
POLSKIE
TOWARZYSTWO
INFORMACJI
PRZESTRZENNEJ

# ROCZNIKI GEOMATYKI 2003

## Podstawy metodyczne i technologiczne infrastruktur geoinformacyjnych

Janusz Michalak

Tom I
Zeszyt 2
Warszawa

**JANUSZ MICHALAK**
**Wydział Geologii Uniwersytetu Warszawskiego**
**Al. Żwirki i Wigury 93, 02-089 Warszawa**
**e-mail: J.Michalak@geo.uw.edu.pl**
**tel. (022) 55-40-529 fax (022) 55-40-001**
**http://netgis.geo.uw.edu.pl**

# Spis treści

# 3. MODELE POJĘCIOWE DANYCH, USŁUG I INTERFEJSÓW

Rozdział ten przedstawia podstawy metodyczne, na których są oparte i przy pomocy których są rozwijane technologie stosowane w infrastrukturach geoinformacyjnych.

Dwa główne międzynarodowe ośrodki zajmujące się standaryzacją w obszarze geomatyki (OGC i ISO/TC 211) przyjęły jako podstawę swoich prac metodykę i język UML *(Unified Modeling Language)*. Ponieważ możliwości tego języka wykraczają znacznie poza potrzeby opisu modeli pojęciowych struktur danych geoprzestrzennych, standard ISO 19103 określa profil (zawężenie) języka UML do tych celów i jednocześnie definiuje reguły jego stosowania.

W rozdziale 2.2.4 przedstawione są główne rodzaje modeli pojęciowych stosowane w zagadnieniach geomatycznych: ontologiczny, ogólny, abstrakcyjny, implementacyjny i aplikacyjny. Tu problematyka modelowania pojęciowego jest opisana bardzie szczegółowo i obejmuje: sposoby agregacji danych geoprzestrzennych, typy hierarchii, zastosowanie oprogramowania narzędziowego do opracowywania i rozwijania modeli. Oddzielnym istotnym zagadnieniem opisanym w tym rozdziale jest pojęcie usługa i powiązania tego pojęcia z innymi pojęciami stosowanymi w modelach opartych na metodykach obiektowych.

## 3.1. Rola standardów w projektowaniu i budowie infrastruktur

Standardy (i specyfikacje) w informatyce są warunkiem koniecznym dla działania systemów i w tym także dla ich współdziałania – interoperacyjności. Ta zasada dotyczy także systemów geoinformacyjnych. Ponieważ informatyka nie zajmuje się aspektem geoprzestrzennym informacji, opracowanie standardów w tym zakresie jest zadaniem geomatyki. Obecnie zajmują się tym dwa międzynarodowe ośrodki – OGC (Open GIS Consortium) i Komitet Techniczny ISO/TC 211. Role tych organizacji w procesie standaryzacji są różne – OGC zajmuje się sprawami technologii i praktycznych implementacji, a ISO bardziej formalną i proceduralną stroną zatwierdzania ich jako oficjalne dokumenty.

**Czego dotyczą specyfikacje OpenGIS i standardy ISO 19100?**
Dotyczą one:

Interoperacyjności (interoperowalności, współdziałania) systemów geoinformacyjnych, czyli tego, co jest podstawą funkcjonowania infrastruktury. Uzyskuje się to przez:

- ujednolicone modele danych,
- zdefiniowane interfejsy,
- języki dostępu i manipulacji danymi w oparciu o te interfejsy,
- automatyczną translacja danych i modeli.

**Czego nie dotyczą specyfikacje OpenGIS i standardy ISO 19100?**

- Nie dotyczą natomiast aspektu tematycznego (dziedzinowego) geoinformacji.
- Nie dotyczą także innych aspektów (warstw przedstawionych w tabeli 1), jakie występują w aplikacjach sieciowych.

Infrastruktura geoinformacyjna to z punktu widzenia informatyki złożona aplikacja sieciowa. W tabeli 1 sześć dolnych warstw nie jest objętych standardami geomatyki – to jest domena informatyki. Jedynie najwyższa warstwa – aplikacyjna – może zawierać elementy, które wymagają oddzielnych standardów – w tym przypadku geomatycznych.

**Tabela1. Siedmiowarstwowy model ISO dla aplikacji sieciowych**

| Numer | Nazwa | Przeznaczenie warstwy |
|---|---|---|
| 7 | **Warstwa aplikacyjna** | **Określa jak dana aplikacja wykorzystuje sieć, np. telnet, ftp, WWW.** |
| 6 | Warstwa prezentacyjna | Sposób reprezentowania danych, np. XDR (eXternal Data Representation), login, password. |
| 5 | Warstwa sesji | Sposób ustanowienia komunikacji, np. RPC (Remote Procedure Call). |
| 4 | Warstwa transportu | Sposób przesyłania danych, np. TCP (Transmission Control Protocol). |
| 3 | Warstwa sieci | Sposoby przydzielania adresów i przesyłania pakietów, np. IP (Internet Protocol). |
| 2 | Warstwa wiązania danych | Sposób dzielenia danych na ramki i ich wysyłania, np. Ethernet. |
| 1 | Warstwa fizyczna | Podstawowe sieciowe urządzenia związane z rodzajem nośnika, np. kabel koncentryczny, skrętka, światłowód. |

**Co jest lepsze: standardy ISO 19100, czy też specyfikacje OpenGIS?**

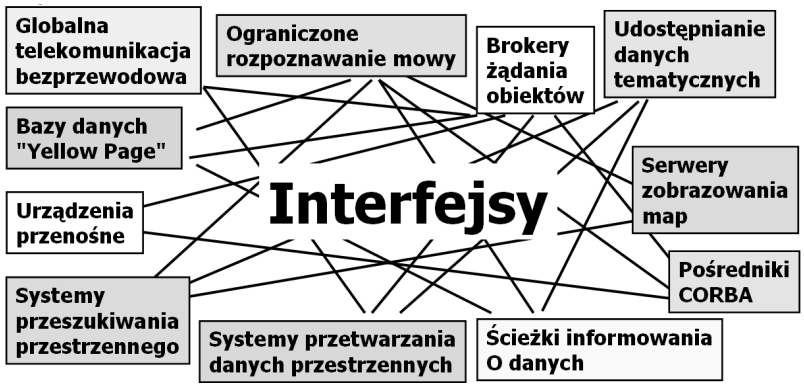
Nie ma między nimi znaczących różnic. Gdy kilka lat temu obie te organizacje rozpoczęły działalność oddzielnie, można było odnieść wrażenie, że wyniki ich działań są różne. Obecnie współpraca jest tak daleko posunięta, że rezultaty prac standaryzacyjnych w tym zakresie są wspólnym dziełem obu ośrodków – znaczna część standardów ISO 19100 to adaptacje specyfikacji OpenGIS.

## 3.2. Podstawowe pojęcia – struktura danych, interfejs i usługa

W raportach grup ekspertów inicjatywy INSPIRE kluczową role w architekturze infrastruktury mają usługi zdefiniowane tu w rozdziale 2.1.1. Pojęcie usługi ma tam charakter funkcjonalny (organizacyjny), to znaczy odnosi się do możliwości wykonywania zadań (świad-

czeń) przez jedną jednostkę (system) dla drugiej (innego systemu lub człowieka). Dla realizacji usługi jest konieczny interfejs, czyli zdefiniowane połączenie tych jednostek. Interfejs to pojęcie bardziej techniczne niż usługa i wiąże się z nim protokół, język komunikacji i określone zasoby techniczne systemu.

Ze względu na dużą różnorodność usług w zakresie geoinformacji, a także różnorodność jednostek uczestniczących w tych usługach, potrzebnych jest wiele różnych interfejsów. Złożoność tą ilustruje rysunek 18.




**Rys. 18.** Interfejsy łączą systemy pełniące różne role w infrastrukturze geoinformacyjnej. Różnorodność tych połączeń wymaga opracowania wielu typów interfejsów. [Źródło: Archiwum OGC]

Protokół związany z danym interfejsem określa między innymi, jakie operacje i na jakich danych mogą być wykonywane w ramach określonej usługi. Ponieważ przedmiotem usług są dane (lub metadane, które też są danymi), specyfikacja protokółu powinna definiować model struktur tych danych lub się odwoływać do jakiegoś modelu zewnętrznego. Z tego względu w standardach geomatycznych punktem wyjścia dla interoperacyjności opartej na usługach i interfejsach są modele danych geoprzestrzennych.

Przy rozpatrywaniu zagadnień infrastruktury, na określony system wchodzący w jej skład patrzy się „z zewnątrz" i w takim przypadku widziany jest tylko jego interfejs, a wewnętrzna organizacja i reguły funkcjonowania są ukryte – taka sytuację informatyka nazywa „hermetyzacją". Wewnętrzny model danych może być ( i często jest) inny niż model danych zdefiniowany w specyfikacji interfejsu. Ponadto, dany system może mieć wiele interfejsów i każdy z nich może posługiwać się innym modelem danych w zależności, jakiego typu usługa jest z nim związana. Istotnym problemem, jaki tu występuje to konwersja danych pomiędzy różnymi ich modelami.

Reguły funkcjonowania interfejsu określone w jego specyfikacji, tak jak inne składniki infrastruktury, są oparte na określonych standardach dotyczących różnych rozwiązań sposobu współdziałania systemów. W tym przypadku stosuje się rozwiązania ogólno-informatyczne z uwzględnieniem specyficznych wymagań geoinformacji. Do najczęściej stosowanych w tym zakresie platform i środowisk interoperacyjnych należą technologie: WWW (HTTP), CORBA, DCOM, SQL i XML.

