Agenda číselníky - analytický model

verze 1.0, build 2

Dynamic Enumerators – Analysis Model

version 1.0, build 2

© autor RNDr. Ilja Kraval

Object Consulting s.r.o.

http://www.objects.cz

Změny oproti předešlému dokumentu Build 1:

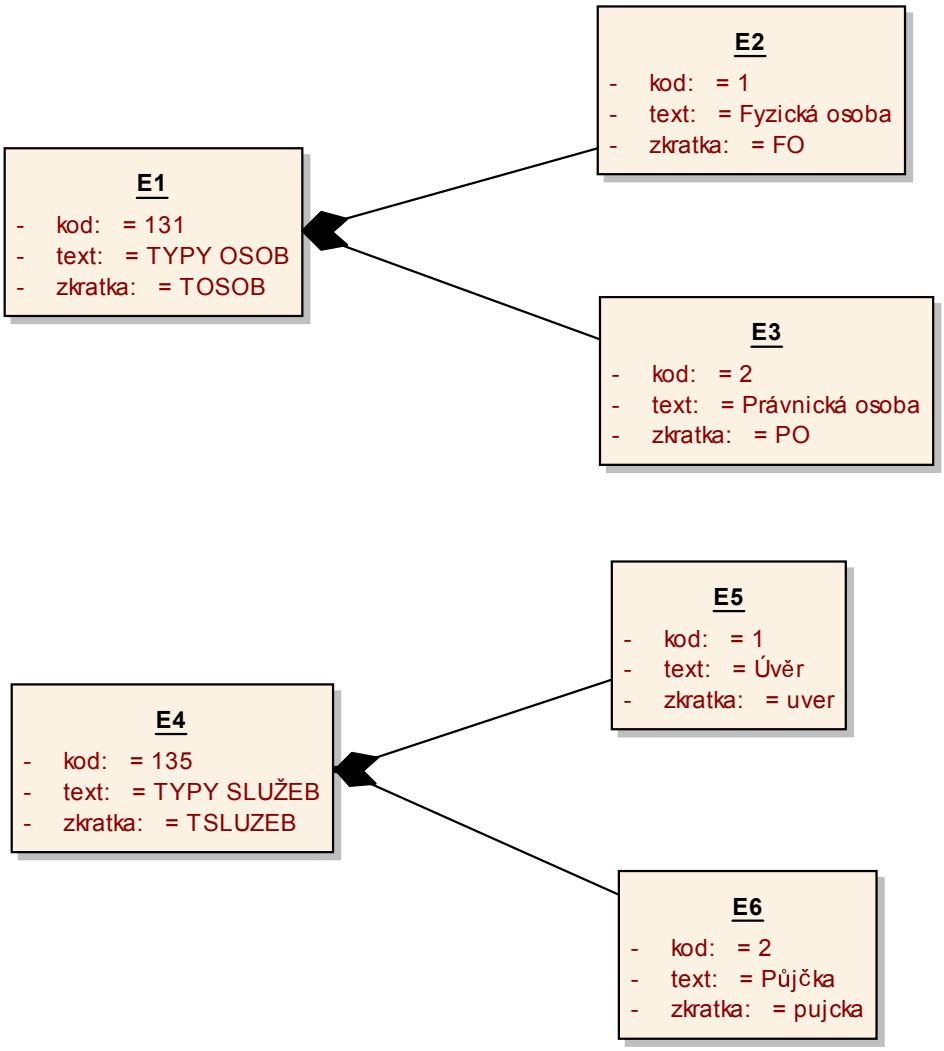
V dokumentu build 1 v textu WORDU (nikoliv v diagramu) bylo chybně uvedeno, že prvek Abstraktní Item obsahuje atributy kód, text, zkratku. Správný je diagram ***obrázek 4 Class diagram***, tj. v textu má být správně uvedeno, že tyto atributy obsahuje prvek Single-Item. Byly provedeny opravy v textu WORDU, diagram se nemění.

Byly provedeny drobné úpravy nalezených pravopisných chyb a překlepů.

1. INSTANCE DIAGRAMS

Prvním cílem agendy je zavést dynamické výčtové typy, tzv. číselníky, což jsou jednoduché entity obsahující kód, text a zkratku (pozn.: podle potřeby může být rozšířeno o další potřebné atributy).

Příklad takové evidence v run-time v běhu v bankovním systému:



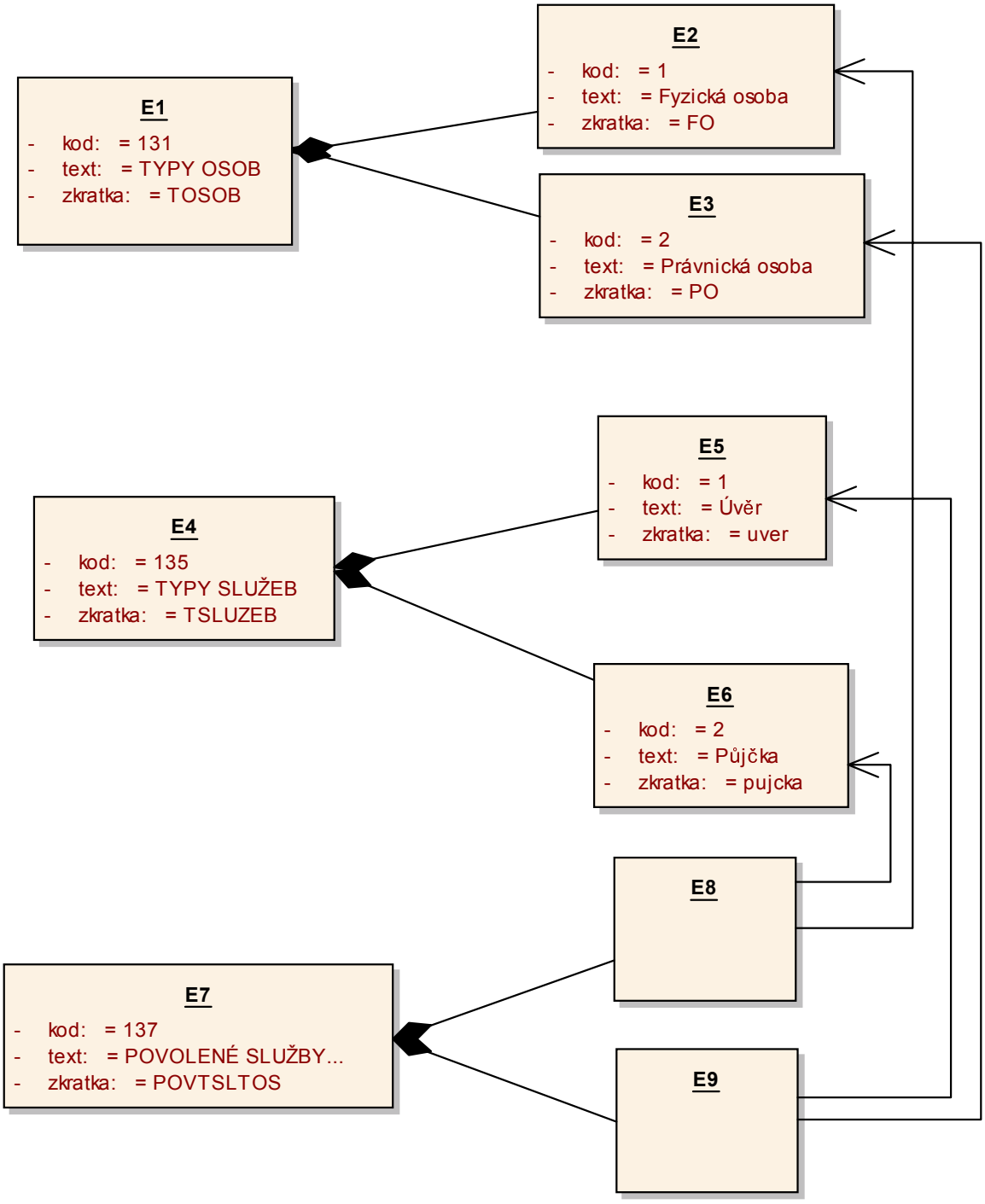
obrázek 1 : Příklad evidence číselníků

Evidovaný element E1 (evidovaná instance) reprezentuje první číselník s názvem (viz hodnota textu) TYPY OSOB, obsahuje dva itemy E2 a E3, druhý element E4 reprezentuje druhý číselník s názvem TYPY SLUŽEB (např. možné typy služeb, které lze poskytnout klientům v bance).

Předešlý příklad evidence vede k zobecnění: Číselníky (zde prvky E1 a E4) lze chápat také jako itemy – tj. prvky číselníku, tzv. číselníku číselníků. Uvedená úvaha vede k dvojúrovňovému stromu.

Kromě těchto jednoduchých entit lze také pracovat s dalším typem číselníku, jehož prvky reprezentují propojení dvou předešlých itemů do asociace.

