Tento projekt je financován se státní podporou
Technologické agentury ČR v rámci
Programu BETA2.
www.tacr.cz
Výzkum užitečný pro společnost.

MINISTERSTVO VNITRA
ČESKÉ REPUBLIKY

Rámcové vymezení objektů NaSaPO včetně referenčních atributů

Výsledek V03 – metodika (Nmet)

Konečný uživatel výsledků: **Ministerstvo vnitra České republiky**

Náměstí Hrdinů 1634/3

140 21 Praha 4 – Nusle

Název projektu: NaSaPO – Národní sada prostorových objektů

Číslo projektu: **TITIMV706**

Řešitel projektu: Západočeská univerzita v Plzni

Další řešitelé: Výzkumný ústav geodetický, topografický a kartografický, v. v. i.

GEOREAL, spol. s r. o.

T-MAPY, spol. s r. o.

Doba řešení: 1. 1. 2020 – 31. 12. 2021

Důvěrnost a dostupnost: veřejně nedostupný

Informace o autorském týmu:

Bc. Jonáš Častulík
(T-MAPY, spol. s r. o.)

Mgr. Jan Fišer
(T-MAPY, spol. s r. o.)

Mgr. Pavla Fraňková
(T-MAPY, spol. s r. o.)

Ing. Jana Waisová
(T-MAPY, spol. s r. o.)

Ing. Radek Makovec
(Výzkumný ústav geodetický, topografický a kartografický, v. v. i.)

RNDr. Eva Sovjáková
(Výzkumný ústav geodetický, topografický a kartografický, v. v. i.)

Ing. David Velhartický
(GEOREAL, spol. s r. o.)

Doc. Ing. Václav Čada, CSc.
(Západočeská univerzita v Plzni, Fakulta aplikovaných věd, katedra geomatiky)

Ing. Karel Janečka, Ph.D.
(Západočeská univerzita v Plzni, Fakulta aplikovaných věd, katedra geomatiky)

Další informace o projektu:

Zpracování terminologického a věcného vymezení objektů NaSaPO nejen pro účely projektu NaSaPO a podporu Strategie rozvoje prostorových informací v ČR do roku 2020 (GeoInfoStrategie).

Tento dokument vznikl v rámci řešení programu BETA2 projektu č. TITIMV706 s názvem „NaSaPO – Národní sada prostorových objektů“ s finanční podporou TA ČR.

Program veřejných zakázek v aplikovaném výzkumu a inovacích pro potřeby státní správy BETA2 byl schválen usnesením vlády České republiky č. 278 ze dne 30. 3. 2016 a je zaměřen na podporu aplikovaného výzkumu a inovací pro potřeby orgánů státní správy. Poskytovatelem finančních prostředků je Technologická agentura ČR.

Tento projekt je financován se státní podporou Technologické agentury ČR v rámci Programu BETA2.
www.tacr.cz
Výzkum užitečný pro společnost.

MINISTERSTVO VNITRA ČESKÉ REPUBLIKY

Obsah

Úvodní slovo **5**

Úvod **6**

1. Scope dokumentu **8**

2. Přehled použitých norem **9**

3. Seznam použitých pojmů a zkratek **10**

3.1 Termíny a definice NaSaPO 10

3.2 Základní termíny ISO norem geografické informace 10

3.3 Termíny a definice z norem – vzhled jevu 12

3.4 Termíny a definice z norem – vztahy a chování 13

3.5 Termíny a definice z norem – data 16

3.6 Termíny a definice z norem – jazyk a pojmy 17

4. Interoperabilita **19**

4.1 Heterogenita geografické informace 19

4.2 Přehled komponent interoperability pro projekt NaSaPO 21

5. Referenční model **22**

5.1 Konceptuální rámec referenčního modelu 23

5.2 Sémantický základ referenčního modelu 25

5.3 Syntaktický základ referenčního modelu 26

5.4 Servisní základ referenčního modelu 26

5.5 Procedurální základ referenčního modelu 27

6. Terminologie **28**

7. Pravidla pro aplikační schémata **28**

7.1 Jazyk konceptuálního schématu 32

7.1.1 Pravidla pro modelování v UML 33

7.2 Katalog vzhledů jevů 39

7.3 Obecný model vzhledu jevů 46

7.3.1 Struktura obecného modelu vzhledu jevů 46

7.3.2 Atributy typů vzhledu jevů 48

7.3.3 Vztahy mezi typy vzhledů jevů 50

7.4 Prostorové a časové aspekty vzhledu jevů 51

8. Identifikátory **52**

9. CityGML **52**
9.1 Application Domain Extension 53
9.2 Koncept prostorů 54
9.2.1 Koncept úrovně detailů 57
10. Závěr **59**
Příloha A. Příkladová studie aplikačních schémat NaSaPO **60**
Příloha B. Zpracování dat podle aplikačních schémat CityGML **73**
Literatura **78**

Úvodní slovo

Koncept je pojem, pod kterým volně rozumíme abstrakci (mentální konstrukci) pojatou lidmi, která zachycuje podstatu obvykle třídy jevů. Pojmem „koncept“ máme tedy na mysli abstrakci souboru podobných jevů do jednoho konceptu.

PŘÍKLAD: Pokud „povýšíme“ naše uvažování od konkrétního mostu tj. od konkrétního jevu, na zástupce množiny mostů, pak jsme „abstrahovali“ od konkrétního jevu do konkrétní koncepce mostu. Když „povýšíme“ naše úvahy ještě dále, například od těch, které se zabývají mosty, tunely nebo budovami, tj. ze souboru konkrétních konceptů, a považujeme jejich instance za stavby, potom považujeme stavby za abstraktní koncept.

Konceptuální model je definován jako model, který definuje pojmy univerza diskurzu [ISO 19101-1:2014]. Je tedy nutné mít na paměti, že pro každou doménu zájmu je možné vytvořit samostatný konceptuální model. Takové řešení je však velmi nákladné, protože každý model by pak mohl mít i svou vlastní geometrii a sémantiku. Z toho důvodu je vhodné vytvářet modely tak, aby je bylo možné použít napříč širším spektrem aplikací.

Úvod

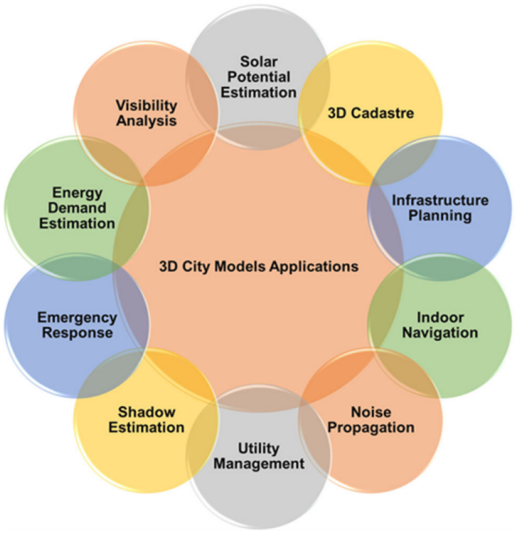
Návrh na vytvoření metodiky pro správu a využívání Národní sady prostorových objektů byl vyhodnocen již ve Strategii rozvoje infrastruktury pro prostorové informace v ČR do roku 2020 (GeoInfoStrategie) jako zcela zásadní. Vyhotovení bude mít dopad na množství agend veřejné správy, které je nutné zohlednit při naplňování koncepce Digitální Česko. Zejména jde o oblasti digitálních agend a digitální ekonomiky jako celku. Národní sada prostorových objektů je přímo vyjmenována jako významný záměr naplňování dílčího cíle v rámci Informační koncepce ČR[1] v části cíle 05 – Efektivní a centrálně koordinované ICT veřejné správy.

NaSaPO (Národní sada prostorových objektů) je komplex dat o území a prostředků pro jejich sběr, správu a využití. Tato poměrně strohá definice je pouze její první úrovní, která je doplněna i o úroveň druhou a třetí, uvedené v samostatném dokumentu Definice NaSaPO. Jedním z úkolů NaSaPO je propojení dat napříč veřejnou správou a kde je to možné, harmonizovat tato data.

Cílem koncepce je navrhnout a realizovat datovou politiku, zajistit interoperabilitu, odstranit duplicity a zpřístupnit prostorové informace ve vlastnictví veřejné správy a ve veřejné sféře s cílem vytvoření veřejných služeb. Je navrhováno propojit centrální sdílené prostorové informace s informacemi ve správě krajských datových center a dalších systémů územních samospráv, které slouží zejména pro územní plánování, krizové řízení a další oblasti využívající geoinformace. Navržená metodika je v souladu se směřováním vládní politiky a klíčovými opatřeními vlády na podporu rozvoje digitálního trhu a digitální ekonomiky České republiky. Nejčastější otázkou, na kterou by měla existovat odpověď i v rámci tohoto dokumentu, je dotaz na využití dat NasaPO.

Obce a další vládní organizace stále více využívají 3D modely měst a krajiny pro plánování a údržbu. Tyto modely často obsahují 3D data, kterými jsou například budovy, silnice, vodní cesty apod. Data jsou shromažďována, udržována a používána v aplikacích pro plánování a simulace v různorodém prostředí (většinou jde o městské prostředí). Níže je uvedeno pouze několik příkladů možností využití takovýchto dat. Problematice využití 3D dat (nejen) ve veřejné správě by mohl být věnován samostatný dokument, který by bylo nutné navíc velmi často aktualizovat. Důvodem jsou dynamicky se rozvíjející technologie, které vedou k rozšiřování možností na poli analýz nad 3D prostorovými daty.

[1] Informační koncepce České republiky (IKČR) je základním dokumentem, který stanovuje na základě zmocnění podle § 5 a odst. 1 zákona 365/2000 Sb., o informačních systémech veřejné správy, cíle České republiky v oblasti informačních systémů veřejné správy (dále také jen ISVS) a obecné principy pořizování, vytváření, správy a provozování informačních systémů veřejné správy v ČR na období 5 let.



Obrázek 1: Znázornění možných aplikací 3D dat

Příklady takových aplikací se pohybují v rozmezí od rozsáhlých studií po mikroanalýzy zaměřené na úroveň budovy. Může jít o studie viditelnosti, simulace stínů vrhaných budovami a vegetací, modelování výstavby nových budov, odhad výšky hladiny řek při povodních, šíření hluku, posouzení dopadů světelných paprsků na střechy pro účel proveditelnosti instalace solárních panelů a mnohé další. Potřeba těchto aplikací nás vede ke zpracování projektu NaSaPO.

1. Scope dokumentu

Tento dokument specifikuje konceptuální model NaSaPO. Pro formální popis konceptuálního modelu se využívá konceptuální schéma. Konceptuální model je zpracováván v libovolné formě, nezřídka běžným jazykem. K přesnému, úplnému a jednoznačnému vyjádření konceptuálního modelu a vytvoření konceptuálního schématu se však používá jazyk konceptuálního schématu (CSL). Pro potřebu konceptuálního modelování NaSaPO byl jako nejvhodnější vybrán jazyk UML. Konceptuální modely slouží k zajištění interoperability mezi jednotlivými doménami na sémantické úrovni. Norma ISO 91103:2015 popisuje čtyři hlavní úrovně abstrakce, přičemž tento dokument se zaměřuje na první tři z nich.

1. Metamodely (nejabstraktnější metamodely jako jsou Obecný model vzhledu jevů (GFM) z ISO 19109 a UML Metamodel z ISO 19505).
2. Abstraktní schémata (jádrové modely se základními pojmy, tj. geometrie a topologie z IS0 19107).
3. Aplikační schémata (stále ještě konceptuální modely, ale specifické pro aplikace, tj. silniční komunikace, budovy).
4. Implementační schémata (schémata pro konkrétní implementace, např. GML aplikační schémata).

Konceptuální model poskytuje rámec, ve kterém jsou na obecné úrovni specifikovány technické aspekty datové části projektu NaSaPO (kromě sběru dat, který je řešen v samostatném dokumentu, jímž je výsledek V06 – Správa využívání NaSaPO). Tento dokument řeší požadavky a doporučení týkající se zejména komponent interoperability, které jsou podrobněji popsány v jednotlivých kapitolách a jejichž shrnutí je součástí kapitoly 4.2 Přehled komponent interoperability. Kromě toho je část dokumentu také věnována problematice CityGML. Standard CityGML byl zvolen jako nejvhodnější pro konceptuální model NaSaPO.

Samotné specifikace jednotlivých typů objektů a jejich vlastností již není součástí tohoto dokumentu. Pro jednotlivá aplikační schémata budou vytvářeny samostatné dokumentace. K tomuto dokumentu nicméně patří i příloha, ve které jsou zpracované ukázky aplikačních schémat a možných nástrojů pro interoperabilitu mezi modelem NaSaPO a daty ve formátech DGN a XML (ve standardu JVF DTM).

2. Přehled použitých norem

V následujícím seznamu je uveden přehled norem, které se problematiky konceptuálního modelování úzce dotýkají. Ne všechny normy jsou citované, ale jejich znalost značně usnadní porozumění problematice řešené v tomto dokumentu.

- ISO 19101-1:2014, Geographic information – Reference model – Part1: Fundamentals
- ISO 19103:2015, Geographic information – Conceptual schema language
- ISO 19104:2016, Geographic information – Terminology
- ISO 19107:2019, Geographic information – Spatial schema
- ISO 19108:2002, Geographic information – Temporal schema
- ISO 19109:2015, Geographic information – Rules for application schema
- ISO 19110:2016, Geographic information – Methodology for feature cataloguing
- ISO 19111:2019, Geographic information – Referencing by coordinates
- ISO 19112:2019, Geographic information – Spatial referencing by geographic identifiers
- ISO 19115-1:2014, Geographic information – Metadata – Part 1: Fundamentals
- ISO 19126:2009, Geographic information – Feature concept dictionaries and registers
- ISO 19131:2007, Geographic information – Data product specifications
- ISO 19136:2007, Geographic information – Geography Markup Language (GML)
- ISO 19136-1:2020, Geographic information – Geography Markup Language (GML) – Part 1: Fundamentals
- ISO 19157:2013, Geographic information – Data quality
- ISO/IEC 19501, Information technology – Open Distributed Processing – Unified Modelling Language (UML) Version 1.4.2

Kompletní přehled norem, asociovaných s problematikou geografických informací, je uveden v příloze C – Přehled ISO norem geografické informace, která je součástí výsledku V02 – Terminologické a věcné vymezení objektů NaSaPO tohoto projektu.

3. Seznam použitých pojmů a zkratek

Pro účely projektu NaSaPO jsou termíny a zkratky, které se využívají napříč výstupy projektu NaSaPO, vedeny v samostatné příloze A výsledku V02 – Terminologický výkladový slovník projektu NaSaPO. Každý dokument, resp. výstup projektu NaSaPO pak obsahuje definované termíny a zkratky určené pro konkrétní text.

3.1 Termíny a definice NaSaPO

jev *(phenomenon)*
jakýkoliv úkaz, který lze pozorovat, nebo vyvodit ze zobrazení operace nad daty.

objekt *(object)*
entita s jednoznačně definovanou hranicí a identitou, která zapouzdřuje stav a chování

[ČSN P ISO/TS 19103:2006]

třída *(class)*
popis množiny objektů, které sdílejí stejné atributy, operace, metody, vztahy a sémantiku

[ČSN P ISO/TS 19103:2006]

katalog tříd objektů NaSaPO / Katalog NaSaPO
katalog obsahující definice a popisy typů vzhledů jevů, atributy vzhledu jevů a vztahů mezi vzhledy jevů vyskytujícími se v jedné nebo více množinách geografických dat NaSaPO, a to zároveň s jakýmikoliv operacemi vzhledů jevů, které mohou být uplatněny

3.2 Základní termíny ISO norem geografické informace

vzhled jevu *(feature)*
abstrakce jevu reálného světa

POZNÁMKA: Vzhled jevu se může vyskytovat jako typ nebo jako instance. ***Typ vzhledu jevu*** *nebo* ***instance vzhledu jevu*** *musí být specifikovány, jestliže je míněna jen jedna z těchto alternativ.*

[ČSN EN ISO 19101-1:2018]

abstraction of real world phenomena

NOTE: A feature can occur as a type or an instance. Feature type or feature instance shall be used when only one is meant.

[ISO 19101-1:2014]

objekt *(object)*
entita s jednoznačně definovanou hranicí a identitou, která zapouzdřuje stav a chování

POZNÁMKA 1: Stav je reprezentován atributy a vztahy, chování je reprezentováno operacemi, metodami a stavovými diagramy. Objekt je instancí třídy.

[ČSN P ISO/TS 19103:2006]

entity with a well defined boundary and identity that encapsulates state and behaviour

[ISO/TS 19103:2005 – upraveno z ISO/IEC 19501]

An object is an instance that originates from a class. In the metamodel Object is a subclass of Instance and it originates from at least one Class. The set of Classes may be modified dynamically, which means that the set of features of the Object may change during its life-time.

PŘEKLAD: Objekt je instance, která pochází ze třídy. V metamodelu je objekt podtřída instance a pochází z alespoň jedné třídy. Sada tříd může být dynamicky modifikována, což znamená, že sada rysů objektu se může během své životnosti změnit.

[OMG/UML UML Semantics, version 1.3, 1997]

POZNÁMKA 2: V UML je anglické homonymum feature vyhrazeno pro vlastnost/rys, jakým je například operace nebo atribut, který je zapouzdřen jako součást seznamu v klasifikátoru, jako je například rozhraní, třída nebo datový typ.

[ČSN P ISO/TS 19104-1:2010]

třída *(class)*
popis množiny objektů, které sdílejí stejné atributy, operace, metody, vztahy a sémantiku

POZNÁMKA: Třída může použít množinu rozhraní ke specifikaci sbírek operací, které poskytuje svému prostředí.

[ČSN P ISO/TS 19103:2006]

description of a set of object that share the same attributes, operations, methods, relationships, and semantics

[ISO/TS 19103:2005 – upraveno z ISO/IEC 19501]

A class is a description of a set of objects that share the same attributes, operations, methods, relationships, and semantics. A class may use a set of interfaces to specify collections of operations it provides to its environment.

PŘEKLAD: Třída je popis sady objektů, které sdílejí stejné atributy, operace, metody, vztahy a sémantiku. Třída může použít sadu rozhraní ke specifikaci sbírek operací, které poskytuje svému prostředí.

[OMG/UML UML Semantics, version 1.3, 1997]

3.3 Termíny a definice z norem – vzhled jevu

katalog vzhledů jevů *(feature catalog)*
katalog obsahující definice a popisy typů vzhledů jevů, atributů vzhledů jevů a asociací vzhledů jevů, jež se vyskytují v jedné nebo více množinách geografických dat, a to zároveň s jakýmikoliv operacemi vzhledů jevů, které se mohou uplatnit

[ČSN EN ISO 19101-1:2018]

catalogue containing definitions and descriptions of the feature types, feature attributes, and feature relationships occurring in one or more sets of geographic data, together with any feature operations that may be applied

[ISO 19101-1:2014]

atribut vzhledu jevu *(feature attribute)*
charakteristika vzhledu jevu

PŘÍKLAD 1: Atribut vzhledu jevu zvaný „barva“ může mít hodnotu atributu „zelená“, jejíž datový typ je „text“.

PŘÍKLAD 2: Atribut vzhledu jevu zvaný „délka“ může mít hodnotu atributu „82,4“, jejíž datový typ je „reálné číslo“.

POZNÁMKA 1: Atribut vzhledu jevu má název, datový typ a doménu hodnot, které jsou k němu přidruženy. Atribut vzhledu jevu pro instanci vzhledu jevu má také hodnotu atributu vybranou z domény hodnot.

POZNÁMKA 2: Atribut vzhledu jevu může v katalogu vzhledů jevů zahrnovat doménu hodnot, ale nespecifikuje hodnoty atributu pro instance vzhledů jevů.

[ČSN EN ISO 19101-1:2018]

characteristic of a feature

[ISO 19101-1:2014]

asociace vzhledů jevů *(feature association)*
vztah, který spojuje instance jednoho typu vzhledu jevů s instancemi téhož nebo odlišného typu vzhledu jevů

[ČSN EN ISO 19109:2020]

relationship that links instances of one feature type with instances of the same or a different feature type

[ISO 19110:2016]

operace vzhledu jevů *(feature operation)*
operace, kterou může provést každá instance typu vzhledů jevů

PŘÍKLAD: Operací vzhledu jevů na ‚přehradě‘ je zvýšení přehrady. Výsledky této operace jsou zvětšení výšky ‚přehrady‘ a zvýšení vodní hladiny v ‚nádrži‘.

POZNÁMKA: Operace vzhledu jevů poskytují někdy základ pro definici typu vzhledů jevů.

[ČSN EN ISO 19109:2020]

operation that every instance of a feature type may perform

[ISO 19110:2016]

3.4 Termíny a definice z norem – vztahy a chování

vztah *(relationship)*
sémantické spojení mezi prvky modelu

POZNÁMKA: Druhy vztahů zahrnují asociaci, zobecnění, metavztah, tok i několik druhů seskupených pod závislostí.

[ČSN P ISO/TS 19103:2006]

semantic connection among model elements

[ISO/TS 19103:2015 – upraveno z ISO/IEC 19501]

chování *(behaviour)*
pozorovatelné účinky operace nebo události včetně jejích výsledků

[ČSN P ISO/TS 19103:2006]

asociace *(association)*
sémantický vztah mezi dvěma nebo více klasifikátory, který specifikuje spojení mezi jejich instancemi

POZNÁMKA: Binární asociace je asociací mezi přesně dvěma klasifikátory (včetně možnosti asociace z klasifikátoru na sebe sama).

[ČSN P ISO/TS 19103:2006]

semantic relationship that can occur between typed instances

[ISO/TS 19103:2015 – upraveno z ISO/IEC 19501]

agregace *(aggregation)*
zvláštní forma asociace, která specifikuje vztah typu „je součástí“ mezi agregátem (celkem) a komponentní součástí

POZNÁMKA: Viz kompozice.

[ČSN P ISO/TS 19103:2006]

special form of association that specifies a whole-part relationship between the aggregate (whole) and component part

[ISO/TS 19103:2015 – upraveno z ISO/IEC 19501]

dědičnost *(inheritance)*
mechanismus, kterým specifičtější prvky přijímají strukturu a chování obecnějších prvků, jež jsou příbuzné chováním

POZNÁMKA: Viz zobecnění.

[ČSN P ISO/TS 19103:2006]

mechanism by which more specific classifiers incorporate structure and behaviour defined by more general classifiers

[ISO/TS 19103:2015 – upraveno z ISO/IEC 19501]

zobecnění *(generalization)*
taxonomický vztah mezi obecnějším prvkem a specifičtějším prvkem, který je zcela bezesporný s obecnějším prvkem a obsahuje dodatečnou informaci

POZNÁMKA 1: Instance specifičtějšího prvku se může použít tam, kde se připouští obecnější prvek.

POZNÁMKA 2: Viz dědičnost.

[ČSN P ISO/TS 19103:2006]

taxonomic relationship between a more general element and a more specific element of the same element type

[ISO/TS 19103:2015 – upraveno z ISO/IEC 19501]

kompozice *(composition)*
forma agregace, která požaduje, aby instance části byla najednou zahrnuta nanejvýše v jedné složenině a aby složený objekt byl zodpovědný za vytvoření a výmaz těchto částí

POZNÁMKA: Části s nepevnou násobností mohou být vytvářeny po samotné složenině, ale jednou vytvořené s ní žijí a umírají (tj. sdílejí doby existence). Takové části také mohou být před smrtí složeniny explicitně vyjmuty. Kompozice může být rekurzivní. Synonymum: složená agregace.

[ČSN P ISO/TS 19103:2006]

aggregation where the composite object (whole) has responsibility for the existence and storage of the composed object (parts)

[ISO/TS 19103:2015 – upraveno z ISO/IEC 19501]

operace *(operation)*
služba, která může požadovat od objektu, aby ovlivnil chování

POZNÁMKA 1: Operace má signaturu, která může omezovat skutečné parametry, jež jsou možné.

POZNÁMKA 2: Definice z UML Reference Manual: Specifikace transformace nebo dotazu, kterým může být objekt volán k provedení.

POZNÁMKA 3: Operace má název a seznam parametrů. Metoda je procedura, která implementuje operaci. Má algoritmus nebo popis procedury.

[ČSN P ISO/TS 19103:2006]

behavioural feature of a classifier that specifies the name, type, parameters, and constraints for invoking an associated behaviour

[ISO/TS 19103:2015 – upraveno z ISO/IEC 19501]

vlastnost *(property)*
aspekt nebo atribut objektu určený jménem

[ČSN EN ISO 19109:2020]

facet or attribute of an object referenced by a name

[ISO 19109:2015]

prvek následníka objektu GML

POZNÁMKA: Odpovídá atributu vzhledu jevů a roli asociace vzhledů jevů v ISO 19109. Jestliže má vlastnost GML vzhledu jevu atribut xlink:href, který odkazuje na nějaký vzhled jevu, pak tato vlastnost reprezentuje roli asociace vzhledů jevů.

[ČSN P ISO/TC 19104-1:2010]

a child element of a GML object

[ISO 19136:2007]

omezení *(constraint)*
sémantická podmínka nebo restrikce

POZNÁMKA: V UML jsou jistá omezení předdefinována, jiná mohou být definována uživatelem. Omezení jsou v UML jedním ze tří mechanismů rozšiřitelnosti. Viz: označená hodnota, stereotyp.

[ČSN P ISO/TS 19103:2006]

condition or restriction expressed in natural language text or in a machine readable language for the purpose of declaring some of the semantics of an element

[ISO/TS 19103:2015 – upraveno z ISO/IEC 19501]

závislost *(dependency)*
vztah mezi dvěma prvky modelování, v němž změna jednoho prvku modelování (nezávislého prvku) ovlivní jiný prvek modelování (závislý prvek)

[ČSN P ISO/TS 19103:2006]

relationship that signifies that a single or a set of model element requires other model elements for their specification or implementation

[ISO/TS 19103:2015 – upraveno z ISO/IEC 19501]

kardinalita *(cardinality)*
počet prvků v množině

POZNÁMKA: Opak: násobnost.

[ČSN P ISO/TS 19103:2006]

number of elements in a set

[ISO/TS 19103:2005 – upraveno z ISO/IEC 19501]

násobnost *(multiplicity)*
specifikace rozsahu přípustných kardinalit, jichž může množina nabývat

POZNÁMKA 1: Specifikace násobnosti mohou být dány pro role v rámci asociací, části ve složeninách, opakování a jiné účely. Násobnost je v podstatě (eventuálně nekonečná) podmnožina nezáporných celých čísel.

[ČSN P ISO/TS 19103:2006]

POZNÁMKA 2: Opak: kardinalita.

[ČSN P ISO/TC 19104-1:2010]

specification of the range of allowable cardinalities that a set may assume

[ISO/TS 19103:2015 – upraveno z ISO/IEC 19501]

klasifikátor *(classifier)*
mechanismus, který popisuje rysy týkající se chování a strukturální rysy

POZNÁMKA: Klasifikátory zahrnují rozhraní, třídy, datové typy a komponenty.

[ČSN P ISO/TS 19103:2006]

mechanism that describes behavioural and structural features in any combination

[ISO/TS 19103:2015 – upraveno z ISO/IEC 19501]

instance *(instance)*
entita, která má jedinečnou identitu, lze na ni aplikovat množinu operací, a stav, který uchovává účinky těchto operací

POZNÁMKA: Viz objekt.