## 3.3. Modele pojęciowe dotyczące geoinformacji

Modelowanie pojęciowe jest obecnie podstawowym narzędziem projektowania i bodowy systemów informatycznych. Stosuje się je we wszystkich etapach tego procesu:

- modele ontologiczne – jako sposób sformalizowanego zapisu naszych wyobrażeń o rzeczywistości,
- modele ogólne i abstrakcyjne – w trakcie opracowywania koncepcji zapisu informacji dotyczącej rzeczywistości w systemach komputerowych,
- modele implementacyjne – na etapie przystosowywania modelu abstrakcyjnego do wybranego środowiska (platformy) implementacyjnej związanej z wybraną technologią,
- modele aplikacyjne – w zastosowaniu modeli abstrakcyjnych, ogólnych i implementacyjnych do konkretnych praktycznych zagadnień.

Powyższe uwagi w pełni odnoszą się także do informacji geoprzestrzennej. Rozdział ten przedstawia podstawowe reguły modelowania pojęciowego w zakresie geoinformacji opare na doświadczeniach i dokumentach OGC i ISO/TC 211.

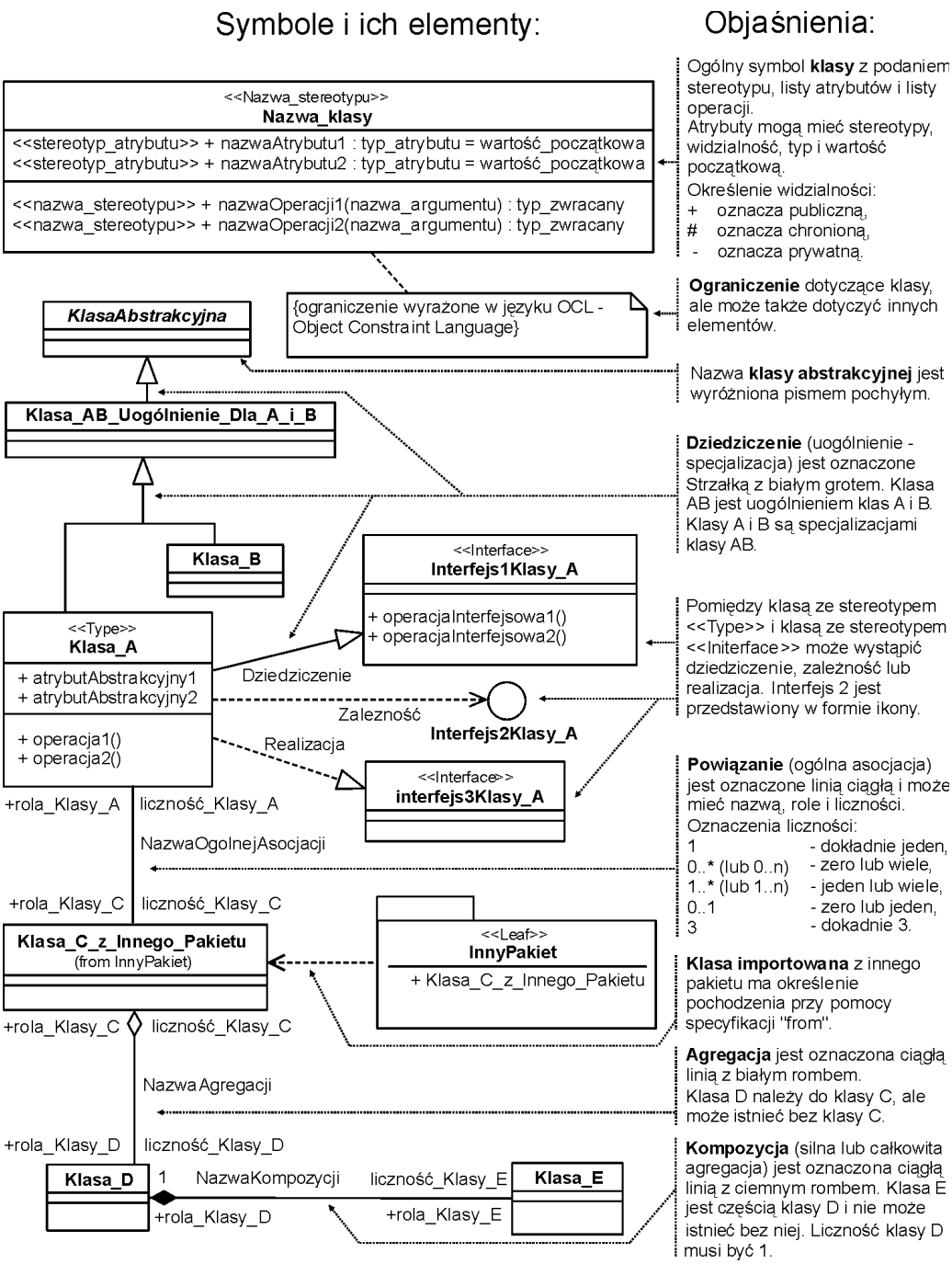
### 3.3.1. Język UML i jego profil dla geomatyki

Podstawę języka UML stanowi osiem rodzajów diagramów graficznych:

- Diagram przypadków użycia – opisują one funkcjonowanie systemu z punktu widzenia użytkownika tego systemu.
- Diagramy klas i pakietów – ich zadaniem jest opisanie struktury danych, związanych z nimi interfejsów i innych komponentów obiektowych projektowanego systemu. Większość diagramów UML przedstawionych w tej pracy są diagramami klas.
- Diagramy sekwencji – opisują cechy dynamiczne w zakresie kolejności przesyłania komunikatów pomiędzy klasami.
- Diagramy kolaboracji (diagramy kooperacji) – również opisują cechy dynamiczne, ale w zakresie sekwencji z uwzględnieniem statycznej struktury obiektów.
- Diagramy stanów – określają cechy dynamiczne w rozłożeniu na poszczególne stany i przejścia pomiędzy nimi.
- Diagramy aktywności – opisują dynamiczny przepływ sterowania z uwzględnieniem równoległości procesów.
- Diagramy komponentów – podają implementacyjne rozłożenie komponentów systemu.
- Diagramy rozproszenia – opisują przestrzenne rozproszenie składników systemu.

Język UML i związana z nim metodyka może być w pełni wykorzystany jedynie przy pomocy odpowiedniego oprogramowania narzędziowego, które pozwala na:

- opracowywanie diagramów zgodnie z regułami tego języka,
- automatyczną bieżącą weryfikację ich poprawności,
- wiązanie poszczególnych elementów występujących w różnych diagramach tego samego typu i pomiędzy różnymi typami,
- łączenie diagramów w jeden model,
- zapis tego modelu w pliku dla przechowania lub przekazania,
- dokumentowanie poszczególnych części modelu,
- konwersję modelu zapisanego w języku UML do różnych języków i platform aplikacyjnych,
- inżynierię odwrotną, czyli konwersję z tych języków i platform do języka UML.



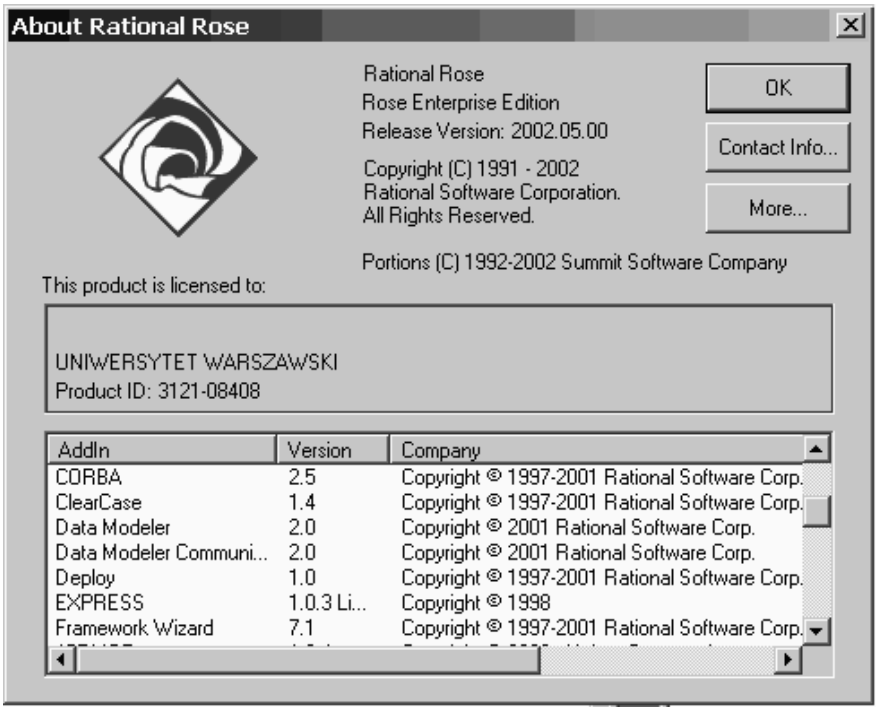

**Rys. 19.** Podstawowe elementy notacji graficznej diagramów klas języka UML stosowane w geomatyce. [Na podstawie opracowań OGC i ISO/TC 211]

Z uwag przedstawionych we wstępie do rozdziału 3 wynika, że znaczenie języka UML w dzisiejszej geomatyce jest zasadnicze. Prawie wszystkie obecnie opracowywane dokumenty standaryzacyjne posługują się nim jako podstawową metodą opisu przedstawianej w nich problematyki. Z tego względu znajomość tego języka jest niezbędna dla zrozumienia ich treści. Rysunek 19 przedstawia najważniejsze elementy notacji graficznej diagramów klas UML. Jednak sam zapis graficzny przy pomocy diagramów nie stanowi pełnego opisu modelu. Wiele szczegółów modelu nie ma reprezentacji graficznej lub nieobecność symboli zakłada, że mają one wartości domyślne.

