ČNB ČESKÁ NÁRODNÍ BANKA

Věcné zadání projektu SDAT

A - Obecné požadavky

Červen 2015

Obsah

1 Úvod

Účelem dokumentu je definovat obecné a společné funkcionality systému SDAT. Tyto funckionality se tak prolínají celým systémem a všemi jeho tzv. vrcholovými oblastmi. Dokumenty, které popisují jednotlivé vrcholové oblasti se tak zaměřují na popis specifických funckionalit systému, které předpokládají existenci společných pravidel a funckionalit. Tato společná pravidla a tyto společné funkcionality jsou popsány právě v tomto dokumentu.

Součástí tohoto dokumentu je tak i definice tzv. nefunkčních požadavků, které tak platí pro celý systém.

2 Obecná pravidla a funkcionality

2.1 Provozní prostředí

Pro zabezpečení všech činností systému SDAT souvisejících s jeho provozem, údržbou a rozvojem je nutná existence několika prostředí systému nebo přesněji instancí jednotlivých komponent systému seskupených do těchto prostředí. Pro potřeby zachycení této problematiky jsou v této části dokumentu definovány následující komponenty:

a) komunikační kanály (viz dokument D – Sběr dat, kapitola Komunikační kanály),
b) zpracování Vstupních zpráv (viz dokument D – Sběr dat, kapitola Zpracování vstupní zprávy),
c) úložiště dat (viz dokument D – Sběr dat, kapitola Objekt Hodnota údaje),
d) výběr dat (viz dokument E – Výběry dat),
e) metapopis (viz dokument B – Metapopis),
f) Registr osob (viz dokument C – Vykazovací povinnosti a Registr osob),
g) Vykazovací povinnosti (viz dokument C – Vykazovací povinnosti a Registr osob).

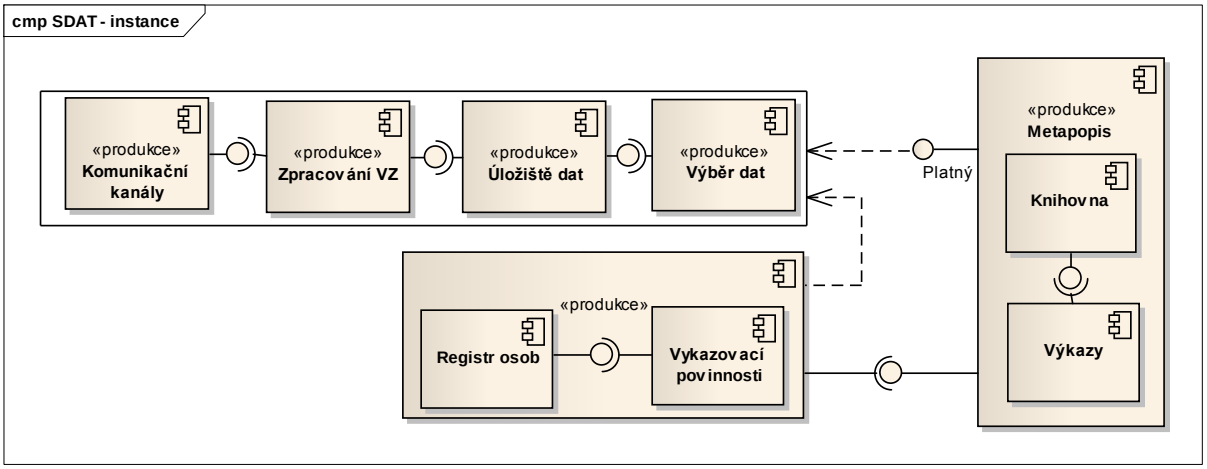
V následujících kapitolách jsou vyjmenována a charakterizována jednotlivá provozní prostředí systému SDAT.

2.1.1 Produkční prostředí

V produkčním prostředí probíhá rutinní provoz systému. Procesy příjmu a zpracování dat probíhají pouze vůči instancím objektů metapopisu ve stavu Platný.

V tomto prostředí jsou provozovány otestované a akceptované verze komponent. Produkční prostředí je dostupné odkudkoli z prostředí Internetu a současně přímo ze systémového prostředí ČNB.

Produkční prostředí bude implementováno v ČNB do geografického clusteru (tj. provoz SDAT ve dvou různých geografických lokalitách).