[ČSN P ISO/TS 19103:2006]

individual entity having its own value and possibly its own identity

[ISO/TS 19103:2015 – upraveno z ISO/IEC 19501]

identifikátor *(identifier)*
lingvisticky nezávislá posloupnost znaků schopná jedinečně a trvale identifikovat to, s čím je sdružena

[ISO 19135:2005 – upraveno z ISO/IEC 11179-3:2003]

rozhraní *(interface)*
pojmenovaná množina operací, které charakterizují chování entity

[ČSN EN ISO 19119:2019]

pojmenovaná množina operací, které charakterizují chování prvku

[ČSN P ISO/TS 19103:2006]

classifier that represents a declaration of set of coherent public features and obligations

Note 1 to entry: An interface specifies a contract; any classifier that realizes the interface must fulfil that contract. The obligations that can be associated with an interface are in the form of various kind of constraints (such as pre- and post-conditions) or protocol specifications, which can impose ordering restrictions on interactions through the interface.

[ISO/TS 19103:2015 – upraveno z ISO/IEC 19501]

3.5 Termíny a definice z norem – data

typ *(type)*
stereotypovaná třída, která specifikuje doménu objektů spolu s operacemi aplikovatelnými na tyto objekty, bez definování fyzické implementace těchto objektů

POZNÁMKA: Typ může mít atributy a asociace.

[ČSN P ISO/TS 19103:2006]

stereotyped class that specifies a domain of objects together with the operations applicable to the objects, without defining the physical implementation of those objects

[ISO/TS 19103:2015 – upraveno z ISO/IEC 19501]

datový typ *(data type)*
specifikace domény hodnot s operacemi přípustnými na hodnotách v této doméně

PŘÍKLAD: Integer, Real, String, Date a SG Point [konverze dat do řady kódů].

POZNÁMKA: Datové typy zahrnují jednoduché předdefinované typy a uživatelem definované typy.

[ČSN P ISO/TS 19103:2006]

specification of a value domain with operations allowed on values in this domain

[ISO/TS 19103:2015]

data *(data)*
opakovaně interpretovatelná formalizovaná podoba informace vhodná pro komunikaci, interpretaci nebo zpracování

[ISO/IEC 2382-1:1993 převzato z ČSN P ISO/TC 19104-1:2010]

aplikace *(application)*
manipulace s daty a jejich zpracování na podporu požadavků uživatele

[ISO 19101:2002]

aplikační schéma *(application schema)*
konceptuální schéma pro data požadovaná jednou nebo více aplikacemi

[ISO 19101:2002]

aplikační schéma UML *(UML application schema)*
aplikační schéma zapsané v UML podle ISO 19109

[ISO 19136:2007]

jednotný identifikátor zdroje; URI *(Uniform Resource Identifier; URI)*
jedinečný identifikátor pro zdroj strukturovaný ve shodě s IETF RFC 2396

POZNÁMKA: Obecná syntaxe je <scheme>::<scheme-specific-part>. Hierarchická syntaxe s prostorem názvů je <scheme>://<authority><path>?<query>. Viz RFC 2396.

[ISO 19136:2007]

3.6 Termíny a definice z norem – jazyk a pojmy

pojem/koncept *(concept)*

1. znalostní jednotka vytvořená jedinečnou kombinací aspektů a/nebo komponent (upraveno podle ISO 1087-1:2000);
2. společné znaky mezi jednotlivými věcmi definované jedním nebo více omezeními popisujícími limity pro zahrnutí jednotlivých věcí odpovídajících danému pojmu

[ZDROJ: ISO 16354:2013]

definice *(definition)*
vyjádření pojmu pomocí deskriptivní výpovědi, která slouží k jeho odlišení od souvisejících pojmů

[ZDROJ: ISO 16354:2013]

konceptuální formalismus *(conceptual formalism)*
množina modelovacích pojmů užitých k popisu konceptuálního modelu

PŘÍKLAD: Metamodel UML, metamodel EXPRESS.

POZNÁMKA: Jeden konceptuální formalismus může být vyjádřen v několika jazycích konceptuálního schématu.

[ISO 19101:2002]

konceptuální model *(conceptual model)*
model, který definuje pojmy univerza diskurzu

[ISO 19101:2002]

konceptuální schéma *(conceptual schema)*
formální popis konceptuálního modelu

[ISO 19101:2002]

jazyk konceptuálního schématu *(conceptual schema language)*
formální jazyk založený na konceptuálním formalismu za účelem reprezentace konceptuálních schémat

PŘÍKLAD: UML, EXPRESS, IDEF1X

POZNÁMKA: Jazyk konceptuálního schématu může být lexikální nebo grafický. Několik jazyků konceptuálního schématu může být založeno na stejném konceptuálním formalismu.

[ISO 19101:2002]

grafický jazyk *(graphical language)*
jazyk, jehož syntaxe je vyjádřena grafickými symboly

[ISO 19101:2002]

metamodel *(metamodel)*
<UML> model, který definuje jazyk pro vyjádření modelu

[ISO/TS 19103:2005 – upraveno z ISO/IEC 19501]

schéma *(schema)*
formální popis modelu

[ISO 19101:2002]

4. Interoperabilita

Interoperabilita je naprosto esenciální část pro hlubší pochopení geografické informace. Je obecně definována jako schopnost systému nebo jeho komponenty umožnit sdílení informace a řídit proces spolupráce mezi aplikacemi. Další definice pocházejí z Institutu pro elektrotechnické a elektronické inženýrství a z mezinárodní normy ISO/IEC 2382-1:1993.

- Způsobilost systému nebo jejich komponent si vyměňovat informace a tyto vyměněné informace používat.
- Schopnost komunikovat, provádět programy nebo přenášet data mezi různými funkčními jednotkami tak, že uživatel vystačí jen s malou znalostí konkrétních vlastností jednotek, popř. je vůbec nepotřebuje znát.

Interoperabilitu lze přirovnat k procesu mezilidské komunikace případně může být popsána jako komunikační proces, který existuje mezi dvěma subjekty, uživatelem a poskytovatelem, mezi nimiž probíhají interakce týkající se geografické informace.

PŘÍKLAD: Uživatel pošle dotaz na poskytovatele prostřednictvím komunikačního kanálu (např. internet). Pro vydefinování dotazu používý uživatel vlastí pojmy a vlastní slovník. Jakmile jeho dotaz dospěje k poskytovateli, je jeho úkolem tento dotaz správně interpretovat. To znamená, že poskytovatel musí správně identifikovat pojmy uživatele a na základě toho zodpovědět dotaz. Odpověď je poté poté odeslána zpět uživateli, který ji opět musí správně interpretovat a vyhodnotit.

Norma ISO 19101-1:2014 stanovuje detailní popis a rámec této komunikace. Této problematice se podrobněji věnuje výsledek V02 – Terminologické a věcné vymezení objektů NaSaPO v kapitole řešící obor úvah ISO norem geografické informace.

4.1 Heterogenita geografické informace

Interoperabilita geografické informace aspiruje na zavedení dobře fungující komunikace mezi systémy a aplikacemi, nakládající s geografickými informacemi a uživateli. Systémy, aplikace a uživatelé se mohou různit, a proto pro dosažení interoperability je důležité překonat heterogenitu, která mezi nimi existuje. Ta je rozdělena do čtyř typů:

- systémová heterogenita,
- syntaktická heterogenita,
- strukturální heterogenita
- sémantická heterogenita

Systémová heterogenita se zaobírá rozdíly v technickém vybavení (hardwarová vybavenost), operačními systémy (převážně interoperabilita mezi OS Windows a Linux) a komunikačními systémy (sítě a protokoly). Nastavení propojení mezi těmito systémy je pro NaSaPO nezbytné. Systémovou heterogenitu se více zaobírá výsledek V04 – Koncepce NaSapo v kapitole týkající se rámcového návrhu informačního systému NaSaPO

Syntaktická heterogenita se týká fyzické reprezentace dat. Vztahuje se k vnější podobě zprávy, ale ne k jejímu obsahu. Syntaktická heterogenita se projevuje v případě, že aplikace komunikuje v odlišném

kódovacím formátu než jiná aplikace (například přijímaná data jsou ve formátu DGN, ale publikovaná data mají být ve formátu GML). Syntaktická heterogenita se také projevuje mezi různými strukturami dat, které jsou použity aplikacemi za účelem zobrazení prostorové informace (vektorová vs rastrová data). Syntaktická heterogenita je řešena v rámci přílohy této dokumentace.

Strukturální heterogenita (nebo také schématická) se týká rozdílů vztažených ke konceptuálním modelům geografických informací. V praxi jde o rozdílnosti na úrovni popisu geografických jevů, které mohou být popsány třídami, atributami a vztahy, přičemž napříč aplikacemi může být jeden jev popsán více způsoby (například ulice jako třída nebo ulice jako vlastnost vztažená k třídě cesta…). Problematikou se částečně zabývá výsledek V02 – Terminologické a věcné vymezení objektů NaSaPO. Aspekty související s touto heterogenitou jsou také velmi podrobně řešeny v tomto dokumentu.

Sémantická heterogenita se zaobírá rozdíly pojmů na úrovni vnímání a reprezentace reality, Způsob vnímání reality více jedinci může vést k různým pojmům (most může být pro jednoho překážkou, pro jiného nezbytným prvkem komunikace). Sémantická heterogenita je podrobně probrána ve výsledku V02 – Terminologické a věcné vymezení objektů NaSaPO.

4.2 Přehled komponent interoperability pro projekt NaSaPO

Tabulka 1 poskytuje přehled komponent relevantních pro podporu interoperability v projektu NaSaPO. jednotlivé komponenty pokrývají různé aspekty, které jsou nezbytné pro tuto podporu. Pro lepší pochopení souvislostí jednotlivých komponent byl doplněna i tabulka znázorňující komponenty ve vztahu k interoperabilitě.

Tabulka 1: Přehled komponent interoperability

Referenční model	Komponenta definuje rámec pro konceptuální modelování a dodává pravidla pro psaní aplikačních schémat.
Terminologie	Komponenta podporuje použití konzistentního jazyka s využitím terminologického výkladového slovníku.
Pravidla pro aplikační schémata	Komponenta specifikuje pravidla pro aplikační schémata a zaobírá se koncepty, které jsou důležité pro zpracování těchto aplikačních schémat.
Jazyk konceptuálního schématu	Komponenta slouží k popisu konceptuálních schémat. Jde o formální jazyk, jež je možný analyzovat jak člověkem, tak i počítačem.
Katalog vzhledů jevů	Katalog vzhledů jevů poskytuje informativní přehled vzhledů jevů a datových typů definovaných v datových specifikacích.
Obecný model vzhledu jevu	Obecný model vzhledu jevů (GFM - General Feature Model) je komponentou, která slouží jako meta-model pro specifikaci a popis typů vzhledu jevů.
Prostorové a časové aspekty objektů	Komponenta je nezbytná pro vyjádření prostorových a časových charakteristik objektů v aplikačních schématech.
Identifikátory	Komponenta slouží k jednoznačné identifikaci prostorových objektů.

5. Referenční model

K zajištění integrity geografické informace i výsledků jejich aplikace se musí vývoj a využití opírat o stanovený referenční model. Referenční model popisuje základní požadavky, kterými je nezbytné se řídit. Popis referenčního modelu se opírá o konceptuální formalismus. Referenční model je abstraktním rámcem nebo ontologií pro konkrétní doménu, sestávající z vzájemně provázané množiny zřetelně definovaných pojmů vytvořených experty nebo skupinami expertů s cílem:

- vytvořit standardy pro objekty a jejich vzájemné vztahy,
- poučit a napomáhat při řešení složitých komplexních problémů jejich rozdělením na více menších problémů, které jsou snadněji uchopitelné, lze je lépe projednat a vylepšovat řešení
- podpořit srozumitelnou komunikaci mezi lidmi
- stanovit jednoznačné role a definovat zodpovědnosti
- umožnit porovnání možných alternativ řešení problémů.

Referenční modely jsou často ilustrovány jako množiny konceptů s nějakou indikací vztahů mezi nimi. Podle OASIS (Organization for the Advancement of Structured Information Standards) je referenční model abstraktním rámcem pro pochopení důležitých vztahů mezi entitami nějakého prostředí a pro vývoj konzistentních standardů nebo specifikací toto prostředí podporujících. Referenční model je založen na nevelkém počtu sjednocujících konceptů a může být použit jako základ pro vzdělávání a vysvětlování těchto standardů neodborníkům. Referenční model není přímo svázán s žádnými standardy, technologiemi nebo jinými konkrétními detaily implementací. Usiluje však o poskytování jednotné sémantiky, která může být použita jednoznačně napříč různými implementacemi i mezi nimi navzájem. Pojetí OASIS bylo převzato jako definice referenčního modelu v normách ISO geografické informace:

Referenční model tedy:

- Sestává z komplexního pohledu poskytujícího abstraktní popis prvků, které mohou utvářet obor geografických informací a jejich vzájemné vztahy.
- Stanovuje základní požadavky kladené na danou činnost, a určí základy kterými je přitom nezbytné se řídit.
- Slouží k zajištění integrity norem pro geografické informace a výsledků jejich aplikace.
- Slouží k definování a popsání interoperability geografické informace tak, že si bude všímat systémové, syntaktické, strukturální a sémantické úrovně.
- Slouží k objasnění role sémantiky v technologiích, jakou je například web a mnoho nově se rozvíjejících způsobů přístupu k němu. Rovněž slouží k objasnění toho, jak sémantický web může podporovat interoperabilitu v oboru geografické informace.
- Popisuje koncepce interoperability a objasňuje, na jakých základech je založena.
- Identifikuje různé aspekty požadované pro normalizace v doméně geografické informace.
- Poskytuje vysokoúrovňový popis těch aspektů geografické informace, kterými se zabývají ISO normy pro geografické informace a popisuje v jakém vzájemném vztahu jsou tyto aspekty.

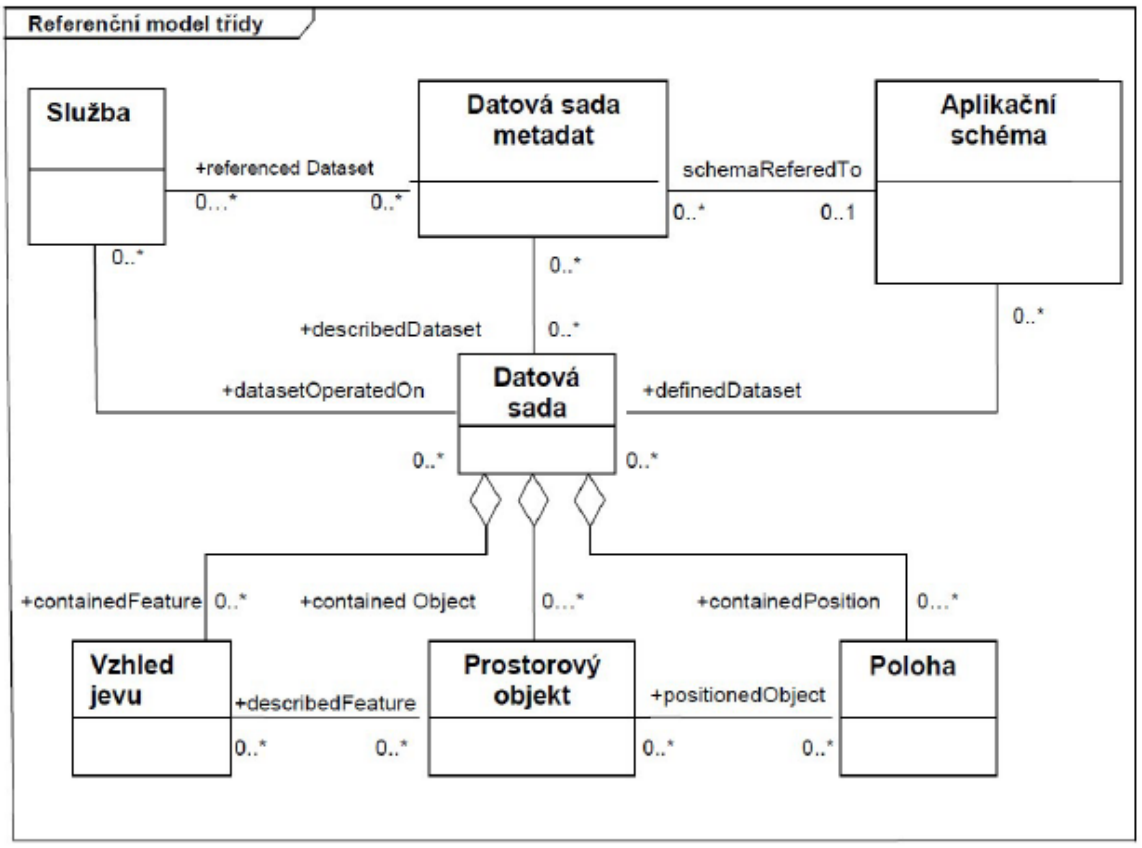
Vysokoúrovňový diagram domény geografické informace obsahuje:

- Datovou sadu , která zahrnuje:
  - Vzhledy jevů včetně jejich atributů, vztahů a operací mezi nimi.
  - Prostorové objekty, které mohou popisovat prostorové aspekty vzhledů jevů, nebo složené datové struktury, které přidružují hodnoty atributů k jednotlivým polohám v definovaném prostoru. Jsou možné dva obecné přístupy k modelování prostorových aspektů geografické informace:
    - vnímat prostor jako prostor obsazený vzhledy jevů, jež jsou popsány použitím vektorových dat,
    - představit si variaci hodnot zájmu nad prostorem jako určitou distribuční funkci (pokrytí).
  - Popisy poloh prostorových objektů v prostoru a čase s použitím měrných jednotek poskytovaných referenčními systémy.
- Aplikační schéma, které poskytuje popis vzhledů jevů určité sady dat. Dále potom identifikuje typy prostorových objektů a referenční systémy požadované pro poskytnutí úplného popisu geografické informace v datové sadě.
- Datovou sadu metadat, která umožňuje uživatelům vyhledávat, hodnotit, porovnávat a uspořádávat geografická data. Popisuje správu, organizaci, obsah a kvalitu geografické informace v datových sadách. Může se odkazovat na aplikační schéma nebo na katalog vzhledů jevů, jenž obsahuje definice pojmů použitých v aplikačním schématu.
- Geografické informační služby jimiž jsou softwarové programy operující nad geografickou informací obsaženou v sadách dat. Služby odkazují na informaci o datové sadě metadat za účelem správného provedení vyhledávacích operací, stejně jako transformací či interpolací. Služby přistupují k datům v prostředí propojeném sítí, v němž jsou datové sady uchovávány v systémech správy distribuovaných databází.

5.1 Konceptuální rámec referenčního modelu

Konceptuální rámec slouží k podpoře popisu referenčního modelu. Je založený na základech interoperability a dělí se na:

- meta-meta úroveň,
- meta úroveň,
- aplikační úroveň,
- instanční úroveň.



Obrázek 2: Vysokoúrovňová doména geografické informace

Tabulka 2: Rámec referenčního modelu

Konceptuální rámec referenčního modelu				
Úroveň /základ	Interoperabilita			Procedurální normy
	Sémantický základ	Syntaktický základ	Servisní základ	
Meta-meta	Meta-meta: Sémantický	Meta-meta: Syntaktický	Meta-meta: Servisní	Meta-meta: Procedurální
Meta	Meta: Sémantický	Meta: Syntaktický	Meta: Servisní	Meta: Procedurální
Aplikace	Aplikace: Sémantický	Aplikace: Syntaktický	Aplikace: Servisní	Aplikace: Procedurální
Instance	Instance: Sémantický	Instance: Syntaktický	Instance: Servisní	Instance: Procedurální

Meta-meta úroveň se zaobírá obecnějšími aspekty ISO norem pro geografické informace. Zabývá se také profilováním, které spočívá ve vytváření balíčků či podmnožin z úplné sady norem. Vytváření těchto balíčků napomáhá rychlé implementaci a proniknutí do uživatelských prostředí díky srozumitelnosti úplných sad norem.

Meta úroveň pokrývá aspekty ISO norem pro geografické informace požadované pro vývoj aplikací. To zahrnuje normy založené na pravidlech a také normy definující prvky jako datové typy, datové struktury, jazyky pro kódování geografické informace atd.

Aplikační úroveň pokrývá aspekty ISO norem pro geografické informace zaměřené na oblasti, které mohou být bezprostředně implementovatelné. Patří sem schémata pro metadata a ontologie, týkající se konkrétních domén a aplikací.

Instanční úroveň pokrývá data, kódovaná data a také služby nebo aplikace zpracovávající data.

5.2 Sémantický základ referenčního modelu

Na meta-meta úrovni se sémantický model musí skládat z norem, které slouží jako základ pro definici a popis základních pojmů podporujících rozvoj geografických aplikací a interoperability mezi nimi.

Zahrnuje referenční model a další mezinárodní normy týkající se jazyků konceptuálního schématu použitých pro definici ISO norem pro geografické informace. Dále také obsahuje normy, týkající se datových toků a kódů, specifikace UML, OCL a OWL.

Ne meta úrovni model sestává z norem, jež definují a popisují základní pojmy, které jsou nezbytné pro definování, popis a reprezentaci geografické informace. Tyto pojmy slouží k vytvoření ontologie a umožňují konzistentní definici vzhledu geografických jevů s jejich konkrétními charakteristikami. Patří sem:

- norma definující GML včetně pravidel pro vývoj konzistentních aplikačních schémat,
- normy související s definováním prostorových pojmů, včetně geometrických a topologických pojmů.
- normy související s definováním jak časových, tak i časoprostorových pojmů,
- normy související s definováním pojmů pro popis prostorových referenčních systémů souřadnicemi, geografickými identifikátory atd.,
- normy související s definováním pojmů pro popis kvality geografické informace,
- normy pro definování specifikací datových produktů a
- normy definující pojmy a pravidla nebo metodologie pro vývoj konzistentních aplikačních ontologií na bázi GFM.

Na aplikační úrovni se model musí skládat z definice pojmů geografické informace, které jsou konkrétně definovány pro nějakou aplikaci. Tyto pojmy jsou definovány ve shodě s meta úrovní a mohou být definovány jako součást mezinárodních norem. Tato úroveň zahrnuje:

- aplikační schémata,
- aplikační ontologie v konkrétních oborech,
- specifikace datových produktů a
- metadatové modely (např. metadata vektorových dat, metadata zobrazení, atd.).

Instalační úroveň musí sestávat z dat, která jsou uchovávána v geografických informačních systémech, databázích, souborech, atd. Tato data vyhovují aplikačním schématům, ontologiím a metadatovému modelu definovanému v aplikační úrovni. Instanční úroveň zahrnuje:

- sadu dat včetně vzhledů jevů,
- prostorové a časové objekty a polohu,
- dokument instancí metadat,
- popis kvality dat.

5.3 Syntaktický základ referenčního modelu

Na meta-meta úrovni se model musí skládat z norem, které slouží jako základ pro definování pravidel a metodologií pro kódování geografické informace do formátů, umožňujících komunikaci nebo výměnu dat. Může jít například o specifikaci XML.

Na meta úrovni se syntaktický základ musí sestávat z norem definujících určité jazyky a pravidla pro kódování geografické informace, stejně jako pravidla pro její znázornění formou map nebo diagramů ve shodě se sémantickým základem. Zahrnuje:

- normy definující kódovací jazyky a pravidla pro geografická data (např. vektorová data, zobrazení, atd.) a
- normy definující pravidla pro znázornění a definici katalogu znázornění.

Aplikační úroveň musí obsahovat definice normalizovaných reprezentací geografické informace. Aplikační základ zahrnuje:

- normy popisující textové a binární kódování,
- normy popisují XML reprezentace dat a
- pravidla znázornění a katalogy.

Instanční úroveň musí sestávat z dokumentů instancí kódovaných nebo znázorněných podle XML aplikačních schémat nebo katalogů a pravidel pro znázornění. Instanční úroveň zahrnuje:

- kódované instance vektorových, obrazových a mřížových dat,
- kódované instance dat pokrytí,
- kódované instance pro metadata,
- mapy (digitální obrazy, papírové mapy atd.) a diagramy.

5.4 Servisní základ referenčního modelu

Meta-meta úroveň musí sestávat z norem, jež slouží jako základ pro definici pravidel a metodologií pro vývoj zpracování geografické informace a služeb pro vyhledání, přístup a zpracování geografické informace, zahrnuje:

- referenční model pro služby,
- referenční model pro služby založené na místu a
- referenční model pro neomezený veřejný přístup.

Meta úroveň musí obsahovat normy definující pravidla a metodologie pro vývoj zpracování geografické informace a služeb. Zahrnuje normy pro služby.

Aplikační úroveň obsahuje normy z definic normalizovaných služeb geografické informace. Aplikační úroveň zahrnuje:

- normy pro geografické služby interakce mezi lidmi a komunikace,
- normy pro geografické služby řízení modelů/informací,
- normy pro geografické služby řízení toku prací a úloh,
- normy pro geografické služby zpracování prostorové, tematické a časové informace a metadata.

Instanční úroveň musí sestávat z instancí služeb (včetně webových služeb) v souladu se službami definovanými jako součást aplikační úrovně.

5.5 Procedurální základ referenčního modelu

Meta-meta úroveň musí sestávat z norem, které slouží jako základ pro definování norem popisujících uspořádané řady kroků ke splnění specifikované úlohy.

Meta úroveň musí obsahovat normy definující rámce nebo struktury, na nichž jsou procedurální normy založeny a vyvíjeny. Zahrnuje normy pro shodu a zkoušení.

Na aplikační úrovni se musí sestávat z definic normalizovanych procedur, které jsou používány v určitém kontextu. Zahrnují:

- normy týkající se terminologie použité v ISO normách geografické informace a registr terminologií,
- normy týkající se definic profilů,
- normy týkající se procedur pro registraci a
- normy týkající se procedur pro hodnocení kvality.

Instanční úroveň musí obsahovat procedury definované pro určitý kontext. Může obsahovat:

- specifikace definující kroky pro sběr vektorových nebo rastrových dat a
- specifikace definující kroky pro zpracování dat k danému účelu.

Na závěr je vhodné dodat, že vzhledem ke kvalitě referenčního modelu zpracovaného v rámci normy ISO 19101-1:2014 nebyl vytvářen explicitní referenční model pro účely NaSaPO.