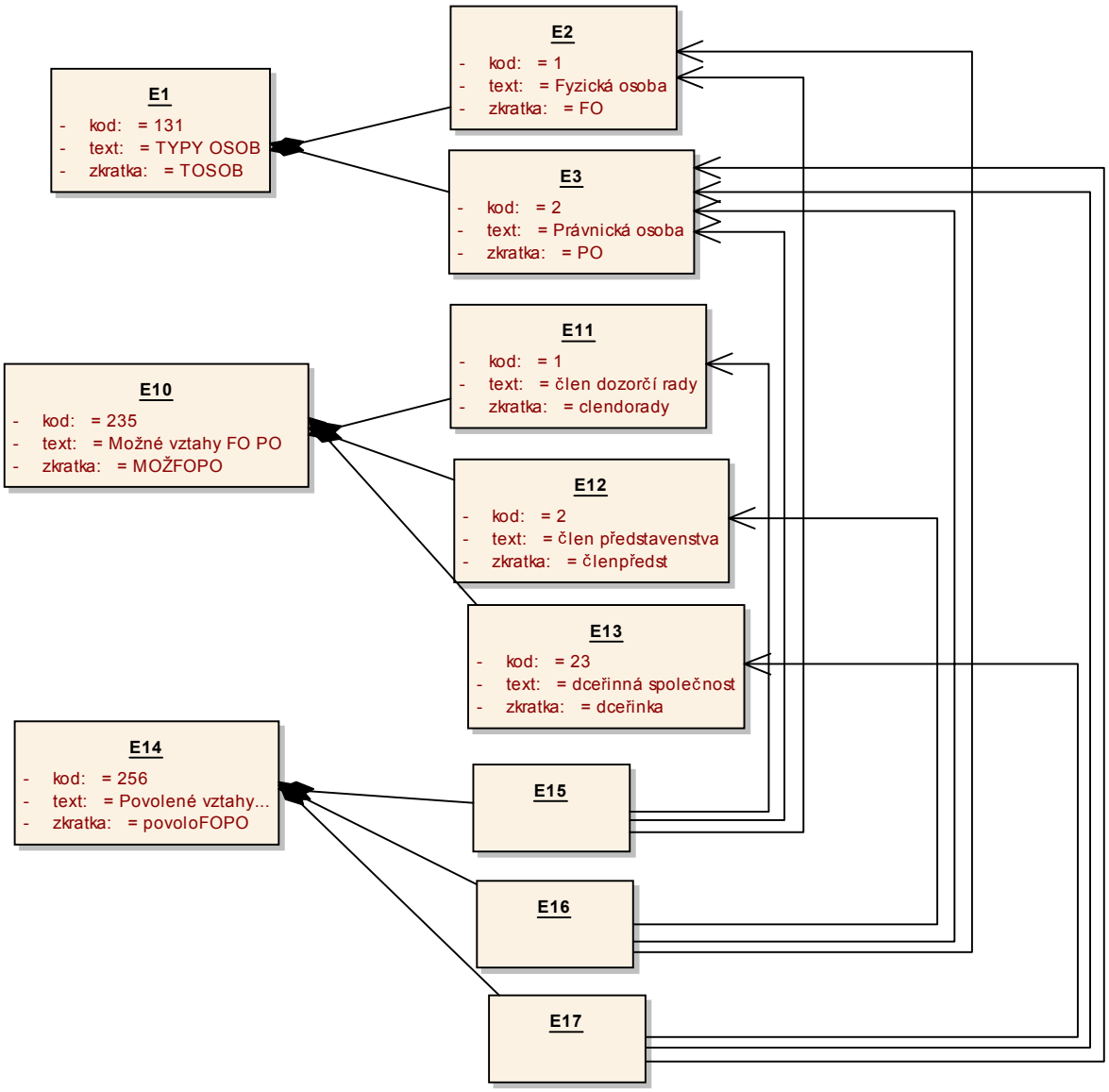
Příklad evidence v bankovním systému, kdy se potřebuje evidovat vztah „Povolené typy služeb pro typy osob“. Zde například fyzická osoba může mít službu typu půjčka a nemůže mít službu typu úvěr (a naopak).



obrázek 2 : Další typ itemu (E8 a E9) propojující dva itemy

V tomto případě číselník E7 Povolené bankovní služby obsahuje dva itemy, které mají jinou povahu, než itemy uvedené na diagramu - *obrázek 1 : Příklad evidence číselníků*. Tyto itemy nemají ani kód, text a zkratku, ale dva odkazy na dva itemy z jiných číselníků. Tyto odkazy musí být v daném číselníku jako dvojice unikátní a realizují tak asociativní vazbu.

Podobně budou potřebné také číselníky s itemy, které propojují tři itemy. Například v evidenci osob „povolené typy vztahů mezi typy osob“:



obrázek 3 Zavedení itemů jako propojení tří prvků

Například prvek E15 propojuje tři itemy číselníků a uvádí povolený vztah: Fyzická osoba – Právnická osoba – Člen dozorčí rady, podobně prvek E17 uvádí povolený vztah: Právnická osoba – Právnická osoba – Dceřinná společnost.

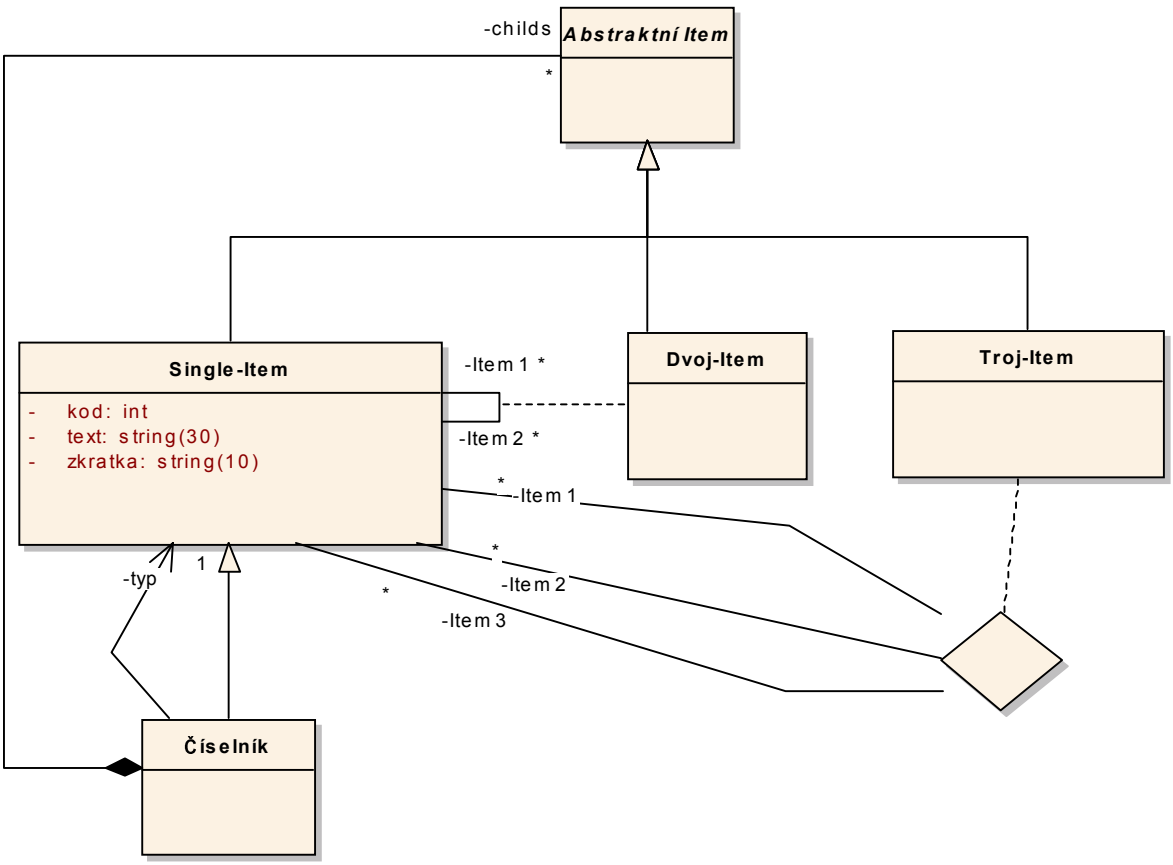
Je třeba podotknout, že číselník E10 zavádí všechny možné vztahy mezi osobami (celá „hromada vztahů“ - což se někdy používá samo o sobě bez dalšího omezení),

navíc však číselník E14 udává povolené kombinace. Také trojice odkazů musí být v daném číselníku unikátní.

Čtyř-itemy nebudeme zavádět, ale pokud budou třeba, je zřejmé, jakou by měly strukturu.

2. CLASS DIAGRAMS

Uvedené tři možné příklady evidence je třeba zobecnit do meta-pravidla, tj. do diagramu tříd:



obrázek 4 Class diagram

Třída ***Abstraktní Item*** je abstraktní. Prvek typu ***Číselník*** může obsahovat buď děti typu ***Single-Item***, nebo typu ***Dvoj-Item***, anebo typu ***Troj-Item***. Třída ***Dvoj-Item*** je asociativní třídou a třída ***Troj-Item*** je násobnou asociativní třídou. Děti v seznamu musí být vždy stejného typu.

