Służby Geodezyjnej i Kartograficznej. W rozdziale dziewiątym dokonany jest przegląd najczęściej popełnianych błędów w budowie modeli UML przeznaczonych do utworzenia schematów bazujących na języku GML. Tematem rozdziału dziesiątego jest zastosowanie metodyki MDA do transformacji modeli UML do struktur relacyjnych baz danych. Rozdział jedenasty zawiera metodyczną analizę różnych przypadków występujących w modelach danych tematów aneksów II i III dyrektywy INSPIRE, w tym porównanie z modelami tematów aneksu I, analizę różnych typów i form danych, jakie tam występują. Dwunasty rozdział to podsumowanie, w którym zwraca się szczególną uwagę na dynamiczny rozwój metod z tego zakresu, zmiany zachodzące w zakresie stosowanej terminologii i skutki, jakie te zmiany za sobą pociągają.

UML GEOSPATIAL DATA MODELS AND THEIR TRANSFORMATION INTO GML SCHEMAS AND DATABASE STRUCTURES

Keywords: geoinformation, geographic information, conceptual model, UML, application schema, GML, relational model, transformation

Abstract

The main objective of the monograph is to present essential methodologies, technologies and software tools dedicated to building conceptual models of geospatial data on conceptual level, and implementation level, and then to be transformed into XSD schemas based on GML language and to encode data bases structures in DDL language. The whole monograph consists of twelve chapters concerning different aspects of models development and their transformation. The introduction familiarizes readers with all issues presented and clarifies broader theoretical context with regard to modeling and exploitation of models in practical applications. The second chapter is dedicated to modern methods of encoding spatial data based on markup languages, in particular on GML language; rules for that encoding are also explained. This chapter contains a short history of GML language and presents other markup languages associated with it. Chapters three and four provide an introduction to spatial information modeling based on MDA methodology with application of UML language and it contains a review of standards and tools dedicated to such modeling. In chapters five and six, essential rules of development of thematic application schemas are presented in the methodology of UML and GML languages. Examples to illustrate them are provided. Chapter seven is dedicated to issues of transformation application schemas from UML to GML, in particular to the requirements and constrains that must be fulfilled and also to available tools. The next chapter eight concerns UML models dedicated to components of the national infrastructure designated for Geodetic and Cartographic Service. In chapter nine, a review of most frequent mistakes committed in drawing up UML models dedicated to generating of schemas based on GML language are presented. The subject of chapter ten is the application of MDA methodology for transformation of UML models into relational databases structures. Chapter eleven contains methodological analysis of various cases occurring in data models for the themes defined in Annex II and III of INSPIRE Directive as well as a comparison with the models for themes defined in Annex I and an analysis of various data forms occuring there. In chapter twelve, the recapitulation is presented, in which dynamic development of methods in this area is taken in consideration. In addition, significant changes in the terminology and the effects of these changes are discussed.

POLSKIE TOWARZYSTWO INFORMACJI PRZESTRZENNEJ
ROCZNIKI GEOMATYKI 2012 ❍ Tom X ❍ Zeszyt 1(51)