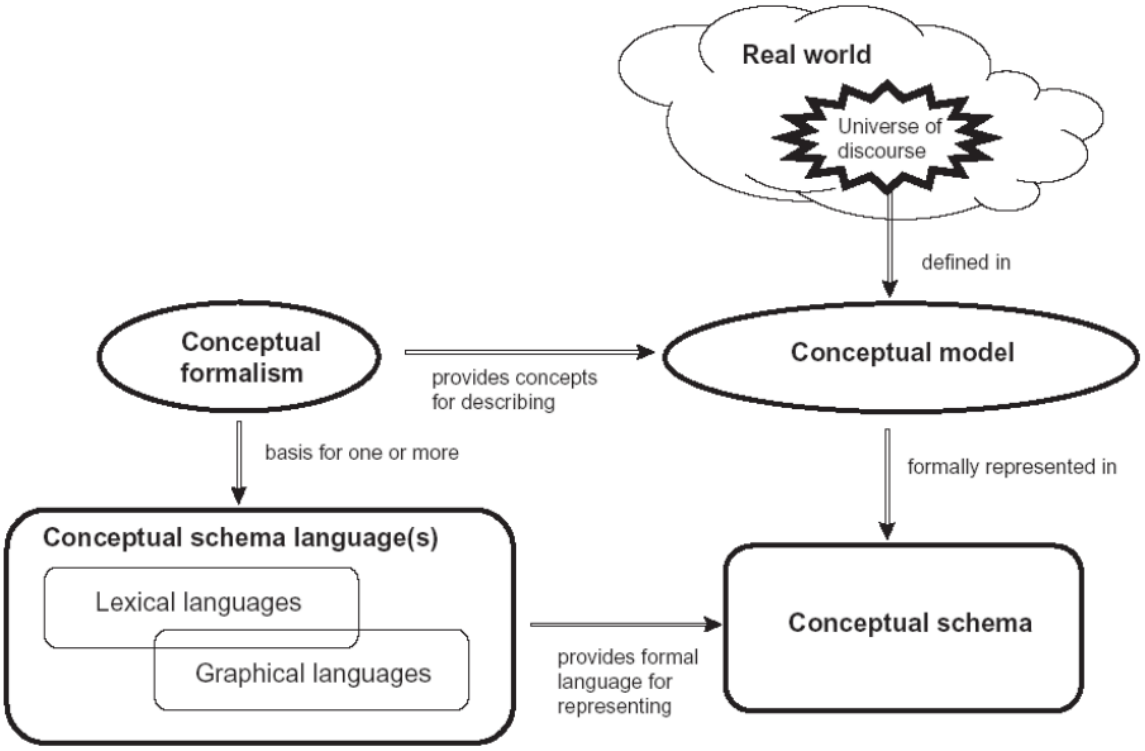
6. Terminologie

Pojmy, které jsou použité v tomto dokumentu i v dokumentech dalších, týkajících se projektu NaSaPO, jsou udržovány v terminologickém výkladovém slovníku NaSaPO, který je veden jako příloha A výsledku V02 – Terminologické a věcné vymezení projektu NaSaPO. Jednotlivé termíny a jejich definice byly převzaty z relevantních ISO norem.

Z hlediska interoperability je zásadní nalezení terminologického konsenzu mezi odbornou veřejností v oblastech informačních technologií (včetně dopravní a technické infrastruktury) a v oblasti řízení staveb. Hlavním problémem je překryv mnoha nejen geografických termínů, přičemž jejich nesprávné pochopení může přinášet negativní důsledky v rámci sběru, správy a sdílení (nejen) prostorových dat. Komplexní problematikou se zabývá dokument Terminologické a věcné vymezení objektů NaSaPO.

7. Pravidla pro aplikační schémata

Schéma na obrázku 4 poskytuje grafického znázornění role konceptuálního modelování tak, že popisuje vztah mezi modelováním reálného světa a jeho digitálního obrazu.



Obrázek 4: Schéma znázorňující přechod z reálného světa do jeho konceptuální podoby

Univerzum diskurzu neboli náhled na reálný svět je vybraná část reálného světa, která je popisována v rámci nějakého modelu. Univerzum diskurzu obsahuje nejen prostorové objekty jako jsou vodní plochy, komunikace, administrativní hranice apod. ale i jejich atributy, operace a vazby které existují mezi těmito prostorovými objekty. Univerzum diskurzu je popsáno konceptuálním modelem.

Konceptuální schéma je podrobný popis konceptuálního modelu pro vybrané univerzum diskurzu. Jazyk konceptuálního schématu popisuje konceptuální schéma. Konceptuální schéma, které definuje, jakým způsobem by mělo být univerzum diskurzu popsáno, se nazývá aplikační schéma. Konceptuální schéma se opírá o konceptuálních formalismus, který poskytuje pravidla pro jazyk konceptuálního schématu.

Tato část specifikuje pravidla pro aplikační schémata NaSaPO a zaobírá se koncepty, které jsou důležité pro zpracování těchto aplikačních schémat. Aplikační schéma je konceptuální schéma pro data požadovaná jednou nebo více aplikacemi. Aplikační schéma musí definovat obsah a strukturu dat a může definovat operace pro manipulaci a zpracování dat a omezení vedoucí k zajištění integrity dat. Účel aplikačního schématu je dvojí:

- Poskytnout počítačově čitelný popis dat definující strukturu dat, čímž je umožněno, aby byl aplikován automatický mechanismus pro správu dat.
- Dosáhnout společného a správného porozumění dat tím, že obsah dat dílčího aplikačního oboru bude dokumentován, čímž bude umožněno zcela jednoznačné získání informace z těchto dat.

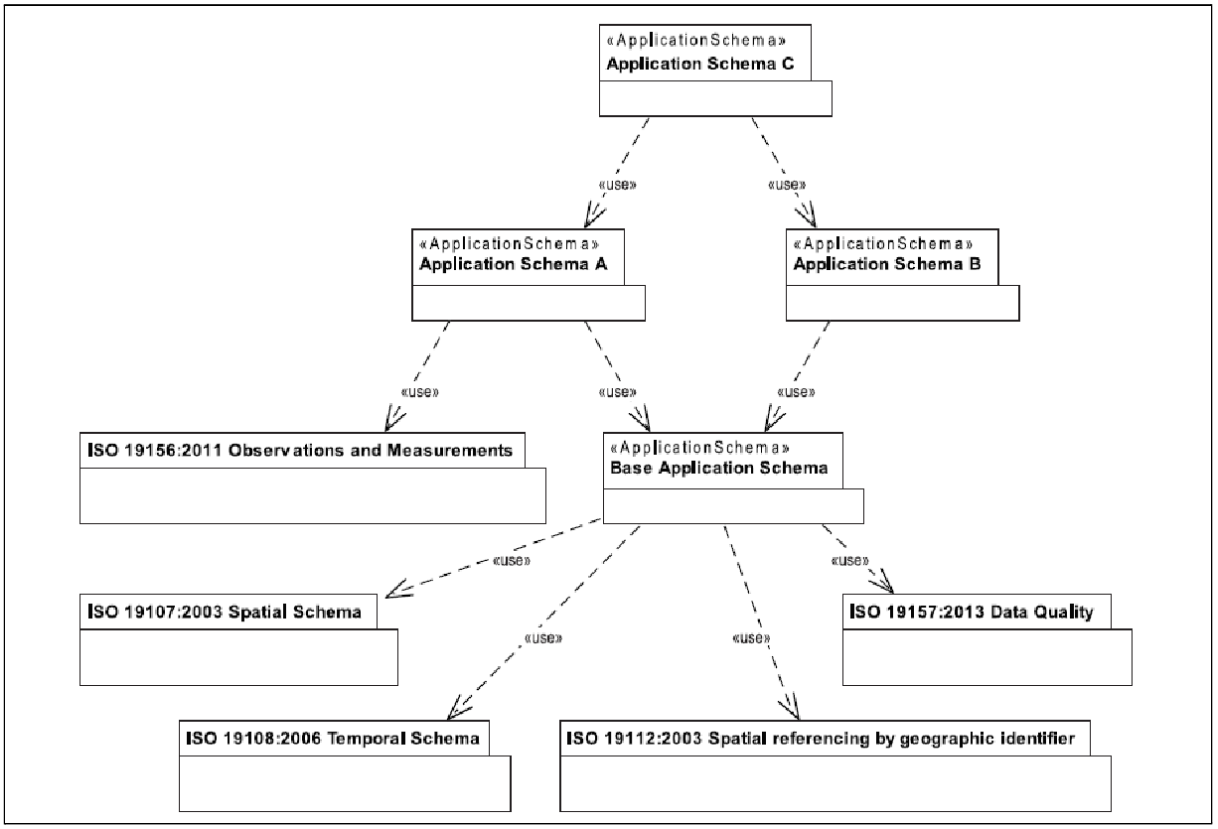
Pro naplnění těchto dvou účelů je potřeba postupného procesu pro vytváření aplikačního schématu. Příslušné kroky mohou být stručně popsány následovně:

1) Prozkoumání požadavků ze zamýšlené oblasti aplikace (univerza diskurzu).
2) Vytvoření konceptuálního modelu této aplikace s pojmy definovanými v Obecném modelu vzhledu jevů. Tento úkol sestává z identifikování typů vzhledu jevů, jejich vlastností a omezení.
3) Popsání aplikačního schématu ve formálním modelovacím jazyku (např. UML a OCL) podle pravidel definovaných v této mezinárodní normě.
4) Integrování formálního aplikačního schématu s dalšími normalizovanými schématy (prostorovým schématem, schématem kvality, atd.) do kompletního aplikačního schématu.

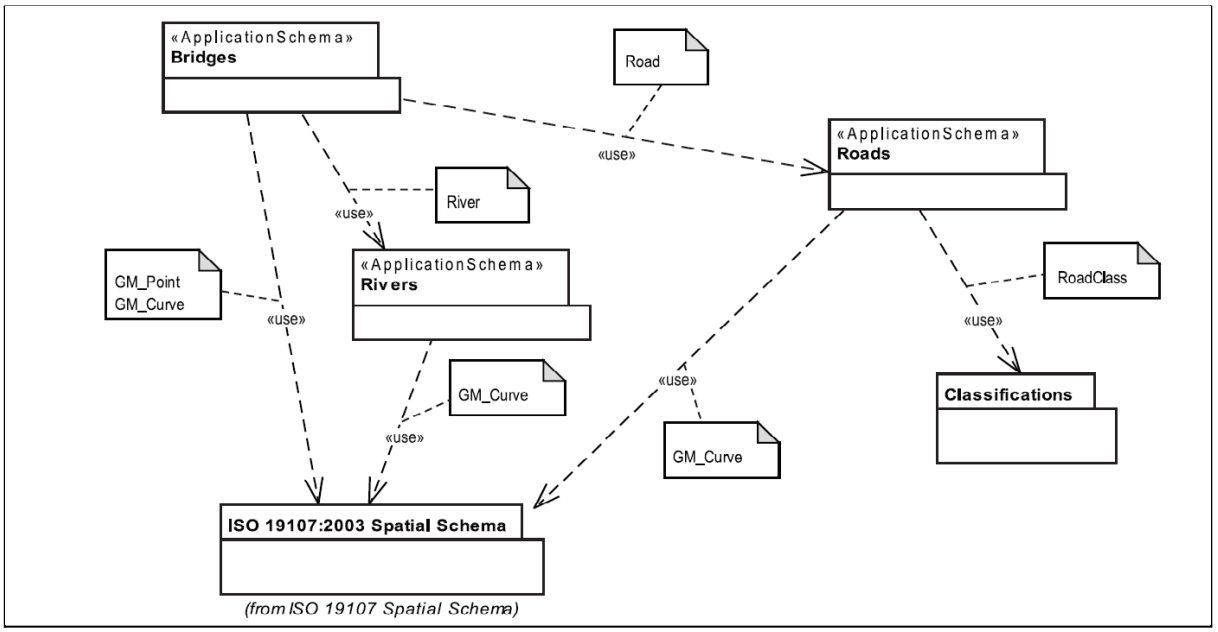
Tento proces může zahrnovat iterace a smyčky vznikající na základě zpětné vazby. Při vývoji velkého informačního modelu (kterým NaSaPO bezesporu je) je práce často rozdělena do nezávislých částí, které mohou být integrovány definovaným rozhraním. Kompletní aplikace může být sestavena z více než jednoho aplikačního schématu. Navíc každé aplikační schéma může odkazovat na standardizovaná schémata. Takováto organizace informačního modelu je využívána, aby bylo zamezeno vytváření velkých a složitých schémat. Další výhodou je možnost jednoduššího delegování změn modelu, které se může týkat různých aspektů, odpovědným orgánům. Úplná definice datové struktury určité aplikace sestává z aplikačního schématu integrovaného s ostatními standardními schématy, na které přímo nebo nepřímo odkazuje.

PŘÍKLAD: Obrázek 5 ukazuje aplikační schéma, které používá prvky ze schémat definovaných v jiných mezinárodních normách ISO geografické informace a z jiných aplikačních schémat.

PŘÍKLAD: Obrázek 6 zobrazuje, že aplikace zaměřená na doménu zájmu obsahující silnice, řeky a mosty využívá více aplikačních schémat, které musí být vzájemně provázané a které odkazují na standardní prostorové schéma.



Obrázek 5: Příklad integrace aplikačního schématu



Obrázek 6: Příklad integrace aplikačního schématu

Dalším pravidlem modelování aplikačních schémat je nutnost popsání každého typu objektu z katalogu v alespoň jednom aplikačním schématu. V něm je také na konceptuální úrovni definováno, jaké atributy, relace a případně operace obsahuje. Aplikační schéma obsahuje úplný a přesný popis všech typů objektů, které jsou v něm obsaženy. Každé aplikační schéma popsané v NaSaPO by mělo odpovídat obecnému modelu typů vzhledu jevů, který je popsaný v kapitole výše a vyspecifikovaný v normě ISO 19109:2015. Tato norma popisuje také veškeré požadavky, které jsou potřebné pro správnou interpretaci aplikačních schémat. Jde zejména o využití a popis jazyku konceptuálního schématu, modelování datových struktur tak, aby byl možný přenos relevantních informací, ustanovuje pravidla pro pojmenovávání a verzování aplikačních schémat, pravidla pro katalog typů vzhledu jevů.

Důležité pro správné zpracování aplikačních schémat je také poskytnutí dostatečné dokumentace pro všechny prvky ve schématu aplikace, s čímž souvisí i poskytnutí odkazů z typu prostorových objektů na definici typu v příslušném katalogu prvků. Proces definování typů prostorových objektů je blíže specifikován ve výsledku V02 – Terminologické a věcné vymezení objektů NaSaPO. Proces definování aplikačních schémat pro jednotlivé typy prostorových objektů je následující:

- Nejprve je identifikován relevantní koncept v rámci tématu pro typ prostorového objektu.
- Provede se analýza, která určí, zda již neexistuje podobný koncept.
- Pokud existuje podobný koncept, je potřeba se rozhodnout, zda je vhodné koncept rozšířit nebo vytvořit nový.

Aplikační schéma může obsahovat stejný typ objektu jako jiné aplikační schéma. Pokud jde v implementační rovině o stejnou instanci objektu, má tento objekt v obou aplikačních schématech shodný identifikátor a liší se pouze atributy použitými pro danou doménu zájmu.

Jak už bylo uvedeno výše, základní jednotka geografické informace je nazývána vzhled jevu. Mezinárodní norma ISO 19109 udávající pravidla pro vytváření aplikačních schémat rozlišuje čtyři aspekty definování vzhledů jevů: definice nebo popis použitý pro jejich seskupení do typů, atributy asociované s každým typem, vztahy mezi typy a chování vzhledů jevů. Vzhledy jevů mají často jedinou hodnotu a jsou statické. V případě vzhledu jevu, jehož vlastnosti se mění spojitě v prostoru a čase, je výhodnější použít v aplikaci model formalizovaný jako pokrytí. Model pokrytí je podrobně popsán v ISO 19123 a aplikační schéma může zahrnovat obojí, a to typy vzhledů jevů stejně jako typy pokrytí. Tato problematika souvisí s obecně známými pojmy vektorového a rastrového přístupu k datům v prostředí GIS.

- Mezinárodní norma ISO 19109 definuje následující pravidla pro tvorbu aplikačních schémat:
- Aplikační schéma musí být vyjádřeno v Jazyku konceptuálního schématu. Musí popsat strukturu a obsah množiny dat, která reprezentuje univerzum diskurzu.
- Vývojář aplikačního schématu může použít definice z katalogů vzhledů jevů, které již existují. Tam, kde aplikační schéma integruje komponenty z jiných schémat, musí být závislosti explicitně zaznamenány použitím mechanismů poskytnutých jazykem konceptuálního schématu.
- Vzhledy jevů musí být modelovány jako instance metatřídy FeatureType, jak bylo definováno na obrázku 2. Vzhledy jevu nesmí být modelovány jako specializace GM_Object (ISO 19107:2003) nebo TM_Object (ISO 19108:2002). Vzhled jevu musí mít název, který je jedinečný v aplikačním schématu.
- Název každého typu vlastnosti by měl být jedinečný v aplikačním schématu, nebo by měl mít stejný záměr s ohledem na CLASS vlastnící vlastnost jako jakákoliv jiná vlastnost se stejným názvem

v aplikačním schématu. Typ vlastnosti může být použit jedním nebo více typy vzhledů jevů, pokud je definice typu vlastnosti ve všech kontextech stejná.

- Atributy vzhledů jevů musí být modelovány jako instance metatřídy AttributeType, jak je definováno na obrázku 2.
- Operace musí být modelovány jako instance metatřídy Operation, jak je definováno na obrázku 2.
- Role asociace vzhledu jevu musí být modelovány jako instance metatřídy FeatureAssociationRole, jak je definováno na obrázku 2.
- Proces „přiřazení hodnoty“ musí být modelován jako instance třídy asociované s metatřídou ValueAssignment.
- V instanci „přiřazení hodnoty“ musí být vlastnost, ke které je přiřazena hodnota, konzistentní s typem vzhledu jevu, s kterým je asociována.
- Asociace vzhledů jevů musí být modelovány jako instance metatřídy FeatureAssociationType, jak je definováno na obrázku 2.
- Vztahy dědičnosti musí být modelovány jako instance metatřídy InheritanceRelation, jak definuje obrázek 2.

Mezinárodní norma ISO 19109:2015 Geografická informace – Pravidla pro aplikační schéma nestandardizuje jakéhokoliv aplikační schémata. Norma má za účel definovat pravidla pro vytváření aplikačních schémat konzistentním způsobem. Zajišťuje usnadnění získávání, zpracování, analyzování, zpřístupňování, prezentování a přenášení geografických dat mezi různými uživateli, systémy a místy. V případě přenosu a výměny dat jsou pravidla této mezinárodní normy použita dodavateli a uživateli geografických dat k tomu, aby:

- vybudovali aplikační schéma přenosu dat pro jejich výměnu;
- interpretovali sémantiky přenášené množiny dat při respektování uživatelských místních dat a obsahu a struktury dat;
- určili nezbytné transformace mezi dvěma datovými sadami.

Pravidla této mezinárodní normy slouží uživatelům aplikací s požadavky na podobná data v procesu vytváření obecného aplikačního schématu pro rozhraní mezi jejich systémy a daty. Mapování z jednoho aplikačního schématu do jiného aplikačního schématu může být obtížné a dokonce nemožné, pokud jsou tato dvě schémata příliš divergentní. Mapování je usnadněno, pokud je aplikační schéma použité v rámci nějakého systému navrženo tak, že zohledňuje také požadavky na výměnu dat. Tato pravidla mohou být použita také pro budování aplikačního schématu v rámci systému, přestože taková aplikační schémata nejsou součástí obsahu této mezinárodní normy. Vytvoření aplikačního schématu je proces. Obsah aplikačního schématu musí být uspořádán v souladu s vnímáním reality v univerzu diskurzu. Aplikační schéma definuje strukturu a obsah dat. To je vyjádřeno v jazyce konceptuálního schématu (UML).

7.1 Jazyk konceptuálního schématu

Jazykem konceptuálního schématu se zaobírá norma ISO/TS 19103:2005, Geographic information – Conceptual schema language. Jazyk konceptuálního schématu se používá k popisu konceptuálního schématu, přičemž jde o formální popis s možností analýzy ze strany člověka i počítače. Jazyk konceptuálního schématu zahrnuje všechny jazykové konstrukty, které jsou nezbytné k vytvoření konceptuálního schématu a k manipulaci s jeho obsahem. Je založen na konceptuálním formalismu, který stanovuje pravidla, omezení,

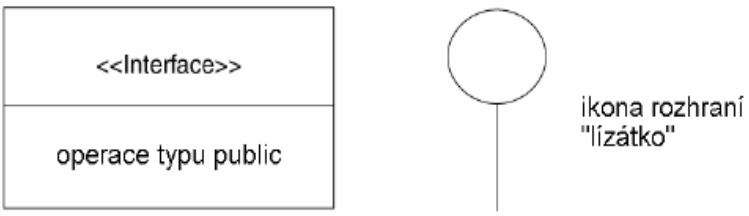
mechanismy dědičnosti, události, funkce, procesy a další prvky, které tvoří jazyk konceptuálního schématu. Tyto prvky mohou být dále využity při tvorbě konceptuálních schémat, která popisují daný informační systém za předpokladu, že jsou zvoleným univerzem diskurzu. Principy konceptuálního formalismu jsou podrobněji rozebírány ve výsledku V02 – Terminologicke a vecne vymezeni objektu NaSaPO. Pro účely projektu NaSaPO byl vybrán jazyk UML (Unified Modeling Language).

7.1.1 Pravidla pro modelování v UML

Norma ISO 19103:2015, Geographic information – Conceptual schema language se dopodrobna zaobírá profilem UML v oboru geografické informace. Stanovuje pravidla a směrnice pro použití UML v této oblasti a definuje specifické aspekty použití UML, které jsou ve shodě s ISO normami pro geografické informace. Norma je založena na normě ISO/IEC 19501:2005. V rámci tohoto výsledku se zaměříme pouze na UML struktury, které se vyskytují v konceptuálních schématech NaSaPO.

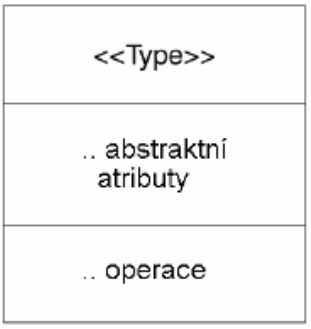
7.1.1.1 Třídy

Třída je popis množiny objektů, které sdílejí stejné atributy, operace, metody, vztahy a sémantiku. Třída reprezentuje modelovaný pojem. Tento pojem může být v závislosti na druhu modelu založen na reálném světě (pro konceptuální model) nebo může být založen na implementaci buď pojmů na platformách nezávislého systému (pro modely specifikací) nebo pojmů pro platformu specifického systému (pro implementační modely). V případě tohoto dokumentu se budeme výhradně bavit o variantě sloužící pro konceptuální model. Klasifikátor je zobecnění třídy, které zahrnuje jiné třídám podobné prvky, jako jsou například datové typy, účastníci a komponenty. Třída UML má název, množinu atributů, množinu operací a omezení. Třída se může zúčastnit asociací. UML definuje pomocí stereotypů <<Interface>> a <<Type>> dva speciální druhy tříd. <<Interface>> je množina operací, která charakterizuje chování prvku. <<Interface>> může být reprezentován takzvaným „lízátkem“ a neobsahuje v UML žádné atributy ani nespecifikuje jeho implementaci.



Obrázek 9: Znázornění stereotypu <<Interface>>

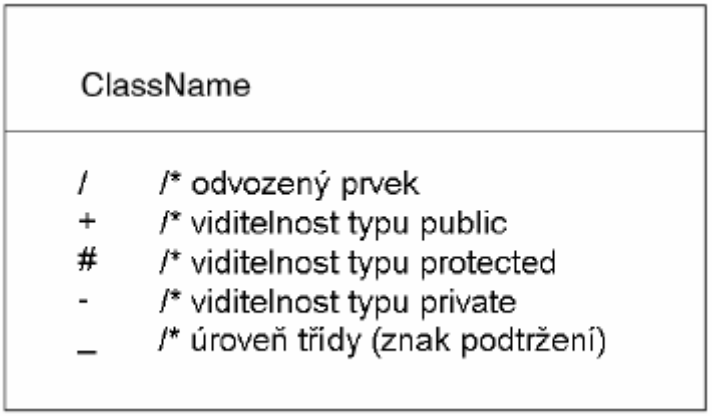
<<Type>> je stereotypovaná třída používaná pro specifikaci domény instancí (objektů) spolu s operacemi aplikovatelnými na tyto objekty. Typ může mít abstraktní atributy a asociace. Abstraktní atribut znamená, že jejich specifikací není myšleno, že mají být konkrétně implementovány jako proměnné instancí. <<Interface>> nemůže mít atributy, ale pojem abstraktní atributy v <<Type>> může být použit jako místo k popisu příslušného abstraktního stavu. Tak například bod může mít abstraktní stav pro souřadnice x a y, ale skutečnou reprezentaci založenou na radiánech a stupních, která je použita k výpočtu x a y. Pro objekty, které mohou být předávány hodnotou, jako je tomu například při přenosu rysů, je nezbytné, aby o abstraktním stavu byla poskytnuta dostatečná informace. To znamená, že nemůže být použita „jen na rozhraní“ založená specifikace. Stereotyp <<Type>> může být použit pro specifikaci rozhraní a abstraktního stavu.



Obrázek 10: Znázornění stereotypu <<Type>>

Rozhraní a typy se mohou používat podle normativních stereotypů UML, znamenajících množinu operací, případně množinu operací a abstraktních atributů. Abstraktní třída definuje třídu polymorfních objektů a nemůže být konkretizována. Abstraktní třída se odlišuje od <<Interface>> tím, že může mít atributy a může implementovat některé z jejích operací. <<Interface>> je ekvivalentem abstraktní třídy bez atributů a bez implementace operací. Abstraktní třída je specifikována tím, že má classname (název třídy) uvedený kurzívou nebo označenou hodnotou {Abstract} umístěnou vedle tohoto classname.

Viditelnost atributu, operace a konce asociace, definovaná v kontextu třídy, může být omezena různými způsoby. Viditelnost typu private („-“) znamená, že atribut nebo operace je viditelná pouze v rámci třídy, v níž je definována. Asociace je dosažitelná pouze ze třídy na jejím zdrojovém konci. Viditelnost typu protected („#“) znamená, že atribut nebo operace je viditelná pouze v rámci třídy, v níž je definována a jejím podtřídám. Asociace je dosažitelná pouze ze třídy na jejím zdrojovém konci a z jejich podtříd. Viditelnost typu public („+“) znamená, že atribut nebo operace je viditelná z vnějšku jakékoliv třídy v rámci téhož balíčku, v jakém je tato třída definována. Asociace je dosažitelná z jakékoliv třídy sdružené s touto třídou, v jejímž rámci je definována. Viditelnosti typu private a protected se v ISO normách geografické informace obyčejně nepoužívají.



Obrázek 11: Přehled symbolů pro definice odvozeného prvků, viditelnosti a úrovně třídy

Viditelnost atributů, operací a konců asociací je reprezentována symbolem, který předchází název prvku ve formátu symbol mezera název symbolu viditelnosti (např. # protected). Atribut nebo operaci, které jsou na úrovni třídy lze definovat jejich podtržením. Pokud prvku modelu předchází symbol lomítka („/“), znamená to, že se jedná o prvek odvozený (vypočtený) z jiného prvku.

7.1.1.2 Atributy

Notace UML pro atribut má následující formu:

[<<stereotype>>][visibility]name[multiplicity][:type][=initial value][{property-string}],

kde jsou volitelné prvky uzavřeny v lomených závorkách a:

- stereotype je název stereotypu, k němuž tento atribut náleží, pokud existuje;
- visibility specifikuje viditelnost tohoto atributu;
- name je znakový řetězec, který identifikuje název tohoto atributu;
- multiplicity specifikuje počet hodnot, který instance třídy může mít pro daný atribut;
- type-expression identifikuje datový typ tohoto atributu;
- initial value specifikuje automaticky určenou hodnotu pro tento atribut;
- property-string obsahuje všechny označené hodnoty použitě pro tento atribut.