Rozlišují se typy číselníků - nikoliv různé třídy, ale instančně odkazem do číselníku typů, viz směrová asociace z třídy ***Číselník*** do třídy ***Single-Item*** s rolí na konci „typ“. Jejich seznam je obsažen v konfiguračním prvku ***Číselník Typy číselníků***.

2.1 Seznam tříd

2.1.1. Abstraktní Item

Abstraktní třída. Vrcholová třída stromu dědičnosti.

2.1.2. Single-Item

Koncový list klasického číselníku. Obsahuje atributy kód, text, zkratku.

2.1.3. Dvoj-Item

Asociativní třída - provazuje dva prvky typu ***Single-Item*** anebo jeho dědiců. Koncový list.

2.1.4. Troj-Item

Násobná asociativní třída - provazuje tři prvky typu ***Single-Item*** anebo typu ***Číselník***. Koncový list.

2.1.5. Číselník

Třída reprezentuje číselníky. Prvek umí pracovat s kódem, textem, zkratkou díky dědičnosti. Každý prvek obsahuje v kompozici prvky typu ***Abstraktní Item***, v nichž jsou prvky typu buď ***Single-Item*** resp. ***Dvoj-Item*** resp. ***Troj-Item*** . Každý prvek si ukazuje do číselníku s názvem Číselník Typy číselníků, čímž se rozlišuje typ číselníku (typy nejsou reprezentovány různými třídami, ale pouze tímto odkazem).

3. CONFIGURATION INSTANCES

3.1 Číselník Typy číselníků

Prvek typu ***Číselník*** obsahující výčet povolených typů číselníků v prvcích typu ***Single-Item***.

Jeho doporučený obsah v textovém tvaru je (pro tuto verzi) následující:

Hlavička

Kód	Text	Zkratka
1	Typy číselníků	TCIS

Děti

Kód	Text	Zkratka
1	Číselník klasický	TC1
2	Číselník pro 2 odkazy	TC2
3	Číselník pro 3 odkazy	TC3

4. CONSTRAINTS

4.1 Děti v každém číselníku jsou všechny téhož typu a to přímí dědicové typu Abstraktní Item: Single-Item resp. Dvoj-Item, resp. Troj-Item (odpovídá odkazu do Číselník Typy číselníků), případně dalších budoucích typů

4.2 Kód v prvku Single-Item je unikátní v dětech číselníku

4.3 Text v prvku Single-Item je unikátní v dětech číselníku

4.4 Kód ani text není prázdný v Single-Item

4.5 Dvojice odkazů na dva itemy v dětech číselníku pro 2 odkazy je unikátní

4.6 Ani jeden z odkazů na dva itemy v prvku Dvoj-Item není prázdný

4.7 Trojice odkazů na tři itemy v dětech číselníku pro 3 odkazy je unikátní

4.8 Ani jeden z odkazů na 3 itemy v prvku Troj-Item není prázdný

4.9 Prvky na koncích asociací asociativních prvků Dvoj-Item a Troj-Item jsou typu Single-Item resp. jeho dědice Číselník

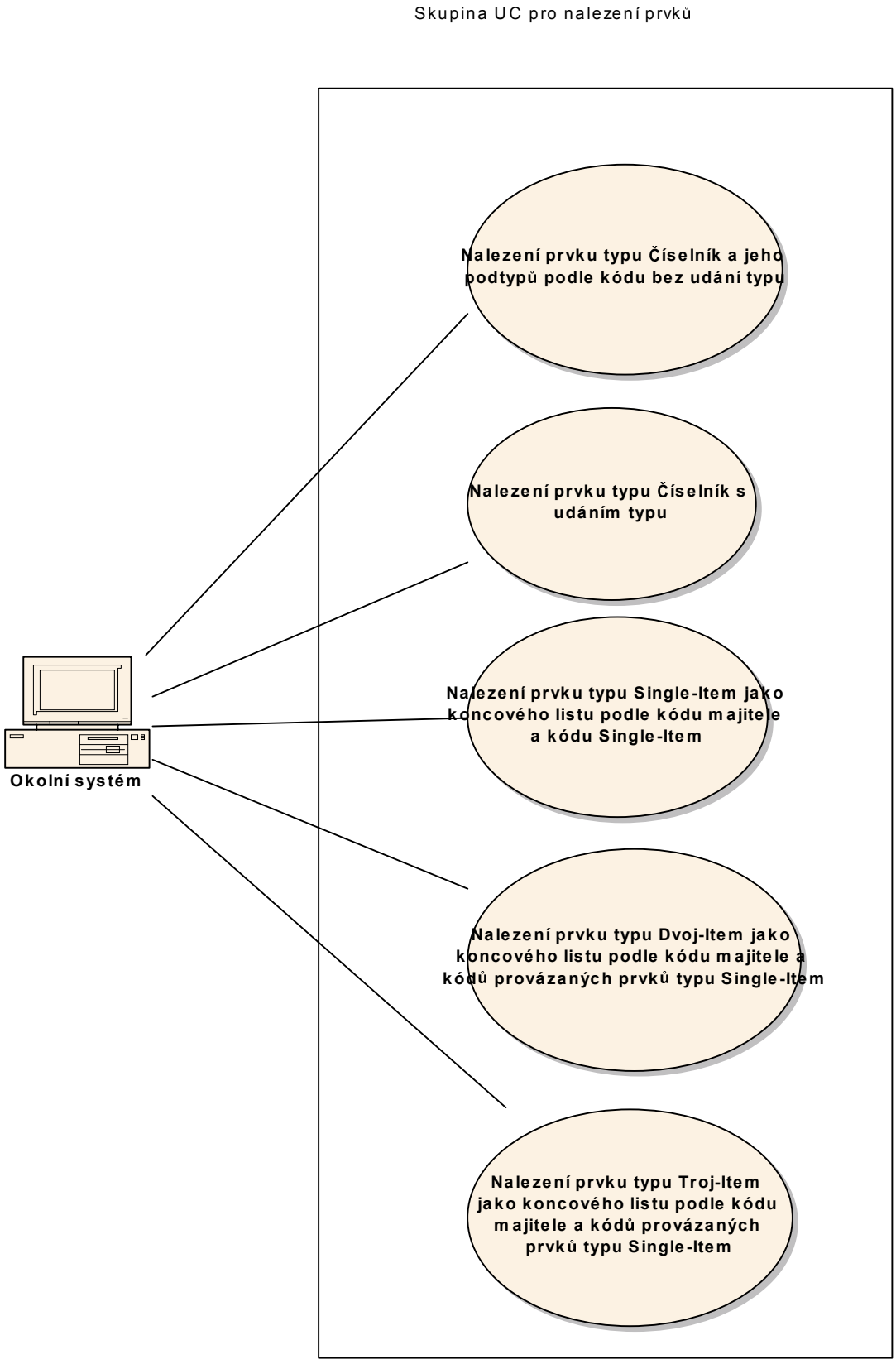
5. USE CASE DIAGRAM

5.1 External Profitable USE CASES

Případy užití, které vznikají jako požadavky na systém díky chování a požadavkům z okolí. Přinášejí přímý užitek pro okolí. Mají povahu podobnou jako „tlačítko pro užití“ na systému.

5.1.1. Skupina UC pro nalezení prvků

DIAGRAM:



obrázek 5 Skupina UC pro nalezení prvků

5.1.1.1 Nalezení prvku typu *Číselník* a jeho podtypů podle kódu bez udání typu

PRECONDITION: Je zadán kód číselníku jako vstup.

SCENARIO:

V seznamu prvků typu ***Číselník*** se nalezne odpovídající prvek podle kódu (bez ohledu na typ).

Vrací se nalezený číselník typu ***Číselník***, pokud prvek nenalezen, tak ERROR.

5.1.1.2 Nalezení prvku typu Číselník s udáním typu

PRECONDITION: Je zadán kód číselníku a kód typu číselníku jako vstup.

SCENARIO:

V seznamu číselníků s omezením odkazu na daný prvek typ se nalezne odpovídající prvek podle kódu.

Vrací se nalezený číselník typu ***Číselník***, pokud prvek nenalezen, tak ERROR.

5.1.1.3 Nalezení prvku typu *Single-Item* jako koncového listu podle kódu majitele a kódu *Single-Item*

PRECONDITION: Je zadán kód pro ***Číselník*** a kód pro ***Single-Item*** z tohoto číselníku jako vstup.

SCENARIO:

Podle kódu se v seznamu číselníků nalezne prvek typu ***Číselník*** a současně podle kódu ***Single-Item*** se nalezne odpovídající prvek v seznamu prvků nalezeného číselníku. Vrací se typ ***Single-Item***, pokud nenalezen, tak ERROR.

5.1.1.4 Nalezení prvku typu *Dvoj-Item* jako koncového listu podle kódu majitele a kódů provázaných prvků typu *Single-Item*

PRECONDITION: Je zadán jednak kód prvku typu Číselník, a dvakrát dvojice kód číselníku a kód prvku ***Single-Item*** z tohoto číselníku jako vstup.