Spis treści

1. Wstęp 11

2. Różnice pomiędzy językiem zapisu danych i jego dziedzinową aplikacją 15

2.1. Podstawy zapisu znacznikowego na bazie języka XML 15

2.2. Wprowadzenie do języka GML 18

2.3. Krótka historia zapisu geoinformacji 24

2.4. Języki oparte na GML i z nim powiązane 25

2.5. Przyszłość języka GML 27

2.6. Modele UML dedykowane zapisom w języku GML 30

3. Wprowadzenie do modelowania informacji przestrzennej – metodyka MDA i diagramy klas UML 35

3.1 Wprowadzenie 35

3.2. Reguły budowy schematów aplikacyjnych w UML 39

4. Przegląd standardów i narzędzi stosowanych do modelowania informacji geograficznej 43

4.1. Model dziedzinowy informacji geograficznej 44

4.2. Funkcjonalność narzędzi do modelowania pojęciowego 45

5. Schematy aplikacyjne UML – reguły budowy i przykłady 49

5.1. Pojęcie schematu aplikacyjnego, jego rola i znaczenie 49

5.2. Proces budowy schematów aplikacyjnych 50

5.3. Przykłady schematów aplikacyjnych UML 52

6. Budowa schematu aplikacyjnego GML – reguły budowy, narzędzia i przykłady 55

6.1. Reguły budowy schematów aplikacyjnych GML 55

6.2. Przykład przekształcenia schematu aplikacyjnego UML na GML 66

7. Transformacja schematu aplikacyjnego UML do schematu aplikacyjnego GML – wymagania, ograniczenia i wybrane narzędzia 69
7.1. Metody transformacji UML do GML 69
7.2. Metoda ręczna 70
7.3. Metoda automatyczna 71
7.4. Podsumowanie 76
8. Przykład zastosowania metod modelowania danych z zakresu Służby Geodezyjno-Kartograficznej 79
8.1. Założenia przyjęte w GUGiK przy opracowywaniu projektów rozporządzeń 79
8.2. Realizacja założeń 80
8.3. Przykłady schematów aplikacyjnych do projektów rozporządzeń 83
9. Najczęściej popełniane błędy w modelach UML dla schematów aplikacyjnych GML 87
9.1. UML jest cierpliwy jak papier 87
9.2. Wymagania dotyczące modeli UML dla INSPIRE 90
10. Zastosowanie metodyki MDA – wybrane zagadnienia transformacji schematów aplikacyjnych UML do struktur relacyjnych baz danych 95
10.1. Transformacja w ujęciu metodyki MDA 95
10.2. Ogólne zasady mapowania pomiędzy modelem obiektowym i modelem relacyjnym 97
10.3. Transformacja schematu aplikacyjnego UML do logicznej struktury relacyjnej bazy danych 101
11. Schematy aplikacyjne tematów aneksów II i III dyrektywy INSPIRE 107
11.1. Nietypowy przypadek – temat Geologia 115
12. Podsumowanie 121
Słownik podstawowych terminów stosowanych w tekście 125
Literatura 131

POLSKIE TOWARZYSTWO INFORMACJI PRZESTRZENNEJ
ROCZNIKI GEOMATYKI 2012 ❍ Tom X ❍ Zeszyt 1(51)

Janusz Michalak

Słownik podstawowych terminów stosowanych w tekście

Abstrakcyjny – obiekt, atrybut, typ, klasa *(abstract – object, attribute, type, class)* – **1:** Określony ogólnie, bez szczegółów związanych z określoną implementacją (uwarunkowaniami technologicznymi) lub z określoną aplikacją (uwarunkowaniami wynikającymi z dziedziny zastosowania). Na przykład wynik pomiaru w znaczeniu ogólnym jako atrybut w modelu pojęciowym nie musi mieć określonego typu. Typ będzie zależał od fizycznego charakteru mierzonego elementu i od typy przyrządu pomiarowego. **2:** Klasa abstrakcyjna w modelu danych to klasa, która nie ma własnych obiektów, a jedynie służy jako klasa bazowa dla innych klas. Użycie takiej klasy jest uzasadnione tylko gdy są (lub mogą być) wyprowadzone z nie przynajmniej dwie klasy.

Atrybut *(attribute)* – Właściwość wyróżnienia lub obiektu określona przez nazwę tej właściwości i zakres wartości, jakie mogą być przypisane tej nazwie dla określenia tej właściwości.

Atrybut geoprzestrzenny *(geospatial attribute)* – Właściwość (cecha) wynikająca z faktu, że wyróżnienie zajmuje pewne miejsca w rzeczywistości w sensie geoprzestrzennym. Najczęściej przez domniemanie przyjmuje się, że określenie geoprzestrzenny obejmuje również czas, czyli jest równoznaczne z określeniem czaso-geoprzestrzenny. Przykładami takich atrybutów są: wielkość, kształt, położenie, przynależność geoprzestrzenna (np. leży w obrębie), relacje geoprzestrzenne względem innych wyróżnień (np. odległość lub rodzaj sąsiedztwa).

Atrybut niegeoprzestrzenny *(non-geospatial attribute)* – Wszystkie pozostałe atrybuty niezwiązane z odniesieniem przestrzennym. Atrybuty te mogą należeć zarówno do wyróżnień geoprzestrzenne jak i do innych obiektów i wystąpień niegeoprzestrzennych.

Cecha *(trait)* – Kategoria klasy, której zadaniem jest (w przypadku modeli danych) dostarczenie innej klasie określonych własności (atrybutów i powiązań z innymi klasami). W tym przypadku klasa ma stereotyp «`trait`». Podobnym mechanizmem pozwalającym na uniknięcie problemów wielokrotnego dziedziczenia jest **domieszka**.

Dane (*data*, w liczbie pojedynczej: *datum*) – **1:** Jednostki informacji, czyli pojedyncze fragmenty informacji. Dane niezorganizowane nie stanowią informacji i często są bezużyteczne. Dane zorganizowane stanowią elementy informacji. Zorganizowanie danych może być jawne, na przykład w językach znacznikowych lub niejawne, na przykład miejsce umieszczenia adresu na kopercie decyduje, czy jest to adres nadawcy czy odbiorcy. **2:** Fakty, statystyki, opinie i przewidywania zebrane z różnych wewnętrznych i zewnętrznych źródeł. Dane bez kontekstu są szumem (Nowicki i Staniszkis, 2002).

Dane geoprzestrzenne *(geospatial data)* – **1:** Dane w sensie zdefiniowanym przez informatykę, ale w odróżnieniu od innych rodzajów danych są one odniesiona do określonego miejsca (fragmentu przestrzeni) i w rezultacie niezbędnymi ich składnikami są dane określające położenie tego miejsca względem Ziemi. **2:** Dane przestrzenne dotyczące Ziemi i wszystkich obiektów przestrzennych z nią związanych (Gaździcki, 2004).

Domieszka *(mixin)* – **1:** Kategoria klasy, której zadaniem jest (w przypadku modeli danych) dostarczenie innej klasie określonych własności (atrybutów i powiązań z innymi klasami). Taka klasa nie ma własnych obiektów, czyli musi być abstrakcyjna. Stosowanie tego rodzaju klasy jest uzasadnione tyko w przypadkach, gdy przynajmniej dwie zwykłe klasy otrzymują w ten sposób własności. Jest to sposób na uniknięcie problemów z implementacją wielokrotnego dziedziczenia. Jedyny przypadek zastosowania klasy *mixin* do języka GML to modele dla niektórych tematów INSPIRE. **2:** Ograniczony sposób dziedziczenia ma pozwalający również na uniknięcie problemów z implementacją wielokrotnego dziedziczenia. W takim przypadku powiązanie dziedziczenia ma stereotyp «mixin». Porównaj: **cecha**.