Obrázek 1 - Schéma produkčního prostředí

2.1.2 Testovací prostředí

Testovací prostředí slouží k testování jednotlivých fází přípravy metadat a dat (Hodnot údajů). Na rozdíl od produkčního prostředí, na testovacím prostředí je možno realizovat proces příjmu a zpracování dat vůči instancím objektů metapopisu ve všech stavech (ne jenom ve stavu Platný, jako je tomu na produkčním prostředí).
Přístup do testovacího prostředí mají jak uživatelé ČNB, tak Osoby. Typickými případy testování jsou následující situace:

a) uživatel v určité fázi projektování potřebuje otestovat navržené kontroly a struktury jednoho nebo více výkazů,
b) v rámci projektování nového výkazu je oslovena Osoba se žádostí o pilotní sestavení výkazu,
c) Osoba testuje sestavení a předání nového výkazu ještě před jeho platností,
d) Osoba, které vznikla nově povinnost vykazovat již platný výkaz, testuje jeho sestavení a předání,
e) Osoba mění komunikační kanál, kterým předává Vstupní zprávy. Např. namísto webové aplikace testuje použití webových služeb včetně zpracování jejich zpětné odezvy,
f) uživatel v ČNB sestavením a zasláním výkazu testuje problematické chování systému, konkrétní kontroly apod.

Sestavit data Výkazu prostředky systému (webové aplikace) a zaslat Vstupní zprávu do testovacího prostředí je možné pokud jsou objekty metapopisu ve stavech Projektovaný, Schválený a Platný. Stav Projektovaný je však nutné před testováním v testovacím prostředí zafixovat mezistavem „Projektovaný, probíhá testování “. Existence tohoto stavu u Výkazu zamezuje přepsání používaných sdílených objektů Knihovny. V případě, že by došlo k potřebě testovat další Výkaz, který používá stejné instance komponenty Knihovna, jako Výkaz, který má stav Projektovaný, probíhá testování, systém neumožňuje import takového Výkazu do testovacího prostředí. Tímto opatřením se má dosáhnout toho, aby se neměnila definice testovaného Výkazu. Jediná možnost, jak dosáhnout importu nového Výkazu používajícího sdílené instance objektů Knihovny je zrušení stavu Projektovaný, probíhá testování.

Přesun instancí objektů z produkčního do testovacího prostředí komponenty Metapopis probíhá v závislosti na jejich stavu:

1) **instance objektů ve stavu Platný:** v rámci procesu synchronizace obsahu testovacího prostředí:
    a. jsou z testovacího prostředí odstraněny totožné instance objektů ve stavu Schválený. Veškerá data přijata během testování výkazu ve stavu Schválený jsou smazána,
    b. bezprostředně poté jsou přesunuty do testovacího prostředí instance objektů ve stavu Platný,
2) **instance objektů ve stavu Schválený:** v rámci procesu synchronizace obsahu testovacího prostředí:
    a. jsou z testovacího prostředí odstraněny totožné instance ve stavu Projektovaný nebo Projektovaný, probíhá testování,
    b. bezprostředně poté jsou přesunuty do testovacího prostředí instance objektů ve stavu Schválený,
3) **instance objektů ve stavu Projektovaný:** na základě ad-hoc na pokynu uživatele:
    a. jsou z testovacího prostředí odstraněny totožné instance objektů ve stavu Projektovaný,
    b. v případě, že se totožné instance objektů nacházejí ve stavu „Projektovaný, probíhá testování“ a zároveň se importované instance liší od existujících instancí (například je importovaná nová Položka číselníku, která zatím na testu není, nebo naopak importovaný Číselník neobsahuje nějakou Položku číselníku, která na testu je), systém tuto skutečnost ohlásí a nabídne uživateli:
        i. zrušení přesunu, tj. neprovede přesun pro žádnou z instancí objektů tak, aby nenarušil právě probíhající testování. Uživatel má možnost vyřešit konflikt tak, že
            1. okamžitě převede konfliktní instance objektů v testovacím prostředí do stavu Projektovaný, aby je mohl systém odstranit při opakování akce přesunu, nebo
            2. nejdříve dokončí testování předchozí podoby instancí objektů a následně převede instance objektů v testovacím prostředí do stavu Projektovaný,
        ii. dokončení přesunu s vynuceným přepisem instancí objektů ve stavu Projektovaný, probíhá testování,
    c. v případě, že kontrola v bodě b. proběhne bez konfliktu, jsou vybrané instance objektů přesunuty do testovacího prostředí a současně jim systém nastaví stav Projektovaný, probíhá testování.