SCENARIO:

Podle kódu prvku typu ***Číselník*** se v seznamu číselníků nalezne odpovídající číselník a současně se nalezne jeho dítě typu ***Dvoj-Item*** a to tak, že se nalezne první i druhý odkaz na prvek typu ***Single-Item*** nalezeného prvku. Vrací prvek typu Dvoj-Item, pokud nenalezen, ERROR

5.1.1.5 Nalezení prvku typu *Troj-Item* jako koncového listu podle kódu majitele a kódů provázaných prvků typu *Single-Item*

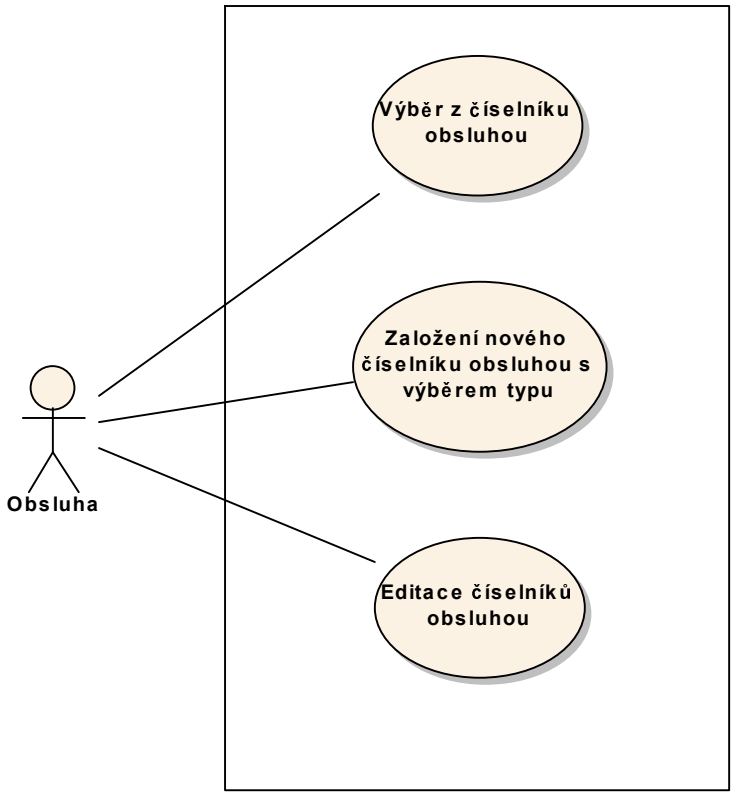
PRECONDITION: Je zadán jednak kód číselníku, a třikrát dvojice kód číselníku a kód ***Single-Item*** z tohoto číselníku jako vstup.

SCENARIO:

Postupuje se obdobně, jako u UC ***Nalezení prvku typu Dvoj-Item jako koncového listu podle kódu majitele a kódů*** , pouze se nalezne prvek s třemi odkazy na prvky typu ***Single-Item*** (nikoliv na dva).

5.1.2. Skupina UC pro práci obsluhy

Diagram:



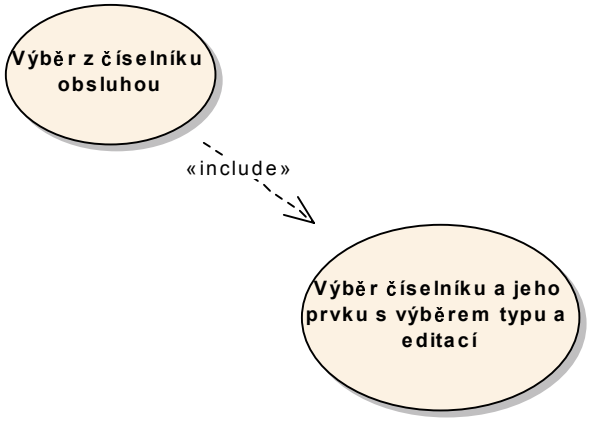
obrázek 6 Skupina UC pro práci obsluhy

5.1.2.1 Výběr z číselníku obsluhou

SCENARIO:

Provede se UC ***Výběr číselníku a jeho prvku s výběrem typu a editací***, přičemž status ***Změny povoleny*** se nastaví na ne, ostatní statusy se ponechají v té hodnotě, jak budou v té chvíli nastaveny (případně z prvního inicializačního nastavení).

Diagram:



obrázek 7 Výběr z číselníku obsluhou

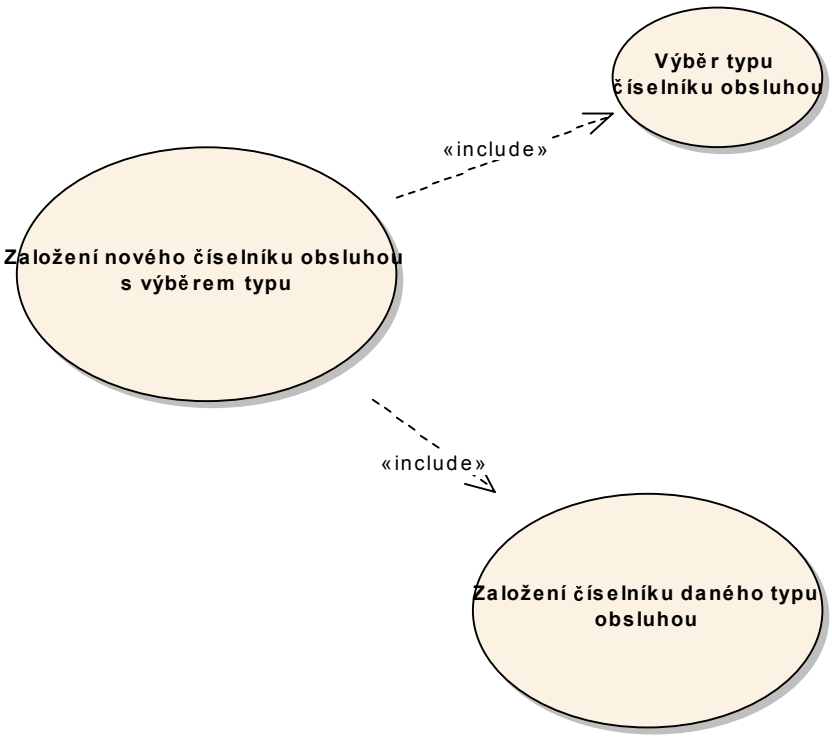
5.1.2.2 Založení nového číselníku obsluhou s výběrem typu

SCENARIO:

Obsluha provede UC ***Výběr typu číselníku obsluhou***.

Dále se pokračuje UC ***Založení číselníku daného typu obsluhou***.

Diagram:



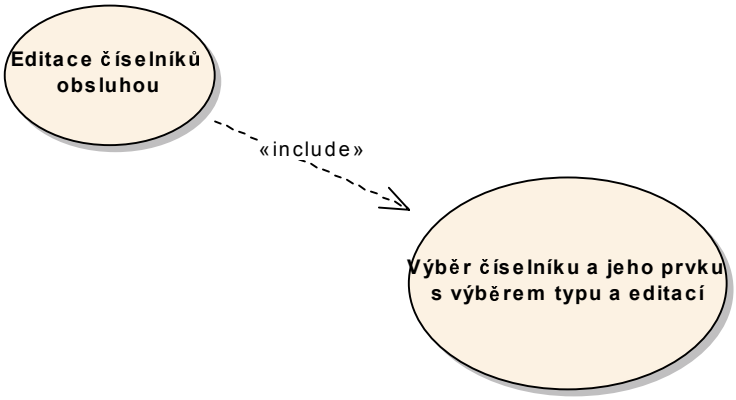
obrázek 8 Založení nového číselníku obsluhou s výběrem typu

5.1.2.3 Editace číselníků obsluhou

SCENARIO:

Provede se UC ***Výběr číselníku a jeho prvku s výběrem typu a editací***, před tím se nastaví status ***Změny povoleny*** na „povoleno“, ***Filtr pro typ*** na „všechny“ a status ***Co vybrat*** na „koncový list“.

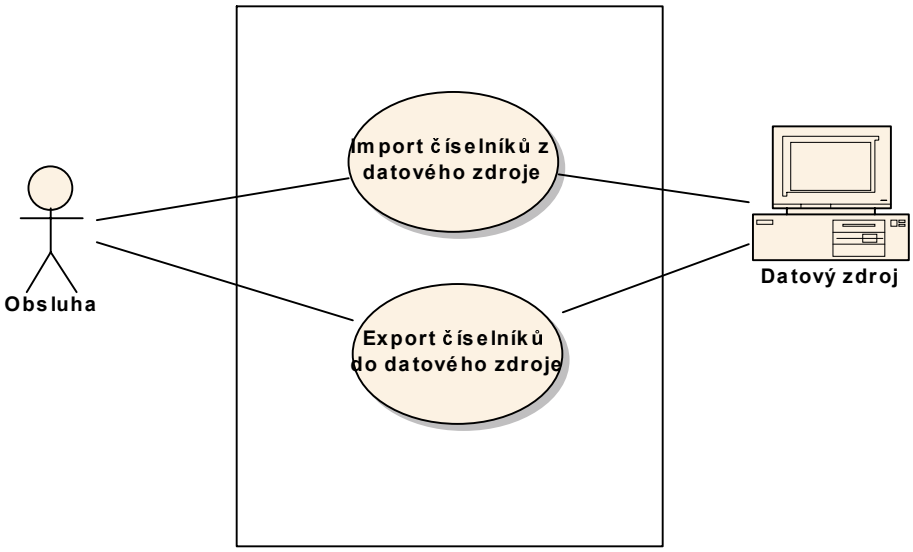
DIAGRAM:



obrázek 9 Editace číselníků obsluhou

5.1.3. Skupina UC pro export a import

DIAGRAM:



obrázek 10 Skupina UC pro export a import

5.1.3.1 Import číselníků z datového zdroje

Obsluha nastaví cestu k datovému zdroji (soubor XML, DB zdroj apod.).