Atribut by měl být v kontextu třídy a jejich nadtypů jedinečný, pokud není atributem redefinovaným z nadtypu. Pokud je atribut potlačen, mělo by se s ním zacházet jako s abstraktním atributem, který může být vypočítán na požádání podle pravidel definovaných pro jeho odvození. Datový typ musí být vždy definovaný, neexistuje žádný automaticky určený typ. Pokud neexistuje žádná explicitní násobnost, předpokládá se, že násobnost je rovna jedné. Atribut může definovat automaticky určenou hodnotu, která se použije tehdy, když se vytváří objekt onoho datového typu. takto určené hodnoty jsou definovány v definici atributu UML explicitními automaticky určenými hodnotami. Tyto hodnoty jsou zapotřebí v některých situacích v konceptuálních modelech, které jsou ve shodě s ISO normami geografické informace.

7.1.1.3 Datové typy

Základní datové typy mohou být rozděleny do tří kategorií uvedených níže, jedná se o:

- Jednoduché typy – základní typy pro reprezentaci hodnot, jako jsou CharacterString (znakový řetězec), Integer (celé číslo), Boolean, Date, Time apod.
- Typy implementací a sbírek – typy pro struktury implementací a reprezentací jako jsou Name, Record a typy pro reprezentaci vícenásobných výskytů jiných typů, například Set, Bag nebo Sequence.
- Odvozené typy – jde o typy měr a jednotek měření.

Základní typy jsou definovány jako abstraktní typy, přičemž název třídy je vždy zobrazen kurzívou. Podrobný popis všech datových typů je součástí normy ISO 19103:2015, Geographic information – Conceptual schema language. Přehled jednoduchých typů, které jsou použity při konceptuálním modelování schémat NaSaPO jsou popsány níže a znázorněny na přiložených obrázcích. Předpokládáme, že množinu těchto typů bude možné s postupem projektu rozšiřovat a je možné, že se objeví v pozdějších verzích tohoto dokumentu.

- Number (číslo) je základní typ pro všechna číselná dat, udávající základní algebraické operace. Number je abstraktní třídou a nadtřídou pro další číselná data.
- Integer (celé číslo) je celé číslo se znaménkem. Délka celého čísla je závislá na zapouzdření a použití a sestává se z přesné celočíselné hodnoty bez mantisy.
- Decimal (desítkové číslo) je typ desítkových dat, v němž toto číslo reprezentuje přesnou hodnotu jako konečnou reprezentaci desítkového čísla. Desítkové číslo se často používá u dat, které pracují s peněžními hodnotami (od reálného čísla se liší tím, že jde o přesné číslo, zatímco reálné číslo je jen přibližná hodnota).
- Real (reálné číslo) je číslo se znaménkem (pohyblivá řádová čárka), sestávající z mantisy a exponentu, který reprezentuje hodnotu preciznosti danou počtem udaných čislic. Nejde však nutně o přesnou hodnotu. Délka reálného čísla je závislá na zapouzdření a použití. běžná implementace dvojkového reálného čísla používá základ 2. Protože reálná čísla mohou aproximovat jakoukoliv míru, kde není absolutní přesnost možná, je tato forma číselného typu nejčastěji používaná pro míry. V případech, kde je přesnost nutná je dávána přednost desítkové reprezentaci. Reálné číslo může být uvažováno jako celočíselná část následovaná mantisou udanou v násobcích mocniny ½.
- Vector (vektor) je množina čísel, které nazýváme souřadnice a které reprezentují polohu v souřadnicovém systému.
- CharacterString (znakový řetězec) je libovolná posloupnost znaků, které se nacházejí v souboru znaků ISO/IEC 10646 nebo ISO 8859.
- Date (kalendářní datum) udává hodnoty pro rok, měsíc a den. Kódování znaků je specifikováno v ISO 8601, ISO 19118 a ISO 19136. Stejně tomu je i u typů Time a DateTime.
- Time (čas) je udán hodinou, minutou a sekundou.
- DateTime je kombinace kalendářního data a času.
- Boolean je specifikován hodnotami TRUE (pravdivá hodnota) a FALSE (nepravdivá hodnota).
- Logical (logická hodnota) je hodnota specifikující TRUE, FALSE nebo MAYBE (neznámá hodnota).
- Probability (pravděpodobnost) je reprezentována jako číslo větší nebo rovné 0 a menší nebo rovné 1.
- Multiplicity (násobnost) definuje dolní a horní hranici počtu možných instancí hodnot.

7.1.1.4 Operace

Operace jsou specifikovány textovým řetězcem, který může být rozdělen do prvků, které popisují vlastnosti operace: *visibility name (parameter-list) : return-type-expression [{ property-string }]*, kde

- volitelné prvky jsou uzavřeny v lomených závorkách,
- visibility specifikuje viditelnost operace,
- name je znakový řetězec, který identifikuje název operace,
- parameter-list je seznam parametrů,
- return-type-expression specifikuje datový typ nebo typy hodnoty vracené operací a
- property-string obsahuje všechny označené hodnoty aplikované na operaci.

Prvek property-list má tvar: *[kind] name : type-expression [= default-value]*, kde

- kind je in, out nebo inout s automatickým určením v případě, že chybí,
- name je název tohoto parametru,
- type-expression identifikuje datový typ tohoto parametru a
- default-value je počáteční hodnota pro tento parametr.

Hodnoty pro kind mají následující významy:

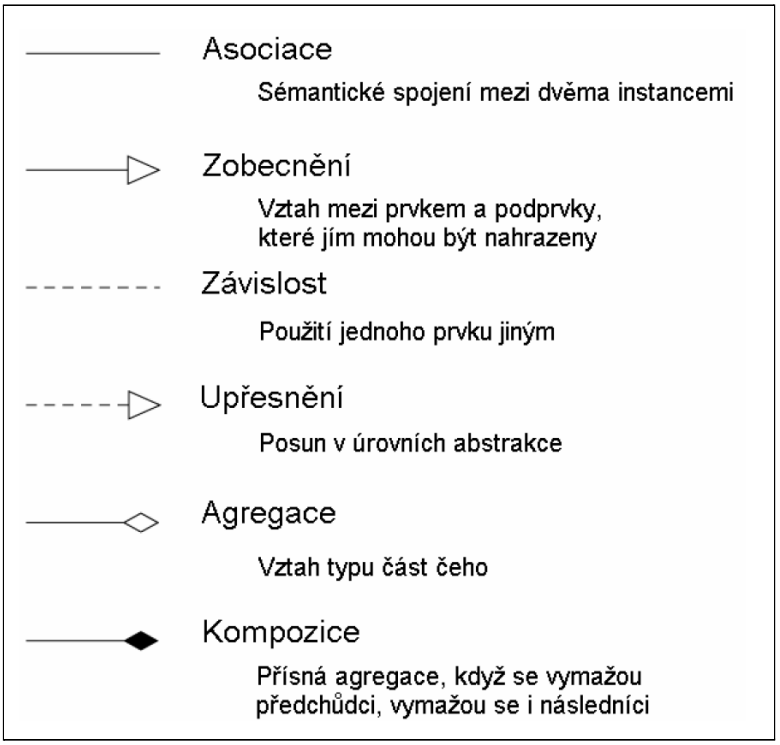
- in je hodnota parametru operací, která se čte, ale nemodifikuje,
- out je hodnota parametru, která se vytváří operací a může číst po návratu operace do místa volání,
- inout je hodnota parametru, která se operací čte, modifikuje a může se číst po návratu operace do místa volání. Pro výjimky se používá stereotyp <<Exception>>.

V UML jsou pro operace definovány čtyři normativní vlastnosti, isQuery, sequential, guarded a concurrent.

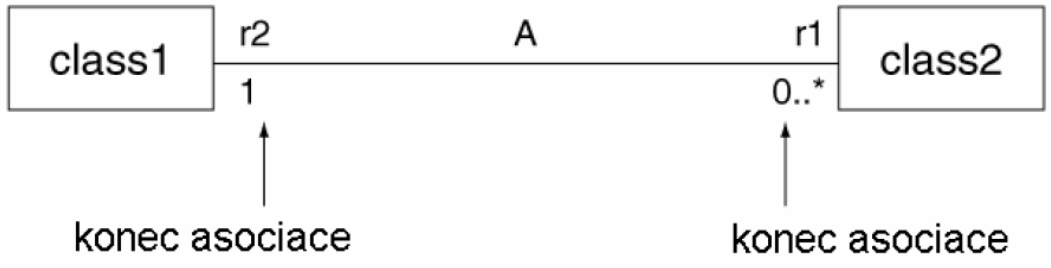
7.1.1.5 Vztahy

Vztah je v UML konkretizované sémantické spojení mezi prvky modelu. Druhy vztahů zahrnují asociaci, zobecnění, agregaci/kompozici a několik druhů seskupených pod závislostí. Je třeba rozlišovat termín vztah a specifičtější termín asociace. Oba jsou definovány pro účely propojení tříd, asociace je ale vyhrazena pro ty vztahy, které jsou v realitě spojkami instance s instancí. Zobecnění, realizace a závislost jsou vztahy dvou tříd, agregace a další vztahy objektů jsou restriktivnější a tedy nazývány asociace.

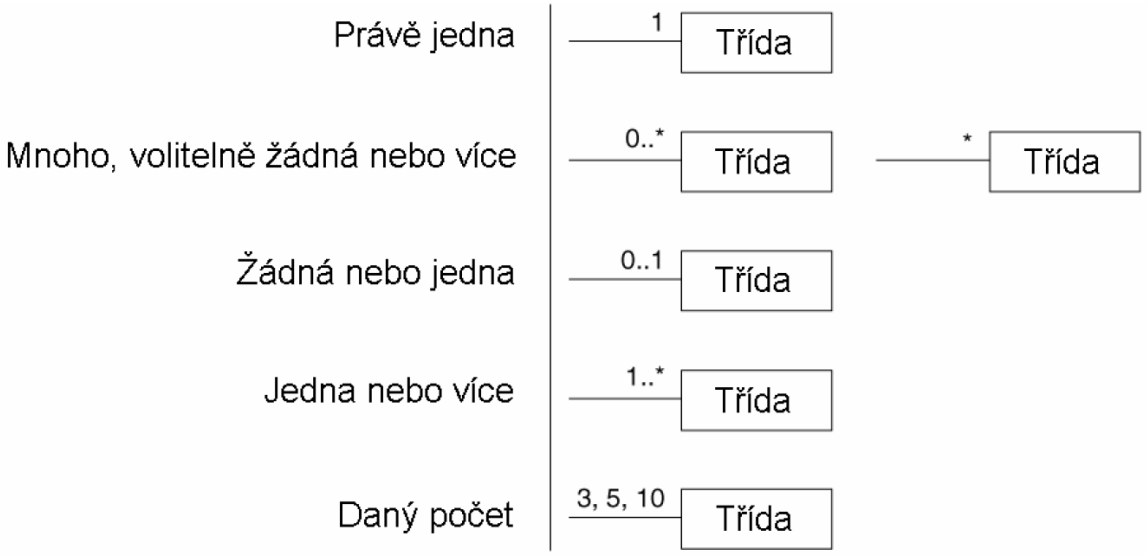
Asociace je v UML sémantický vztah mezi dvěma nebo více klasifikátory, který se týká spojení mezi jejich instancemi. Asociace se používá k popisu vztahu mezi dvěma nebo více třídami. UML definuje kromě běžné asociace dva speciální typy asociací, které se nazývají agregace a kompozice. Tyto tři typy mají rozdílné sémantiky. Běžná asociace by se měla používat k reprezentován celkového vztahu mezi dvěma třídami. Asociace agregace a kompozice by se měly používat k vytváření vztahů mezi dvěma třídami „je součástí“. Binární asociace má název a dva konce asociace. Konec asociace má název role, údaj o násobnosti a volitelná symbol agregace. Ke třídě by měl být vždy připojen konec asociace.



Obrázek 12: Přehled druhů vztahů

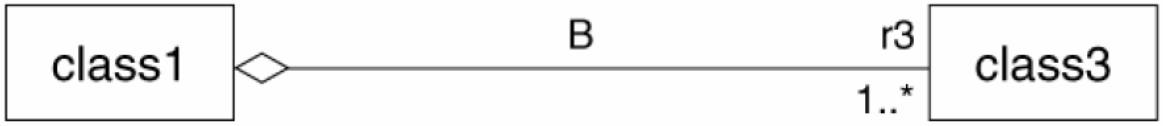


Obrázek 13: Ukázka asociace, kdy A je asociace a r1, r2 jsou názvy rolí



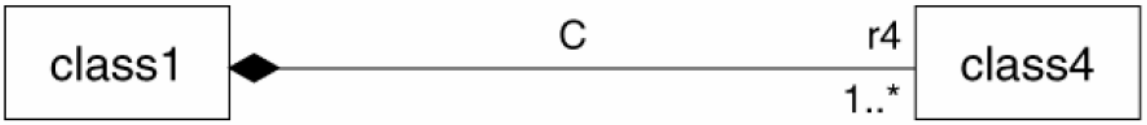
Obrázek 14: Znázornění kardinality v UML

Asociace agregace je vztah mezi dvěma třídami, v němž jedna z těchto tříd hraje roli kontejneru a druhá hraje roli obsahu kontejneru. Symbol asociace, znázorněný prázdným kosočtvercem, indikuje, že třída X je agregací sestávající z třídy Y. Jinými slovy to znamená, že třída Y je součásti třídy X (a v modelu instancí to znamená, že objekty třídy X budou obsahovat jeden nebo více objektů třídy Y).



Obrázek 15: Znázornění agregace, kde B je asociace, r3 je název role a 1..* udává kardinalitu

Asociace kompozice je přísná agregace. Pokud je v asociaci kompozice vymazán objekt kontejneru, vymažou se také všechny objekty obsahu kontejneru. Asociace kompozice by se měla použít tehdy, když objekty, reprezentující části objektu kontejneru, nemohou existovat bez objektu kontejneru.



Obrázek 16: Znázornění kompozice, kde C je asociace, r4 je název role a 1..* udává kardinalitu

Norma ISO 19103:2015 včetně jejích příloh jde ještě do větších podrobností, nicméně pro účely tohoto dokumentu jsme se snažili využít jen ty části, které jsou nezbytné pro pochopení konceptuálního modelu NaSaPO. V případě rozšíření modelu předpokládáme, že proběhne i revize tohoto dokumentu a následné rozšíření o další kapitoly. Jedná se zejména o podrobnější popis stereotypů, omezení, balíčků apod., které mohou být také součástí UML. Stávající popis UML mi měl vždy odrážet potřebnou úroveň podrobnosti pro plné pochopení jednotlivých modelů v přílohách dokumentace.

7.2 Katalog vzhledů jevů

Katalogy vzhledů jevů (feature catalogues) definují typy vzhledů jevů, jejich operace, atributy a asociace, které jsou nezbytné pro přeměnu dat na použitelné informace. Problematika týkající se katalogů vzhledů jevů je částečně rozebrána ve výsledku V02 – Terminologicke a vecne vymezeni objektu NaSaPO. Ačkoliv předpokládáme, že samotné katalogy budou vytvářeny pro jednotlivé obory zájmů a budou vznikat v rámci návazných projektů, byla vytvořena tabulka s návrhem základních okruhů, skupin a samotných typů vzhledů jevů. Inspirací pro tabulku byl datový model DTM, přičemž bylo myšleno na 3D charakteristiky datového modelu NaSaPO. Je třeba zdůraznit, že se jedná opravdu o návrh, jehož struktura se bude s velkou pravděpodobností zásadně měnit na základě navazujících analýz a jejich výstupů. Níže je zpracován návrh vzhledu jevu zpracovaného v rámci katalogu vzhledů jevů NaSaPO. Samotný katalog by měl být poté zpracován a pojat jako aplikace nebo součást informačního systému, který bude obsahovat přehled všech objektů s informacemi o nich. I na výstupu je vidět, proč je to vhodnější varianta. Důvodem je dědění atributů a vazeb, které se prolínají celým modelem, přičemž přehlednost katalogu v papírové formě je z toho důvodu menší než ve formě digitální. Pro příklad byla vybrána budova.

Tabulka 3:Základní informace o vzhledu objektu

Název objektu: Budova
Definice: Samostatně stojící, trvalá zastřešená stavba, která je obvykle zděná a lze do něj vstoupit lidmi. Je určena pro obsazení lidmi (např. bydlení, místo práce nebo rekreace).
Aplikační schéma: Budovy
Nadtřída: AbstractBuilding
Typ: «TopLevelFeatureType»

Základní informace obsahují pouze název typu vzhledu jevu, jeho definici, umístěná v aplikačním schématu nebo schématech, nadtřídu a typ. Typ vzhledu jevu budova nemá vlastní atributy ani vazby. Níže je sepsán seznam všech děděných atributů a vazeb v rámci modelu CityGML.

Tabulka 4: Zděděné atributy a asociace z třídy AbstractBuilding

Třída dědění: AbstractBuilding	
Atributy	**Název:** buildingClassValue **Popis:** Označuje konkrétní typ členění budovy **Četnost:** [0..1]
	Název: buildingFunctionValue **Popis:** Určuje zamýšlené účely budovy **Četnost:** [0..*]
	Název: buildingUsageValue **Popis:** Určuje skutečné použití budovy **Četnost:** [0..*]
	Název: roofType **Popis:** Označuje tvar střechy budovy **Četnost:** [0..1]
	Název: storeysAboveGround **Popis:** Označuje počet podlaží umístěných nad úrovní terénu **Četnost:** [0..1]
	Název: storeysBelowGround **Popis:** Označuje počet podlaží umístěných pod úrovní terénu **Četnost:** [0..1]
	Název: storeyHeightsAboveGround **Popis:** Uvádí výšky každého nadzemního podlaží. První hodnota v seznamu označuje výšku podlaží nejblíže úrovni terénu, poslední hodnota označuje výšku nejvzdálenější **Četnost:** [0..1]

	Název: storeyHeightsBelowGround **Popis:** Uvádí výšku každého podlaží pod zemí. První hodnota v seznamu označuje výšku podlaží nejblíže úrovni terénu, poslední hodnota označuje výšku nejvzdálenější **Četnost:** [0..1]
Asociace	**Název:** buildingInstallation **Popis:** Vazba na instalace v budově **Četnost:** [1..*]
	Název: buildingFurniture **Popis:** Vazba na nábytek v budově **Četnost:** [1..*]
	Název: buildingConstructiveElement **Popis:** Vazba na konstrukční prvky budovy **Četnost:** [1..*]
	Název: address **Popis:** Vazba na adresy budovy **Četnost:** [1..*]
	Název: buildingRoom **Popis:** Vazba na pokoje v budově **Četnost:** [1..*]
	Název: buildingSubdivision **Popis:** Vazba na logické části budovy **Četnost:** [1..*]

Tabulka 5: Zděděné atributy a asociace z třídy AbstractConstruction

Třída dědění: AbstractConstruction	
Atributy	**Název:** conditionOfConstruction **Popis:** Označuje stav životního cyklu stavby **Četnost:** [0..1]
	Název: constructionEvent **Popis:** Popisuje konkrétní události v průběhu životnosti stavby **Četnost:** [0..*]
	Název: dateOfConstruction **Popis:** Označuje datum, kdy byla stavba dokončena **Četnost:** [0..1]
	Název: dateOfDemolition **Popis:** Označuje datum, kdy byla stavba zbořena **Četnost:** [0..1]
	Název: elevation **Popis:** Určuje kvalifikované výšky stavby ve vztahu k dobře definovanému povrchu **Četnost:** [0..*]

	Název: height **Popis:** Určuje kvalifikované výšky konstrukce nad zemí nebo pod zemí **Četnost:** [0..*]
	Název: occupancy **Popis:** Poskytuje kvalifikované informace o pobytu osob, zvířat nebo jiných movitých věcí ve stavbě **Četnost:** [0..*]
Asociace	**Název:** boundary **Popis:** Vztahuje se k povrchům, které spojují konstrukci **Četnost:** [1..*]

Tabulka 6: Zděděné atributy a asociace z třídy AbstractOccupiedSpace

Třída dědění: AbstractOccupiedSpace	
Asociace	**Název:** lod3ImplicitRepresentation **Popis:** Vztahuje se k implicitní geometrii, která představuje obsazený prostor v úrovni detailu 3 **Četnost:** [0..1]
	Název: lod2ImplicitRepresentation **Popis:** Vztahuje se k implicitní geometrii, která představuje obsazený prostor v úrovni detailu 2 **Četnost:** [0..1]
	Název: lod1ImplicitRepresentation **Popis:** Vztahuje se k implicitní geometrii, která představuje obsazený prostor v úrovni detailu 1 **Četnost:** [0..1]

Tabulka 7: Zděděné atributy a asociace z třídy AbstractSpace

Třída dědění: AbstractSpace	
Atributy	**Název:** area **Popis:** Určuje kvalifikované oblasti související s prostorem **Četnost:** [0..*]
	Název: spaceType **Popis:** Určuje stupeň otevřenosti prostoru **Četnost:** [0..1]
	Název: volume **Popis:** Určuje kvalifikované objemy související s prostorem **Četnost:** [0..*]

Asociace	**Název:** boundary **Popis:** Vzrahuje se k povrchům, které ohraničují prostor **Četnost:** [1..*]
	Název: lod0MultiCurve **Popis:** Vztahuje se ke geometrii 3D MultiCurve, která představuje prostor v úrovni detailu 0 **Četnost:** [0..1]
	Název: lod2MultiCurve **Popis:** Vztahuje se ke geometrii 3D MultiCurve, která představuje prostor v úrovni detailu 2 **Četnost:** [0..1]
	Název: lod3MultiCurve **Popis:** Vztahuje se ke geometrii 3D MultiCurve, která představuje prostor v úrovni detailu 3 **Četnost:** [0..1]
	Název: lod0MultiSurface **Popis:** Vztahuje se ke geometrii 3D MultiSurface, která představuje prostor v úrovni detailu 0 **Četnost:** [0..1]
	Název: lod2MultiSurface **Popis:** Vztahuje se ke geometrii 3D MultiSurface, která představuje prostor v úrovni detailu 2 **Četnost:** [0..1]
	Název: lod3MultiSurface **Popis:** Vztahuje se ke geometrii 3D MultiSurface, která představuje prostor v úrovni detailu 3 **Četnost:** [0..1]
	Název: lod0Point **Popis:** Vztahuje se ke geometrii 3D bodu, která představuje prostor v úrovni detailu 0 **Četnost:** [0..1]
	Název: lod1Solid **Popis:** Vztahuje se ke geometrii 3D tělesa, která představuje prostor v úrovni detailu 1 **Četnost:** [0..1]
	Název: lod2Solid **Popis:** Vztahuje se ke geometrii 3D tělesa, která představuje prostor v úrovni detailu 2 **Četnost:** [0..1]
	Název: lod3Solid **Popis:** Vztahuje se ke geometrii 3D tělesa, která představuje prostor v úrovni detailu 3 **Četnost:** [0..1]

Tabulka 8: Zděděné atributy a asociace z třídy AbstractPhysicalSpace

Třída dědění: AbstractPhysicalSpace	
Asociace	**Název:** pointCloud **Popis:** Vztahuje se na 3D PointCloud, který představuje fyzický prostor **Četnost:** [0..1]
	Název: lod1TerrainIntersectionCurve **Popis:** Vztahuje se na 3D geometrii MultiCurve, která představuje terénní průsečíkovou křivku fyzického prostoru v úrovni detailu 1 **Četnost:** [0..1]
	Název: lod2TerrainIntersectionCurve **Popis:** Vztahuje se na 3D geometrii MultiCurve, která představuje terénní průsečíkovou křivku fyzického prostoru v úrovni detailu 2 **Četnost:** [0..1]
	Název: lod3TerrainIntersectionCurve **Popis:** Vztahuje se na 3D geometrii MultiCurve, která představuje terénní průsečíkovou křivku fyzického prostoru v úrovni detailu 3 **Četnost:** [0..1]

Tabulka 9: Zděděné atributy a asociace z třídy AbstractCityObject

Třída dědění: AbstractCityObject	
Atributy	**Název:** relativeToTerrain **Popis:** Popisuje vertikální polohu objektu vzhledem k okolnímu terénu **Četnost:** [0..1]
	Název: relativeToWater **Popis:** Popisuje vertikální polohu objektu vzhledem k okolní vodní hladině **Četnost:** [0..1]
Asociace	**Název:** appearance **Popis:** Vztahuje vzhledy k objektu **Četnost:** [1..*]
	Název: genericAttribute **Popis:** Vztahuje obecné atributy k objektu **Četnost:** [1..*]
	Název: generalizesTo **Popis:** Vztahuje zobecněné reprezentace stejného objektu reálného světa v různých úrovních detailu k objektu **Četnost:** [1..*]
	Název: externalReference **Popis:** Odkazuje na externí objekty v jiných informačních systémech, které mají vztah k objektu **Četnost:** [1..*]

	Název: relatedTo **Popis:** Popisuje, jak spolu objekty souvisí **Četnost:** [1..*]
	Název: dynamizer **Popis:** Vztahuje objekty z modulu Dynamizer k ostatním objektům **Četnost:** [1..*]

Tabulka 10: Zděděné atributy a asociace z třídy AbstractFeatureWithLifespan

Třída dědění: AbstractFeatureWithLifespan	
Atributy	**Název:** creationDate **Popis:** Označuje datum, kdy byl objekt přidán **Četnost:** [0..1]
	Název: terminationDate **Popis:** Označuje datum, kdy byl objekt odebrán **Četnost:** [0..1]
	Název: validFrom **Popis:** Označuje datum, kdy objekt začal existovat v reálném světě **Četnost:** [0..1]
	Název: validTo **Popis:** Označuje datum, kdy objekt přestal existovat v reálném světě **Četnost:** [0..1]

Tabulka 11: Zděděné atributy a asociace z třídy AbstractFeature

Třída dědění: AbstractFeature	
Atributy	**Název:** description **Popis:** Poskytuje další informace o objektu **Četnost:** [0..1]
	Název: featureID **Popis:** Určuje jedinečný identifikátor objektu v dané instanci **Četnost:** [1..1]
	Název: identifier **Popis:** Určuje jedinečný identifikátor na globální úrovni **Četnost:** [0..1]
	Název: name **Popis:** Určuje název objektu **Četnost:** [0..*]

Obdobným způsobem je potřeba postupovat u všech vzhledů jevů, které jsou součástí katalogu vzhledů jevů NaSaPO. Předpokladem pro úspěšné zpracování katalogu je shoda při definování vzhledů jevů, které budou součástí NaSaPO a následné zpracování dokumentace, která bude popisovat tyto vzhledy jevů v rámci katalogu vzhledů jevů NaSaPO. Řešitelský tým také doporučuje, aby byl v navazujících činnostech zpracován návrh informačního systému pro tento katalog (a ten byl následně implementován).