### 3.3.2. Programy narzędziowe dla języka UML (Rational Rose)

Małe i proste modele w języku UML mogą być opracowywane „ręcznie na papierze", lecz w przypadku modeli dużych i skomplikowanych jest to praktycznie niemożliwe, ponieważ sprawdzenie poprawności i niesprzeczności tysięcy elementów i ich powiązań przerasta możliwości takiego sposobu. Z tego powodu powszechnie są stosowane do tego celu programy narzędziowe i pierwsze miejsce w śród nich zajmuje program Rational Rose (rys. 20) firmy Rational, będącej obecnie częścią koncernu IBM. Podstawowe cechy tego programu są nastepujące:

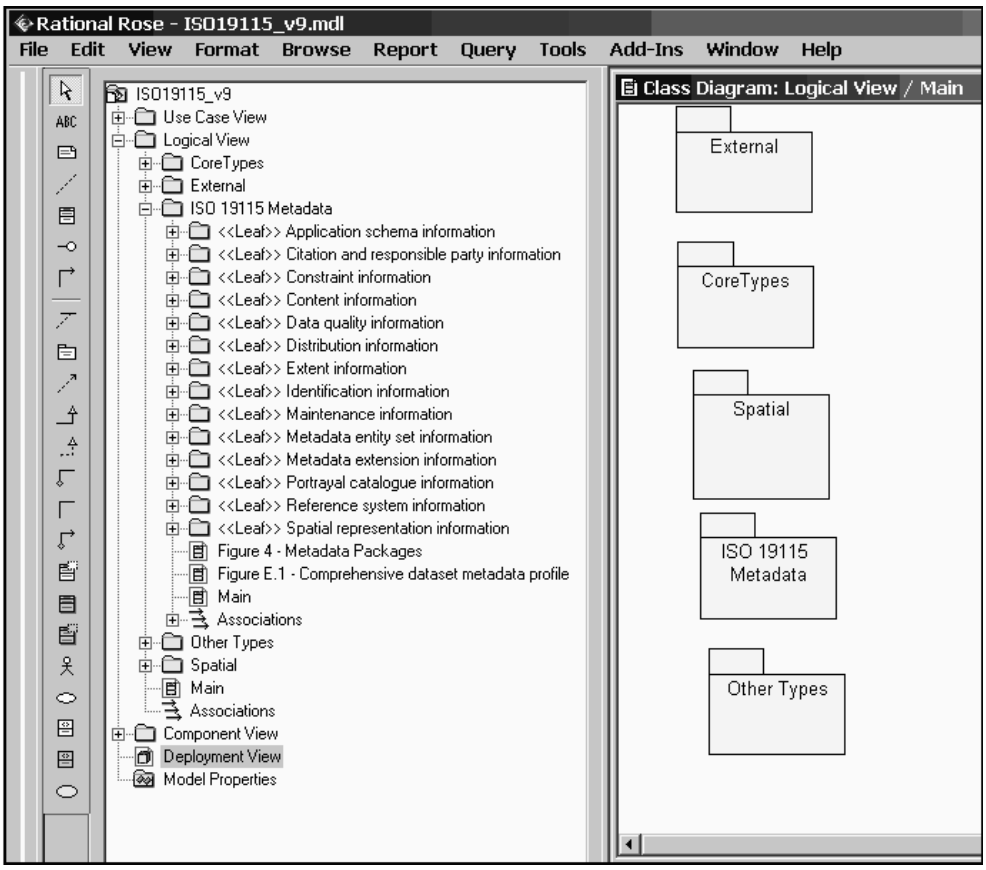
- dynamiczna weryfikacja poprawności modelu złożonego z wielu pakietów;
- określanie elementów modelu, atrybutów i powiązań tych elementów poprzez wybór z listy, w zależności od ustalonego typu modelu i/lub środowiska implementacyjnego;




**Rys. 20.** Okno informacyjne programu narzędziowego Rational Rose Enterprise 2002 z dodatkowymi modułami rozszerzeń implementacyjnych.

- możliwość definiowania własnych typów, stereotypów i elementów w ramach specyfikacji języka UML dla potrzeb aplikacyjnych,
- możliwość automatyzowania procesu budowy lub konwersji modelu przy pomocy języka skryptowego;
- możliwość konwersji modeli do wielu środowisk implementacyjnych, między innymi do: CORBA, C++, EXPRESS, ODQL, Oracle, COM i XML.

Skomplikowany model może się składać z setek oddzielnych diagramów zawierających tysiące klas. Istotny problem stanowi wzajemna zgodność poszczególnych diagramów cząstkowych. Z tego względu (zgodnie z regułami języka UML) w programie tym model jest dzielony na hierarchiczną strukturę pakietów, w których (najczęściej w pakietach należących do najniższego poziomu i ze stereotypem <<Leaf>>) umieszczane są klasy. Przykład zorganizowania modelu w pakiety przedstawia rysunek 21.

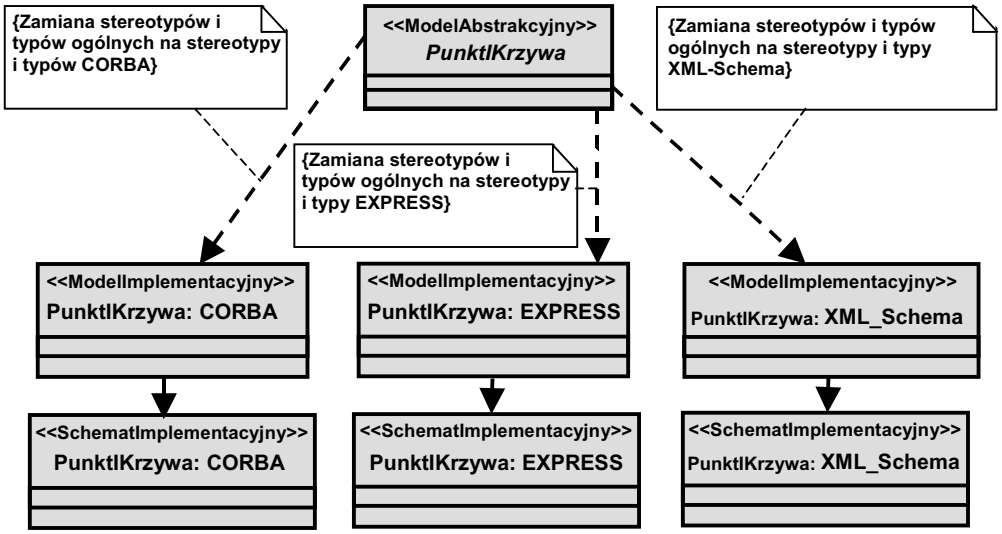



**Rys. 21.** Fragment okna programu Rational Rose przedstawiający pakiety modelu metadanych zgodnego ze standardem ISO 19115. Po stronie prawej widoczne są symbole pakietów wyższego poziomu. Po stronie lewej w oknie katalogowym jest widoczna lista pakietów najniższego poziomu należących do pakietu „ISO 19115 Metadata".

### 3.3.3. Modele abstrakcyjne i implementacyjne

Model implementacyjny to model dostosowany do potrzeb określonego środowiska realizacji systemu geoinformatycznego. Najczęściej stosowanymi środowiskami w geomatyce są: Java, CORBA, C++, ODQL, SQL, COM/DCOM , XML DTD i XML Schema.

Model abstrakcyjny stanowi podstawę dla opracowania modeli implementacyjnych, przy pomocy których są generowane schematy implementacyjne. W modelu implementacyjnym stosuje się stereotypy i typy zgodne ze specyfikacją określonej implementacji. W modelu abstrakcyjnym stosuje się wyłącznie stereotypy określone w specyfikacji języka UML lub inne ich odpowiedniki (ogólne) abstrakcyjne. Przejście z modelu abstrakcyjnego do modelu implementacyjnego przedstawia rysunek 22.