Synchronizace obsahu produkčního a testovacího prostředí pro instance objektů ve stavu Platný a Schválený probíhá procesem, který je naplánován na okrajové (noční hodiny). Testovací prostředí po dobu běhu tohoto procesu hlásí provozní odstávku, během které není možné provádět testování. Synchronizaci nelze navázat na procesy schválení a zplatnění instancí objektů, protože riziko porušení integrity systému je v takovém případě vysoké (testujícímu se mohou změnit podmínky v průběhu testování; předpokládáme, že v nočních hodinách k testování nebude docházet nebo k němu bude docházet v minoritním rozsahu).

Pro testování instancí objektů ve stavech Projektovaný a Schválený platí, že provedené testy vlivem následných změn obsahu těchto instancí na produkčním prostředí (zde pokračuje

proces projektování) nebudou splňovat integritní pravidla systému a např. uložené Hodnoty údajů jsou tak nedostupné. V rámci údržby testovacího prostředí tak mohou být odstraněny.

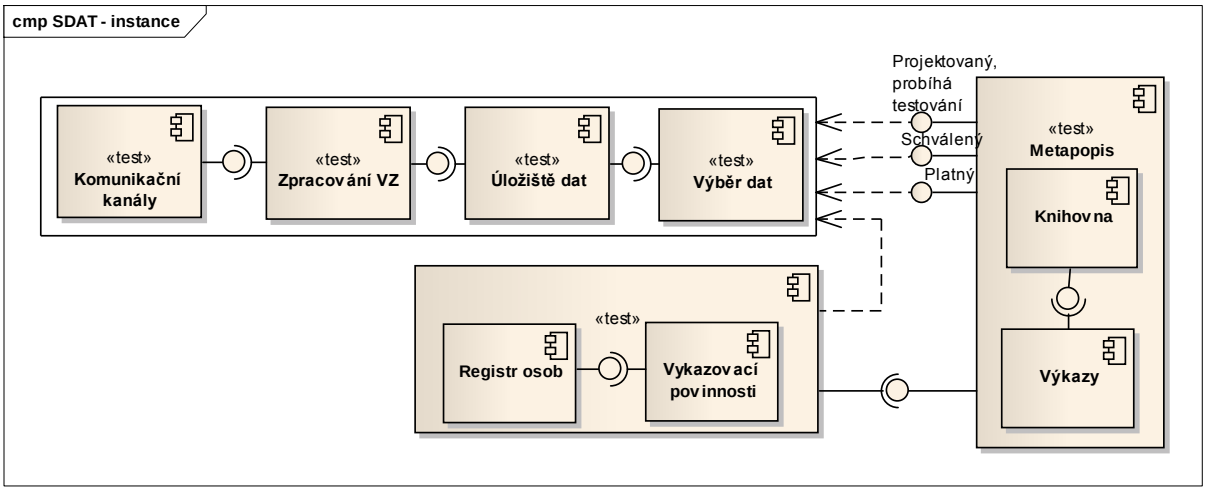
Odlišností testovacího prostředí od produkčního je možnost vytvořit Výskyt výkazu na stav Projektovaný, Probíhá testování a Schválený. Výskyt výkazu si na testovacím prostředí vytváří Osoba (v případě, že je to ona, kdo má potřebu provádět testování). Tyto Výskyty výkazu nejsou tedy automaticky generovány systémem, a to platí i pro stav Platný. Výjimečně však může být Výskyt výkazu na testovacím prostředí připraven ze strany ČNB (zejména tehdy, pokud ČNB žádá nějakou z Osob o součinnost při testování; toto se bude dít nejčastěji u nově projektovaných výkazů).

Testovací prostředí odpovídá z pohledu verzí programového vybavení komponent produkčnímu prostředí.