Encja *(entity)* – Pojęcie z modelu encja-związek, oznaczające konkretny lub abstrakcyjny byt wyróżnialny w modelowanej rzeczywistości. W odróżnieniu od obiektu, encja nie jest kojarzona z metodami (Subieta, 1999a).

GML *(Geography Markup Language)* – Język znaczników geograficznych, aplikacja języka (metajęzyka) XML przeznaczona do zapisu geoinformacji w celu przesyłania jej pomiędzy różnymi systemami – on-line, niezależnie od platformy sprzętowo-systemowej i niezależnie od charakteru i technologii systemu geoinformacyjnego (Gaździcki, 2004).

Informacja *(information)* – **1:** Dane komputerowe, które są zorganizowane i przedstawione w usystematyzowanej formie dla zrozumiałości ich podstawowego znaczenia. Związki pomiędzy informacją i danymi wyjaśnia definicja danych. **2:** Dane interpretowane w kontekście określonego celu (Nowicki i Staniszkis, 2002). **3:** Wiedza uzyskiwana w drodze interpretacji danych, która w ustalonym kontekście ma określone znaczenie i dotyczy obiektów, takich jak fakty, zdarzenia, przedmioty, zjawiska, procesy i idee (Gaździcki, 2004).

Informacja geograficzna – patrz: **informacja geoprzestrzenna**.

Informacja geoprzestrzenna *(geospatial information)* – **1:** Informacja w sensie zdefiniowanym przez informatykę, ale w odróżnieniu od innych rodzajów informacji jest ona odniesiona do określonego miejsca (fragmentu przestrzeni) i w rezultacie niezbędnymi jej składnikami są dane określające położenie tego miejsca względem Ziemi. **2:** Informacja uzyskiwana w drodze interpretacji danych geoprzestrzennych (Gaździcki, 2004).

Instancja *(instance)* – Synonim egzemplarza stosowany w normach PN-EN ISO 19100 (Gaździcki, 2011).

Klasa *(class)* – Pojęcie klasy jest używane w trzech dość bliskich znaczeniach: **(1)** zbiór obiektów o zbliżonych własnościach; **(2)** byt semantyczny rozumiany, jako miejsce przechowywania takich cech grupy podobnych obiektów, które są dla nich niezmienne (np. zestawu atrybutów, nazwy, metod, ograniczeń dostępu); **(3)** wyrażenie językowe specyfikujące budowę obiektów, dozwolone operacje na obiektach, ograniczenia dostępu, wyjątki, itd. Zwykle klasy wiąże się ze sobą poprzez hierarchię (lub inną strukturę) dziedziczenia (Subieta, 1999a).

MDA – 1: *(Model Driven Approach)* Podejście oparte na modelu: pojęciowym, logicznym i fizycznym. Niezależny od implementacji schemat aplikacyjny zostaje odwzorowany na różne specyfikacje (wykorzystujące różne technologie, np. usługi sieciowe, relacyjne bazy danych, XML), a te z kolei mogą zostać zaimplementowane (wdrożone) na różnych platformach sprzętowo-programowych (CEN, 2006). **2:** *(Model Driven Architecture)* Zbiór metod porządkujących proces tworzenia systemów informatycznych opartych na budowie modeli i ich transformacji. Koncepcja MDA została opracowana przez międzynarodową organizację OMG, której celem jest rozwiązywanie problemów związanych z integracją systemów informatycznych pochodzących od różnych dostawców oraz działających na różnych platformach informatycznych (OMG, 2003).

Metamodel *(metamodel)* – W założeniu, model definiujący składnię, semantykę i pragmatykę wprowadzonego modelu, notacji lub diagramu. Metamodel proponowany przez autorów UML ustala pewne elementy składni diagramów, ograniczenia typologiczne, klasyfikację pojęć oraz związki pomiędzy pojęciami (Subieta, 1999a).

Metka *(tagged value)* – Inaczej wartość etykietowana. Obok stereotypów i ograniczeń, to jeden z mechanizmów rozszerzenia semantyki języka UML. Pozwala dołączyć do elementu modelu UML dodatkowe właściwości. Metka to para *klucz=wartość*.

Metodyka *(methodology)* – Zestaw pojęć, notacji, modeli formalnych, języków i sposobów postępowania służący do analizy rzeczywistości (stanowiącej przedmiot projektowanego systemu informatycznego) oraz do projektowania pojęciowego, logicznego i/lub fizycznego. Zwykle metodyka jest powiązana z odpowiednią notacją (diagramami) służącymi do zapisywania wyniku poszczególnych faz projektu, jako środek wspomagający ludzką pamięć i wyobraźnię i jako środek komunikacji w zespołach oraz pomiędzy projektantami i klientem (Subieta, 1999a).

Model pojęciowy *(conceptual model)* – Model procesów lub model struktury danych odwołujący się do ludzkiej percepcji i wyobraźni, mający za zadanie zrozumienie problemu, udokumentowanie wyniku analizy lub projektu w czytelnej i abstrakcyjnej formie językowej oraz ułatwienie komunikacji w zespołach ludzkich (Subieta, 1999a).