7.3 Obecný model vzhledu jevů

Obecný model vzhledu jevů (GFM – General Feature Model) je meta-modelem pro specifikaci a popis typů vzhledu jevů. Je vyjádřen v CSL, kterým je UML podle normy ISO 19103. Typy vzhledu jevů mají vlastnosti, kterými jsou jejich atributy, operace a asociační role. Vztahy mezi typy vzhledu jevů jsou definovány jejich asociacemi a dědičností. Typy vzhledu jevů mohou být dokumentovány v katalogu typů vzhledu jevů, přičemž obecný model vzhledu jevů může sloužit jako meta-model struktury katalogu vzhledu jevů. Obecný model vzhledu jevů také slouží jako definiční model pro vytváření aplikačních schémat. Vzhledy jevů reálného světa jsou reprezentovány dle ISO 19109 jako třídy konceptuálního modelu. Modelované objekty jsou pak instancemi vzhledu jevů. Každá instance vzhledu jevů má unikátní identifikátor (featureID), který přetrvává po celou dobu existence objektu a je shodný pro všechny verze téhož objektu reálného světa.

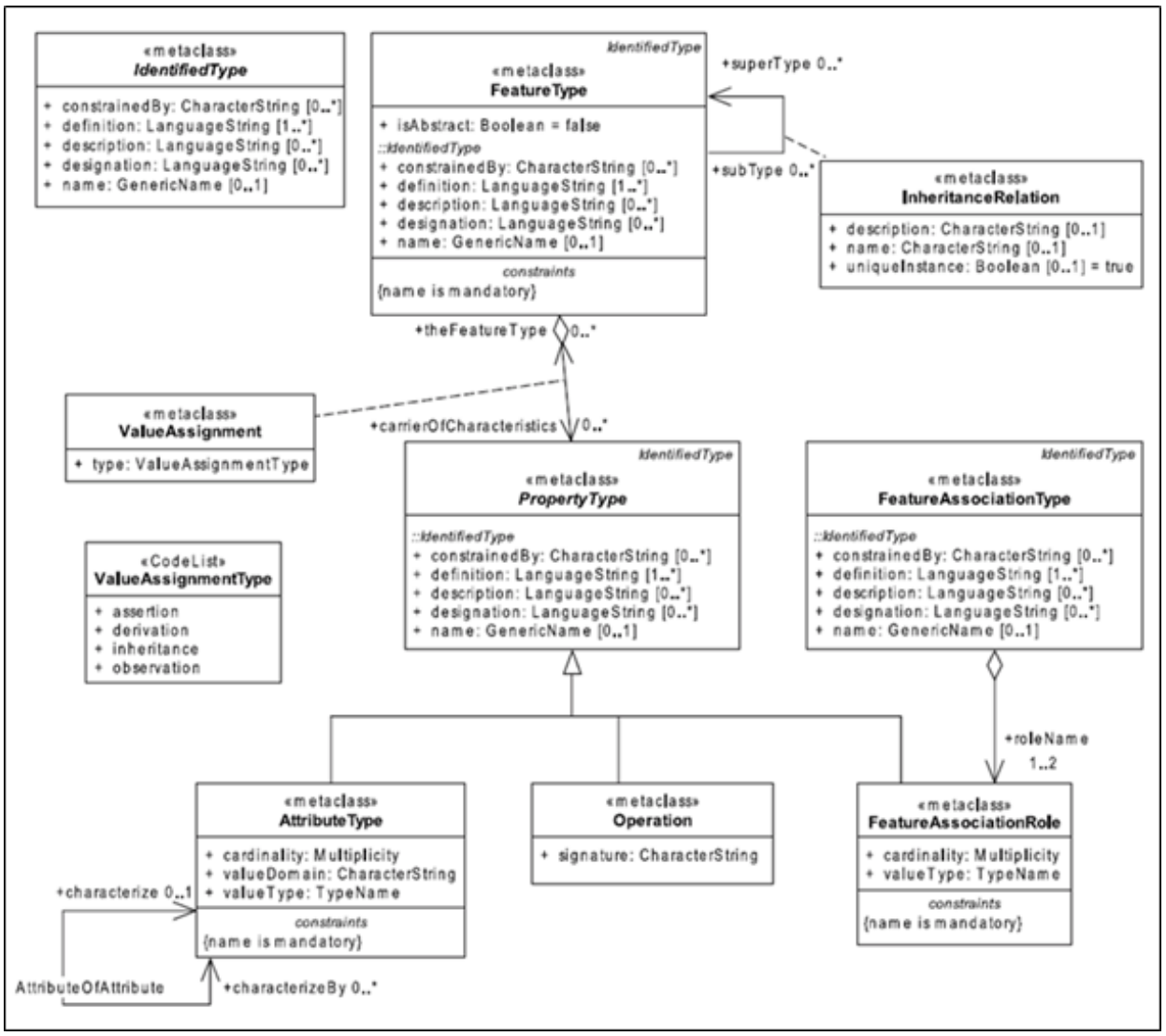
7.3.1 Struktura obecného modelu vzhledu jevů

Struktura obecného modelu vzhledu jevů je znázorněna na obrázku 17. Základní metatřídou obecného modelu vzhledu jevů je **IdentifiedType**. Jedná se o metatřídu, ze které ostatní klíčové třídy obecného modelu vzhledu jevů dědí některé společné identifikační a popisné atributy. Je nadtypem pro metatřídy FeatureType, PropertyType a FeatureAssociationType. Popisné atributy metatřídy IdentifiedType jsou:

- Name (název prvku) – obecně volitelný atribut, který je v některých specializacích požadován.
- Definition (definice) – stručná definice prvků (je mandatorní).
- Designation (ustanovení) – jazykové označení prvků, které slouží k doplnění názvu.
- Description (popis) – jedná se o popis prvku nad rámec informací vyžadovaných pro stručnou definici. Tento popis může pomoci objasnit rozsah a použití daného prvku.
- ConstrainedBy (omezení) – popis omezení v IdentifiedType nebo v dalších třídách.

FeatureType nebo-li vzhled jevu je abstrakcí jevů reálného světa. FeatureType je metatřída, která je konkretizovaná jako třída, které reprezentuje jednotlivé typy vzhledu jevů. Určitý typ vzhledu jevů je třídou pro všechny instance tohoto typu vzhledu jevů. Instance třídy, které reprezentují jednotlivý typ vzhledu jevů, jsou instancemi vzhledu jevů. Vzhledy jevů musí být modelovány jako instance metatřídy FeatureType. FeatureType dědí atributy z abstraktní metatřídy IdentifiedType atributy pro identifikaci a popis s tím, že name je povinný atribut, který jedinečný v aplikačním schématu. FeatureType navíc obsahuje atributy a asociační role:

- IsAbstract (identifikátor abstrakce) – povinný atribut vyjádřený datovým type boolean. Pokud má hodnotu true, potom typ vzhledu jevů funguje jako nadtyp.
- CarrierOfCharacteristics (nositel charakteristiky) – jedná se o asociační roli pro specifikování jakékoliv operace vzhledu jevu, jakýkoliv typ atributu vzhledu jevu a jakoukoliv asociační roli vzhledu jevu, která obsahuje charakteristiky typu vzhledu jevu.
- SuperType (identifikátor nadtypu) – asociace pro specifikování nadtypu typu vzhledu jevů.



Obrázek 17: Obecný model vzhledu jevů převzatý z normy ISO 19109

PropertyType je abstraktní nadtřída pro AtributType, Operation a FeatureAssociationRole, které poskytují charakteristiky, které mohou být asociovány s typem vzhledu jevů. PropertyType dědí z abstraktní metatřídy IdentifiedType atributy pro identifikaci a popis. PropertyType nepřidává žádné další atributy nebo asociace. Typy vlastností jsou specifikovány nezávisle na jakémkoli konkrétním typu vzhledu jevů. Instance vlastnosti a její hodnota vzniká v souvislosti se vzhledem jevu, i když v některých případech je tento vzhled jevu velmi obecný a nemusel by být v datové sadě explicitně uveden. Vlastnosti budou často popisovány nezávisle na vzhledech jevů. Je proto nejlepší, pokud jsou názvy typů vlastností v aplikaci jedinečné.

AttributeType je metatřídou pro definice atributů typů vzhledu jevů. Atributy vzhledu jevů musí být modelovány jako instance metatřídy AttributeType. AttributeType dědí atributy a asociace z abstraktní metatřídy PropertyType, přičemž attribute name je povinný. Dále přidává další atributy a asociace:

- ValueType – povinný atribut, vyjádřený datovým typem hodnoty atributu (ISO 19103 definuje některé datové typy, které mohou být použity).
- ValueDomain – množina povinných hodnot (množina čísel, GM_Object a jeho podtypy…).
- Cardinality – povinný atribut, vyjadřující počet instancí atributu, který může být asociovaný s jednotlivou instancí vzhledu jevů.
- Characterize – typ atributu, který je popsán tímto typem atributu (v případě atributu atributu).
- CharacterizeBy – typ atributu, který popisuje typ tohoto atributu (v případě atributu atributu).

Chování vzhledu jevů je popsáno v rámci metatřídy Operation a může být vykonáno na nebo prostřednictvím všech instancí typu vzhledu jevů. Operace musí být modelovány jako instance metatřídy Operation. Kromě atributů a asociací zděděných z abstraktní metatřídy PropertyType přidává Operation následující atributy:

- Signature – popis, označující název argumenty a navrácené hodnoty operací. Jde o povinný atribut.

FeatureAssociationRole je metatřídou pro třídy rolí, které jsou částí FeatureAssociationType. Na instanci FeatureAssociationRole, jež poskytuje roli pro jeden typ vzhledu jevů,se dá nahlížet jako na část tohoto typu vzhledu jevů. Role asociace vzhledu jevů musí být modelovány jako instance metatřídy FeatureAssociationRole. FeatureAssociationRole dědí atributy a asociace z abstraktní metatřídy PropertyType. Name je povinný pro roli asociace. Dále přidává atributy valueType a cardinality.

ValueAssignment je asociační metatřídou pro jakýkoli proces, kterým jsou hodnoty přiřazeny k vlastnosti, a v případě potřeby instance této metatřídy podporují ověřování důkazů pro přiřazení hodnoty vlastnosti. Proces přiřazení hodnoty musí být modelován jako instance třídy asociované s metatřídou ValueAssignment. Příkladem může být přiřazení hodnoty na základě prohlášení, v němž je hodnota vlastnosti přiřazena kompetentním orgánem (majitel pozemkové parcely je zpravidla prohlášen autoritou pozemkového registru). Metatřída ValueAssignment obsahuje jediný povinný atribut označující příslušnou obecnou třídu a tím je type (sloužící k přiřazení hodnoty).

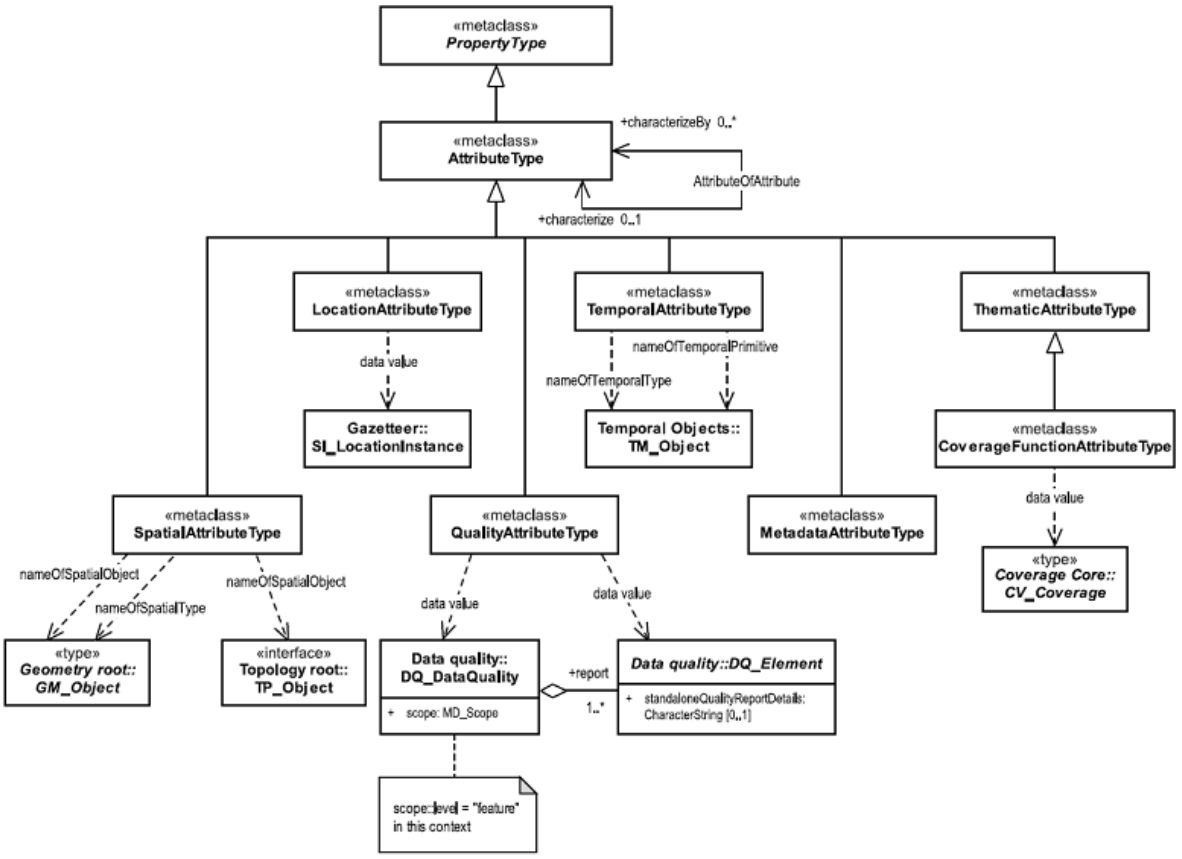
FeatureAssociationType je metatřídou pro popis asociací mezi vzhledy jevů. Kromě atributům a asociacím zděděným z abstraktní metatřídy IdentifiedType připojuje FeatureAssociationType asociaci roleName. RoleName je název konkrétní role asociované s FeatureAssociationType. Jde o povinnou asociaci, které mohou být přítomny až dvě.

InheritanceRelation je třídou pro vzájemné vztahy mezi více obecným typem vzhledu jevů (nadtypem) a jedním specializovaným typem vzhledu jevů. Každá instance specializovaného typu vzhledu jevů je také instancí obecného typu vzhledu jevů. InheritanceRelation přidává atribut uniqueInstance. Jedná se o atribut s datovým typem boolean, přičemž hodnota true vyjadřuje, že instance nadtypu nesmí být instancí více než jednoho podtypu.

7.3.2 Atributy typů vzhledu jevů

Vedle atributů zděděných z nadtypů (PropertyType), má typ atributu s ním asociovaný typ (valueType), doménu (valueDomain) a kardinalitu (Multiplicity).

SpatialAttributeType reprezentuje prostorový atribut, který se používá pro vyjádření prostorových charakteristik typu vzhledu jevů. Typ prostorového atributu má jako typ hodnoty prostorový typ nebo GM_Object nebo TP_Object (struktury těchto typů hodnot jsou definovány v ISO 19107).



Obrázek 18: Model pro atributy obecného modelu vzhledu jevů převzatý z normy ISO 19109

TemporalAttributeType reprezentuje časový atribut, který se používá jako referenční charakteristika času vzhledu jevů. Typ časového atributu má jako typ hodnoty časový typ nebo TM_Object (struktura je definována v ISO 19108).

QualityAttributeType reprezentuje atributy, které obsahují informaci o kvalitě. Tyto atributy jsou použity, když jsou v datové množině zahrnuty data charakteristiky jakosti vzhledu jevů nebo jeho vlastností. Atributy kvality mohou mít typ hodnoty podle definice DQ_Element nebo DQ_DataQuality definované v ISO 19157.

LocationAttributeType reprezentuje atributy, které obsahují prostorové reference na vzhled jevu vyjádřené geografickým identifikátorem (například poštovní oblast může být identifikována kódem, ale její poloha je získána z geografického rejstříku).

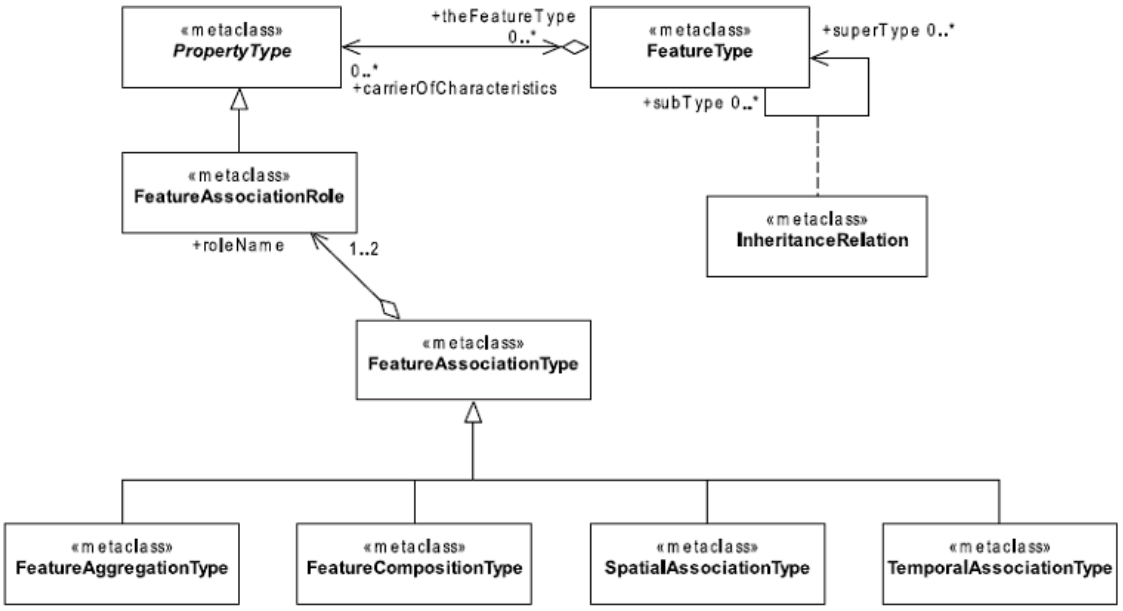
MetadataAttributeType reprezentuje atributy, které obsahují metadatovou informaci, pokud je taková informace zahrnuta jako data v množině dat.

ThematicAttributeType reprezentuje atributy, které obsahují jakoukoliv popisnou charakteristiku vzhledu jevů s výjimkou metadatových a prostorových atributů. Mohou být použity jak základní (viz ISO 19103), tak uživatelem definované datové typy.

CoverageFunctionAttributeType reprezentuje tematické atributy, jejichž hodnoty se mění jako funkce prostorovo-časového místa v předmětném rámci vzhledu jevů. Jejich typy hodnot a domény hodnot jsou definovány uživatelem nebo aplikační oblastí a poskytují rangeType pro CV_Coverage (definováno v normě ISO 19123), který poskytuje hodnoty dat pro atribut.

7.3.3 Vztahy mezi typy vzhledů jevů

Asociace je druhem vztahu, který může existovat jak mezi typy vzhledu jevů, tak mezi instancemi typů vzhledu jevů. Níže jsou tyto vztahy detailně popsány.



Obrázek 19: Model pro asociace obecného modelu vzhledu jevů převzatý z normy ISO 19109

InheritanceRelation reprezentuje specializaci a zobecnění, které specifikuje podtypy a nadtypy typů vzhledu jevů. Tyto vztahy existují jen mezi typy vzhledu jevů a nikoliv mezi instancemi vzhledu jevů. Typickou a vlivnou vlastností tohoto druhu vztahu je, že podtyp bude dědit všechny vlastnosti svého nadtypu.

FeatureAssociationType reprezentuje všechny další typy asociací mezi typy vzhledu jevů. Tyto vztahy se budou vyskytovat jako typy asociací, pokud jsou definovány, a jako instance v množině dat. Typ asociace může být charakterizován svými vlastními vlastnostmi, například má své vlastní atributy.V oblasti geografických informací existuje více druhů asociací, které musí být zpracovávány podle pravidel definovaných v mezinárodní normě ISO 19109.

- FeatureAggregationType – reprezentuje asociace mezi typy, které jsou určitým druhem vztahu „celek–část“, kde části mohou existovat dokonce i tehdy, když je agregát zničen. Agregace se používá ke specifikování typů, které tvoří složité typy.
- FeatureCompositionType – reprezentuje vztahy jako „celek–část“, v němž jsou vzhledy jevů jednoho typu vlastněny vzhledy jevů jiného typu, takže části budou zničeny společně se složeným celkem.

- SpatialAssociationType – reprezentuje prostorové vztahy nebo topologické vztahy, které mohou existovat mezi vzhledy jevů. Tyto vztahy závisí na relativní prostorové poloze a rozsahu vzhledu jevů.
- TemporalAssociationType – reprezentuje časové asociace, které mohou existovat mezi vzhledy jevů.

7.4 Prostorové a časové aspekty vzhledu jevů

Základní charakteristikou prostorového objektu je, že je spojen nejen s prostorovým, ale často také s časovými charakteristikami. Všechny prostorové a časové charakteristiky jsou přímo nebo nepřímo spojeny s referenčním systémem popisující například interpretaci hodnot souřadnic. Ač je toto nezbytné například pro správné nastavení transformačních služeb, nebudeme se tím do hloubky v této dokumentaci zaobírat. Pro účely správného pochopení konceptuálního modelu stačí pouze vědět, že v implementační úrovni bude nutné řešit i tyto náležitosti, kterým se podrobně věnuje norma ISO 19107:2019, Geographic information – Spatial schema a norma ISO 19108:2002, Geographic information – Temporal schema.

Dalším důvodem pro zahrnuté prostorových aspektů do komponent interoperability je nezbytnost vyjádření prostorových charakteristik prostorového objektu v aplikačním schématu jedním z následujících způsobů (v závislosti na požadavcích):

- zadáním vlastností typu prostorových objektů hodnotou, která je prostorovou geometrií;
- zadáním vlastností typu prostorových objektů hodnotou, která je geografickým identifikátorem;
- zadáním vlastností typu prostorových objektů hodnotou, která je funkcí pokrytí;
- zadáním odkazů na jiné prostorové objekty.

Čas je dimenze analogická prostorové dimenzi. Stejně jako prostor má i čas své vyjádření pomocí geometrie a topologie. Bod v čase zaujímá pozici, kterou lze vztáhnout k dočasnému referenčnímu systému. Časové charakteristiky prostorového objektu musí být vyjádřeny v aplikačním schématu jedním z následujících způsobů v závislosti na požadavcích:

- zadáním vlastností typu prostorových objektů hodnotou, která je dočasnou geometrií;
- zadáním vlastností typu prostorových objektů hodnotou, která je jedním ze základních typů vyjadřujících čas (popsaných v kapitole 7.1.1.3. datové typy).

8. Identifikátory

Jedinečná identifikace prostorových objektů je zajištěna identifikátory objektů, tj. identifikátory zveřejněné odpovědným poskytovatelem údajů s úmyslem, že mohou být použity třetími stranami. Identifikátor je nástrojem pro jednoznačnou identifikaci, přičemž v oblasti geografické informace je identifikátor chápán jako prostředek prostorového určení vzhledu jevu.

Problematikou jednoznačné identifikace objektů prostorových dat pro účely NaSaPO se zaobírá komplexně výsledek V07 projektu NaSaPO.

9. CityGML

Rozsáhlá analýza prováděná v rámci zpracování projektu NaSaPO vyhodnotila CityGML jako nejvhodnější standard pro modelování měst a krajiny. V návaznosti na toto vyhodnocení se konceptuální model a i další kapitoly koncepce NaSaPO odkazují na CityGML a hlavně na verzi 3.0. CityGML je zavedený mezinárodní standard Open Geospatial Consortium (OGC) pro modelování, uchovávání a výměnu sémantických virtuálních 3D modelů měst. Specifikace CityGML definuje datový model pro ukládání a výměnu 3D modelů měst a krajiny, který lze implementovat na různých databázových platformách (např. Oracle, PostGIS) a také výměnný formát pro přenos dat mezi těmito platformami. Vzhledem k důrazu na 3D pojetí NaSaPO, je třeba brát v potaz CityGML jakožto etablovaný formát na poli 3D GIS. CityGML byl původně vyvinut členy skupiny zvláštního zájmu 3D (SIG 3D) iniciativy Geodata Infrastructure North-Rhine Westphalia (GDI NRW) v Německu. Nyní je to však otevřený standard vyvinutý a udržovaný Open Geospatial Consortium (OGC). CityGML definuje způsoby, jak popsat geometrii a atributy většiny běžných 3D prvků a objektů nacházejících se ve městech, jako jsou budovy, silnice, řeky, mosty, vegetace a městský mobiliář. Ty mohou být doplněné texturami a/nebo barvami pro lepší dojem z jejich vzhledu. Specifické vztahy mezi různými objekty lze také uložit pomocí CityGML, například tak, že budova je rozdělena na tři části nebo že budova má přístřešek pro auto i balkon. CityGML definuje různé standardní úrovně detailů (LOD) pro 3D objekty. Ty poskytují možnost reprezentovat objekty pro více různých aplikací a účelů. Typy objektů uložených v CityGML jsou seskupeny do různých modulů. První verze CityGML byla jako OGC standard přijata již v roce 2008. Následovala verze 2.0 v roce 2011. V současné době je dostupná verze CityGML 3.0, která předchozí verzi výrazně překonává. Vývoj CityGML 3.0 probíhal s důrazem na lepší interoperabilitu s Industry Foundation Classes (podle normy ISO 16739:2013), INSPIRE, IndoorGML, Land Administration Domain Model (ISO 19152:2012) apod. a vychází z norem řady ISO 19100. Stejně tak je myšleno na propojení na data a technologie sémantického webu jako je například Resource Description Framework (RDF). Ve srovnání se standardy BIM (jako je například IFC) jsou CityGML modely obvykle méně podrobné, ale pokrývají mnohem větší prostorový rozsah. Výhodou je také koncept LoD, který umožňuje pracovat s objekty ve čtyřech různých úrovních podrobnosti. Tomuto tématu je níže v dokumentu věnována samostatná kapitola. Konceptuální model CityGML 3.0 je založen na třídní hierarchii a dědění vlastností a vztahů. Typy vzhledu jevů jako jsou budovy nebo tunely jsou podtřídami mnohem obecnějších tříd, ze kterých dědí všechny jejich vlastnosti. Vzhledy jevů se tak v modelu nachází na pomyslných osách specializace – generalizace, přičemž nejvýše umístěným, a tedy nejobecnějším vzhledem jevu je třída

AnyFeature. Významnou roli hrají v CityGML také vztahy mezi objekty. Zejména komplexní objekty, jako jsou budovy, se obvykle skládají z částí, které rovněž představují vzhledy jevů a mohou být případně dále děleny. Některé typy vzhledu jevů jsou v modelu označeny jako TopLevelFeatureType – např. budova, most, vodní těleso. Jedná se o hlavní objekty modelu města a často bývají vrcholem agregační hierarchie. Pouze tyto vzhledy jevů mohou být přímými členy objektu CityModel. Vztahy mohou být rozšířeny o atributy potřebné například pro bližší určení typu vztahu. Vztahy se v CityGML používají zejména pro obecné provázání souvisejících objektů (relatedTo), agregace do komplexních objektů a externí odkazy na vnější zdroje, jako jsou katastr nemovitostí nebo BIM. Mnoho vztahů v konceptuálním modelu je specifických pro konkrétní typy vzhledu jevů – například mezi Prostorem (Space) a Hranicí prostoru (SpaceBoundary). Jak již bylo popsáno, CityGML slouží i jako výměnný formát a jako zdroj dat pro vizualizace, a to buď ve specializovaných aplikacích, nebo ve webovém prohlížeči. Lze jej také použít pro velké množství prostorových analýz, jako jsou studie viditelnosti a slunečního potenciálu a mnohé další (některé příklady jsou již uvedeny v úvodní kapitole tohoto dokumentu). Každá z těchto aplikací může vyžadovat konkrétní sémantická data. Díky své struktuře a podpoře těchto sémantických informací představuje CityGML silnou platformu pro podporu aplikací.