Není prováděna synchronizace obsahu testovacích a produkčních instancí komponent. Výjimku představuje výše popsaná synchronizace a plnění komponenty Metapopis a komponenta Registr osob, která je automaticky aktualizována v určitém časovém okamžiku např. v nočních hodinách. Nedochází však k plné synchronizaci (to znamená, že na testovací prostředí jsou pouze přidávány nové Osoby, které vznikly v produkci, ale nejsou odstraňovány Osoby, které jsou na testovacím prostředí a nejsou na produkčním), protože v testovací instanci Registru osob mohou být zavedeny Osoby pouze pro potřeby testování, např. pro SW firmy.

Testovací prostředí je dostupné odkudkoli z prostředí Internetu a současně přímo ze systémového prostředí ČNB.

Testovací prostředí bude implementováno v ČNB do geografického clusteru (tj. provoz SDAT ve dvou různých geografických lokalitách).



Obrázek 2 – Schéma testovacího prostředí

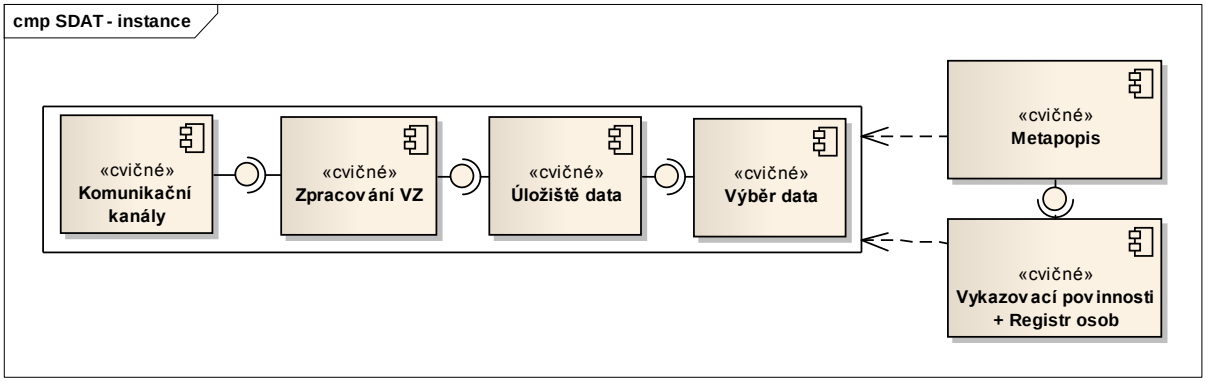
2.1.3 Cvičné prostředí

Toto prostředí slouží jako „cvičné hřiště“ pro interní uživatele (správu systému, metodiky výkazu), pokud je třeba si vyzkoušet nový postup při návrhu výkazu, chování systému v různých situacích, testování kontrol apod.

Cvičné prostředí odpovídá z pohledu verzí programového vybavení komponent produkčnímu prostředí.

Je prováděna synchronizace obsahu cvičných a produkčních instancí komponent, a to na ad-hoc bázi v závislosti na tom, jaké aktivity na prostředí aktuálně probíhají.

Toto prostředí je dostupné pouze v rámci systémového prostředí ČNB.



Obrázek 3 - Schéma cvičného prostředí

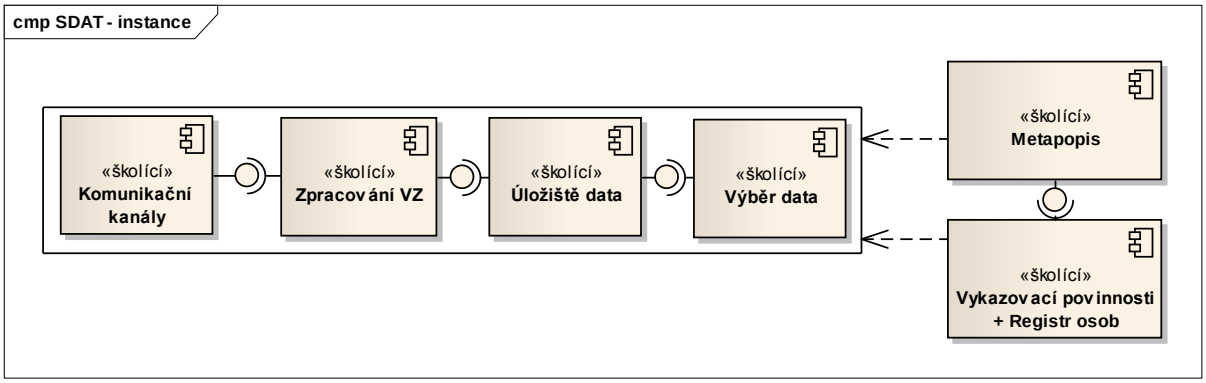
2.1.4 Školicí prostředí

Na školicím prostředí probíhají organizovaná školení uživatelů systému, zejména metodiků výkazů.