Model semantyczny *(semantic model)* – Zestaw pojęć, technik i notacji mający na celu odwzorowanie semantyki danych, czyli ich znaczenia w świecie zewnętrznym. Modele semantyczne wprowadzają w tym celu pojęcia, takie jak: generalizacja, specjalizacja, asocjacja, agregacja, klasyfikacja, własności temporalne, zdarzenia, własności behawioralne, itd. Przykładem prostego modelu semantycznego jest model encja-związek. Niekiedy terminem "model semantyczny" określa się również konkretny diagram (lub inną formę językowo-graficzną) odwzorowującą rzeczywistość opisywaną przez dane (Subieta, 1999a).

Norma *(standard)* – **1:** Dokument przyjęty na zasadzie konsensu i zatwierdzony przez upoważnioną jednostkę organizacyjną, ustalający – do powszechnego i wielokrotnego stosowania – zasady, wytyczne lub charakterystyki odnoszące się do różnych rodzajów działalności lub ich wyników i zmierzający do uzyskania optymalnego stopnia uporządkowania w określonym zakresie (Ustawa, 2002). **2:** Polska Norma – jest normą o zasięgu krajowym, przyjętą w drodze konsensu i zatwierdzoną przez krajową jednostkę normalizacyjną (Polski Komitet Normalizacyjny), powszechnie dostępną, oznaczoną – na zasadzie wyłączności – symbolem PN (PKN, 2010). Zobacz: **normy ISO serii 19100, standard, standardy OGC.**

Normy ISO serii 19100 *(ISO 19100 series of International Standards)* – Rodzina norm ISO w dziedzinie informacji geograficznej. Wynik prac Komitetu Technicznego ISO/TC211, który pracuje nad wieloma projektami standaryzacji informacji przestrzennej w bardzo szerokim zakresie tej problematyki. Zobacz: **norma, standard, standardy OGC**.

Obiekt *(object*) – **1:** W teorii informacji – konkretny lub abstrakcyjny byt (wystąpienie) wyróżnialny w modelowanej rzeczywistości, posiadający nazwę, jednoznaczną identyfikację, wyraźnie określone granice, atrybuty i inne właściwości takie jak rodzaj struktury wewnętrznej lub struktury danych z nim związanych. Te składniki obiektu określają: jego stan (poprzez wartości atrybutów i powiązania) i jego zachowanie się (poprzez operatory i funkcje, czyli metody) (Subieta, 1999a). **2:** W geomatyce przyjmuje się, że obiekt jest wystąpieniem klasy i jest to oparte na paradygmacie obiektowości wywodzącym się z języka UML, który jest przyjęty do opisu modeli pojęciowych (OMG, 2001). **3:** Termin stosowany w różnych znaczeniach; dla uniknięcia wątpliwości, jeśli jego znaczenie nie wynika z kontekstu, powinien być uzupełniony dodatkowym określeniem (Gaździcki, 2004).

Rola *(role)* – W języku UML jedna z możliwości opisu powiązania. Pozostałe to nazwa powiązania oraz krotność. Każda klasa biorąca udział w powiązaniu ogrywa w nim określoną rolę. Inaczej jest to „oblicze", które klasa przy jednym końcu powiązania prezentuje klasie przy drugim jego końcu.

Schemat *(schema)* – **1:** Opis logicznej struktury bazy danych lub innego systemu związanego z danymi, np. interfejsu wymiany danych (XML Schema). **2:** Opis atrybutów wyróżnień *(feature)*, lub bardziej dokładnie – specyficzny model atrybutów dla wyróżnień określony za pomocą elementarnych typów danych i ograniczeń dotyczących tych typów (Buehler, McKee, 1996).

Schemat aplikacyjny *(application schema)* – Schemat przeznaczony dla konkretnego systemu lub dla konkretnej dziedziny zastosowań.

Schemat implementacyjny *(implementation schema)* – Schemat uwzględniający technologiczne środowisko, w którym będzie realizowana jego aplikacja. Na przykład zapisany w formie schematu XML.

Specyfikacja *(specification)* – **1:** Abstrakcyjny opis bytu programistycznego (procedury, modułu, klasy, obiektu, bazy danych, itp.) określający reguły użycia lub ustalający podstawowe założenia jego implementacji (Subieta, 1999a). **2:** Dokument lub opis, który określa w sposób kompletny, precyzyjny i sprawdzalny wymagania, projekt lub charakterystykę systemu lub jego fragmentu, a często także procedury dla określenia czy te wymagania są spełnione.