Nejběžnější implementace, která se obecně používá k distribuci a výměně dat, datové sady CityGML sestává ze sady souborů ve formátu XML (a případně některých doprovodných obrazových souborů, které se používají jako textury). Každý takový soubor může představovat část datové sady, například konkrétní oblast, objekty konkrétního typu (například budovy) nebo objekty v konkrétním LoD. Struktura souboru CityGML je hierarchická, která prostupuje od obecných (abstraktních) objektů až k jednotlivým konkrétním objektům a jejich atributům. Tyto objekty mají geometrii popsanou pomocí Geografického značkovacího jazyka (GML). Další důležitou implementací CityGML je 3D City Database (3D City DB). Jedná se o schéma databáze s otevřeným zdrojovým kódem, které implementuje standard CityGML nad standardní prostorovou relační databází (Oracle a PostGIS). Obsah 3D City DB lze exportovat do formátů KML, COLLADA a glTF pro vizualizaci v široké škále aplikací, jako jsou Google Earth, ArcGIS a Cesium Virtual Globe založené na WebGL. Při implementaci je potřeba mít na paměti, že mnohé koncepty použité v konceptuálním modelu CityGML nejsou pro implementátory nezbytně důležité. Ve spoustě případů se totiž jedná o abstraktní třídy důležité z pohledu konceptuálního, nikoliv implementačního. Pro implementátory jsou podstatné typy objektů nejvyšší úrovně jako jsou budovy nebo konstrukční prvky.

9.1 Application Domain Extension

Bohužel ani CityGML nepokrývá veškeré sémantické informace požadované určitými aplikacemi. Při využití CityGML v rámci NaSaPO bude nezbytné využití objektů a atributů, které nejsou v CityGML explicitně modelovány nebo nespadají do žádné z tematických tříd modelu. Takové informace specifické pro doménu mohou být modelovány pomocí schématu ADE (Application Domain Extension). ADE je způsob rozšíření modelu CityGML. Každé ADE je reprezentováno vlastním konceptuálním schématem v UML s jedinečným jmenným prostorem. Pomocí ADE lze stávající třídy CityGML rozšířit o vlastnosti nebo vztahy. Dále může uživatel definovat i nové třídy. Biljecki a kol. (2018) uvádí dva hlavní způsoby použití ADE:

- Nové objekty definované v rámci ADE jsou odvozeny od objektů existujících v rámci CityGML, takže automaticky dědí vlastnosti příslušných nadtříd.

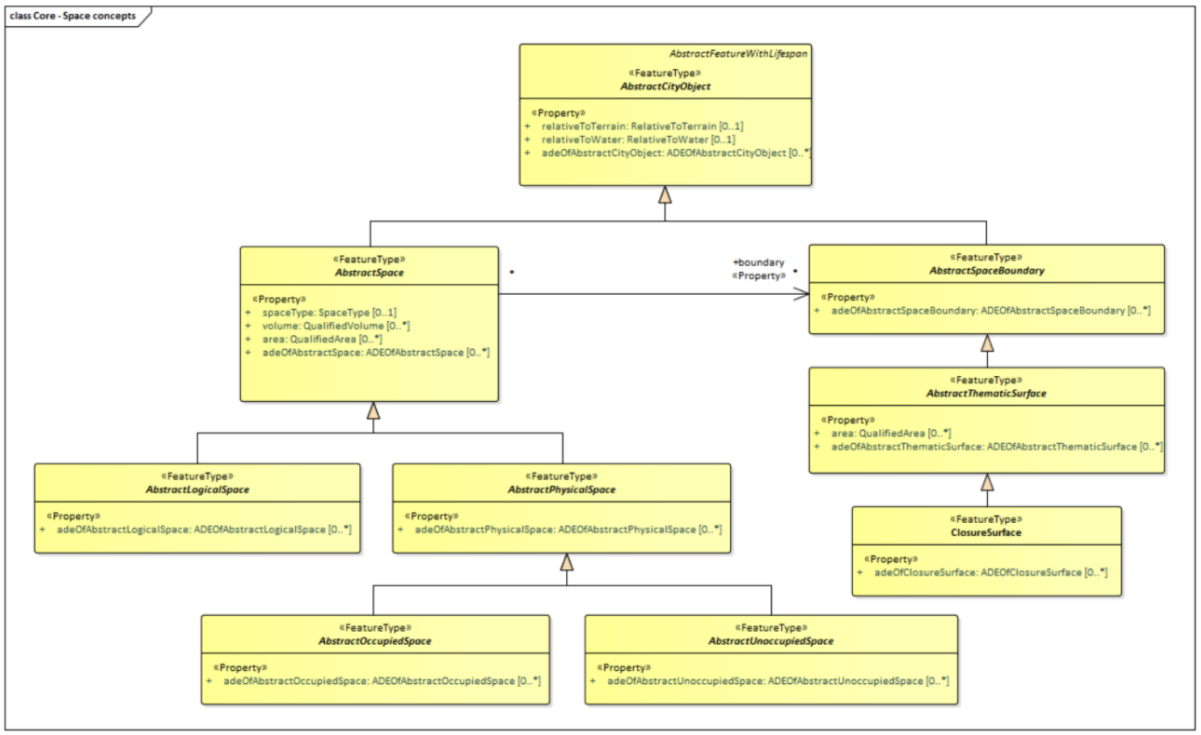
- Existující objekty CityGML mohou být rozšířeny o další vlastnosti, datové typy, geometrie nebo nové vztahy k dalším objektům. Každý stávající typ vzhledu jevu v CityGML má vlastnost nesoucí název adeOfFeatureTypeName, kterou je možné využít například pro připojení nového atributu k existujícímu vzhledu jevů. Tento způsob rozšíření je označován jako hák („hook“).

Druhým používaným způsobem rozšíření modelu CityGML pomocí ADE je doplnění nových vzhledů jevů jakožto podtříd stávajících objektů CityGML, což umožňuje dědění jejich vlastností. To zachovává využití principů CityGML 3.0 jako je například koncept prostorů i pro nově vytvořené třídy. Při reprezentaci vzhledu jevu včetně geometrie by měla být nově vzniklá třída odvozena, přímo nebo nepřímo, ze tříd Core::AbstractSpace nebo Core::AbstractSpaceBoundary. Použití vlastních datových typů není omezeno.

9.2 Koncept prostorů

V CityGML 3.0 bylo představeno nové sémantické rozdělení prostorových prvků pomocí konceptu prostorů (AbstractSpace) a hranic prostoru (AbstractSpaceBoundary). Datový model NaSaPO přebírá ze CityGML toto rozdělení. Prostor je entitou s objemovým rozsahem v reálném světě. Příkladem jsou budovy, místnosti, dopravní prostory, vegetace, vodní útvary apod. Tyto prvky jsou modelovány jako prostory, přesněji jako konkrétní podtřídy abstraktní třídy pro prostory. Hranice prostoru je entita s plošným rozsahem v reálném světě. Hranice prostoru vymezuje a propojuje prostory. Příkladem mohou být povrchy stěn a střech, které spojují budovu (tedy vymezují prostor), vodní plocha (propojení mezi vodním prostorem a vzduchem), povrch vozovky (hranice mezi zemí a dopravním prostorem) nebo i digitální model terénu, který představuje hranici mezi nadzemním a podzemním prostorem. Vztah mezi abstraktními třídami prostorů a hranic prostorů je v modelu UML reprezentován prostřednictvím asociace mezi oběma třídami. Klasifikace objektů v reálném světe do prostorů a hranic prostoru je založena pouze na sémantice těchto objektů a nikoliv na použitém typu geometrie. To je z důvodu možnosti definování objektu různým geometrickým znázorněním (například budova může být reprezentována 3D tělesem nebo i bodem). Koncept prostorů podporuje vyjádření explicitních topologických, geometrických a tematických vztahů mezi jednotlivými prostory a mezi jejich hranicemi. Implementace kontrol geometricko topologické konzistence bude snazší, protože většinu kontrol lze provést v jádrovém aplikačním schématu a poté automaticky aplikovat na všechny další schémata.

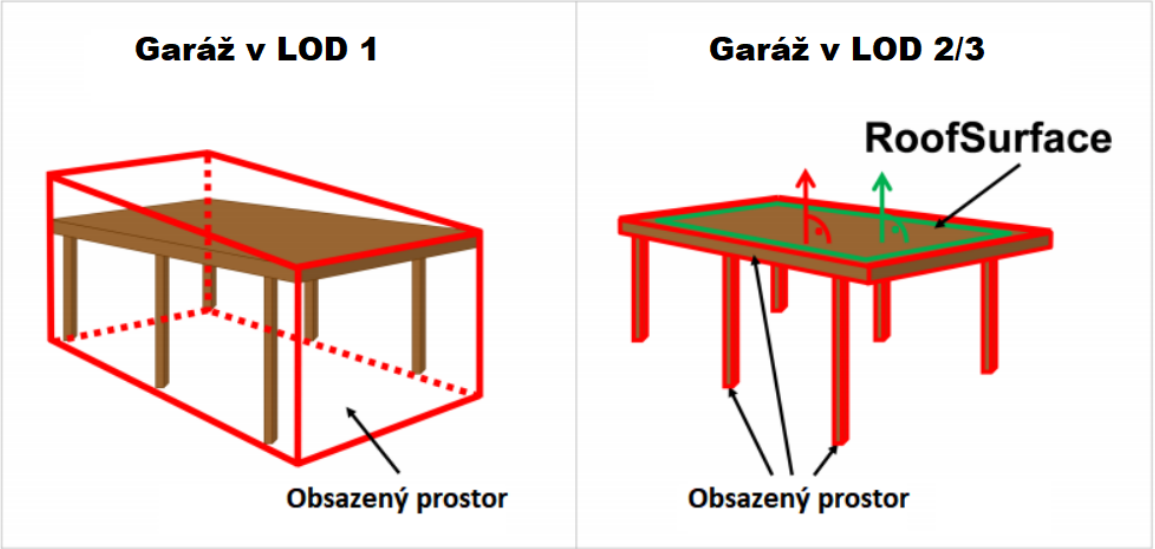
Prostory se dále dělí na abstraktní třídy fyzických a logických prostorů. Fyzické prostory (AbstractPhysicalSpace) jsou prostory, které jsou zcela nebo částečně ohraničené fyzickými objekty. Fyzickým prostorem je například budova, či místnost, protože jsou ohraničené stěnami. Dalším příkladem mohou být dopravní prostory silnic, které jsou ohraničeny povrchem vozovky a zemí. Logické prostory (AbstractLogicalSpace) jsou naopak prostory, které nejsou nutně ohraničené fyzickými objekty, ale jsou definovány podle tématických úvah. Logické prostory mohou být také ohraničeny virtuálními hranicemi. Například bytová jednotka je virtuálním prostorem, který agreguje konkrétní místnosti a byty. Dalším příkladem může být městská část, která je ohraničena virtuálními správními hranicemi. Fyzické prostory se dále dělí na prostory obsazené a neobsazené. Obsazené prostory (AbstractOccupiedSpace) představují fyzické objekty, které zabírají prostor. Příkladem mohou být budovy, mosty, stromy, vodní útvary atd. Obsazení prostoru znamená, že určitý prostor je blokován nějakým objektem o určitém objemu.



Obrázek 20: Ukázka konceptuálního modelu pro koncept prostorů v modulu Core v CityGML 3.0

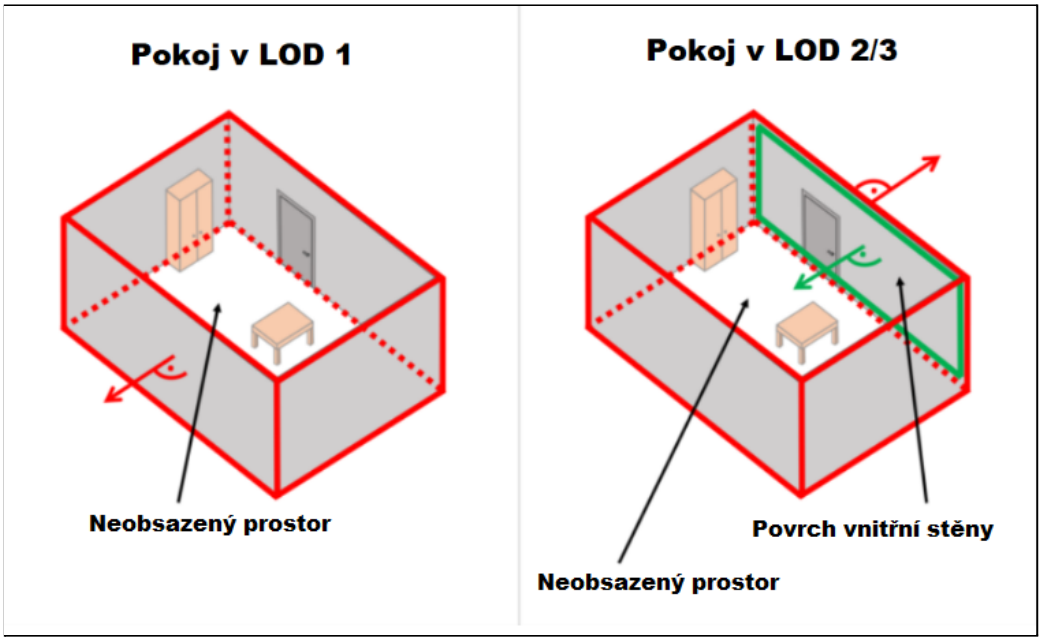
Například prostor, který je blokován budovou nelze již použít pro možnost průjezdu tímto prostorem, popřípadě umístěním jiného objektu do tohoto prostoru. Protějškem obsazených prostorů jsou prostory neobsazené (AbstratUnoccupiedSpace). Ty představují fyzické entity, které nezabírají prostor (žádný prostor neblokují). Příkladem neobsazených prostorů mohou být místnosti nebo dopravní prostory (podobný přístup se využívá v robotice – navigace mobilních robotů využívá mapu obsazených a neobsazených prostorů, ve kterých se robot může pohybovat).

Klasifikace typů prvků do obsazeného a neobsazeného prostoru také definuje sémantiku geometrií připojených k příslušným prvkům. Pro obsazené prostory musí normálové vektory povrchu směřovat ve stejném směru jako povrchy vnějšího obalu objektu. Pro neobsané prostory musí normálové vektory ukazovat v opačném směru. Znamená to, že z pohledu pozorovatele musí povrchové normálové vektory vždy směřovat k pozorovateli (v případě obsazených prostorů je tedy pozorovatel umístěn mimo prostor a v případě neobsazeného prostoru musí být umístěn uvnitř prostoru). Klasifikace do obsazeného a neobsazeného prostoru nemusí být vždy na první pohled patrná. Příkladem může být garáž, která je součástí obsazeného prostoru i v případě, že není kompletně uzavřená. Toto platí však pouze v LOD1, kde je garáž vnímána v podstatě stejně jako budova. V LOD1 je obsazený prostor definován celým tělesem garáže zatímco v dalších LOD2 a LOD3 je garáž reprezentována samostatnými objekty jako je střecha a stěny. V tomto případě je pak prostor garáže mezi těmito objekty definován jako neobsazený prostor (viz obrázek 21).



Obrázek 21: Ukázka reprezentace garážového přístřešku v odlišných LoD – červeně jsou naznačené solidy, zeleně plocha

Druhým příkladem je místnost, která je vedena jako neobsazený prostor. Místnost je reprezentována třídou pro místnosti (BuildingRoom) což je podtřída třídy pro neobsazené prostory. V LoD1 by celá místnost byla reprezentována neobsazeným prostorem. V LoD2/3 by tomu bylo tak také, nicméně místnost bude obsahovat i jednotlivé povrchy, například povrchy stěn (InteriorWallSurface).



Obrázek 22: Ukázka reprezentace pokoje v jednotlivých LoD

9.2.1 Koncept úrovně detailů

Jednou z hlavních charakteristik CityGML je koncept úrovní detailů (Level of Detail – LoD). Ve verzi CityGML 3.0 byl oproti verzi 2.0 vypuštěn LoD 4, který se používá pro interiér objektů. V nové verzi je naproti tomu možné modelovat např. vnitřní uspořádání budov již od LoD 0. Je dokonce možné modelovat i například vnější plášť budovy v méně podrobném LoD a interiéry v podrobnějším LoD. Problematikou přepracovaného konceptu LoD se podrobněji zabývá *Lowner a kol. (2006)*. Každý prostorový objekt může být tedy reprezentován ve čtyřech definovaných úrovních detailu (LoD 0–3) a to nezávisle uvnitř a vně objektu. Koncept LoD tak nabízí možnost sběru a evidence dat v různých mírách podrobnosti od méně podrobného až po velmi podrobný popis. To může být vhodné hlavně u dat, u nichž lze předpokládat, že je s postupným vývojem a aplikací přesnějších technologií sběru dat bude nutné upravovat a zpřesňovat. Díky tomuto konceptu je možné reprezentovat v rámci CityGML jednotlivé budovy, areály, mestské čtvrti, celá města a dokonce i regiony. Díky tomu je vhodný pro využití v celé řadě aplikací, jejichž potřeba již byla zmíněna v úvodní kapitole tohoto dokumentu. Jde například o simulace a mapování šíření hluku, modelování distribuce jemného prachu, městské a telekomunikační plánování, nebo simulace v reálném čase pro výcvik nouzového řízení. Výhodou tohoto konceptu je i podpora aplikací v oblastech jako je nouzové řízení nebo vnitřní navigace, které vyžadují informace o interiéru budovy. Všechny tyto modely se mohou lišit s ohledem na geometrickou a sémantickou složitost a na stupeň odchylky od odpovídajících objektů reálného světa. Koncept úrovně detailu (LoD) CityGML je široce přijímán na trhu i ve vědecké komunitě. Pojem „model LoDX“ (přičemž X odpovídá jednotlivým úrovním LOD) se často používá k řešení složitosti stávajících modelů měst a jejich vhodnosti pro konkrétní aplikace.

Prostory a hranice prostoru mohou mít různé geometrické reprezentace v závislosti na jejich LoD. Prostory mohou být prostorově reprezentovány jako jednotlivé body v LOD0, povrchy v LOD0/2/3, tělesa v LOD1/2/3 a křivky v LOD2/3. Hranice prostoru lze reprezentovat jako multi-povrchy v LOD0/2/3 a jako multi-křivky v LOD2/3. Různé úrovně podrobností jsou definovány následujícím způsobem:

LOD 0: Objemové objekty reálného světa (v rámci konceptu prostorů může označit jako prostory) lze prostorově reprezentovat jediným bodem, pomocí sady křivek nebo sady povrchů. Plošné objekty v reálném světě (hranice prostorů) mohou být prostorově reprezentovány v LOD0 sadou křivek nebo sadou povrchů. Povrchy LOD0 reprezentace jsou obvykle výsledkem projekce prostoru na rovinu rovnoběžnou se zemí, tedy je možné si je představit například jako plán místností uvnitř budovy. Reprezentace křivky LOD0 jsou buď výsledkem projekce svislého povrchu (např. povrchu stěny) na zemní rovinu nebo kostra volumetrického tvaru podélného rozsahu, jako je silnice nebo řeka.

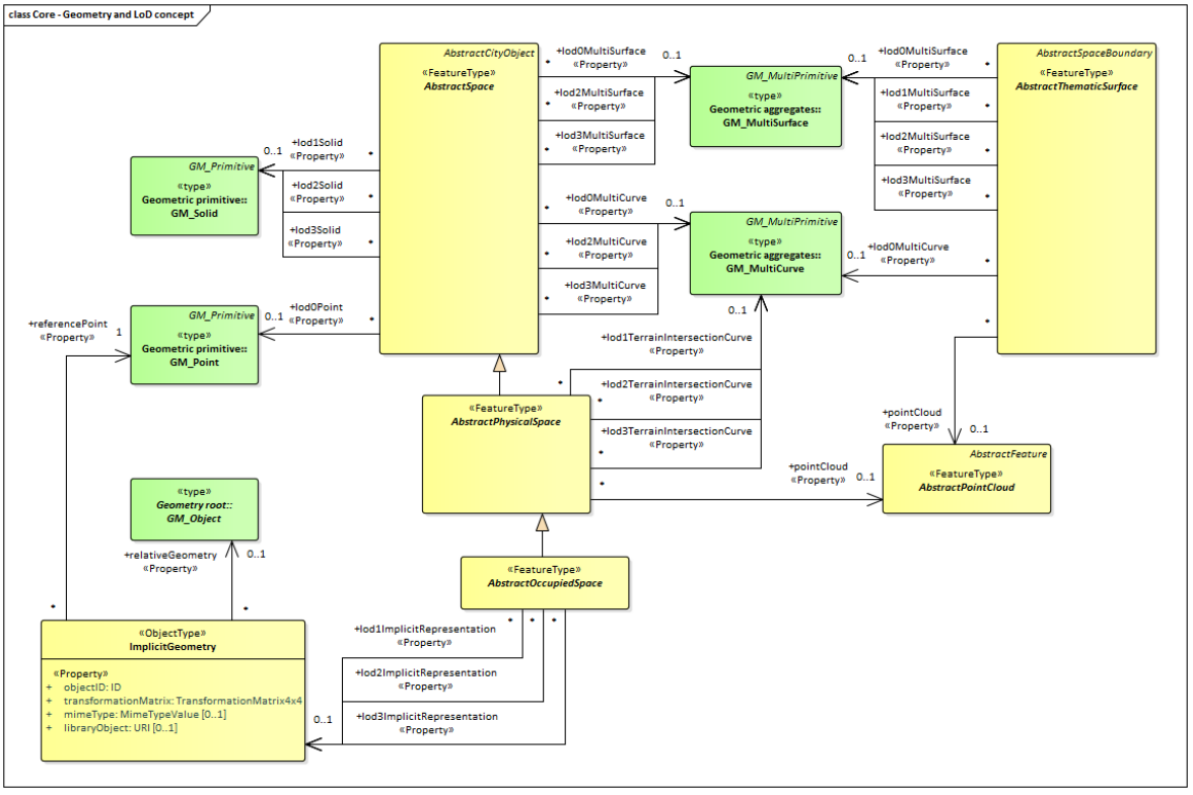
LOD 1: Objemové objekty v reálném světě (prostory) jsou prostorově reprezentovány svislým vytlačováním tělesa, tj. těleso vytvořené z vodorovné stopy svislým vytlačováním. Objekty reálného světa (hranice prostoru) lze v LOD1 prostorově reprezentovat množinou horizontálních nebo svislých ploch.

LOD 2: Objemové objekty v reálném světě (prostory) mohou být prostorově reprezentovány množinou křivek, sada ploch nebo jediná geometrie tělesa. Plošné objekty v reálném světě (hranice vesmíru) mohou být prostorově reprezentován v LOD2 sadou povrchů. Tvar objektu reálného světa je zobecněný v LOD2 a menší detaily (např. boule, promáčkliny, parapety, ale také struktury jako např. balkony nebo vikýře budov)

jsou obvykle zanedbávány. Reprezentace křivky LOD2 jsou kostry objemových tvarů podélného rozsahu jako anténa nebo komín.

LOD 3: Objemové objekty v reálném světě (prostory) mohou být prostorově reprezentovány množinou křivek, sada ploch nebo jediná geometrie tělesa. Plošné objekty v reálném světě (hranice vesmíru) mohou být prostorově reprezentován v LOD3 sadou povrchů. LOD3 je nejvyšší úroveň detailů a příslušné geometrie zahrnují všechny dostupné detaily tvaru.

Tematické třídy, jako je budova, tunel, silnice, využití půdy, vodní útvar nebo městský nábytek, jsou definovány jako podtřídy prostoru a tříd hranice prostoru v rámci tematických modulů. Od té doby všechny městské objekty v tematických modulech představují podtřídy prostoru a hranice prostoru třídy automaticky zdědí geometrie definované v modulu Core.



Obrázek 23: Ukázka použití geometrie v modulu Core v CityGML 3.0

10. Závěr

Zpracovatelé metodiky doufají, že materiál zpracovaný v rámci této dokumentace poskytne návodný postup pro vytvoření modelu NaSaPO. Tím je na mysli jak jeho konceptuální tak i fyzická úroveň. Metodika zahrnuje velmi podrobné (v některých fázích až abstraktní) doporučení a postupy, které je třeba zohlednit při tvorbě konceptuálního modelu. Zpracovatelé se rozhodli pro toto abstraktní zpracování, protože jsou přesvědčeni o potřebě stabilního základu, na kterém by byla NaSaPO postavena. Komponenty interoperability postupně popisují na jaké prvky konceptuálního modelování se zaměřit a kapitola zaměřená na CityGML a následné přílohy doplňují tyto abstraktní pohledy o konkrétní postupy.

Metodika by měla být základem pro navazující projekty na projekt NaSaPO. Základem pro tyto projekty bude rozšíření a ukotvení návrhu katalogu objektů a následné zapracování těchto objektů do struktury, kterou poskytuje standard CityGML. Metodika není zaměřena na nástroje, které bude nutné použít pro toto zapracování, nicméně zpracovatelský tým podnikl v tomto případě vlastní výzkum, kdy bylo provedeno zpracování budovy ze standardu IFC do standardu CityGML. Předpokladem zpracovatelského týmu je, že ačkoliv nástroje pro zpracování budou odlišné, samotné metody budou velmi podobné (i napříč vstupními formáty).

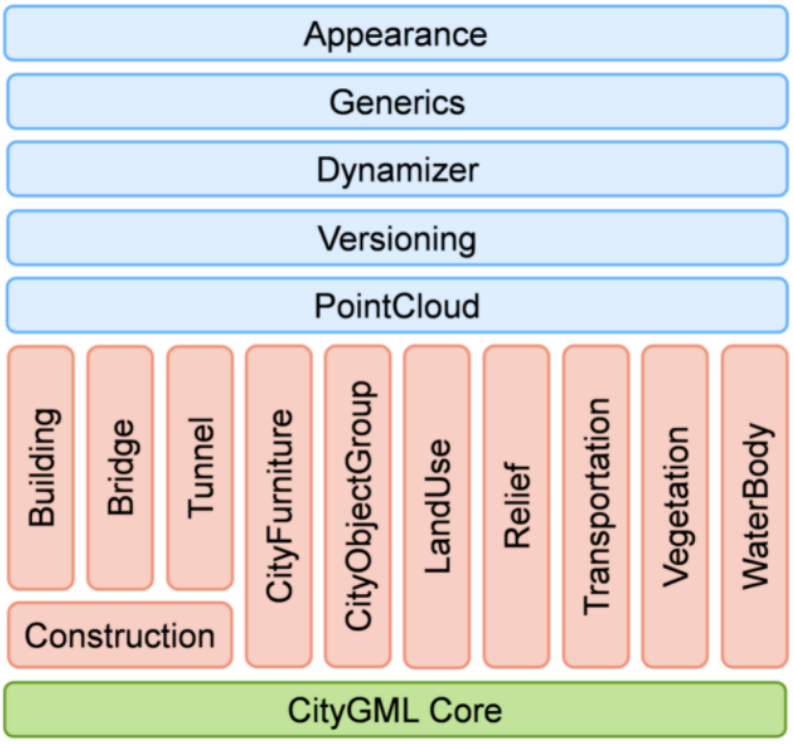
Na úplný závěr je třeba zdůraznit, že základem jakékoliv datové sady je kvalitně popsaný konceptuální model, přičemž tato metodika by měla být právě i jakýmsi průvodcem pro vytvoření takového modelu.