Školicí prostředí odpovídá z pohledu verzí programového vybavení komponent produkčnímu prostředí.

Je prováděna synchronizace obsahu školicích a produkčních instancí komponent, a to na ad-hoc bázi v závislosti na tom, jaké aktivity na prostředí aktuálně probíhají.

Toto prostředí je dostupné pouze v rámci systémového prostředí ČNB. Data na tomto prostředí jsou v rámci procesu synchronizace anonymizovaná (Registr osob, uživatelé).



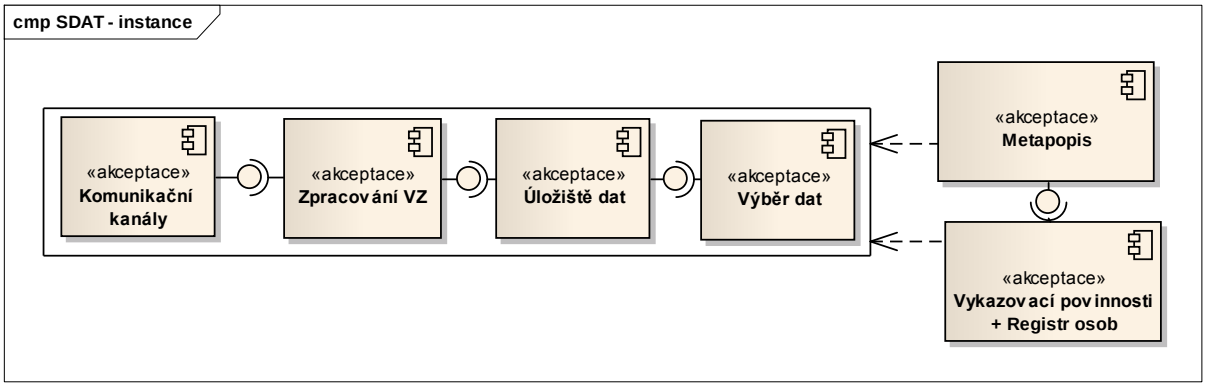
Obrázek 4 - Schéma školicího prostředí

2.1.5 Akceptační prostředí

Akceptační prostředí podporuje proces nasazování nových verzí komponent systému SDAT do prostředí ČNB. Každá změna komponenty provedená dodavatelem v jeho vývojovém prostředí je nejdříve nasazena do akceptačního prostředí pro realizaci akceptačních testů interními uživateli ČNB. Po úspěšném provedení testů a akceptaci jejich výsledků jsou nové verze komponent umístěny do ostatních prostředí (tzv. deployement).

Pokud charakter SW změn dovolí, je prováděna synchronizace obsahu z produkčních do akceptačních instancí komponent, a to na ad-hoc bázi.

Toto prostředí je dostupné pouze v rámci systémového prostředí ČNB.



Obrázek 5 - Schéma akceptačního prostředí

2.1.6 Vývojové prostředí

Vývojové prostředí je prostředí, které používá ČNB během fáze realizace projektu (tímto není dotčeno jakékoli separátní vývojové prostředí na straně dodavatele). Jeho smyslem je, aby existovalo prostředí, kam bude dodavatel nasazovat nové funkcionality tak, jak je bude postupně vyvíjet. Předpokladem je, že k vývojovému prostředí mají přístup klíčoví uživatelé ČNB, kteří participují na vývoji systému s dodavatelem a v podstatě mu tak pomáhají ověřovat jednotlivé funkcionality v okamžiku jejich vzniku. Vývojové prostředí v prostředí

ČNB existuje pouze po dobu realizace projektu a zaniká v okamžiku ukončení projektu, kdy jsou všechny funkcionality dokončeny.

Toto prostředí je dostupné pouze v rámci systémového prostředí ČNB.