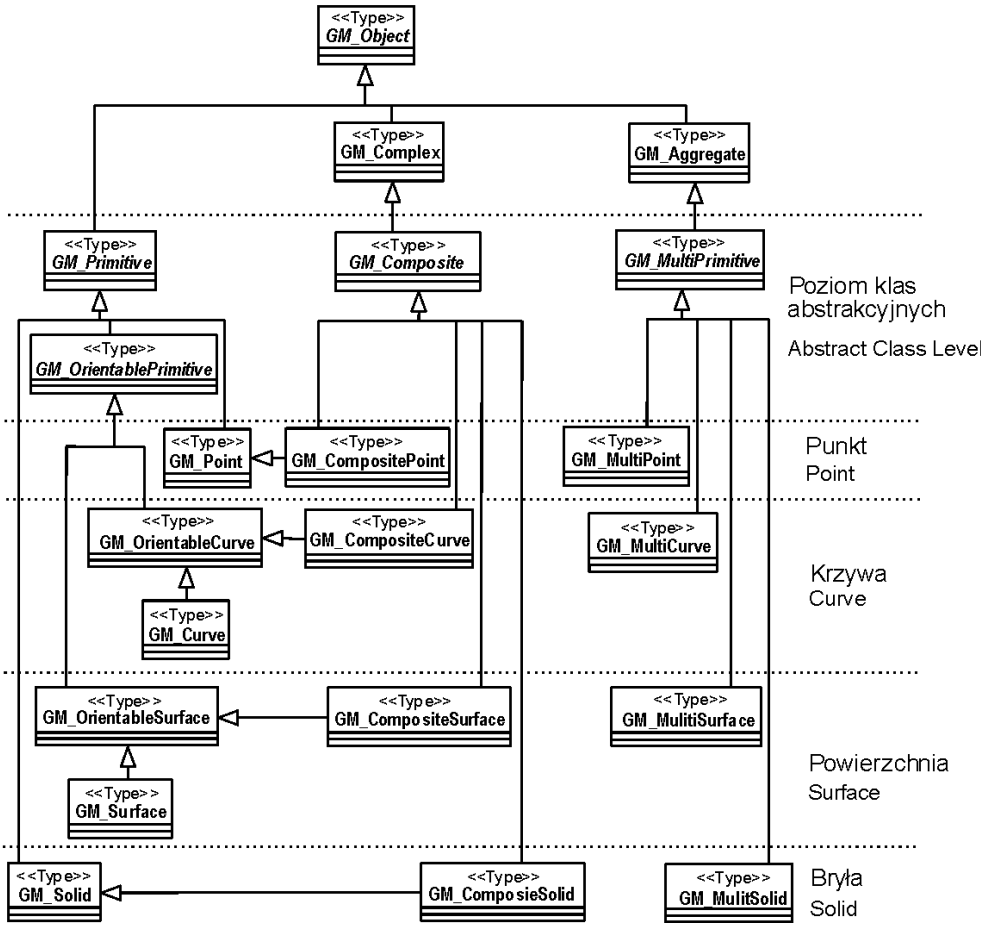

**Rys. 22.** Relacje pomiędzy modelem abstrakcyjnym i odpowiadającymi mu modelami implementacyjnymi (dla platformy CORBA, schematu XML i języka EXPRESS).

**Przykłady modeli abstrakcyjnych**

Rysunki 23 do 27 przedstawiają przykłady diagramów klas modeli abstrakcyjnych zdefiniowanych w specyfikacjach standardów grupy ISO 19100. Wszystkie modele zawarte w tych standardach (z wyjątkiem przykładów dotyczących implementacji i aplikacji) są modelami abstrakcyjnymi. Tu trzeba podkreślić różnicę pomiędzy modelem abstrakcyjnym a abstrakcyjnym elementem modelu, na przykład abstrakcyjną klasą, która nie może mieć swoich wystąpień, to znaczy obiektów należących do tej klasy.

Rysunek 23 jest przykładem diagramu klas modelu abstrakcyjnego, ponieważ użyte w nim stereotypy należą wyłącznie do języka UML (stereotyp <<Type>>). Diagram ten jest jednocześnie ogólny, ponieważ są tam określone jedynie nazwy klas bez elementów, jakie te klasy zawierają i w rezultacie nie można ustalić, co dziedziczą klasy pochodne od klas pierwotnych.

Na rysunku 24 hierarchia klas jest wyprowadzana z klasy bazowej, która jest klasą abstrakcyjną – nie może ona posiadać własnych obiektów. Taka klasa służy jedynie do wyprowadzania z niej innych klas.

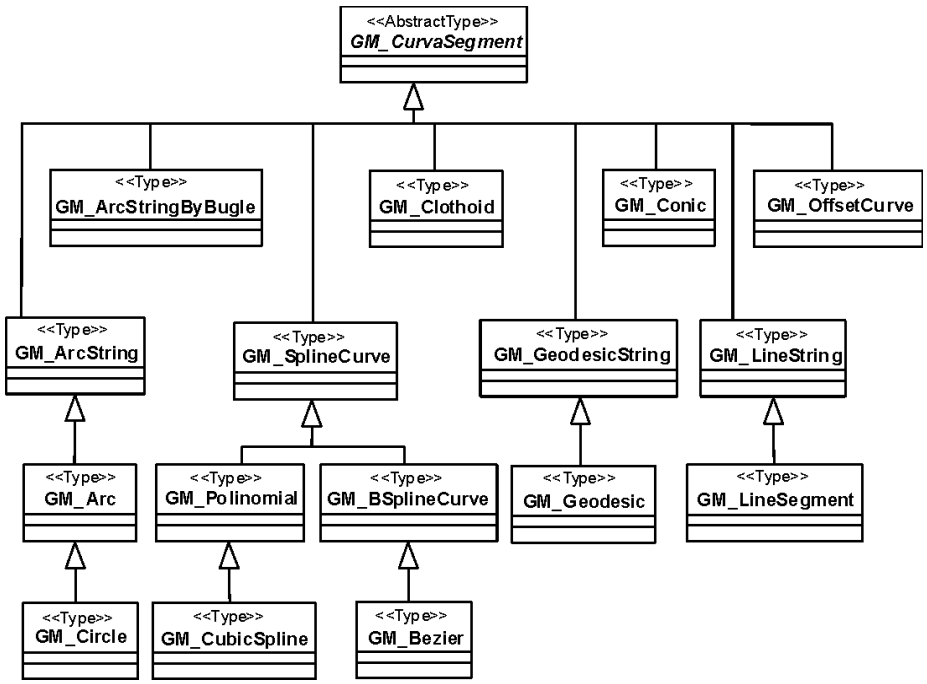



**Rys. 23.** Hierarchia klas definiujących typy geometrii przestrzeni. (diagram opracowany na podstawie ISO 19107 programem narzędziowym Rational Rose)

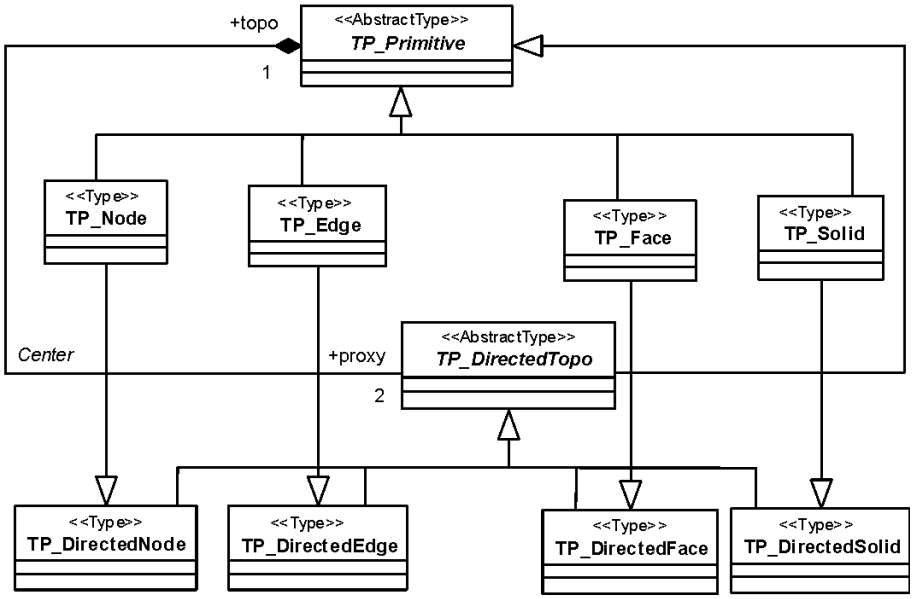
Rysunek 25 przedstawia problem podwójnego dziedziczenia – klasa pochodna jest wyprowadzona jednocześnie z dwóch klas pierwotnych. Podwójne dziedziczenie prowadzi do wielu komplikacji polegających na niejednoznaczności składu elementów zawartych w klasie pochodnej w wyniku dziedziczenia. Z tego względu wiele implementacyjnych środowisk obiektowych nie dopuszcza do wielokrotnego dziedziczenia.

Rysunek 26 jest przykładem użycia ograniczenia w OCL (Object Constrain Language). Jeżeli język UML jest niewystarczający do pełnego zdefiniowania modelu to specyfikacja tego języka zaleca stosowanie wyrażeń w OCL dla określenia dodatkowych warunków i ograniczeń.

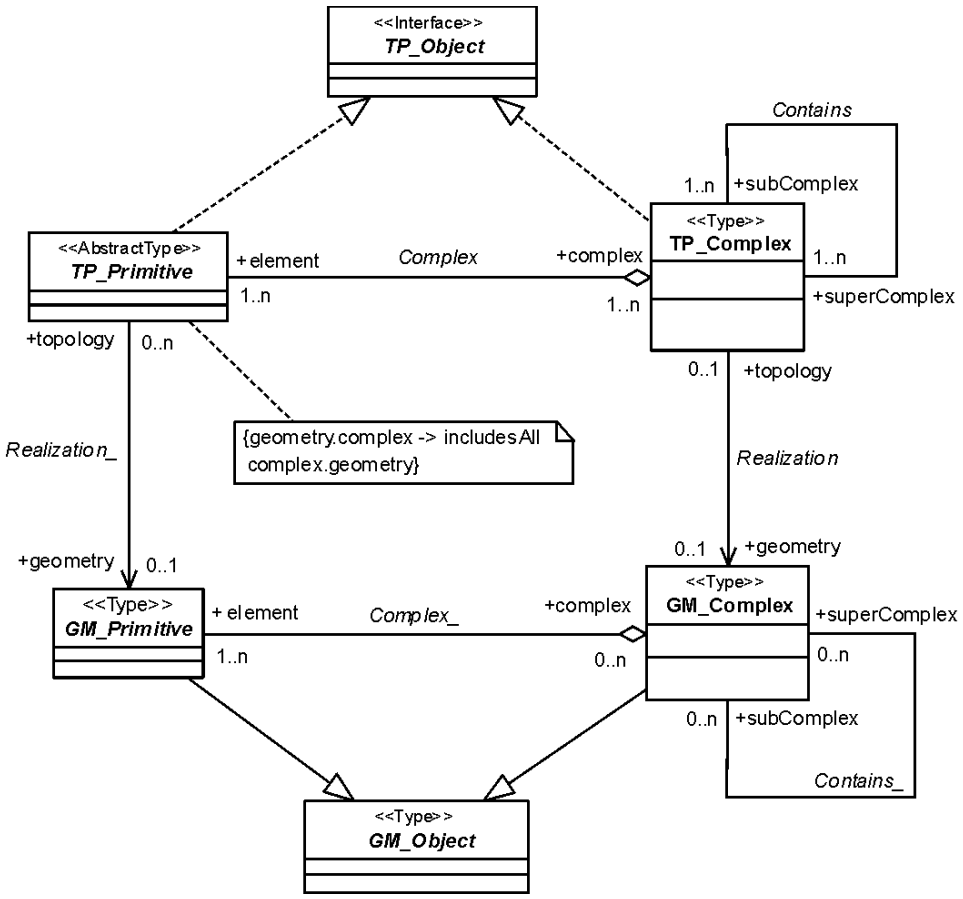
Diagram przedstawiony na rysunku 27 zawiera dwie klasy interfejsowe określające, jakie operacje mogą być wykonywane na klasach związanych z tymi interfejsami. Klasy interfejsowe mogą posłużyć w modelu implementacyjnym do zdefiniowania operacji związanych z usługami. Diagram ten zawiera także klasę, która określa, jakie wartości może przybierać jeden z argumentów innej klasy tego diagramu (Enumeration).