Standard *(standard)* – Wzorzec rozwiązania sprzętowego lub programowego zatwierdzony przez instytucję normalizacyjną lub przyjęty nieformalnie wskutek dużego upowszechnienia, w przypadku standardów informatycznych najczęściej o zasięgu światowym. Do najważniejszych instytucji opracowujących standardy należą: ISO, IEEE, ANSI. Przykładami standardów są: RS-232-C (fabryczny standard interfejsu sprzętowego), ANSI C++ (oficjalny standard języka programowania), POSIX (standard IEEE przenośnego systemu uniksowego), CORBA (standard obiektowych systemów rozproszonych) (Płoski, 1999). Zobacz: **standardy OGC, norma, normy ISO serii 19100.**

Standardy OGC *(OGC standards)* – Techniczne dokumenty specyfikujące interfejsy i reguły zapisu danych geoprzestrzennych. Stanowią one główne rezultaty działalności OGC *(Open Geospatial Consortium)* i są opracowywane przez zespoły złożone z członków OGC dla rozwiązywania różnorodnych problemów dotyczących interoperacyjności. Wszystkie publiczne dokumenty OGC są łatwo dostępne bez żadnych opłat. OGC ma ponad 400 członków, w tym ponad połowa to wyższe uczelnie i instytucje naukowe, także prawie połowę stanowią członkowie europejscy. Standardy OGC dzielą się na specyfikacje abstrakcyjne i standardy implementacyjne. Wiele z tych standardów zostało przyjęte przez komitet ISO/TC 211 jako normy ISO, na przykład: 19107, 19115, 19119, 19123, 19125, 19128, 19136, 19139, 19142, 19143, i 19156. Ze standardami OGC powiązane są inne oficjalne dokumenty OGC, na przykład: OGC *Reference Model* (ORM), *Engineering Reports* lub nieoficjalne, na przykład *Best Practices Documents* i *Discussion Papers*. Zobacz: **standard, norma, normy ISO serii 19100.**

Stereotyp *(stereotype)* – W terminologii UML, klasyfikacja elementu modelu posiadająca semantyczne konsekwencje. Stereotypy mogą być predefiniowane lub zdefiniowane przez użytkownika (Subieta, 1999a).

Struktura *(structure)* – Termin w C++ (także w innych językach) na oznaczenie zestawu nazwanych wartości, w innych językach odpowiada jej zapis lub rekord (Subieta, 1999a).

Tabela *(table)* – Struktura danych implementowana w relacyjnych bazach danych, często nazywana relacją. Tabela składa się z wierszy lub inaczej krotek. Należy zwrócić uwagę, że pomiędzy relacją (w sensie matematycznym) i tabelą występują dość istotne różnice koncepcyjne. Tabela jest wyposażona w nazwy kolumn (które niosą informację semantyczną) (Subieta, 1999a).

Tożsamość *(identity)* – Tożsamość obiektu oznacza, że obiekt istnieje i jest odróżnialny niezależnie od jego aktualnego stanu (wartości atrybutów), który może się zmieniać; możliwe są dwa różne obiekty o identycznych wartościach atrybutów. Praktycznie, tożsamość oznacza istnienie unikalnego wewnętrznego (nieczytelnego dla użytkownika) identyfikatora obiektu, który nie ulega zmianie podczas życia obiektu (Subieta, 1999a).

Unia *(union)* – Typ struktury, rekordu lub obiektu, który może mieć alternatywnie dwa lub więcej zestawów atrybutów. Przykładowo, jeżeli właścicielem samochodu może być osoba lub firma, to obiekt *Samochód* może posiadać alternatywnie albo atrybut *Nazwisko Właściciela* albo atrybut *WłasnośćFirmy*. Unia może mieć związany dyskryminator *(discriminator)*, tj. atrybut, którego wartość określa, z którym wariantem mamy do czynienia. Może też nie mieć dyskryminatora; wówczas odpowiedzialność za rozróżnianie wariantów spada na programistę (tak jest np. w C i C++). Brak dyskryminatora w unii podkopuje koncepcję mocnej kontroli typów i stwarza okazję do bardzo trudnych błędów (Subieta, 1999a).

Walidator (validator) – Program komputerowy sprawdzający poprawność dokumentu (np. XML) o określonej składni.

Wyróżnienie geoprzestrzenne *(geospatial feature)* (w literaturze polskiej termin *feature* jest często tłumaczony jako obiekt) – **1:** Podstawowy fragment (atom) informacji geoprzestrzennej. Posiada atrybuty geoprzestrzenne (geometryczne i topologiczne) np. kształt, rozciągłość, położenie, relacje z innymi wyróżnieniami. Często pojęcie wyróżnienie jest my-

lone z pojęciem obiekt, jednak wyróżnienie może być obiektem, ale też może nim nie być (Mark i in., 2001). Ponieważ w geomatyce wszystkie wyróżnienia są geoprzestrzenne, przymiotnik geoprzestrzenny jest na ogół pomijany i używa się krótszego terminu wyróżnienie. **2:** Cyfrowa reprezentacja zjawiska (bytu) świata rzeczywistego lub jego abstrakcja w modelu pojęciowym. Wyróżnienie ma określone miejsce w przestrzeni i czasie jako jego atrybuty (Buehler, McKee, 1996). Przykładem wyróżnienia może być prawie wszystko co może być umieszczone w przestrzeni i czasie: stół, budynek, miasto, drzewo, fragment lasu, ekosystem, trasa przejazdu lub wyż atmosferyczny jako obszar wysokiego ciśnienia powietrza. **3:** Abstrakcja zjawiska świata rzeczywistego. Termin wyróżnienie może odnosić się do typu zjawiska lub jego konkretnego wystąpienia (ISO/TC 211, 2002a), np. „rzeka" i „Wisła".