2.2 Sledování historie instancí objektů

Aby bylo možno splnit základní princip pro uchovávání dat v systému SDAT, tj. že data se při jejich změně v databázi ve většině případů nepřepisují (existují případy, kdy je přepis hodnoty údaje povolen, viz další text), je zaveden systém sledování historie instancí jednotlivých objektů. Různé objekty z podstaty své existence vyžadují různé přístupy sledování historie. Systém SDAT rozlišuje tyto základní přístupy ke sledování historie instancí objektů:

a) Bez sledování historie,
b) Bez sledování historie – stavy,
c) Sledování historie – časová platnost,
d) Sledování historie – časová platnost + stavy,
e) Sledování historie – časová platnost na každém atributu (tzv. dynamické atributy).

Standardně se předpokládá, že každý objekt v rámci SDAT je z hlediska sledování historie změn řízen přístupem „Bez sledování historie“. V případě, že nějaký objekt má podléhat sledování historie, bude u popisu objektu uvedeno, jakým způsobem se změny u tohoto objektu budou sledovat.

2.2.1 Číslo verze a varianty instance objektu

V případě, že je třeba pro některý z objektů sledovat historii jeho instancí (tedy přístupy c) až e) uvedené výše), jsou tyto instance vždy vymezeny **časovou platností** (atributy „platnost_od“ a „platnost_do“) a navíc je ke každé takové instanci objektu uvedeno **číslo verze a varianty instance objektu**.

Číslo verze a varianty instance objektu nemá v systému jiný účel než lidsky čitelnou identifikaci určitého časového úseku, kdy platila nějaká konkrétní instance objektu. Hlavním identifikátorem platnosti instance objektu tak nadále zůstává vymezení časové platnosti dané instance pomocí atributů „platnost_od“ a „platnost_do“, číslo verze a varianty tento vymezený časový úsek jednoznačně a lidsky čitelně identifikuje. V případě, že uživatel zná číslo verze a varianty konkrétní instance objektu, systém je schopen jednoznačně odpovědět na otázku „Od kdy a do kdy platila tato instance“.

Číslo verze a varianty instance objektu je atribut typu VARCHAR, který se skládá z čísla verze a čísla varianty ve formátu VVV.vvv (V = číslo verze, v = číslo varianty).

Číslo verze a varianty instance objektu je jedinečné pro každý jeden časový okamžik a sledovaný objekt. Každá instance objektu (u objektu, jehož instance podléhají sledování historie), začíná označením 001.000 (Verze = 1, Varianta = 0 + vodící nuly kvůli správnému řazení).

V případě, že dojde k vytvoření nové verze nebo varianty instance nějakého objektu a objekt podléhá sledování historie podle písmen c) až e), vždy dojde k vytvoření nové instance daného objektu a (z hlediska správné identifikace časové platnosti) k:

- vytvoření nové dvojice časových bodů, které vymezují časový úsek, po který daná instance platí (atributy „platnost_od“ a „platnost_do“),
- vytvoření nového čísla verze nebo varianty instance objektu. Pravidla popisující přesný algoritmus vytváření nového čísla verze nebo varianty jsou uvedena v dokumentu B – Metapopis, kapitola Vazby mezi jednotlivými objekty.

2.2.1.1 Pravidla pro vytváření nové verze nebo varianty

Pro vytvoření nové verze a varianty platí následující pravidla:

1. nová verze/varianta se vytváří:
    a. z poslední verze/varianty, která je ve stavu Platný, pokud za ní neexistuje žádná další verze/varianta ve stavu Schválený,
    b. z poslední verze/varianty ve stavu Schválený,
    c. z jakékoliv verze/varianty ve stavu Projektovaný,
2. nová verze/varianta bezprostředně navazuje na verzi/variantu, ze které byla vytvořena (tedy v okamžiku založení neexistuje mezi nimi žádná jiná verze/varianta),
3. pokud je nová verze/varianta zařazena před již existující verzi/variantu, systém zajistí přečíslování čísla verze/varianty tak, aby číslo verze/varianty odráželo reálné pořadí verzí/variant.

2.2.2 Objektový model pro sledování historie instancí objektů

Systém SDAT je složen z velkého počtu objektů, které jsou mezi sebou navzájem propojeny vazbami a vytvářejí tak kompletní objektový model systému. Jak bylo uvedeno výše, z hlediska naplnění byznys podmínek existují různé přístupy pro sledování historie, od situace, kdy se historie nesleduje, až po situaci, kdy se sleduje historie každé jedné instance objektu včetně sledování stavů, kterými daná instance během své časové platnosti prošla.