**Rys. 24.** Diagram klas przedstawiający typy krzywych zdefiniowanych w standardzie ISO. Wszystkie te typy są pochodne od abstrakcyjnego typu *GM_CurvaSegment.* (diagram opracowany na podstawie ISO 19107 programem Rational Rose)




**Rys. 25.** „Podwójna" hierarchia obiektów definiujących typy elementów topologii przestrzeni (diagram opracowany na podstawie ISO 19107 programem Rational Rose).

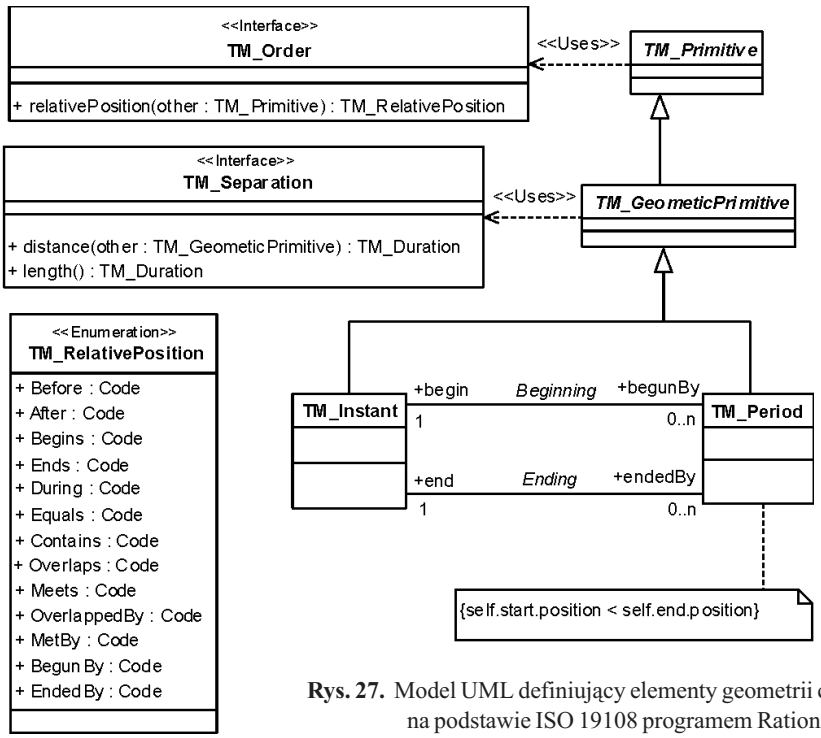



**Rys. 26.** Model UML definiujący powiązania pomiędzy elementami geometrii i topologii przestrzeni (opracowany na podstawie ISO 19107 programem Rational Rose)

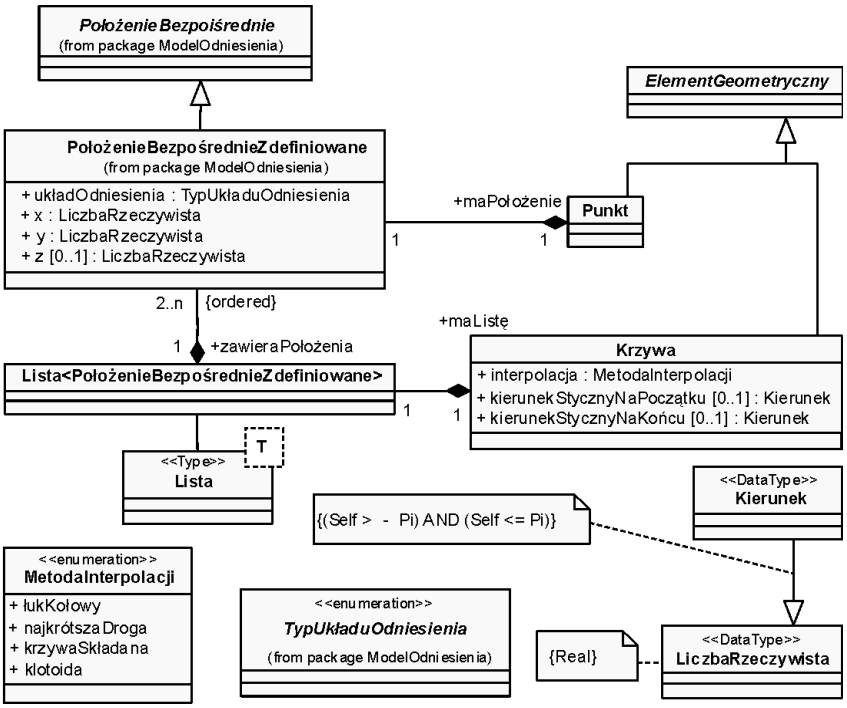
**Porównanie modelu abstrakcyjnego z implementacyjnym na przykładzie języka EXPRESS**

Podstawowa różnica pomiędzy modelem abstrakcyjnym i implementacyjnym polega na zastosowaniu w tym pierwszym stereotypów abstrakcyjnych, domyślnych lub własnych języka UML (na przykład: <<DataType>> i <<Enumeration>> lub domyślny <<Class>>). W drugim przypadku stosuje się stereotypy odpowiednie dla tej implementacji (na przykład: <<EXPRESS Entity>> i <<EXPRESS Type Declatation>>).

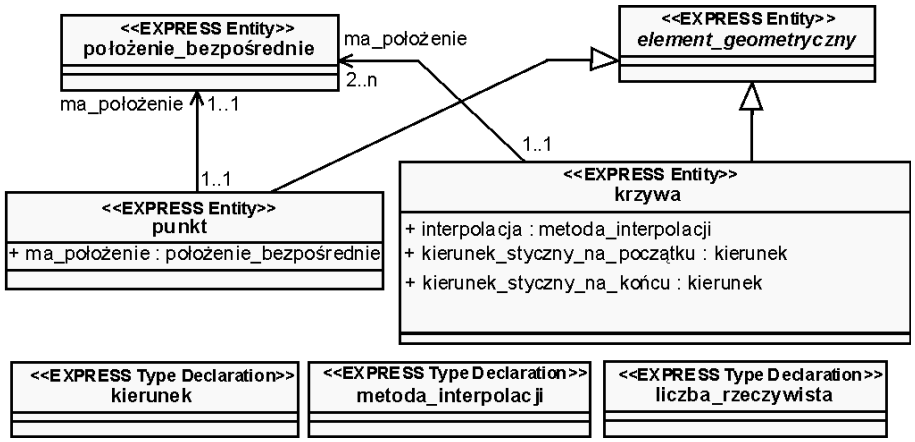
Przedstawiony tu przykład nie dotyczy standardów ISO i jest związany z krajowymi rozwiązaniami z zakresu tej problematyki. Rysunek 28 zawiera prosty model abstrakcyjny dla elementów geometrycznych punkt i krzywa opracowany zgodnie z zasadami określonymi w standardzie ISO 19103. Na rysunku 29 jest przedstawiony odpowiadający (w ograniczonym stopniu) mu model w konwencji reguł języka EXPRESS spełniający wymagania dawnych standardów europejskich wyrażonych w pre-normach komitetu CEN/TC 287.




**Rys. 27.** Model UML definiujący elementy geometrii czasu (opracowana na podstawie ISO 19108 programem Rational Rose)




**Rys. 28.** Przykład prostego modelu abstrakcyjnego dla punktu i krzywej – niezależnego od określonej platformy implementacyjnej.




**Rys. 29.** Przykłady modelu implementacyjnego odpowiadającego modelowi abstrakcyjnemu z rys. 28 i przeznaczonego dla języka EXPRESS związanego z relacyjnymi bazami danych.