Związek *(relationship)* – **1:** W języku UML i w konsekwencji także w normach grupy ISO 19100 – semantyczne połączenie pomiędzy elementami modelu. Przykładami związków są agregacje, kompozycje (agregacje całkowite), powiązania i uogólnienia. **2:** W modelu encji-relacji – powiązanie pomiędzy encjami (Michalak, 2005a).

Literatura

AB ORMSC (Architecture Board ORMS), 2001: Model Driven architecture (MDA). Document number ormsc/2001-07-01.
URL: http:// http://www.enterprise-architecture.info/Images/MDA/MDA%20Technical.pdf

Altova, XMLSpy. URL: http://www.altova.com/xml-editor

BGWM (Biuro Geodety Województwa Mazowieckiego), 2009: Opis koncepcji identyfikatorów, wersjonowania zmian, stosowania reguły *nil reason*.
URL: http://www.geointegracja.gov.pl/ download/file.php?id=80&sid=f2c1f79e942a2cf12ed12a99aee5eec0

Biron P. V., Permanente K., Malhotra A. (W3C), 2004: XML Schema Part 2: Datatypes. Second Edition. W3C Recommendation 28 October 2004. URL: http://www.w3.org/TR/xmlschema-2

Boisvert E., Brodaric B., 2008: GroundWater Markup Language Specification v. 1.0.
URL: http://ngwd-bdnes.cits.rncan.gc.ca/service/api_ngwds/en/gwml.html

Booch G., Rumbaugh J., Jacobson I., 2002: UML – przewodnik użytkownika. Z serii: Inżynieria oprogramowania. Wydanie polskie. Wydawnictwa Naukowo-Techniczne, Warszawa.

Brink L., Portele C., Vretanos P. A. (OGC), 2011: Geography Markup Language (GML) simple features profile (with Corrigendum). OpenGIS Implementation Standard Profile.
URL: http://portal.opengeospatial.org/files/?artifact_id=42729

Buechler K., McKee L. (ed.), 1996: The OpenGIS Guide – Introduction to Interoperable Geoprocessing – Part I of the Open Geodata Interoperability Specification (OGIS). OGIS TC Document 96-001, Open GIS Consortium, Wayland.

Burggraf D., 2011: Input to the GML 4 workshop.
URL: http://external.opengeospatial.org/twiki_public/GML/Gml4WorkshopInput

CEN, 2006: prCEN/TR 15449, Geographic information – Standards, specifications, technical reports and guidelines, required to implement Spatial Data Infrastructure.

Chojka A., 2006: Przegląd metod, środków formalnych i narzędzi programowych wspomagających modelowanie pojęciowe informacji geograficznej. Część I – Modelowanie pojęciowe. *Magazyn Geoinformacyjny Geodeta*, nr 5 (132).

Cox S. (ed.) (OGC), 2010: Geographic Information: Observations and Measurements – OGC Abstract Specification Topic 20. URL: http://portal.opengeospatial.org/files/?artifact_id=41579

Cox S., 2011: Hollow World: a GML application schema template. Solid Earth and Environment GRID (SEE GRID community website). URL: https://www.seegrid.csiro.au/wiki/AppSchemas/HollowWorld

CTWG-O&M (INSPIRE Cross Thematic Working Group on Observations & Measurements), 2011: D2.9 Guidelines for the use of Observations & Measurements and Sensor Web Enablement-related standards in INSPIRE Annex II and III data specification development, Version 1.0.
URL: http://inspire.jrc.ec.europa.eu/documents/Data_Specifications/D2.9_O&M_Guidelines_V1.0.pdf

DT_DS (INSPIRE Drafting Team „Data Specifications"), 2008: D2.6: Methodology for the development of data specifications, Version 3.0.
URL: http://inspire.jrc.ec.europa.eu/reports/ImplementingRules/DataSpecifications/D2.6_v3.0.pdf

DT_DS (INSPIRE Drafting Team „Data Specifications"), 2010a: D2.5: Generic Conceptual Model, Version 3.3. URL: http://inspire.jrc.ec.europa.eu/documents/Data_Specifications/D2.5_v3_3.pdf

DT_DS (INSPIRE Drafting Team „Data Specifications"), 2010b: D2.7: Guidelines for the encoding of spatial data, Version 3.2. URL: http://inspire.jrc.ec.europa.eu/documents/Data_Specifications/D2.7_v3.2.pdf

EC (European Commission), 2009: Guidance Document No. 22 – Updated Guidance on Implementing the Geographical Information System (GIS) Elements of the EU Water policy. Common Implementation Strategy for the Water Framework Directive (2000/60/EC). Technical Report – 2009 – 028. URL: http://circa.europa.eu/Public/irc/ env/wfd/library?l=/framework_directive/guidance_documents/guidance-no22-_nov08pdf_1/_EN_1.0_&a=d

EP&CEU (European Parliament and Council of the European Union), 2007: Directive 2007/2/EC of the European Parliament and of the Council of 14 March 2007 establishing an Infrastructure for Spatial Information in the European Community (INSPIRE).

Fowler M., Scott K., 2002: UML w kropelce. Oficyna wydawnicza LTP, Warszawa.

Gaździcki J., 2004: Leksykon geomatyczny – Lexicon of geomatics. Polskie Towarzystwo Informacji Przestrzennej, Warszawa.