Obsluha nastaví status na „udpate povolen ano/ne“. Pokud je povolen „update“, tak se při importu mohou údaje v již existujících číselnících měnit, pokud je status „update“ nastaven na „ne“, lze importem pouze přidávat prvky (nikoliv měnit údaje již existujících prvcích).

Import se provádí v kontextu diagramu tříd, viz ***CLASS DIAGRAMS***.

Obsluha může nastavit status požadovaná transakce: První volba je transakce na úrovni číselníku (implicitní), v tom případě se při nezdaru importu nezapisuje pouze daný číselník, ale ostatní „úspěšné“ číselníky se importují, anebo druhá volba je transakce na celý import, v tom případě se při nezdaru neimportuje ani jeden prvek z žádného číselníku.

Nejprve se pro daný číselník načte do systému tzv. „hlavička“ (pole kód, text, zkratka), využije se dědičnosti Číselník ze ***Single-Item*** (hlavička je děděním de-facto ***Single-Item***). Pokud je v polích pro hlavičku uveden kód typu číselníku (odpovídá kódům ***Číselník Typy číselníků***), načte se typ, pokud není uveden, implicitně se nastaví na „klasický číselník“.

Podle načteného pole kódu a pole typu se nalezne číselník v seznamu číselníků (viz ***Nalezení prvku typu Číselník s udáním typu***). Pokud číselník není nalezen,

provede se přidání číselníku. Pokud je nalezen a je povolen „update“, provede se změna v hlavičce (pozor na CONSTRAINTS, viz konec kapitoly).

Dále se N-krát provede import dětí (childs) typu ***Abstraktní Item*** podle typu dědice.

Pro typ číselníku „klasický“ se pro dítě typu ***Single-Item*** načte pole kód, text, zkratka, pokud je nalezen prvek dítě daného číselníku podle kódu a je povolen update, provedou se změny (viz CONSTRAINTS), v textu a zkratce, pokud není nalezen, provede se přidání.

Pro typ dítěte ***Dvoj-Item*** se z datového zdroje načtou dvakrát dvojice polí - kód číselníku a kód dítěte (pozn.: kód číselníku může případně být kódem pro „root“jako seznam všech číselníků, například „0“, „nic“ apod., druhý kód je potom kódem číselníku v seznamu číselníků) a najde se podle této „čtveřice kódů“ dítě v číselníku prvek typu ***Dvoj-Item***. Pokud neexistuje, přidá se nový prvek ***Dvoj-Item***, pokud existuje, neprovádí se nic.

Pro typ dítěte ***Troj-Item*** se postupuje podobně, pouze se jedná o tři dvojice kódů.

Všechny kroky importu se zapisují do logovacího souboru.

CONSTRAINTS

Jsou splněny všechny omezující podmínky uvedené v kapitole

CONSTRAINTS.

DIAGRAM: Není doložen (vše v textovém tvaru).

5.1.3.2 Export číselníků do datového zdroje

Export se provádí ve stejném kontextu jako import, pouze s obráceným tokem dat:

Obsluha může nastavit status požadovaná transakce: První volba je transakce na úrovni číselníku (implicitní), v tom případě se při nezdaru exportu neexportuje pouze daný číselník, ale ostatní se exportují, anebo druhá volba je transakce na celý export – v případě nezdaru se neexportuje žádný prvek.

Obsluha nastaví cestu k datovému zdroji (soubor XML, DB zdroj apod.). Obsluze se zobrazí seznam číselníků, obsluha vybere číselníky pro export, může zvolit „všechny“.

Po odsouhlasení se u všech vybraných číselníků provede zápis jednak hlavičky (kód, text, zkratka, typ) a poté se provede N - krát export dětí podle typu číselníku:

Pro klasický číselník N krát zápis ***Single-Item*** - text, kód, zkratka.

Pro prvky typu ***Dvoj-Item*** N krát dvojice kód číselníku a kód dítěte.

Pro prvky typu ***Troj-Item*** N-krát trojice kód číselníku a kód dítěte.

Všechny kroky exportu se zapisují do logovacího souboru.

CONSTRAINTS: Jsou splněny všechny omezující podmínky uvedené v kapitole

CONSTRAINTS

DIAGRAM: Není doložen (vše v textovém tvaru).

5.2 Internal Unprofitable USE CASES

5.2.1. Založení číselníku daného typu obsluhou

PRECONDITION: Je vybrán item odpovídajícího typu číselníku, který má být založen.

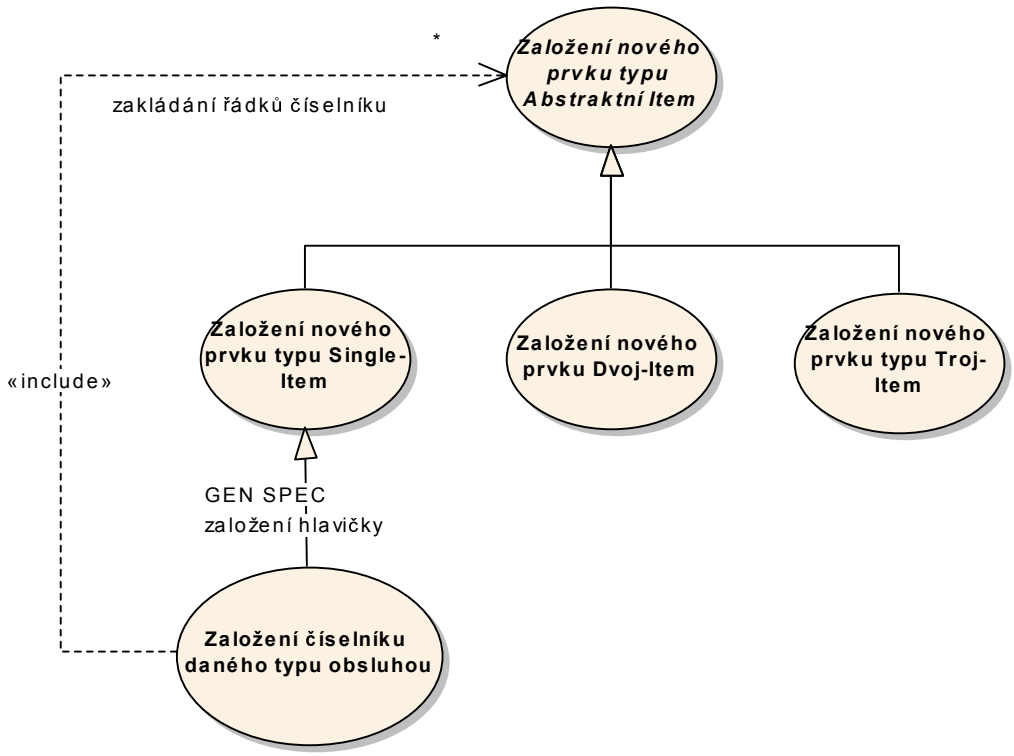
Založí se nový prvek ***Číselník*** a prováže se odkazu „typ“ (viz ***obrázek 4 Class diagram***) do již vybraného prvku – typu v číselníku ***Číselník Typy číselníků***

Pro založení hlavičky se použije z GEN-SPEC UC ***Založení nového prvku typu Single-Item***, kde majitelem bude seznam všech číselníků.

Dále se N-krát založí odpovídající dědic typu ***Abstraktní Item***, tj. odpovídající UC zakládání těchto dědiců (***Založení nového prvku typu Single-Item***, ***Založení nového prvku Dvoj-Item***, atd.).

Nový prvek se vloží do seznamu dětí číselníku.

DIAGRAM:



obrázek 11 Založení číselníku daného typu obsluhou

5.2.2. Založení nového prvku typu Abstraktní Item

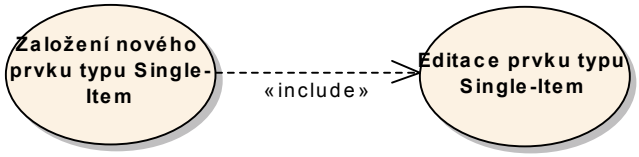
Abstraktní případ užití, který je poděděn.

5.2.3. Založení nového prvku typu *Single-Item*

PRECONDITION : Je vybrán majitel prvku.

Založí se nový prvek typu ***Single-Item*** resp. jeho dědice a provede se ***Editace prvku typu Single-Item*** s tím, že Možnost editace kódu se zapne na ano.

DIAGRAM:



obrázek 12

5.2.4. Založení nového prvku Dvoj-Item

Poznámka: Tento UC je dědicem UC Založení Abstraktní Item.