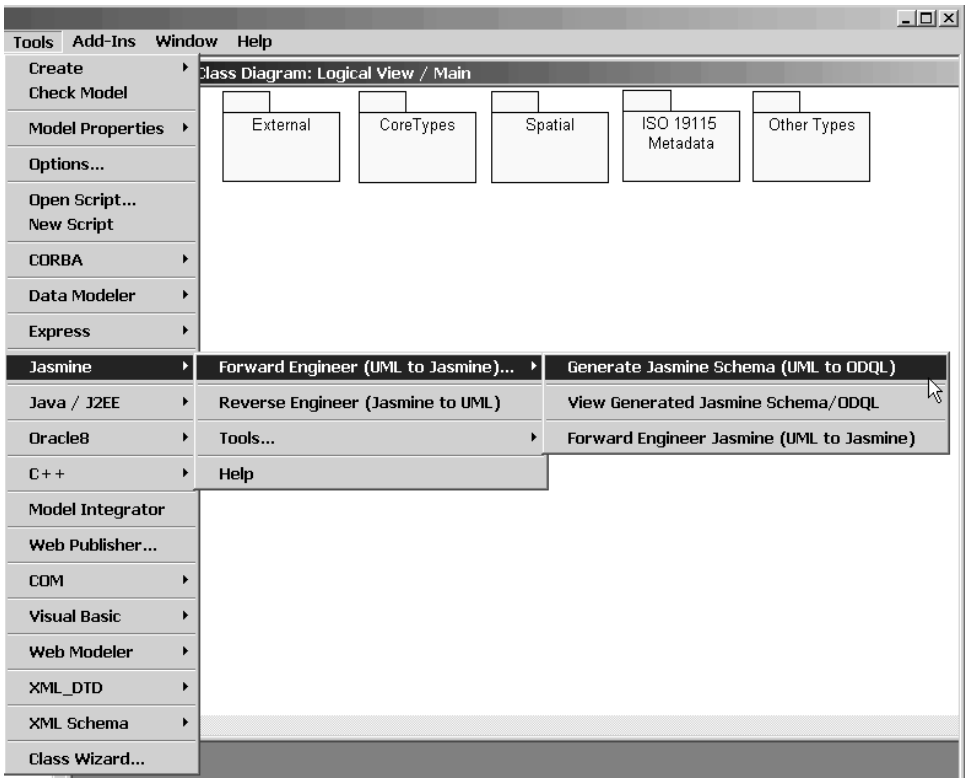
### 3.3.4. Konwersja modeli pojęciowych do języków opisu struktur danych

Metody konwersji modeli do języków opisu struktur danych należą obecnie do najintensywniej rozwijanych w informatyce. Są one szczególnie ważne w sytuacji, gdy buduje się na wielką skalę liczne nowe systemy rozproszone, często przy z góry założonej ich heterogeniczności. W takich warunkach stosowane do tego technologie wymagają, aby konwersja była dokonywana automatycznie. Program Rational Rose jest w stanie sprostać tym potrzebom jedynie częściowo, a przyczyna leży w istotnych różnicach pojęciowych pomiędzy środowiskami i językami implementacyjnymi, do których dokonywana jest konwersja.

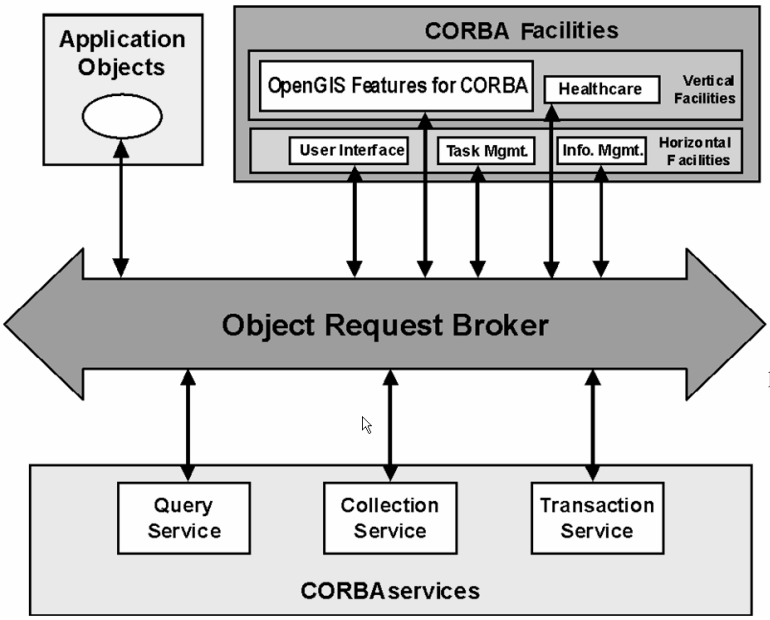
Rysunek 30 przedstawia jeden z prostszych przypadków takiej konwersji – z języka UML do języka ODQL *(Object Definition and Query Language)*. Zadanie to jest stosunkowo łatwe, ponieważ oba te języki oparte są na bardzo podobnym paradygmacie obiektowości. Stosunkowo łatwa jest również konwersja do platformy przetwarzania rozproszonego CORBA *(Common Object Request Broker Architecture)*. OGC opracowało specyfikację implementacyjną dla tej platformy w zakresie prostych wyróżnień geoprzestrzennych. Schematyczną strukturę elementów platformy CORBA dla geoinformacji przedstawia rysunek 31. Rysunek 32 pokazuje przebieg tej konwersji – w oknie katalogowym (po lewej stronie) widoczne są klasy modelu ze stereotypem <<CORBAStruct>>. Po prawej stronie jest widoczny komponent o nazwie „Baza_CORBA", do którego należą te klasy.

Nie wszystkie środowiska implementacyjne są jednakowo zgodne z językiem UML. Dla szeregu z nich zadanie konwersji jest trudne ze względu na różnice paradygmatu obiektowości, na których te środowiska są oparte. Dotyczy to szczególnie tych przypadków, gdzie obiekowość jest ograniczona ze względu na technologiczne podstawy, na przykład w systemach obiektowo-relacyjnych baz danych. Do trudnych przypadków należy także konwersja pomiędzy UML a XML Schema, jednak tu przyczyną jest stosunkowo krótki okres czasu od upowszechnienia się metody zapisu struktury dokumentu XML przy pomocy schematów. Obecnie są rozwijane metody tej konwersji za pośrednictwem XMI (rozdz. 3.4):

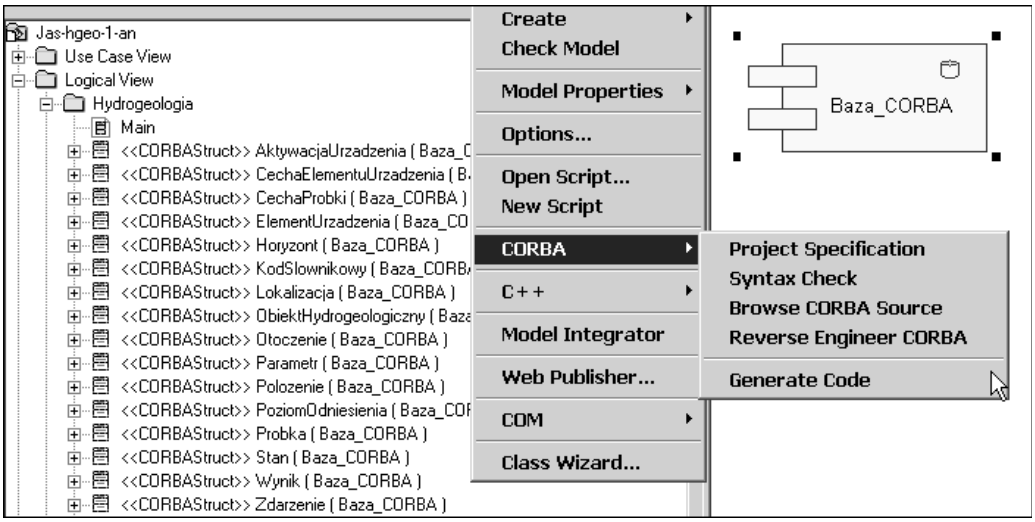
model w UML ↔ XMI ↔ XML Schema




**Rys. 30.** Fragment okna programu Rational Rose przedstawiające konwersję modelu (ISO 19115 – metadane) do schematu platformy ODQL dla obiektowych baz danych (w tym przypadku dla bazy Jasmine).




**Rys. 31.** Schematyczne przedstawienie elementów platformy implementacyjnej CORBA z uwzględnieniem składników dotyczących wyróżnień geoprzestrzennych zdefiniowanych w specyfikacji OpenGIS. [Źródło: Archiwum OGC]




**Rys. 32.** Fragment okna programu Rational Rose przedstawiające konwersję modelu implementacyjnego do schematu platformy CORBA.

### 3.3.5. Modele ogólne i aplikacyjne (dziedzinowe lub tematyczne)

Obok podziału na modele abstrakcyjne i implementacyjne, inny istotny podział modeli pojęciowych to modele ogólne i aplikacyjne.

W geomatyce model ogólny to model nie związany z określoną dziedziną zastosowań geoinformacji – modele takie można nazwać ogólnogeomatycznymi. W modelach tych, jak to widać na przykładach opartych na standardach ISO 19100, definiowana jest geometria, lokalizacja i topologia, a elementy związane z aspektem tematycznym są pozostawione do uwzględnienia w wyprowadzanych z nich modelach aplikacyjnych. Z tego powodu modele ogólne w zasadzie nie maja praktycznego zastosowania, ponieważ geoinformacja bez części tematycznej jest bezużyteczna – jaki jest praktyczny pożytek z przebiegu linii, jeżeli nie wiemy czy jest to droga, czy rzeka.