Gaździcki J., 2011: [W:] (red.) Gaździcki J. Internetowy leksykon geomatyczny. URL: http://www.ptip.org.pl/auto.php?page=Encyclopedia&enc=1

Githaiga J., 2010: Project Overview – FullMoon. Solid Earth and Environment GRID (SEE GRID community website). URL: https://www.seegrid.csiro.au/wiki/Siss/FullMoon

Huang C-H., Chuang T-R., Deng D-P., Lee H-M, 2009: Building GML-native web-based geographic information systems. *Computers&Geosciences*, no 35, 1802-1816. URL: http://www.iis.sinica.edu.tw/papers/trc/8843-F.pdf

IGW-CGI-IUGS (Commission for the Management and Application of Geoscience Information – CGI, Interoperability Working Group – IWG, International Union of Geological Sciences – IUGS), 2008: GeoSciML Cookbook – How To Map Data to GeoSciML, Version 2. URL: http://www.geosciml.org/geosciml/2.0/cookbook/GeoSciML_Data_CookBook_V2.pdf

ISO/TC 211 (Geographic Information/Geomatics), 2002a: ISO 19101: Geographic information – Reference model. URL: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=26002

ISO/TC 211 (Geographic Information/Geomatics), 2002b: ISO 19108:2002 Geographic information – Temporal schema. URL: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=26013

ISO/TC 211 (Geographic Information/Geomatics), 2003: ISO 19107:2003 – Geographic information – Spatial schema. URL: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=26012

ISO/TC 211 (Geographic Information/Geomatics), 2005a: ISO 19103 Technical Specification, Geographic information – Conceptual schema language. URL: http://www.iso.org/iso/catalogue_detail.htm?csnumber=37800

ISO/TC 211 (Geographic Information/Geomatics), 2005b: ISO 19109:2005 Geographic information – Rules for application schema. URL: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=39891

ISO/TC 211 (Geographic Information/Geomatics), 2006: ISO 19110 – Geographic information – Methodology for feature cataloguing. URL: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=39965

ISO/TC 211 (Geographic Information/Geomatics), 2007a: ISO 19136:2007 – Geographic information – Geography Markup Language (GML). URL: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=32554

ISO/TC 211 (Geographic Information/Geomatics), 2007b: ISO 19139 Technical Specification, Geographic information – Metadata – XML schema implementation. URL: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=32557

ISO/TC 211 (Geographic Information/Geomatics), 2009: ISO 19104:2009 Technical Specification, Geographic information – Terminology URL: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=32557

ISO/TC 211 (Geographic Information/Geomatics), 2011: ISO 19118, Geographic information – Encoding. URL: http://www.iso.org/iso/iso_catalogue/catalogue_ics/catalogue_detail_ics.htm?csnumber=44212

ISO/TC 211 (Geographic Information/Geomatics), 2012: Introduction:Welcome to the ISO/TC 211 Harmonized Model Web server. URL: http://www.isotc211.org/hmmg/HTML/root.html

Lake R., Burggraf D., Trninic M., Rae L., 2004: Geography Markup Language: Foundation for the Geo-Web. Wiley (w znacznej części dostępna bezpłatnie)
URL: http://media.wiley.com/product_ancillary/47/04708715/DOWNLOAD/Lake.zip

Mark D. M., Skupin A., Smith B., 2001: Features, Objects, and other Things: Ontological Distinctions in the Geographic Domain. Spatial Information Theory, Proceedings of COSIT 2001, Springer.
URL: http://wings.buffalo.edu/philosophy/faculty/smith/articles/COSIT01MSS.pdf

Michalak J., 2003a: Modele pojęciowe hydrogeologicznych danych geoprzestrzennych – podstawy metodyczne. Biuletyn PIG – *Hydrogeologia*, z. V, nr 406, monografia.

Michalak J., 2003b: Podstawy metodyczne i technologiczne infrastruktur geoinformacyjnych. *Roczniki Geomatyki*, t. 1, z. 2. PTIP, Warszawa, monografia.

Michalak J., 2003c: Geomatics in hydrogeology. *Geological Quarterly*, 47(1): 69-76.

Michalak J., 2005a: Terminologia polska w zakresie technologii interoperacyjnych w geomatyce. [W:] (red.) Gaździcki J. Internetowy Leksykon Geomatyczny.
URL: http://www.ptip.org.pl/auto.php?page=Encyclopedia&enc=1

Michalak J., 2005b: HGLML – HydroGeoLogical Markup Language – znacznikowy język wymiany geoinformacji hydrogeologicznej. Współczesne Problemy Hydrogeologii, t. XII: 499-504.

Michalak J., Nawalany M., Sadurski A., (red.) 2011: Schematyzacja warunków hydrogeologicznych na potrzeby numerycznego modelowania przepływu w JCWPd. Wyd. PIG – PIB, Warszawa.
URL: http://www.psh.gov.pl/plik/id,6091,v,artykul_4556.pdf

Michalak J, 2012: Testowanie roboczych wersji specyfikacji danych tematów załączników II i III INSPIRE. *Roczniki Geomatyki*, t. 10, z. 2, PTIP, Warszawa.

Nowicki B., Staniszkis W., 2002: Inteligentny system zarządzania wiedzą – prezentacja projektu. [W:] Mat. Konferencji eDemocracy, VI Konf. Miasta w Internecie, Zakopane.