Založí se prvek typu ***Dvoj-Item*** a provede se jeho editace, viz UC ***Editace prvku typu Dvoj-Item***. Nový prvek se přidá do seznamu dětí číselníku.

CONSTRAINTS:

Dvojice odkazů na dva itemy v dětech číselníku pro 2 odkazy je unikátní.

Ani jeden z odkazů na dva itemy v prvku Dvoj-Item není prázdný

DIAGRAM: Není přiložen, stejný kontext jako na diagramu - viz obrázek 12

5.2.5. Založení nového prvku typu *Troj-Item*

Postupuje se obdobně jako v ***Založení nového prvku Dvoj-Item***, pouze s tím rozdílem, že se nevyplňují dva konce vztahu, ale tři konce vztahu.

CONSTRAINTS:

Ani jeden z odkazů na 3 itemy v prvku Troj-Item není prázdný

Trojice odkazů na tři itemy v dětech číselníku pro 3 odkazy je unikátní

DIAGRAM: Není přiložen, stejný kontext jako na diagramu - obrázek 12

5.2.6. Editace prvku Abstraktní Item

Abstraktní prázdný UC, je děděn.

5.2.7. Editace prvku typu Single-Item

PRECONDITION:

Je vybrán prvek typu ***Single-Item*** nebo typu jeho dědic.

Je zapnut status Možnost editace pro kód na ano/ne.

SCENARIO:

Obsluha může změnit text a zkratku, kód může měnit pouze v případě, že je zapnuta Možnost editace na ano.

CONSTRAINTS:

Kód v prvku Single-Item je unikátní v dětech číselníku

Text v prvku Single-Item je unikátní v dětech číselníku

Kód ani text není prázdný v Single-Item

DIAGRAM: Není, prvek nepoužívá žádný jiný prvek.

5.2.8. Editace prvku typu Dvoj-Item

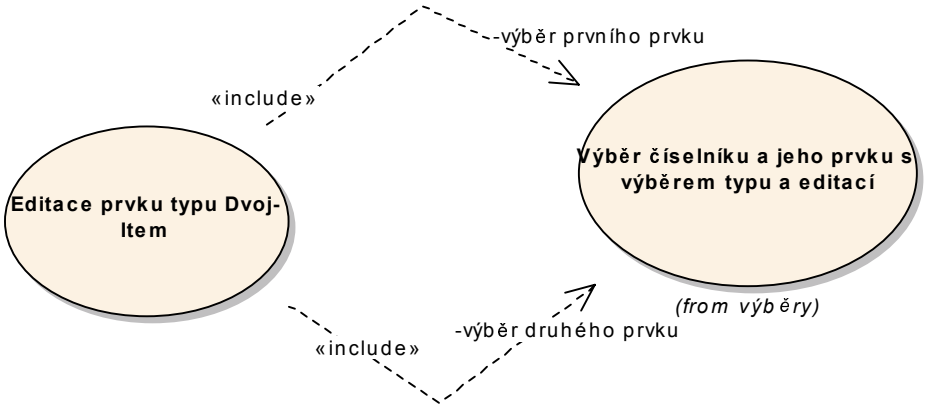
Obsluha může změnit jeden nebo oba odkazy na prvky na koncích vazeb vybraného prvku výběrem prvků z klasických číselníků. Přitom je povoleno do konce vazeb dosadit odkaz na číselník (nejenom na koncový list). Provede se to pomocí dvojího použití UC ***Výběr číselníku a jeho prvku s výběrem typu a editací*** s nastaveným statusem ***Změny povoleny*** na nepovoleno a ***Filtr pro typ*** nastaveným na klasický číselník, přičemž zda se jedná o výběr koncového listu nebo číselníku má možnost obsluha změnit (viz uvedený UC).

CONSTRAINTS:

Ani jeden z odkazů na dva itemy v prvku Dvoj-Item není prázdný

Dvojice odkazů na dva itemy v dětech číselníku pro 2 odkazy je unikátní

DIAGRAM:



obrázek 13

5.2.9. Editace prvku typu Troj-Item

Provádí se stejně jako u ***Editace prvku typu Dvoj-Item*** pouze s tím rozdílem, že se mohou změnit tři odkazy a nikoliv pouze dva.

CONSTRAINTS:

Ani jeden z odkazů na 3 itemy v prvku Troj-Item není prázdný

Trojice odkazů na tři itemy v dětech číselníku pro 3 odkazy je unikátní

DIAGRAM: Stejný kontext jako diagram - ***obrázek 13***, pouze se použijí tři interakce <<include>>.

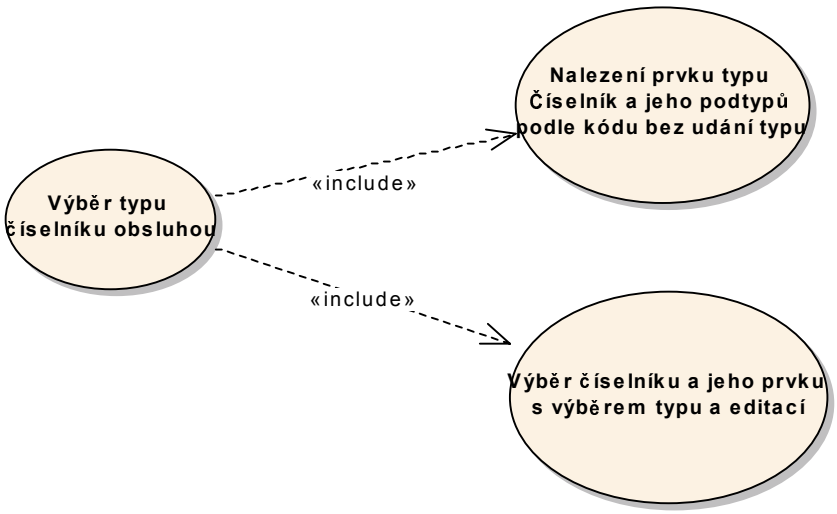
5.2.10. Výběr typu číselníku obsluhou

Nalezne se prvek ***Číselník Typy číselníků*** podle konfiguračního kódu tohoto číselníku tak, že se nastaví kód pro ***Číselník Typy číselníků***, což je konfigurační hodnota (např. 1), a provede se UC ***Nalezení prvku typu Číselník a jeho podtypů podle kódu bez udání typu***.

Provede UC ***Výběr číselníku a jeho prvku s výběrem typu a editací*** s tím, že jako Implicitní číselník se nastaví tento nalezený, status ***Možnost změny číselníku*** se nastaví na ne a status ***Změny povoleny*** také na ne.

POSTCONDITION: Z číselníku ***Číselník Typy číselníků*** vybrán item číselníku odpovídající požadovanému typu číselníku.

DIAGRAM:



obrázek 14 Výběr typu číselníku obsluhou

5.2.11. Výběr číselníku a jeho prvku s výběrem typu a editací

PRECONDITIONS

Mohou být nastaveny tyto statusy:

5.2.11.1 Filtr pro typ

Odpovídá mu jedna z hodnot Číselník Typy číselníků anebo „všechny typy“, první nastavení je na „všechny typy“.

5.2.11.2 Implicitní číselník

Prvek typu Číselník, první nastavení je na první v seznamu, jinak vždy poslední nastavený. Nastavuje se automaticky výběrem číselníku.

5.2.11.3 Možnost změny číselníku

Status nastaven na ano/ne. Pokud je nastaven na ne, obsluha může vybírat pouze z daného implicitního číselníku a nemůže jej změnit.

5.2.11.4 Co vybrat

Status s hodnotou „číselník / koncový list“. Buď je výsledkem výběru číselník anebo koncový list, implicitní je poslední nastavený status, první nastavení je na „koncový list“. Je obsluhou měnitelný.

5.2.11.5 Změny povoleny

Status Změny povoleny na „ano/ne“. První nastavení je na nepovoleno. Status není obsluhou měnitelný, udává, zda lze v číselnících a jeho dětech provádět změny.

SCENARIO:

Pokud je nastaven status ***Možnost změny číselníku*** na ano, obsluha má možnost změnit parametry statusů: Vybrat omezení ***Filtr pro typ*** (použije se UC ***Výběr typu číselníku obsluhou***, k tomu má možnost také zvolit položku „všechny typy“) a status ***Co vybrat*** (koncový list nebo číselník).

Pokud je status ***Možnost změny číselníku*** nastaven na ano, obsluze se zobrazí seznam všech číselníků (kód, text, zkratka) podle nastaveného statusu ***Filtr pro typ***. Pokud je nastaven na konkrétní typ, zobrazí se číselníky pouze tohoto typu. Nastaví se číselník ve výběru jako ***Implicitní číselník***. Obsluha může vybrat číselník ze zobrazeného seznamu.

Pokud je status ***Možnost změny číselníku*** nastaven na ne, předchozí výběr číselníku se nenabízí a jako vybraný se nastaví přímo ***Implicitní číselník*** (bez možnosti jeho změny).

Vybraný číselník se dosadí do prvku ***Implicitní číselník***.