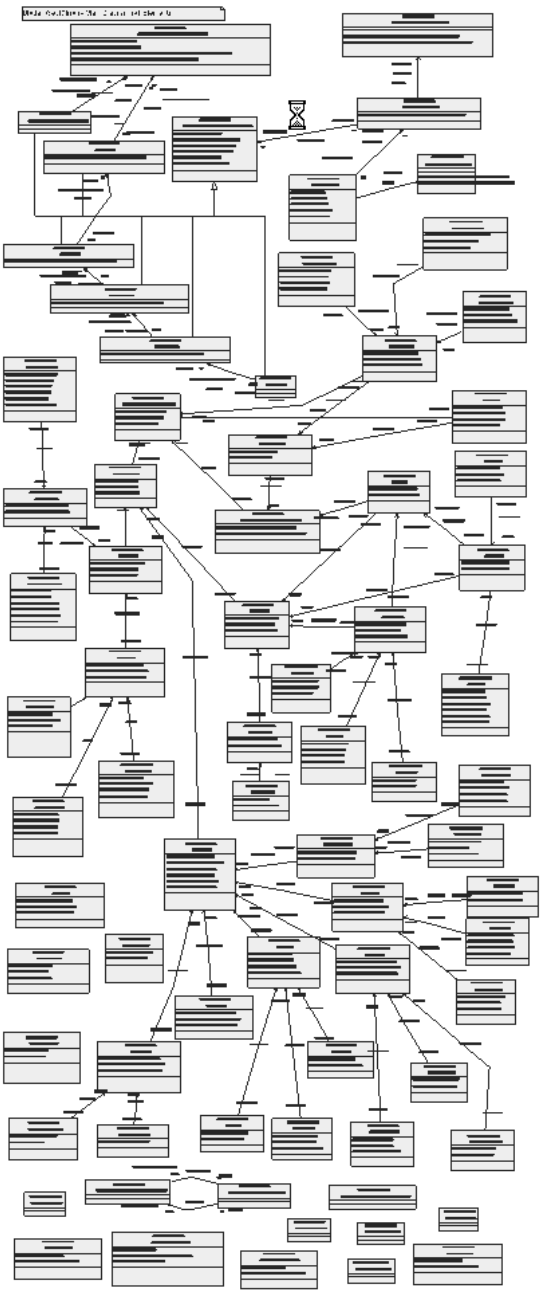
Z powyższego wynika, że model aplikacyjny to rozszerzenie modelu ogólnego o te elementy, które są różne w różnych zastosowaniach. Nawiązując do pojęć generalizacji i specjalizacji (dziedziczenia) przedstawionych w rozdziale 2.2, można powiedzieć, że modele aplikacyjne są opracowywane przez specjalizację (dziedziczenie) z modeli ogólnych. Jednak trzeba tu odróżnić dziedziczenie odnoszące się do całego modelu od dziedziczenia dotyczącego poszczególnych klas wchodzących w jego skład. Model aplikacyjny powstaje przez rozszerzenie modelu ogólnego (przez wzbogacenie go o nowe elementy). To rozszerzenie może być dokonane w różny sposób, jednym z nich jest tworzenie nowych klas składowych przez wyprowadzanie ich (dziedziczenie) z istniejących klas zawartych w modelu ogólnym. Inny sposób polega na utworzeniu oddzielnej hierarchii klas związanych z daną dziedzina i umieszczaniu w tych nowych klasach klas już istniejących i pochodzących z modelu ogólnego. W takim przypadku klasy modelu ogólnego są składnikami (komponentami) nowych klas modelu aplikacyjnego. Trzecim sposobem może być mieszanie obu metod – „wyprowadzania z ogólnych klas" i „zawierania ogólnych klas".

Przykładami modeli ogólnych z zakresu geomatyki są wszystkie modele ogólnogeomatyczne zawarte w standardach, a w tym przedstawione na rys. 23 do 27.

Przykłady modeli aplikacyjnych zawierają rysunki 11, 15 i 32.

### 3.3.6. Stopień złożoności modeli i harmonizacja diagramów

Model zawierający wiele elementów, ze względów praktycznych, dzielony jest na szereg odrębnych diagramów. Jeżeli pomiędzy klasami występującymi na tych diagramach są jakieś powiązania to w domyśle powstaje jeden olbrzymi diagram, który najczęściej jest zbyt skomplikowany, aby można było go analizować, poprawiać lub rozwijać. Przykład takiego diagramu przedstawia rysunek 33. Program Rational Rose przechowuje diagram całościowy pod nazwą main i każda zmiana dokonana w diagramie cząstkowym jest dynamicznie uwzględniana w diagramie całościowym.



Inny problem występuje w przypadku, gdy model całościowy ma powstać z połączenia modeli cząstkowych opracowanych oddzielnie, na przykład przez różnych autorów, różne zespoły, różne ośrodki lub w różnych okresach opracowywania projektu systemu. Łączenie modeli cząstkowych jest trudniejszym zadaniem i wymaga procesu nazywanego harmonizacją modelu. Taki przypadek wystąpił przy opracowywaniu modelu zbiorczego wszystkich modeli zawartych w standardach grupy ISO 19100. Model zbiorczy zawiera ponad 300 skomplikowanych i ściśle ze sobą powiązanych modeli cząstkowych, w dodatku modele te były opracowywane w różnych zespołach rozproszonych po całym świecie. Proces harmonizacji tych modeli jeszcze się nie zakończył, ponieważ nie jest to tylko proste usunięcie błędów powodujących niezgodności, lecz często ma podłoże głębsze – różne podejście do sposobu widzenia geoinformacji. Można powiedzieć, że jest to problem semantyczny, którego podstawy leżą w ontologii geoinformacji.

**Rys. 33.** Przykłady skomplikowanego modelu klas w UML opracowanych programem narzędziowym Rational Rose.

# 3.4. Zapis modelu UML z zastosowaniem języka XML

W ostatnich latach język XML robi zawrotną karierę. Przyczynami tego są:

- jego prostota polegająca na umieszczaniu znaczników w zwykłym tekście,
- jego wyjątkowa elastyczność wynikająca w możliwości definiowania typów dokumentów dla zapisu niemal wszystkiego,
- jest językiem zrozumiałym zarówno przez człowieka jak i maszynę (w tym przypadku system informatyczny).

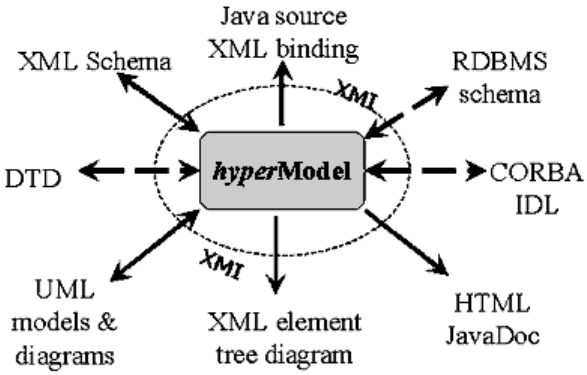
Z powyższych powodów język XML znalazł także zastosowanie do zapisu modeli pojęciowych. Aplikacja XML przeznaczona do tego celu to XMI *(XML Metadata Interchange)*. Tu trzeba wyjaśnić, że termin metadane ma szersze znaczenie, niż to, jakie mu się przypisuje – model pojęciowy opisujący strukturę danych to także „dane o danych" .

Wiele programów narzędziowych przeznaczonych do modelowania pojęciowego może generować i czytać zapisy tych modeli w języku XMI.