OMG (Object Management Group), 2001: OMG Unified Modeling Language Specification, version 1.4. OMG Document Repository. URL: http://cgi.omg.org/docs/formal/01-09-67.pdf

OMG (Object Management Group), 2003: Object Management Group, Model Driven Architecture Guide Version 1.0.1 URL: http://www.omg.org/cgi-bin/doc?omg/03-06-01.pdf

OMG (Object Management Group), 2010: Object Constraint Language.
URL: http://www.omg.org/spec/OCL/2.2

Pachelski W., Parzyński Z. ,2007: Aspekty metodyczne wykorzystania norm serii ISO 19100 do budowy geodezyjnych składników krajowej infrastruktury danych przestrzennych. *Roczniki Geomatyki*, t.5, z.3, PTIP, Warszawa.

Peng Z_R., Zhang C., 2004: The roles of geography markup language (GML), scalable vector graphics (SVG), and Web feature service (WFS) specifications in the development of Internet geographic information systems (GIS).
URL: http://gis.geog.uconn.edu/personal/paper1/journal%20paper/3%202004%20GeographicalSystem1.pdf

PKN (Polski Komitet Normalizacyjny), 2010: Informacje podstawowe o PN.
URL: http://www.pkn.pl/informacje-podstawowe-o-pn

Płoski Z., 1999: Słownik Encyklopedyczny – Informatyka. Wyd. Europa, Warszawa.

Portele C., 2007: OpenGIS Geography Markup Language (GML) Encoding Standard. OpenGIS Standard.
URL: http://portal.opengeospatial.org/files/?artifact_id=20509

Portele C., 2008a: Mapping UML to GML Application Schemas. Guidelines and Encoding Rules. Interactive Instruments GmbH.
URL: http://www.interactive-instruments.de/ugas/UGAS-Guidelines-and-Encoding-Rules.pdf.

Portele C., 2008b: Mapping UML to GML Application Schemas. ShapeChange – Architecture and Description. Interactive Instruments GmbH. URL: http://www.interactive-instruments.de/ugas/ShapeChange.pdf

Portele C., 2012: OGC Geography Markup Language (GML) – Extended schemas and encoding rules. OpenGIS Implementation Standard. URL: https://portal.opengeospatial.org/files/?artifact_id=46568

Refsgaard J. C., Henriksen H. J., 2004: Modelling guidelines – terminology and guiding principles. *Advances in Water Resources* 27 (2004): 71-82.

Schmuller J., 2003: UML dla każdego. Helion, Gliwice.

Ustawa z dnia 12 września 2002 r. o normalizacji.
URL: http://www.pkn.pl/sites/default/files/ustawa_o_normalizacji_2.pdf

Ustawa z dnia 4 marca 2010 r. o infrastrukturze informacji przestrzennej, Dz.U. 2010 Nr 76, poz. 489.
Skogan D., 1999: UML as a Schema Language for XML based data Interchange. Materiały konferencji UML'99. URL: http://xml.coverpages.org/skoganUMLpaper-pdf.gz
Sparx System, Enterprise Architect. URL: http://www.sparxsystems.com.au
Subieta K., 1998: Obiektowość w projektowaniu i bazach danych. Akademicka Oficyna Wydawnicza PLJ, Warszawa.
Subieta K., 1999a: Słownik terminów z zakresu obiektowości. Akademicka Oficyna Wyd. PLJ, Warszawa. URL: http://www.ipipan.waw.pl/~subieta/artykuly/slownik_obiektowosci/hasla_slownika.html
Subieta K., 1999b: Wprowadzenie do obiektowych metodyk projektowania i notacji UML. Jedenasta Górska Szkoła PTI Szczyrk.
Tennakoon W., 2003: Visualization of GML data using XSLT. URL: http://www.itc.nl/library/Papers_2003/msc/gim/tennakoon.pdf
TWG GE (INSPIRE Thematic Working Group – Geology), 2011: D2.8.II.4 INSPIRE Data Specification on Geology – Draft Guidelines. Version 2.9.1. URL: http://inspire.jrc.ec.europa.eu/documents/Data_Specifications/INSPIRE_DataSpecification_GE_v2.01.pdf
TWG-CP (INSPIRE Thematic Working Group – Cadastral Parcels), 2010: D2.8.I.6 INSPIRE Data Specification on Cadastral Parcels – Guidelines, version: 3.0.1. URL: http://inspire.jrc.ec.europa.eu/documents/ Data_Specifications/INSPIRE_DataSpecification_CP_v3.0.1.pdf
WFD WG GIS (Working Group 3.1 – GIS), 2003: Guidance Document No 9 – Implementing the Geographical Information System Elements (GIS) of the Water Framework Directive. Water Framework Directive (WFD) – Common Implementation Strategy. URL: http://www.ec-gis.org/docs/F2305/GIS-GD.PDF
Woolf A., 2009: Enterprise Architect instructions, STFC Rutherford Appleton Laboratory. URL: http://wiki.services.eoportal.org/tiki-download_wiki_attachment.php?attId=732
Zhang C., Peng Z-R., Li W., Day M. J., 2003: GML-Based Interoperable Geographical Databases. URL: http://www.ucgis.org/summer03/studentpapers/chuanrongzhang.pdf