Příloha A. Příkladová studie aplikačních schémat NaSaPO

Příkladová studie aplikačních schémat NaSaPO slouží pro plnohodnotnější pochopení konceptuálního modelu. Nevýhodou jakékoliv koncepce je úroveň její abstrakce. Uživatel který chce aplikovat poznatky získané z koncepce musí překonat „skok“ mezi abstraktní a konkrétní rovinou.

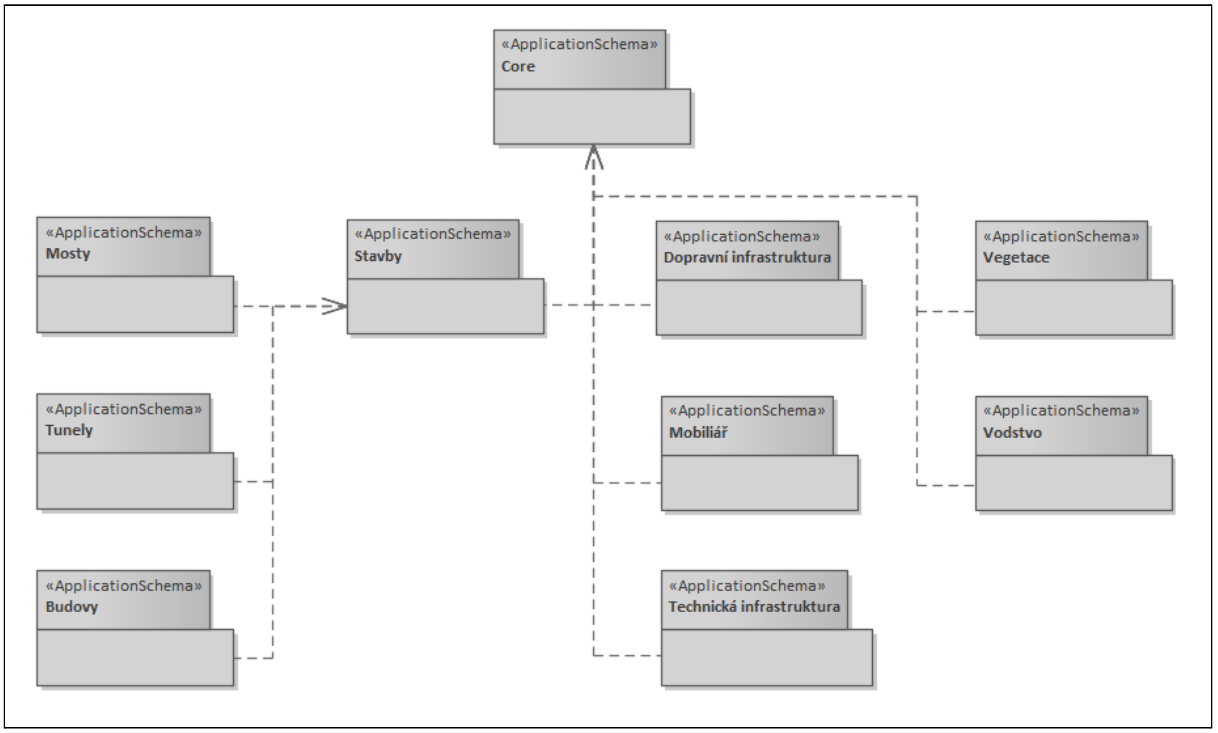
Aplikační schéma NaSaPO

Na základě analýzy stávajících datových modelů se řešitelský tým rozhodl pro využití struktury použité v datovém modelu CityGML, verze 3.0. Konceptuální model CityGML je rozčleněn do modulů (viz obrázek 1). Tento přístup umožňuje při praktickém nasazení využít pouze část modulů CityGML. Například pokud pracujeme s daty budov, postačí nám použít pouze moduly Core, Construction a Building. Části použitých modulů mohou být dále omezeny. Konceptu modulů (aplikačních schémat) CityGML využívá představovaný model NaSaPO.



Obrázek 1: Přehled modulů CityGML – vertikální položky znázorňují jednotlivé tématické moduly, horizontální položky specifikují koncepty, aplikovatelné na tyto moduly

Konceptuální model NaSaPO je tematicky rozčleněn do aplikačních schémat (viz obrázek 2). Jádrové (Core) aplikační schéma obsahuje základní (společné) prvky modelu NaSaPO a jako takový musí být implementován ve všech odvozených systémech. Dále jsou zde tematická aplikační schémata, která z Core přebírají společné informace. Tematický model nebo také diagram aplikačních schémat sestává z jednoduchého popisu jednotlivých aplikačních schémat, které jsou v NaSaPO použity. Tato schémata byla vytvořena za účelem pokrytí základních 3D geografických dat nejvyšší úrovně přesnosti a podrobnosti, které jsou zároveň součástí propojeného datového fondu veřejné správy a které jsou pořizovány geodetickými metodami sběru dat. Vybraným schématům (core, stavby, budovy) jsou v projektu věnovány nadcházející kapitoly.



Obrázek 2: Přehled aplikačních schémat NaSaPO

Model CityGML byl ve své třetí verzi přepracován tak, aby zahrnoval i rostoucí potřebu interoperability s nejpoužívanějšími standardy v oboru. Těmi jsou Industry Foundation Classes (podle normy ISO 16739:2013), IndoorGML, Land Administration Domain Model (ISO 19152:2012) a INSPIRE. Stejně tak je myšleno na propojení na data a technologie sémantického webu jako je například Resource Description Framework (RDF). Řešitelský tým dále upravuje model CityGML tak, aby splňoval potřeby interoperability na datové sady veřejné správy ČR. Návrh datového modelu NaSaPO aplikuje přístup založený na prvotním vytvoření modelu. Tento postup je standardním při tvorbě geoprostorových aplikačních schémat a zahrnuje dva kroky. Prvním je vydefinování datového modelu na konceptuální úrovni. Pro tvorbu modelu je použit modelovací jazyk UML (ISO 19505-2 2012). Druhým krokem je automatické odvození formátů přenosu z tohoto UML modelu pomocí předdefinovaných pravidel transformace. V rámci projektu NaSaPO se řešitelský tým zaměřuje na první krok tohoto přístupu. Datové modely v UML v doméně geografických

informací jsou většinou postavené na standardech z řady ISO 19100. Stejně tomu je i v modelu NaSaPO. Geometrie jsou převzaty z normy ISO 19107:2019, Geographic information – Spatial schema a datové typy z normy ISO 19103:2015, Geographic information – Conceptual schema language. Pro aplikační schémata v UML je použita norma ISO 19109:2019, Geographic information – Spatial schema.

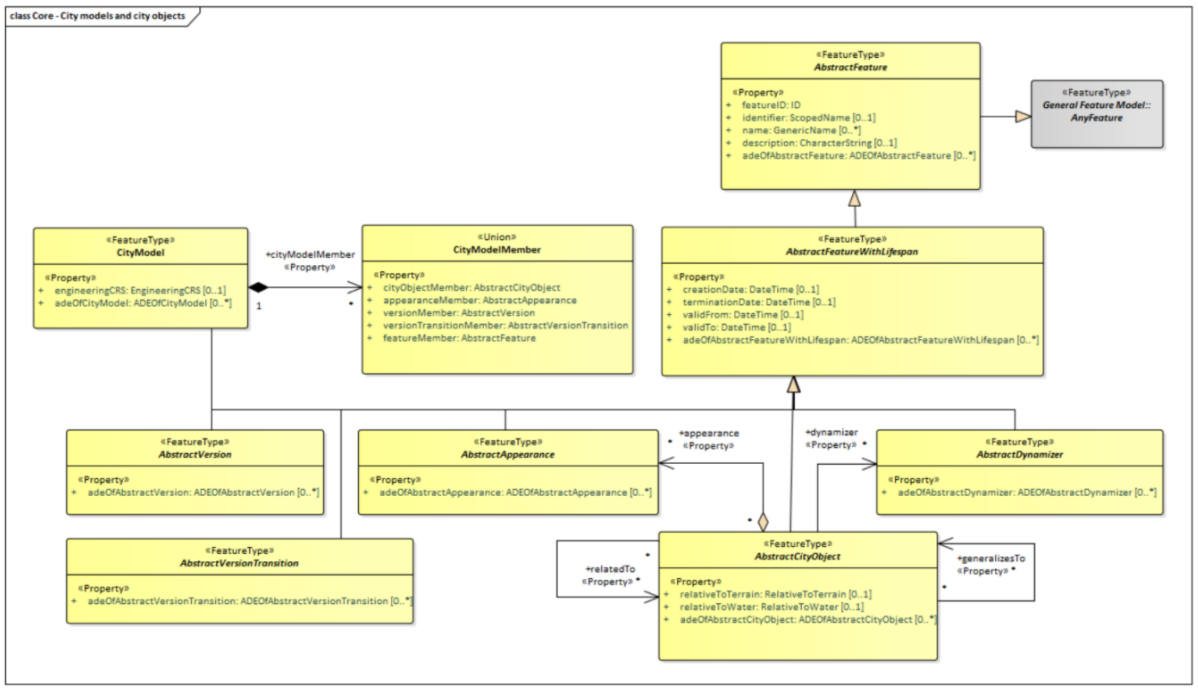
Aplikační schéma – Core

Core aplikační schéma definuje základní komponenty datového modelu NaSaPO. Poskytuje abstraktní typy vzhledu jevů, které jsou nezbytné pro další aplikační schémata (která z nich dědí atributy). Aplikační schéma Core je definováno abstraktními objekty, ze kterých ostatní aplikační schémata čerpají, konceptem prostorů a konceptem úrovní LoD. Aplikační schéma Core přebírá základní atributy z obecného modelu vzhledu jevů a rozšiřuje je o mnoho dalších. Abstraktní typ vzhledu jevů (AbstractFeature) přebírá z obecného modelu vzhledu jevů atributy název a popis. Pro uchování historie objektů v čase používá CityGML 3.0 a tedy i NaSaPO bitemporální časové známky. Jedná se o atributy creationDate a terminationDate, které byly již ve verzi CityGML 2.0 a odkazují na časový úsek, kdy je verze objektu součástí modelu. Tyto atributy jsou použity v abstraktní třídě AbstractFeatureWithLifespan (je tedy zřejmé, že u objektů bude muset probíhat historizace přičemž smazání objektu není smazáním v pravém slova smyslu, ale jen změnou parametru). Nově mohou mít navíc všechny objekty seatributy validFrom a validTo, které popisují životní úseky objektu v reálném světě. To nám dovoluje zpětně určit nejen platné položky modelu, ale také verze objektů reálného světa relevantní v konkrétním časovém okamžiku. Dále je potřeba u každého typu objektu (vzhledu jevů) definovat garanta, což je v aplikačním schématu zobrazeno pomocí povinného atributu s nutností vyplnění alespoň jednoho garanta. V neposlední řadě je nutné u každého objektu evidovat informaci o tom, zda proběhla harmonizace ve smyslu de iure a de facto. Oproti CityGML Core modulu se tak NaSaPO Core liší jak jednotlivými typy objektů tak i jejich atributy.

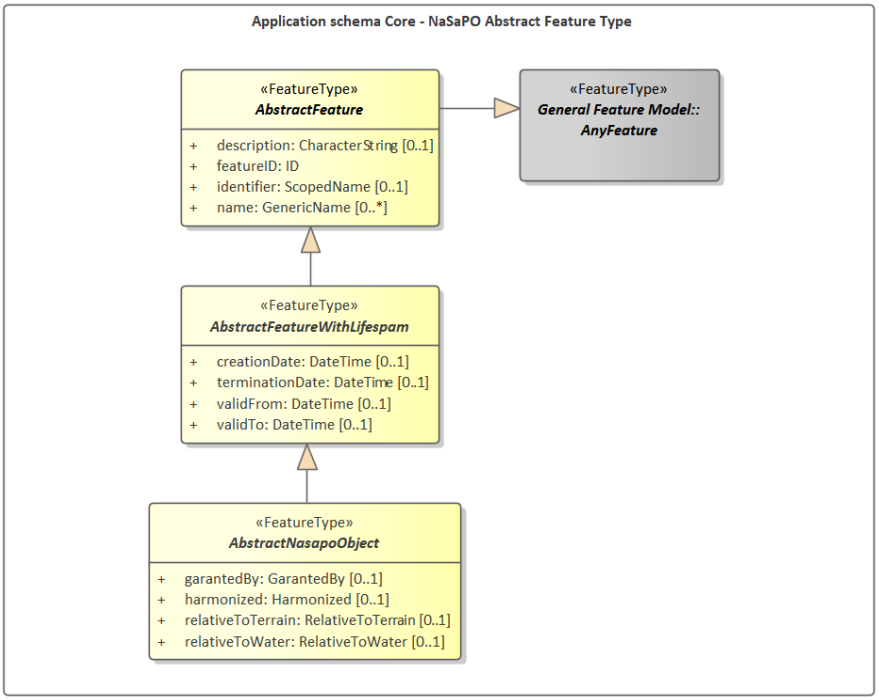
V tabulce 1 jsou popsány jednotlivé třídy definované v aplikačním schématu Core v jeho základní části.

Tabulka 1: Přehled tříd aplikačního schématu Core

AbstractFeature «FeatureType»	AbstractFeature je abstraktní nadtřída všech typů objektů v rámci konceptuálního modelu NaSaPO.
AbstractFeatureWithLifespan «FeatureType»	AbstractFeatureWithLifespan je základní třída pro všechny typy objektů NaSaPO. Tato třída umožňuje uchovávat volitelnou informaci o objektu v reálném světě.
AbstractNasapoObject «FeatureType»	AbstractCityObject je abstraktní nadtřídou pro všechny tématické třídy v rámci konceptuálního modelu CityGML.



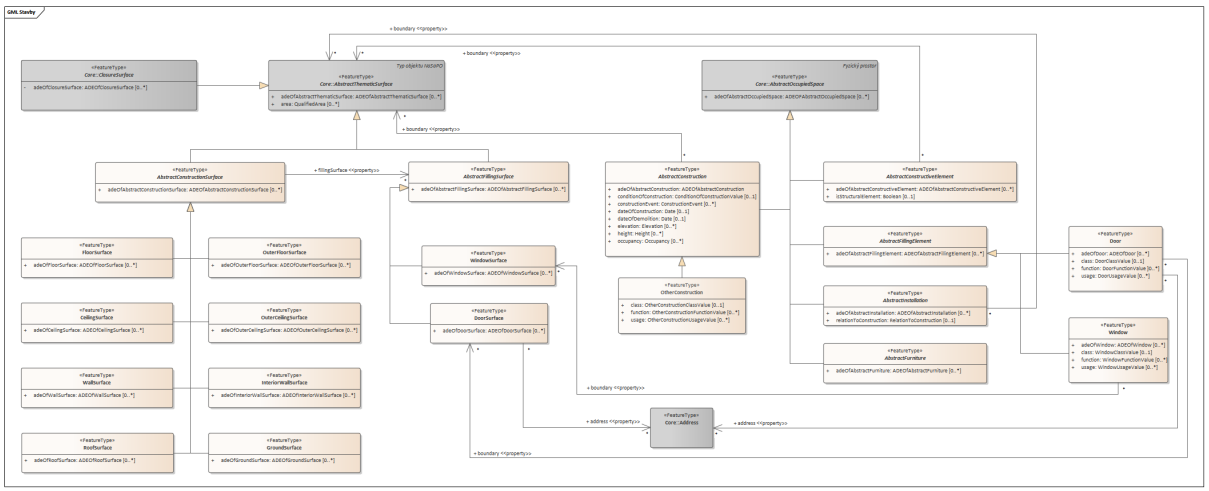
Obrázek 3: Ukázka konceptuálního modelu pro CityObject v modulu Core v CityGML 3.0



Obrázek 4: Ukázka abstraktního objektu NaSaPO z aplikačního schématu Core v CityGML 3.0

Aplikační schéma pro stavby

Aplikační schéma pro stavby definuje koncepty, které jsou společné pro všechny druhy staveb jako jsou budovy, mosty a tunely. Dá se tedy říci, že je nezbytným „rozcestníkem“ do dalších aplikačních schémat. Stavby jsou objekty, které jsou vyrobeny člověkem ze stavebních materiálů a jsou připojeny trvale k zemi. Pokud mluvíme o stavbách pohybujeme se v doméně definované pojmy, které jsou společné pro jakékoliv stavby či části staveb vytvořené člověkem. Aplikační schéma staveb integruje všechny třídy, které by jinak byly definovány samostatně ve schématech pro jednotlivé typy staveb. Jedná se zejména o třídy různých tematických povrchů jako jsou povrchy střech, stěn a podlah a dále také otvory jako jsou dveře a okna. Aplikační schéma pro stavby definuje abstraktní třídu staveb (AbstractConstruction), která je podtřídou abstraktní třídy obsazených prostorů. Třídy, obsahující druhy staveb (budovy, mosty, tunely), jsou definovány jako podtřídy abstraktní třídy staveb. Díky tomuto uspořádání tyto třídy automaticky dědí i přidružení s tématickými povrchy, což vede k podstatnému zjednodušení UML modelů aplikačních schémat pro všechny druhy staveb (například budov). Pro méně časté stavební objekty jako jsou například tovární komíny nebo hradby byla zavedena speciální třída ostatních staveb (OtherConstruction), která je stejně jako ostatní druhy staveb zahrnuta pod abstraktní třídu staveb. Další abstraktní třídy pro stavební prvky. jako jsou vnitřní zařízení (AbstractInstallation) nebo nábytek (AbstractFurniture), jsou také definovány v tomto aplikačním schématu. Vnitřní zařízení jsou trvalou součástí stavby a ovlivňují vnější nebo vnitřní vzhled stavby (například balkony, komíny, nebo schody). Pro uchovávání těchto objektů je vytvořena abstraktní třída obsahující konstrukční prvky stavby (AbstractConstructiveElement), která se dále dělí na prvky budovy, mostu, tunelu ap. podle toho o jakou třídu se jedná. Model znázorňující aplikační schéma NaSaPO pro stavby je uveden na obrázku 5.



Obrázek 5: Konceptuální model pro aplikační schéma staveb

Aplikační schéma pro stavby umožňuje rozdělit stavbu na menší komponenty, jako jsou například stěny, trámy nebo desky, které jsou odděleny hranicemi prostorů tvořenými různými typy povrchů. Konstrukce a konstrukční prvky mohou být omezeny různými typy povrchů. Tímto způsobem může být vnější struktura

konstrukcí a konstrukčních prvků sémanticky diferencovaná na povrchy stěn, povrchy střech, povrchy půd, vnější povrchy podlah, a vnější povrchy stropu, zatímco viditelný povrch vnitřních prostor lze strukturovat do vnitřní povrchy stěn, povrchy podlah a povrchy stropů. Prostory a hranice prostorů tak vytvářejí explicitní spojení mezi (nejen) konstrukčními prvky a jejich tematickými hraničními plochami. Otvory ve stavbách, jako jsou okna (Window) a dveře (Door), jsou dále vyjádřeny jako „výplňové“ objekty spadající do vlastních tříd, které jsou součástí nadtřídy pro výplně (AbstractFillingElement). Přehled povolených povrchů pro jednotlivé třídy je zobrazen v tabulce 2.

Tabulka 2: Přehled povrchů pro aplikační schéma staveb

Třída objektů (vyjádřená prostorem)	**Povolené třídy hranic prostorů**
AbstractConstruction	Construction::AbstractConstructionSurface a podtřídy této třídy, např. Construction::GroundSurface, Construction::RoofSurface, Construction::CeilingSurface, Construction::OuterCeilingSurface, Construction::FloorSurface, Construction::OuterFloorSurface, Construction::WallSurface, Construction::InteriorWallSurface Core::ClosureSurface Je možné přidávat další třídy z ADE
AbstractConstructiveElement	Construction::AbstractConstructionSurface a podtřídy této třídy, např. Construction::GroundSurface, Construction::RoofSurface, Construction::CeilingSurface, Construction::OuterCeilingSurface, Construction::FloorSurface, Construction::OuterFloorSurface, Construction::WallSurface, Construction::InteriorWallSurface Core::ClosureSurface Je možné přidávat další třídy z ADE
AbstractFillingElement	Nejsou povoleny žádné hranice prostorů
AbstractFurniture	Nejsou povoleny žádné hranice prostorů
AbstractInstallation	Construction::AbstractConstructionSurface a podtřídy této třídy, např. Construction::GroundSurface, Construction::RoofSurface, Construction::CeilingSurface, Construction::OuterCeilingSurface, Construction::FloorSurface, Construction::OuterFloorSurface, Construction::WallSurface, Construction::InteriorWallSurface Core::ClosureSurface Je možné přidávat další třídy z ADE

Door	Construction::DoorSurface Core::ClosureSurface Je možné přidávat další třídy z ADE
OtherConstruction	Construction::AbstractConstructionSurface a podtřídy této třídy, např. Construction::GroundSurface, Construction::RoofSurface, Construction::CeilingSurface, Construction::OuterCeilingSurface, Construction::FloorSurface, Construction::OuterFloorSurface, Construction::WallSurface, Construction::InteriorWallSurface Core::ClosureSurface Je možné přidávat další třídy z ADE
Window	Construction::WindowSurface Core::ClosureSurface Je možné přidávat další třídy z ADE

Veškeré zmíněné třídy umožňují mapování konstrukčních prvků z datových sad BIM uvedených ve standardu IFC (např. třídy IFC IfcWall, IfcRoof, IfcBeam, IfcSlab atd.). V tabulce 3 je popsána definice jednotlivých tříd v aplikačním schématu pro stavby.

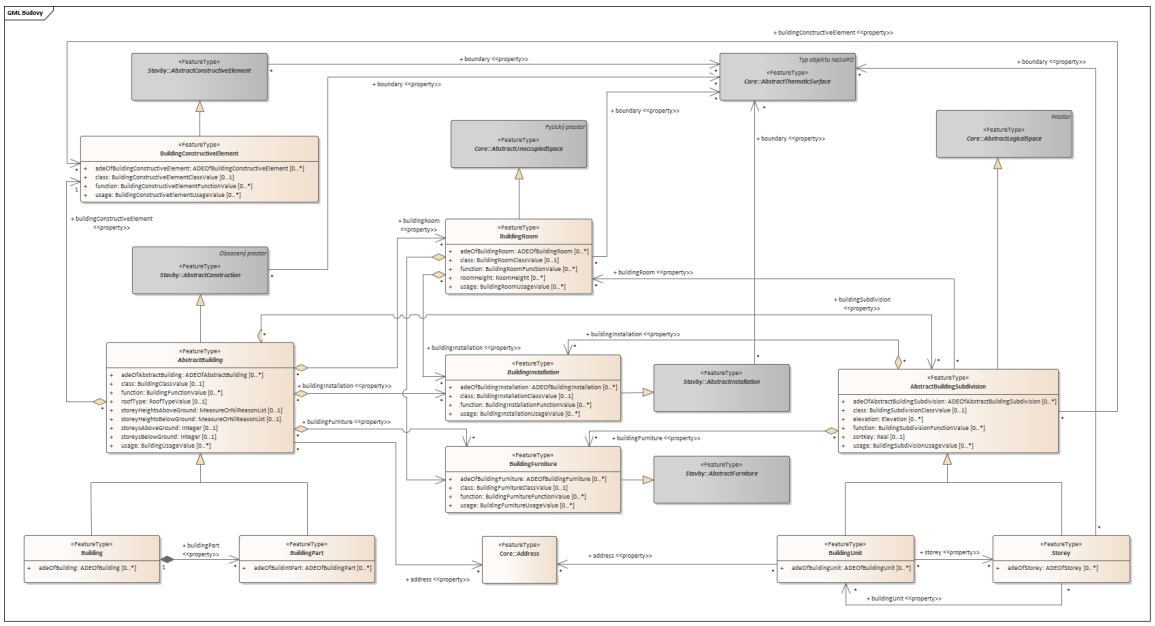
Tabulka 3: Přehled tříd aplikačního schéma pro stavby

Třída objektů	**Definice**
AbstractConstruction «FeatureType»	AbstractBuilding je abstraktní nadtřída reprezentující běžné atributy a asociace pro třídy Building a BuildingPart.
AbstractConstructionSurface «FeatureType»	AbstractBuildingSubdivision je abstraktní nadtřída pro různé druhy logického dělení budov.
AbstractConstructiveElement «FeatureType» «TopLevelFeatureType»	Budova je samostatně stojící, trvalá zastřešená stavba, která je obvykle zděná a lze do něj vstoupit lidmi. Je určena pro obsazení lidmi (např. bydlení, místo práce nebo rekreace).
AbstractFillingElement «FeatureType»	BuildingConstructiveElement je prvek budovy, který je zásadní ze strukturálního hlediska. Příkladem jsou stěny, desky, schodiště nebo trámy.

AbstractFillingSurface «FeatureType»	BuildingFurniture je zařízení pro použití obyvateli budovy, obvykle neoddělitelné do budovy.
AbstractFurniture «FeatureType»	BuildingInstallation je zařízení, které je trvalou součástí budovy, příkladem jsou schody, antény nebo balkony.
AbstractInstallation «FeatureType»	BuildingPart je fyzické nebo funkční dělení budovy.
CeilingSurface «FeatureType»	BuildingRoom je prostor v budově nebo části budovy určený pro lidské obsazení.
Door «FeatureType»	BuildingUnit je logické členění budovy, vytvořené podle nějaké homogenní vlastnosti, jako je funkce, vlastnictví, správa nebo přístupnost.
DoorSurface «FeatureType»	Podlaží je obvykle vodorovná část budovy, která není definována podle stavební konstrukce, ale může být definována podle logických úvah.
FloorSurface «FeatureType»	FloorSurface je povrch, který představuje vnitřní podlahu konstrukce. Příkladem je podlaha místnosti.
GroundSurface «FeatureType»	GroundSurface je povrch, který představuje základní desku konstrukce. Polygon vymezující základovou desku je shodný s půdorysem konstrukce.
InteriorWallSurface «FeatureType»	InteriorWallSurface je povrch, který je viditelný zevnitř konstrukce. Příkladem je stěna místnosti.
OuterCeilingSurface «FeatureType»	OuterCeilingSurface je povrch, který patří k vnějšímu plášti budovy s orientací směrem dolů. Příkladem je strop lodžie.
OuterFloorSurface «FeatureType»	OuterFloorSurface je povrch, který patří k vnějšímu konstrukčnímu plášti s orientací nahoru. Příkladem je podlaha lodžie.
RoofSurface «FeatureType»	RoofSurface je povrch, který vymezuje hlavní střešní části konstrukce.
WallSurface «FeatureType»	WallSurface je povrch, který je součástí fasády budovy viditelný zvenčí.
WindowSurface «FeatureType»	WindowSurface je buď hraniční povrch prvku okna nebo povrch, který utěsňuje otvor vyplněný oknem.