Pokud je nastaven status ***Změny povoleny*** na povoleno, obsluha může editovat tzv. hlavičku, viz UC ***Editace prvku typu Single-Item***.

Pokud je nastaven status ***Co vybrat*** na „číselník“, nelze ve výběru pokračovat dále a obsluha má možnost pouze výběr ukončit (je vybrán číselník) resp. storno výběru) příp. vybrat jiný číselník.

Pokud je nastaven status ***Co vybrat*** na „koncový list“, obsluze se zobrazí seznam dětí vybraného číselníku - pro každý typ dítěte jinak: ***Single-Item*** jako kód text zkratka, ***Dvoj-Item*** jako dvojice kódů, textů a zkratek, podobně ***Troj-Item*** jako trojice kódů, textů a zkratek.

Obsluha vybere daný prvek dítěte.

Pokud je nastaven status ***Změny povoleny*** na ano, obsluha může dále editovat N-krát dědice prvku ***Abstraktní Item***, tj. daný konkrétní UC se děje podle podtypu Itemu.

Obsluze je dovoleno jakkoliv se vracet v postupu průchodu mezi číselníky a jejich dětí.

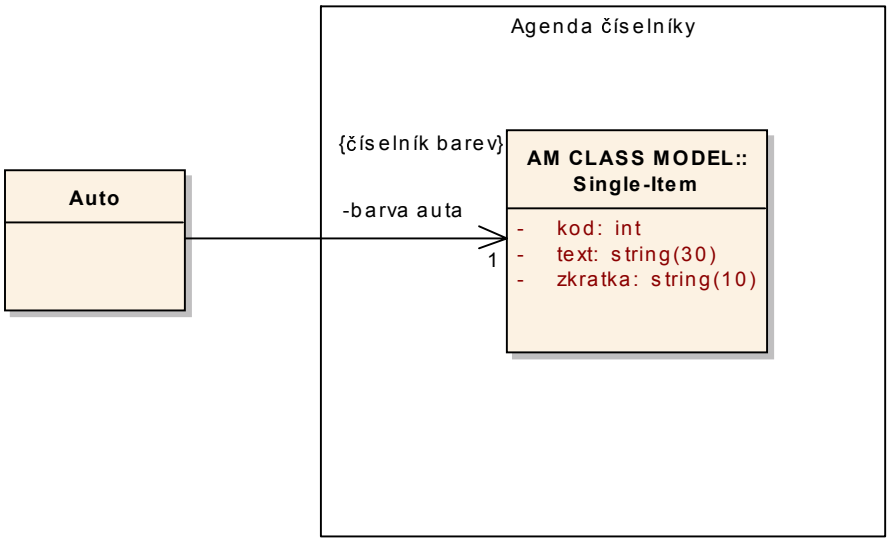
DIAGRAM: není doložen, pouze textový popis.

5.3 Příloha

5.3.1. Použití agendy pro prvky typu Single-Item

Uvedená konstrukce číselníků umožňuje několikeré použití. V prvé řadě se jedná o „klasické“ použití číselníků podle vzoru „Auto má Barvu“. Prvek ze třídy Auto si „ukazuje“ na jeden z prvků koncového listu číselníků, konkrétně číselníku Barev (číselník, který má koncové listy typu ***Single-Item***)

Vzhledem k flexibilní konstrukci agendy (dvojúrovňový strom), může teoreticky dojít k ukázání si na prvek z jiného číselníku, než je žádáno, protože prvky ***Single-Item*** všech takovýchto číselníků jsou téhož typu. Toto omezení je třeba zohlednit na úrovni instancí, nikoliv na úrovni tříd, což je v analytickém modelu vyjádřeno elementem typu CONSTRAINT, viz diagram:



obrázek 15 Auto má Barvu z číselníku Barev

V některých systémech (nikoliv v této agendě) je každý číselník řešen jako samostatná entita a proto se programuje znovu a znovu. Číselník se oslovuje volbou typu (příslušná třída v OOP resp. tabulka atd.), například v pseudokódu (design pro OOP) takto:

```
MyBarva = colBarvy.GetItem(kód=3);
```

kde prvek `colBarvy` reprezentuje seznam barev ze třídy `CcolBarvy`. Pro jiný číselník je třeba založit novou třídu. Za hranicí datového objektu dojde pro vytěžení příslušných dat ke klasickému příkazu SELECT do příslušné tabulky (opět „svůj specifický“ typ) pro daný číselník, např. takto (pseudokód) :

```
SELECT * FROM TBARVY WHERE column_Kod = 3
```

Oproti tomu v řešení, které nabízí tato agenda, jsou všechny číselníky v jedné třídě a k oslovení dochází přes volbu instance číselníku (nikoliv typu):

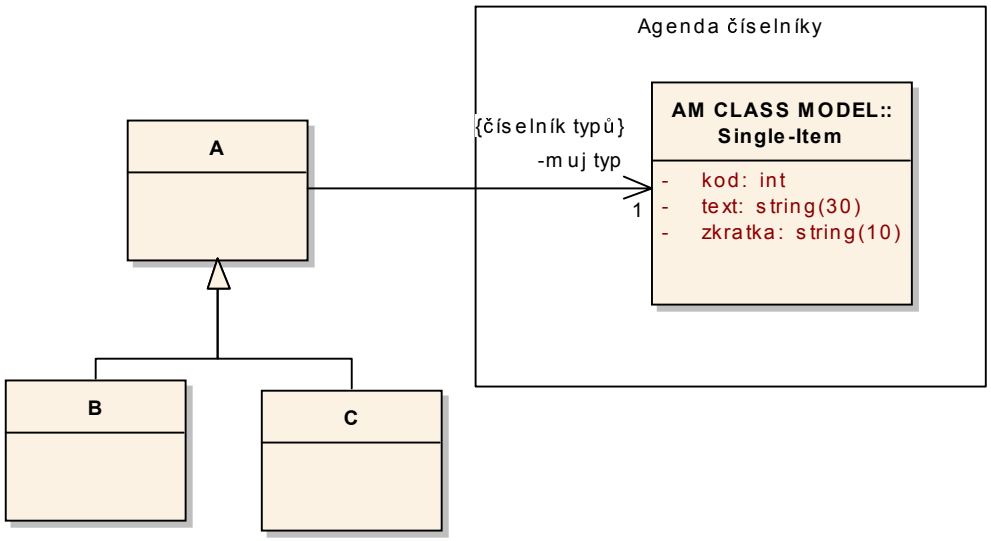
```
MyBarvy  = colCis.GetItem(kód = cBarvy);
MyBarva = MyBarvy.GetItem(kód= 3);
```

anebo „přímo“ takto:

```
MyBarva = colCis.GetListItem(kód = cBarvy, kód= 3);
```

5.3.2. Zvláštní použití agendy pro prvky Single-Item jako ukazatele do typu

Klasické číselníky (s prvky dětí typu ***Single-Item***) by se měly použít také ve zvláštním případě ukazatele do typu, kdy si instance si ukazuje na svůj typ do číselníku typů“ (nejčastěji na úrovni nejvyšší třídy ve stromu GEN-SPEC):



obrázek 16 Instance si ukazuje na svůj typ do číselníku typů

Také v této agendě byl tento způsob použití zaveden v mírně modifikované podobě: Každý číselník rozlišuje svůj typ ukazatelem na jeden z prvků ze seznamu ***Číselník Typy číselníků***.

5.3.3. Použití agendy pro prvky typu Dvoj-Item a Troj-Item

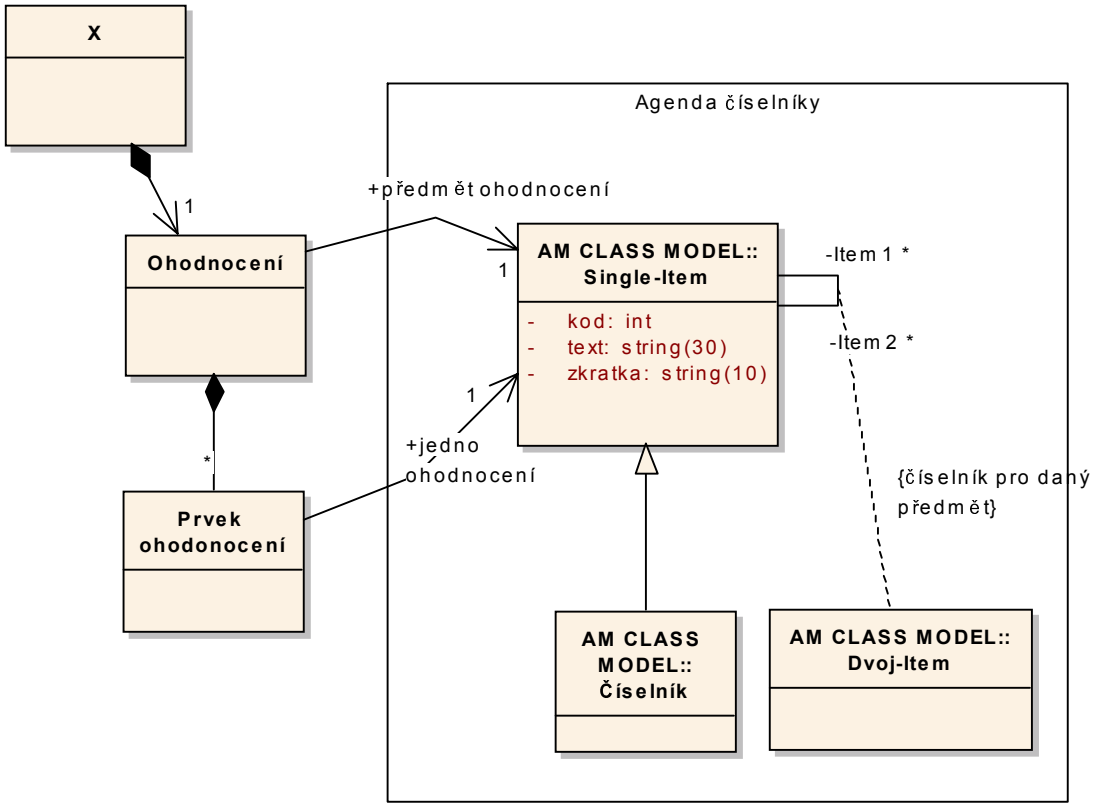
Agendu lze použít také pro číselníky s prvky typu ***Dvoj-Item*** resp. ***Troj-Item***. Jejich použití je podrobněji ukázáno v úvodních kapitolách dokumentu, viz ***INSTANCE DIAGRAMS.***