### 3.4.1. XMI – XML dla wymiany metadanych o modelach pojęciowych

Przedstawiony poniżej początkowy fragment dokumentu XML jest przykładem zapisu modelu pojęciowego UML w języku XMI. Dokument ten został wygenerowany przez program Rational Rose dzięki dodatkowemu rozszerzeniu „Unisys XMI Exporter" i zawiera zapis modelu porządkowego układu odniesienia czasowego opartego na standardzie ISO 19108:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- <!DOCTYPE XMI SYSTEM 'UMLX13-11.dtd' > -->
<XMI xmi.version="1.1" xmlns:UML="href://org.omg/UML/1.3"
  timestamp="Wed Sep 18 16:06:03 2002">
  <XMI.header>
      <XMI.documentation>
          <XMI.exporter>Unisys.JCR.1</XMI.exporter>
          <XMI.exporterVersion>1.3.4</XMI.exporterVersion>
      </XMI.documentation>
      <XMI.metamodel xmi.name="UML" xmi.version="1.3"/>
  </XMI.header>
  <XMI.content>
<!-- ==================== GeolChron    [Model] ======================== -->
      <UML:Model xmi.id="G.0" name="GeolChron" visibility="public"
                  isSpecification="false"isRoot="false" isLeaf="false" isAbstract="false">
          <UML:Namespace.ownedElement>
<!-- ========= GeolChron::Temporal Objects (ISO)    [Package] =========== -->
              <UML:Package xml:link="simple"
              href="GeolChron\Package.Temporal Objects (ISO).xml|S.260.1605.55.1"/>
<!-- ==================== Code    [DataType] ========================== -->
              <UML:DataType xmi.id="G.280" name="Code" visibility="public"
              isSpecification="false" isRoot="false" isLeaf="false" isAbstract="false"/>
<!-- ================== Integer    [DataType] ======================== -->
              <UML:DataType xmi.id="G.281" name="Integer" visibility="public"
              isSpecification="false" isRoot="false" isLeaf="false" isAbstract="false"/>
<!-- ==== GeolChron::Temporal Reference System (ISO)    [Package] ======== -->
              <UML:Package xml:link="simple"
              href="GeolChron\Package.Temporal Reference
          System (ISO).xml|S.260.1605.55.46"/>
<!-- ... -->
```




**Rys. 34.** Schemat ilustrujący możliwości konwersji za pośrednictwem XMI przy pomocy programu HyperModel modeli pojęciowych pomiędzy różnymi platformami implementacyjnymi i językiem UML.
[Źródło: (D. Carlson, 2001)]

*Przykład 3.*

Przedstawiony powyżej przykład pokazuje wiele szczegółowych elementów modelu, które najczęściej na diagramach graficznych nie są widoczne. Zapis modelu w XMI jest kompletny i szczegółowy – wprowadzenie go ponowne do programu Rational Rose przy pomocy operacji importu daje taki sam model, jaki był eksportowany – operacje te są odwracalne.

Język XMI stanowi ogniwo pośrednie pomiędzy modelami i schematami danych kilku najważniejszych platform implementacyjnych. Dzięki temu znalazł wiele zastosowań do przenoszenia między tymi środowiskami modeli pojęciowych danych. Rysunek 34 ilustruje te możliwości na przykładzie programu narzędziowego HyperModel.

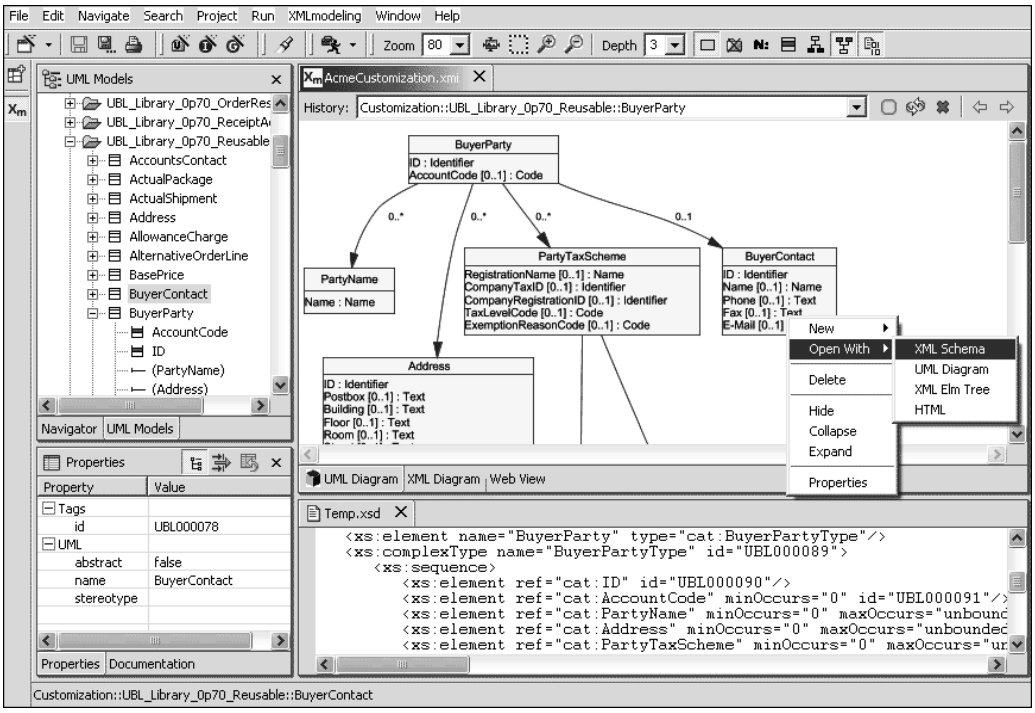
### 3.4.2. Program narzędziowy HyperModel

Program narzędziowy HyperModel firmy Ontogenics Corp. jest realizacją koncepcji pozwalającej przy pomocy języka XMI na dokonywanie konwersji modeli pojęciowych pomiędzy różnymi środowiskami. Właściwości programu narzędziowego HyperModel 1.2 (beta):

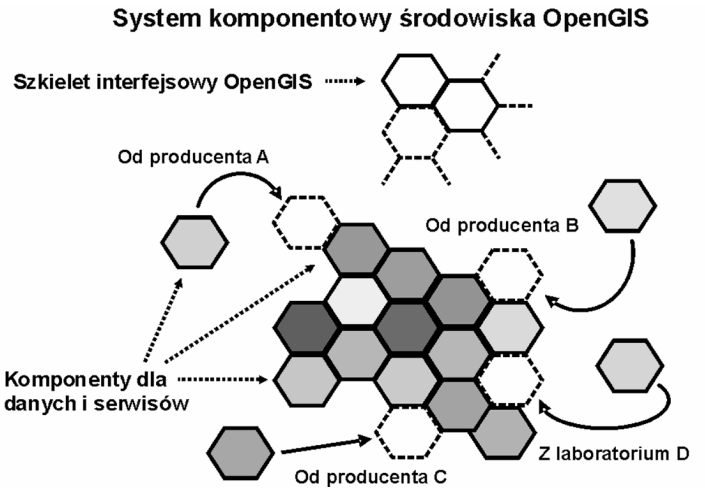
- importowanie schematów XML Schema do UML;
- generowanie schematów XML Schema z modeli UML;
- pełne dostosowanie projektu modelu UML do wymagań schematów XML Schema;
- tworzy dynamiczne i interaktywne diagramy UML;
- współpracuje z innymi narzędziami dla UML i XML:
- UML: Rational Rose, Gentleware Poseidon, ArgoUML, TogetherSoft, MagicDraw, Visio 2002 (tylko export) i szeregiem innych;
- XML: XML Spy.

## 3.5. Technologie komponentowe w geomatyce

Budowanie systemów wchodzących w skład infrastruktury geoinformacyjnej jest zadaniem bardzo złożonym, między innymi z powodu konieczności opracowywania wielu interfejsów dla wielu usług. Realizacja tego zadania stosując tradycyjne systemy monolityczne jest prawie niemożliwa. Z tego względu, jak to ilustruje rysunek 8 rozwój tych systemów zmierza do dzielenia ich na wiele modułów. Technologia komponentowa znacznie upraszcza to zadanie, ponieważ pozwala budować systemy z fragmentów pełniących określone zadania i zgodnych ze standardami związanymi z tymi zadaniami. Standaryzacja w tym przypadku pozwala, aby fragmenty te pochodziły od różnych producentów oprogramowania. Takie zunifikowane




**Rys. 35.** Okna programu HyperModel przedstawiające jego podstawowe możliwości: lewe górne – okno katalogowe modeli, lewe dolne – okno dokumentacji, prawe górne – okno diagramu klas, prawe dolne – okno edytora XML Schema. [Źródło: dokumentacja programu HyperModel]

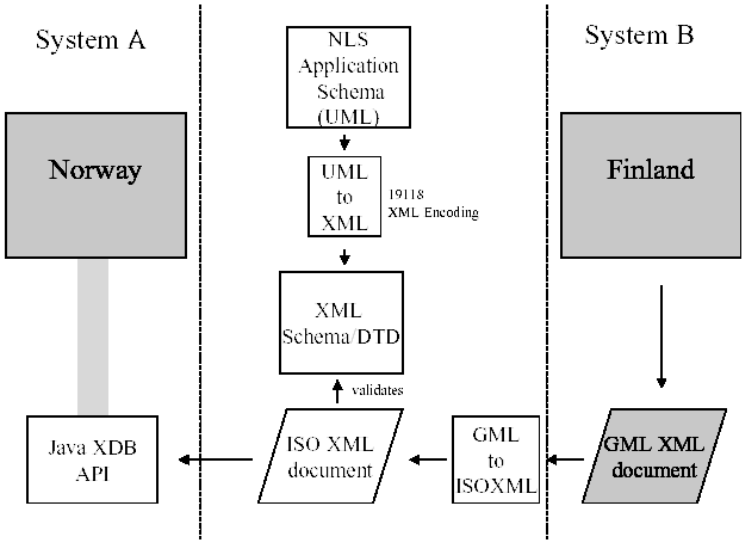



**Rys. 36.** Schemat systemu geoinformacyjnego zbudowanego z komponentów spełniających standardy OpenGIS. [Źródło: Archiwum OGC]

fragmenty nazywane są komponentami i między innymi mogą pełnić rolę standardowych interfejsów. Schemat budowy systemu komponentowego przedstawia rysunek 36.

## 3.6. Rola języka XML w interoperacyjności infrastruktury geoinformacynej

Rozdział 3.4 przedstawił w zarysie rosnące ostatnio znaczenie języka XML w systemach geoinformacyjnych. Tu można zwrócić uwagę na jego rolę w interoperacyjnej wymianie danych pomiędzy różnymi systemami wchodzącymi w skład infrastruktury geoinformacyjnej. Zapis danych geoprzestrzennych w XML, a ściślej w jego aplikacji – GML, jest opisany w rozdziale 5. Jednak sam „czysty" język GML nie wystarczy do komunikowania się systemów przy przesyłaniu bardzo różnorodnej tematycznie informacji geoprzestrzennej. W wielu dziedzinach posługujących się tą informacją stosowane są bardzo różne modele danych (przykłady zawiera rozdział 2.2.5). Aby móc w sposób dynamiczny określić, jaka jest struktura danych, które są w określonym przypadku przesyłane, trzeba znać model pojęciowy stanowiący podstawę definicji tej struktury. Rysunek 37 przedstawia schemat transmisji danych, których struktura jest dynamicznie określona na podstawie modelu zapisanego w UML powiązanego z tymi danymi. Od momentu, gdy model w języku UML zostaje przetransformowany na XML Schema, wszystkie zapisy (dokumentów) dotyczące danych i metadanych (zapis modelu) są dokonywane w języku XML.




**Rys. 37.** Schemat przedstawiający transmisję danych pomiędzy dwoma systemami infrastruktury geoinformacyjnej z dynamicznym określeniem struktury tych danych na podstawie modelu w języku UML powiązanego z tymi danymi. [Źródło: raporty „Grupy Nordyckiej"]