Aplikační schéma pro budovy

Aplikační schéma pro budovy poskytuje reprezentaci tematických a prostorových rysů budov. Budovy jsou vnímány jako volně postavené samonosné stavby, které jsou zastřešené a obvykle zděné, a do kterých mohou vstoupit lidé. Jsou obvykle navržené tak, aby trvale stály na jednom místě. Budovy jsou určené pro různé účely, jakými jsou například rekreace, místo vykonávání pracovní činnosti nebo bydlení (a mnoho dalších). Abstraktní třída budov (AbstractBuilding) dědí z abstraktní super třídy staveb a sama je abstraktní třídou pro budovy (Building) a části budov (BuildingPart). Budovy jsou v konceptuálním modelu reprezentovány nejvyšší úrovní typu objektu budova. Budovy je dále možné dělit podle fyzických prostorů na jednotlivé místnosti (BuildingRoom) a podle logických prostorů na podlaží (Storey) a bytové jednotky (BuildingUnit). Nadtřídou pro podlaží a bytové jednotky je abstraktní třída pro dělení budov (AbstractBuildingSubdivision). Budovy lze navíc rozdělit na konstrukční prvky (BuildingConstructiveElement), jako jsou stěny, desky, schodiště nebo i trámy. Vnitřek budov je reprezentován místnostmi, což umožňuje virtuální přístup k budovám. Budovy mohou dále také obsahovat vnitřní zařízení (BuildingInstallation) a nábytek (BuildingFurniture). Ačkoliv třída pro nábytek nebude nejspíše v nejbližším období využita, aplikační schéma pro budovy ji přebírá ze CityGML 3.0 také s výhledem na zlepšující se technologie. Zařízení jsou statickou a trvalou součástí budovy a silně ovlivňují vnější nebo i vnitřní vzhled budovy. Příkladem jsou balkony, komíny, vikýře nebo schody. Modul budov pro aplikační schéma NaSaPO je znázorněn na obrázku 6.



Obrázek 6: Konceptuální model pro aplikační schéma pro budovy

Budovy mohou být ohraničeny různými typy povrchů. Ty lze sémanticky rozlišit na povrchy stěn, povrchy střech, povrchy terénu, povrchy podlah a povrchy stropů. Je možné také rozlišovat povrchy vnitřních zdí, venkovních podlah nebo vnější povrch stropů. V rámci aplikačního schémata pro budovy lze také rozlišit okna, dveře a jejich povrchy. Jednotlivé možnosti použití povrchů jsou znázorněny v tabulce 4.

Definice tříd pro aplikační schéma NaSaPO je popsána v tabulce 5.

Tabulka 4: Přehled povrchů pro aplikační schéma budov

Třída objektů (vyjádřená prostorem)	**Povolené třídy hranic prostorů**
AbstractBuilding	Construction::AbstractConstructionSurface a podtřídy této třídy, např. Construction::GroundSurface, Construction::RoofSurface, Construction::CeilingSurface, Construction::OuterCeilingSurface, Construction::FloorSurface, Construction::OuterFloorSurface, Construction::WallSurface, Construction::InteriorWallSurface Core::ClosureSurface Je možné přidávat další třídy z ADE
AbstractBuildingSubdivision	Nejsou povoleny žádné hranice prostorů
Building	Construction::AbstractConstructionSurface a podtřídy této třídy, např. Construction::GroundSurface, Construction::RoofSurface, Construction::CeilingSurface, Construction::OuterCeilingSurface, Construction::FloorSurface, Construction::OuterFloorSurface, Construction::WallSurface, Construction::InteriorWallSurface Core::ClosureSurface Je možné přidávat další třídy z ADE
BuildingConstructiveElement	Construction::AbstractConstructionSurface a podtřídy této třídy, např. Construction::GroundSurface, Construction::RoofSurface, Construction::CeilingSurface, Construction::OuterCeilingSurface, Construction::FloorSurface, Construction::OuterFloorSurface, Construction::WallSurface, Construction::InteriorWallSurface Core::ClosureSurface Je možné přidávat další třídy z ADE
BuildingFurniture	Nejsou povoleny žádné hranice prostorů

BuildingInstallation	Construction::AbstractConstructionSurface a podtřídy této třídy, např. Construction::GroundSurface, Construction::RoofSurface, Construction::CeilingSurface, Construction::OuterCeilingSurface, Construction::FloorSurface, Construction::OuterFloorSurface, Construction::WallSurface, Construction::InteriorWallSurface Core::ClosureSurface Je možné přidávat další třídy z ADE
BuildingPart	Construction::AbstractConstructionSurface a podtřídy této třídy, např. Construction::GroundSurface, Construction::RoofSurface, Construction::CeilingSurface, Construction::OuterCeilingSurface, Construction::FloorSurface, Construction::OuterFloorSurface, Construction::WallSurface, Construction::InteriorWallSurface Core::ClosureSurface Je možné přidávat další třídy z ADE
BuildingRoom	Construction::AbstractConstructionSurface a podtřídy této třídy, např. Construction::GroundSurface, Construction::RoofSurface, Construction::CeilingSurface, Construction::OuterCeilingSurface, Construction::FloorSurface, Construction::OuterFloorSurface, Construction::WallSurface, Construction::InteriorWallSurface Core::ClosureSurface Je možné přidávat další třídy z ADE
BuildingUnit	Core::ClosureSurface Je možné přidávat další třídy z ADE
Storey	Construction::AbstractConstructionSurface a podtřídy této třídy, např. Construction::GroundSurface, Construction::RoofSurface, Construction::CeilingSurface, Construction::OuterCeilingSurface, Construction::FloorSurface, Construction::OuterFloorSurface, Construction::WallSurface, Construction::InteriorWallSurface Core::ClosureSurface Je možné přidávat další třídy z ADE

Tabulka 5: Přehled tříd aplikačního schéma pro budovy

Třída objektů	**Definice**
AbstractBuilding «FeatureType»	AbstractBuilding je abstraktní nadtřída reprezentující běžné atributy a asociace pro třídy Building a BuildingPart.
AbstractBuildingSubdivision «FeatureType»	AbstractBuildingSubdivision je abstraktní nadtřída pro různé druhy logického dělení budov.
Building «TopLevelFeatureType»	Budova je samostatně stojící, trvalá zastřešená stavba, která je obvykle zděná a lze do něj vstoupit lidmi. Je určena pro obsazení lidmi (např. bydlení, místo práce nebo rekreace).
BuildingConstructiveElement «FeatureType»	BuildingConstructiveElement je prvek budovy, který je zásadní ze strukturálního hlediska. Příkladem jsou stěny, desky, schodiště nebo trámy.
BuildingFurniture «FeatureType»	BuildingFurniture je zařízení pro použití obyvateli budovy, obvykle neoddělitelné do budovy.
BuildingInstallation «FeatureType»	BuildingInstallation je zařízení, které je trvalou součástí budovy, příkladem jsou schody, antény nebo balkony.
BuildingPart «FeatureType»	BuildingPart je fyzické nebo funkční dělení budovy.
BuildingRoom «FeatureType»	BuildingRoom je prostor v budově nebo části budovy určený pro lidské obsazení.
BuildingUnit «FeatureType»	BuildingUnit je logické členění budovy, vytvořené podle nějaké homogenní vlastnosti, jako je funkce, vlastnictví, správa nebo přístupnost.
Storey «FeatureType»	Podlaží je obvykle vodorovná část budovy, která není definována podle stavební konstrukce, ale může být definována podle logických úvah.

Příloha B. Zpracování dat podle aplikačních schémat CityGML

Hlavní způsob uplatnění CityGML spočívá v uložení minulého nebo stávajícího stavu prostředí, což je v souladu s potřebami NaSaPO. Vytvořený sémantický model přírodního nebo vystavěného prostředí zachycuje nejen jeho fyzický stav, ale také ustálené vlastnosti jevů – rozlišitelné lidským vnímáním. Výsledkem je virtuální 3D model krajiny, jehož data mohou být dále využita pro vizualizaci, navigaci, modelování šíření prostorových jevů a prostorové analýzy. Modely mohou být tvořeny v různých měřítcích a úrovních podrobnosti. Předem definované úrovně podrobnosti (LOD) umožňují současné uložení téhož prvku ve vícero LOD a zároveň usnadňují práci s daty v situaci, kdy máme pro různá území rozdílně přesná data.

Přestože verze 3.0 formátu CityGML doposud nebyla dokončena (07/2021), zásadní změny se již neočekávají. Konceptuální model je veřejně dostupný na stránce

- http://docs.ogc.org/DRAFTS/20-010.pdf.

Předpokládá se následné přijetí CityGML 3.0 jakožto standardu OGC. Ve verzi 3.0 byl vylepšen koncept LOD, usnadněna implementace nebo byly rozšířeny možnosti interoperability zejména s důrazem na BIM. Oproti verzi CityGML 2.0 není kódování závislé na formátu GML. Konceptuální model CityGML 3.0 může být kromě GML implementován například pomocí JSON nebo SQL.

Tato kapitola má za cíl představit některé principy CityGML 3.0 při praktické implementaci. Základní zdroj informací pro implementátory představuje uživatelská příručka CityGML 3.0, která obsahuje technické pokyny pro použití konceptuálního modelu CityGML.

- http://docs.ogc.org/DRAFTS/20-066.pdf

Prozatím je dostupné kódování v jazyce GML (od července 2018), o které se zasloužil tým Technické univerzity v Mnichově.

- https://github.com/opengeospatial/CityGML-3.0Encodings/tree/master/CityGML

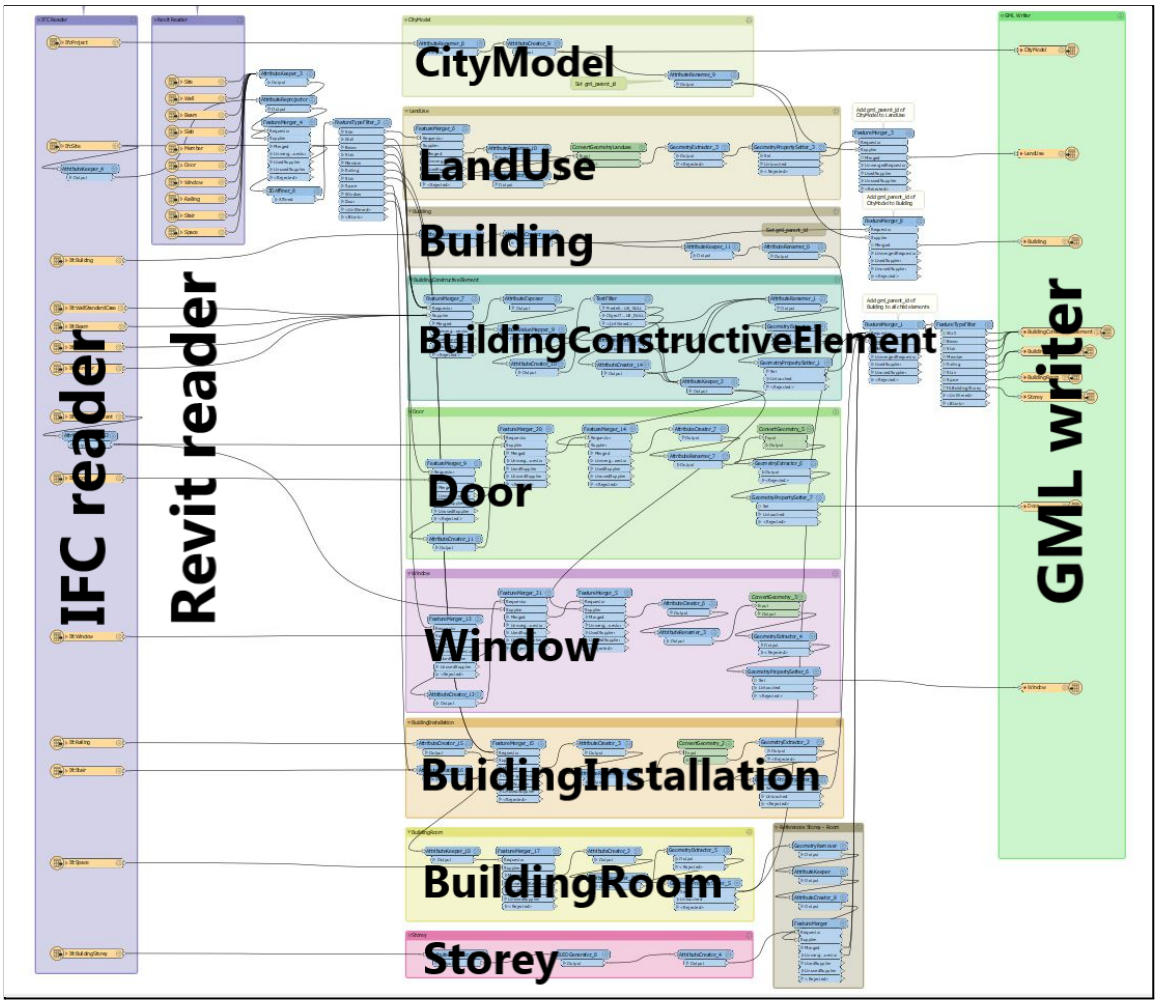
Na stejné stránce je pak možné najít příklady použití CityGML 3.0 včetně odkazů ke stažení dat. Vybrané příklady zde budou představeny blíže.

Zajímavou ukázku tvorby datasetu CityGML 3.0 představuje nástroj ifc-to-citygml3 (https://github.com/tum-gis/ifc-to-citygml3) vyvinutý na Technické univerzitě v Mnichově. Jedná se o nástroj vytvořený v prostředí programu FME (Safe Software) pro konverzi dat mezi IFC a CityGML 3.0. Jelikož FME zatím nepodporuje export do CityGML verze 3.0, byl pro zápis použit obecný GML Writer se zadanými XML schématy CityGML ve formátu XSD. Jako vstupní data byl použit model domu ve formátu IFC včetně zařízení interiéru. Podrobnost výstupní budovy odpovídá LOD 3. Mapování typů vzhledu jevů mezi IFC a CityGML je zachyceno v následujícím obrázku.

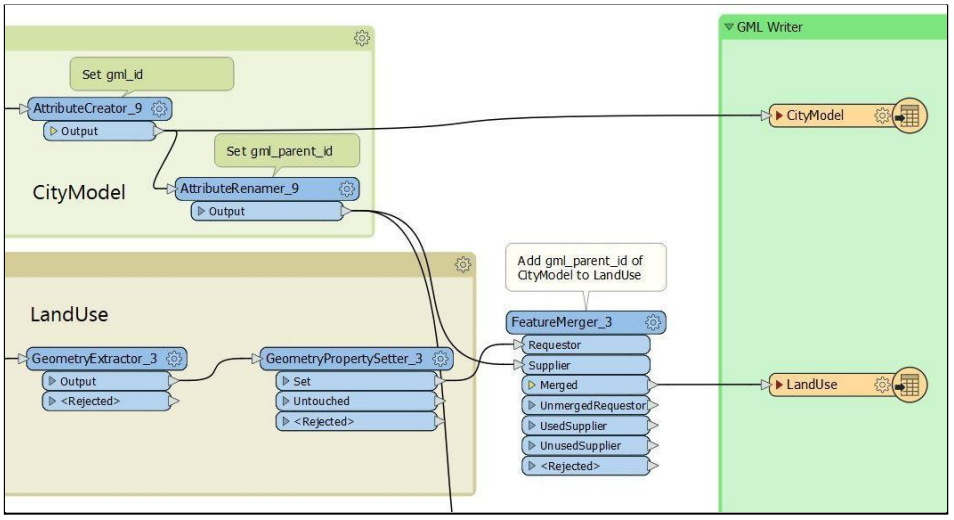
IFC objects	CityGML 3.0 objects
IfcProject	CityModel
IfcSite	LandUse
IfcBuilding	Building
IfcBuildingStorey	Storey
IfcSpace	BuildingRoom
IfcWallStandardCase	BuildingConstructiveElement
IfcBeam	BuildingConstructiveElement
IfcSlab	BuildingConstructiveElement
IfcMember	BuildingConstructiveElement
IfcDoor	Door
IfcWindow	Window
IfcRailing	BuildingInstallation
IfcStair	BuildingInstallation

Obrázek 1: Mapovací tabulka IFC-CityGML3

Celkový přehled nástroje je zobrazen na obrázku 2. V levé části nástroje můžeme vidět dva readery sloužící k načtení dat. IFC reader načítá atributy a Revit reader geometrie typu solid. Oba readery zpracovávají totožný soubor formátu IFC. Dle mapovací tabulky jsou pak vzhledy jevů napojeny na výstupní vzhledy jevů CityGML. Na podrobnějším výřezu nástroje (obrázek 3) můžeme vidět princip hierarchizace objektů CityGML. Objektu typu CityModel je přiřazeno unikátní gml_id, které je následně uloženo jakožto gml_parent_id všem podřízeným objektům. Díky tomu nesou všechny instance vzhledu jevu LandUse informaci o příslušnosti do konkrétního nadřazeného CityModelu. Obdobná situace nastává v případě vztahu Budova-část budovy (např. Building-Door, Building-BuildingConstructiveElement apod.), kdy stěny budovy nesou informaci o tom, které budovy jsou součástí. Tato provázanost brání redundanci uložení prostorových i popisných dat. Zatímco některé atributy se vztahují k celé budově, jiné mohou být navázány na konkrétní část budovy. Geometrie by měly být vždy uloženy pouze v nejjednodušších vzhledech jevů. Tedy vzhled jevu typu budova nemusí obsahovat žádnou geometrii, protože nese informaci o částech budovy ze kterých se skládá.

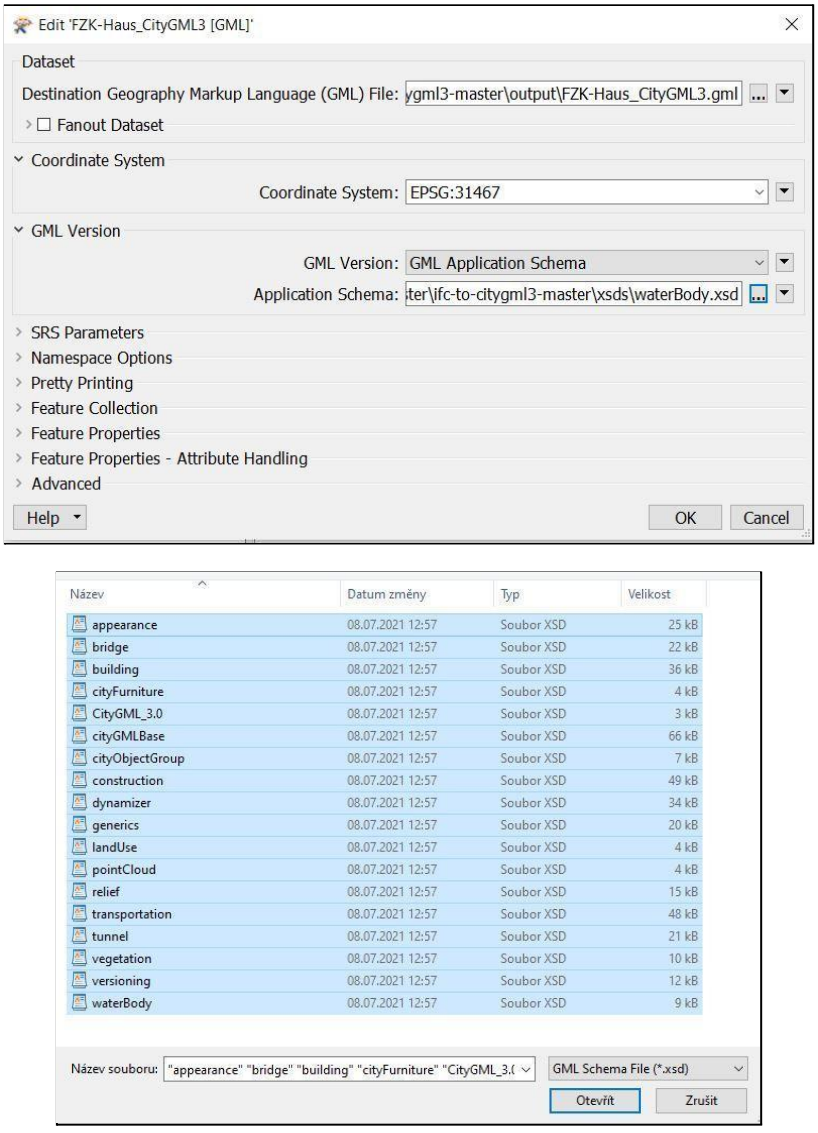


Obrázek 2: ifc-to-citygml3 – celkový přehled

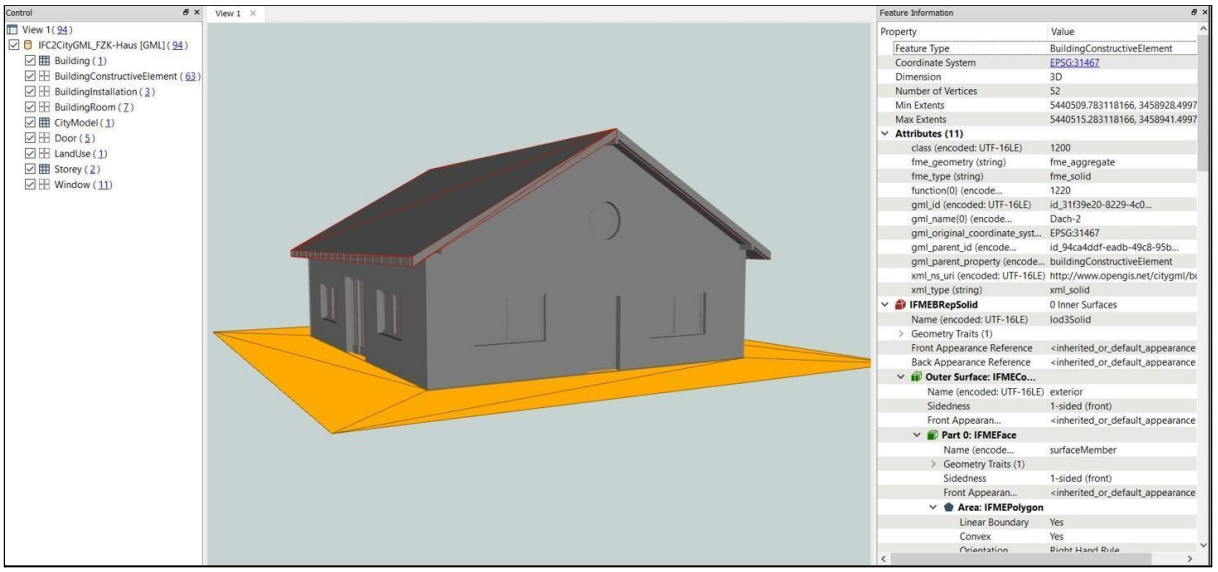


Obrázek 3: ifc-to-citygml3 – hierarchizace a zápis geometrie

Bližší pozornost si zaslouží rovněž princip zápisu geometrie. Prostorová data načtená ze vstupních souborů jsou pomocí kombinace transformerů GeometryExtractor a GeometryPropertySetter uložena do příslušného vzhledu jevů CityGML, přičemž musí být naplněna informace o typu LOD (citygml_lod_name), neboť objekty mohou zpravidla obsahovat více typů geometrií (např. lod0MultiSurface, lod1Solid apod.). Tento poměrně složitý krok odpadne v případě doplnění podpory třetí verze CityGML do CityGML writeru programu FME. Na obrázku 4 je zaznamenán způsob úpravy obecného GML writeru s využitím aplikačních schémat CityGML 3.0. Aplikační schémata ve formátu XSD jsou dostupná online (https://github.com/tum-gis/ifc-to-citygml3/tree/master/xsds). Na obrázku 5 je výstupní soubor ve formátu CityGML 3.0 zobrazený v programu FME Data Inspector.



Obrázek 4: gml writer – načtení aplikačních schémat



Obrázek 5: ifc-to-citygml3 – výstupní soubor FZK-Haus_CityGML3

Literatura

MILES, Alistair; BECHHOFER, Sean. SKOS simple knowledge organization system reference. *W3C recommendation*, 2009.

GRÖGER, Gerhard; PLÜMER, Lutz. CityGML–Interoperable semantic 3D city models. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2012, 71: 12-33.

KUTZNER, Tatjana; CHATURVEDI, Kanishk; KOLBE, Thomas H. CityGML 3.0: New functions open up new applications. *PFG–Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, 2020, 88.1: 43-61.

DE LAAT, Ruben; VAN BERLO, Leon. Integration of BIM and GIS: The development of the CityGML GeoBIM extension. In: *Advances in 3D geo-information sciences*. Springer, Berlin, Heidelberg, 2011. p. 211-225.

BILJECKI, Filip; KUMAR, Kavisha; NAGEL, Claus. CityGML application domain extension (ADE): overview of developments. *Open Geospatial Data, Software and Standards*, 2018, 3.1: 1-17.

KOLBE, Thomas H.; GRÖGER, Gerhard; PLÜMER, Lutz. CityGML: Interoperable access to 3D city models. In: *Geo-information for disaster management*. Springer, Berlin, Heidelberg, 2005. p. 883-899.

GRÖGER, Gerhard, et al. OpenGIS city geography markup language (CityGML) encoding standard, version 1.0. 0. 2008.

Open Geospatial Consortium, accessed 1 October 2020, <https://www.ogc.org>

LÖWNER, Marc-O., et al. Proposal for a new LOD and multi-representation concept for CityGML. *ISPRS Ann Photogramm Remote Sens Spatial Inf Sci*, 2016, 4: 3-12.

VAN DEN BRINK, Linda; STOTER, Jantien; ZLATANOVA, Sisi. Establishing a national standard for 3D topographic data compliant to CityGML. *International Journal of Geographical Information Science*, 2013, 27.1: 92-113.

SOON, K. H.; KHOO, V. H. S. CityGML modelling for Singapore 3d national mapping. *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, 2017, 42.

KUTZNER, Tatjana; KOLBE, Thomas H. CityGML 3.0: sneak preview. In: *PFGK18-Photogrammetrie-Fernerkundung-Geoinformatik-Kartographie, 37. Jahrestagung in München 2018*. 2018. p. 835-839.

OHORI, Ken Arroyo, et al. Modeling cities and landscapes in 3D with CityGML. In: *Building information modeling*. Springer, Cham, 2018. p. 199-215.

KOLBE, Thomas H., et al. OGC City Geography Markup Language (CityGML) Part 1: Conceptual Model Standard. 2021.

GRÖGER, Gerhard; KUTZNER, Tatjana; KOLBE, Thomas H. A CityGML-based encoding for the INSPIRE Data Specification on Buildings. In: *INSPIRE conference*. 2013.

TEAM, Drafting. INSPIRE Generic Conceptual Model. *Generic Conceptual Model of the INSPIRE data specifications*, 2010

TEAM, INSPIRE Drafting. Data Specification": *INSPIRE Generic Conceptual Model. Technischer Bericht, INSPIRE*, 2008.

UML, OMG. Unified modeling language. *Object Management Group*, 2001, 105.

PILONE, Dan; PITMAN, Neil. *UML 2.0 in a Nutshell*. O'Reilly Media, 2005.

Tento projekt je financován se státní podporou
Technologické agentury ČR v rámci
Programu BETA2.

www.tacr.cz
Výzkum užitečný pro společnost.

MINISTERSTVO VNITRA
ČESKÉ REPUBLIKY