5.3.4. Prvky flexibilně ohodnocené číselníky

Jako velmi zajímavé použití této agendy je možné uvést další novou agendu zavádějící prvky, které vyžadují několikanásobné ohodnocení číselníky. Příklady lze nalézt v bankovních systémech anebo v systémech pro pojišťovny.

Existují prvky, které vyžadují, aby obsluha vyplnila N číselníků. Lze zavést jednotné řešení obecně pro všechny případy, navíc toto řešení je velmi flexibilní.

Diagram:



obrázek 17 Násobné ohodnocení číselníky

Pro vysvětlení předešlého diagramu přejdeme k příkladu v instancích.

Nechť například existuje číselník předmětů ohodnocení (klasický číselník) pro pojištění, například obsahuje tyto položky:

1 auto

2 dům

3 život

Pomocí číselníku typu Dvoj-Item s názvem „Číselníky ohodnocení přiřazené k předmětům ohodnocení“ je provázáno ke každému prvku předešlého číselníku N klasických číselníků typu Single-Item (poznámka: na konci vazby prvku Dvoj-Item může být díky dědičnosti také číselník). Například k prvku „auto“ jsou takto provázány číselníky Kubatura vozidla (s texty rozsahů „od … do …“), Typ vozidla.. atd. K prvku dům jsou přes prvky Dvoj-Item přiřazeny číselníky Rozloha (s textem „rozsah od do“).. atd. Je třeba si uvědomit, že tyto prvky včetně vazebních vznikají v run-time a nejsou dány staticky, ale dynamicky, například vyplněním obsluhou nebo importem.

Samotný proces ohodnocení předmětu pracuje tak, že na počátku je dán prvek Ohodnocení (který je vložen do prvku typu X vyžadující dané ohodnocení) a je zvolen jeho Předmět (buď výběrem anebo volbou v programu konstantou – tj. dohodou). Přes vazbu prvku typu Dvoj-Item se postupně zobrazují jednotlivé číselníky, ze kterých je třeba zvolit prvek typu Single-Item. Postupně se při každém výběru přikládají do prvku Ohodnocení jeho děti – tj. Prvky ohodnocení (podobně jako „řádky ohodnocení“), do kterých se vkládá odkaz na právě vybraný prvek z daného číselníku.

Agenda je dynamická jak pro přidání předmětů ohodnocení, tak pro přidání číselníků, ze kterých je nutno pro daný předmět vybrat daná ohodnocení.

5.3.5. Zakázané změny v číselnících

Některé záležitosti tato agenda neřeší. Existují číselníky, ve kterých je zakázáno provádět změny, ať už po celou dobu života systému anebo některé číselníky vyžadují provádět změny pouze oprávněnou osobou. To lze vyřešit několikerým způsobem podle náročnosti požadavků.

Jedna z nejjednodušších variant je zavést pomocný atribut „Možnost editace“ s hodnotami „všichni, pouze oprávněná role, nikdo“. Složitější způsoby řešení se nabízejí použitím plnohodnotné agendy přístupových práv (není zde řešeno).

5.3.6. Mazání prvků

Je zřejmé, že nelze jen tak vymazat prvky číselníků, na které si ukazují jiné prvky.

Tento problém lze řešit jednoduše při použití relační databáze se zapnutou integritou a pomocí odchycení příslušné výjimky. Ale tento přístup neřeší problém, kdy je povolen prázdný odkaz na právě vymazaný prvek.

Lze zavést odpovídající řešení pomocí vzoru OBSERVER, kdy prvky spravující seznamy prvků, z nichž vedou odkazy na v budoucnu mazaný prvek, se registrují k danému v budoucnu mazanému prvku a ten těmto seznamům posílá jako call-back svůj identifikátor těsně před vymazáním.

V tomto případě vzor OBSERVER by se měl použít dvakrát pro dvě služby call-backu:

Poprvé se použije první služba pro zavolání všem registrovaným seznamům takových prvků, které nesmí mít prázdný odkaz na mazaný prvek. Pokud dopadne u všech předešlých volání všech registrovaných seznamů jako odpověď kladně (tj. je možné mazat, protože v žádném prvku v žádném registrovaném seznamu není odkaz na mazaný prvek), tak podruhé se jako druhá služba posílá všem registrovaným seznamům prvků, které mají povolen prázdný odkaz, příkaz k změně odkazu vymazaného prvku na prázdný odkaz . Uvedené postupy musí být samozřejmě v jedné transakci. Vstupním parametrem obou služeb je vždy ID daného mazaného prvku. Není třeba zdůrazňovat, že vzor OBSERVER díky polymorfismu umožňuje flexibilní řešení volání potřebných seznamů.

5.3.7. Doporučení pro fázi Modelování designu a Implementaci

Uvedený model patří do úrovně abstrakce nazývané Analytické modelování a je chápán jako zadání pro další fázi – Modelování designu a následné kódování - Implementaci. Znamená to, že model ve fázi designu nemusí „vypadat úplně shodně“ jako analytický model zde navržený (např. přesně takto s třídami v OOP apod.), dokonce se může hodně lišit, ale v každém případě by měl splňovat toto zadání.

Jinak bude vypadat design pro strukturální analýzu, kdy zde uvedené analytické třídy budou mapovány na funkce a tabulky, jinak bude vypadat návrh v objektové databázi anebo v hybridním systému (OOP + RDB). V každém případě však každý design bude splňovat zde uvedené analytické konstrukce.

Rád bych zde uvedl své osobní (tj. nezávazné) představy o možném designu v prostředí kombinace silný jazyk OOP plus silná relační databáze (hybridní systém).

1. Datový model může být navržen pomocí optimalizačního postupu „sloučení entit“ ve stromu GEN-SPEC. Nakonec všechny třídy budou mít pouze jednu tabulku. V této jediné tabulce jsou všechny údaje pro všechny prvky, jak „hlavičky číselníků“, tak jejich dětí všech tří typů.
2. Pro vazbu mezi daty hlavičky číselníku a jejich dětí lze zvolit jako klíč přímo kódy nebo systémové ID. První varianta vede k přehledný datovým strukturám (je možné je prohlížet a interpretovat přímo v tabulce), ale vede ke složitější podmínce výběru prvku díky složenému klíči. Druhá varianta při vazbě přes ID vede k nepřehlednější datové struktuře (hůř se prohlíží přímo v tabulce), ale jednodušším konstrukcím výběru (někdy stačí pouze jedno ID).

Uvedeme obě varianty.

Možný návrh tabulky - cizí klíč je přes ID:

TAB_CISELNIK:

ID	ID_typ	kod	text	zkratka	ID_parent	ID_Item1	ID_Item2	ID_Item3

Pro prvky typu Single-Item jsou relevantní pole ID, ID_typ, kod, text, zkratka a ID_Parent (ostatní irelevantní).

Pro řádek hlavičky číselníku jsou relevantní tytéž pole, ale ID_parent = null.

Pro prvky typu Dvoj-Item jsou relevantní pole ID, ID_typ, ID_parent, ID_Item1, ID_Item2 (ostatní irelevantní) a pro prvky typu Troj-Item jsou relevantní pole ID, ID_typ, ID_parent, ID_Item1, ID_Item2 a ID_Item3 (ostatní irelevantní).

Ve variantě se složenými klíči je namísto pole ID_parent pole kod (kód číselníku). Pokud je kod_parent roven nule, jedná se o hlavičku číselníku. Odkaz z dítěte na hlavičku je dán podmínkou: „Moje hlavička = řádek, který má stejný kod jako můj kod_parent a má kod_parent roven nule“.

Navíc u varianty odkaz přes kód jsou prvky ID_Item1, ID_Item2 a ID_Item3 nahrazeny dvojicemi klíčů „kód číselníku plus kód dítěte z tohoto číselníku“.