Pokud bychom měli ke každému objektu, který vyžaduje nějaké sledování historie, kreslit do objektového modelu objekty, které slouží pro toto sledování historie (nebo historie a stavů), stal by se objekt velmi nepřehledným.

Místo toho je definován model pro sledování historie (historie a stavů) na obecné úrovni s tím, že reálný model bude odvozen ze znalosti základního modelu a informace, jaký přístup pro sledování historie platí pro ten který objekt.

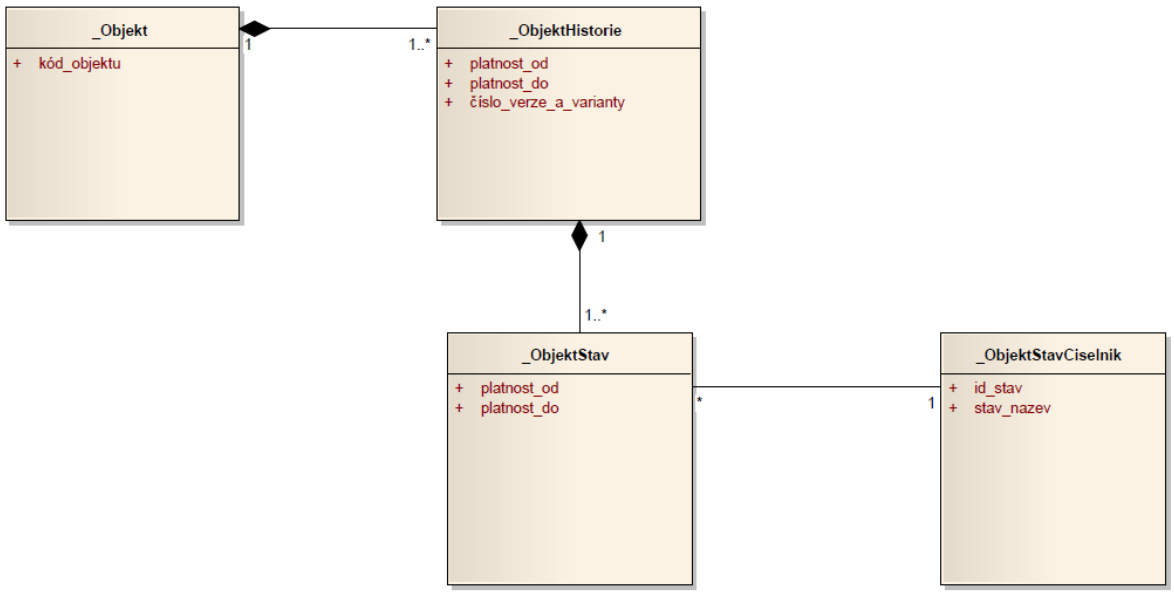
Objekty použité v obecném modelu pro sledování historie (historie a stavů) začínají prefixem „_“, aby bylo patrné, že se jedná o obecné objekty, které v praxi musejí být nahrazeny reálnými objekty toho kterého objektu, ke kterému se váží.

Objekt _Objekt tak zastupuje jakýkoli reálný objekt systému (místo objektu _Objekt si tak lze představit například objekt Výkaz, Vykazovací povinnost, Doména osob, resp. jakýkoli další objekt zachycený v objektovém modelu). Dále platí:

- objekt _ObjektHistorie zastupuje jakýkoli reálný objekt, který uchovává kompletní historii instance jakéhokoli objektu. Pokud bude třeba sledovat historii u objektu Výkaz, pak by měl existovat objekt VýkazHistorie, který bude sloužit pro evidenci všech historických verzí/variant jedné instance objektu Výkaz,
- objekt _ObjektStav zastupuje jakýkoli reálný objekt, který uchovává kompletní historii stavů, kterými prošla vybraná instance objektu _Objekt,
- objekt _ObjektStavCiselnik je objekt, který uchovává seznam všech možných stavů, kterými může projít během svého života instance třídy _Objekt. Pokud bude třeba sledovat stavy u objektu Výkaz, pak by měl existovat objekt VýkazStavCiselnik, kde budou uvedeny všechny stavy, kterými může projít během svého života instance třídy Výkaz.

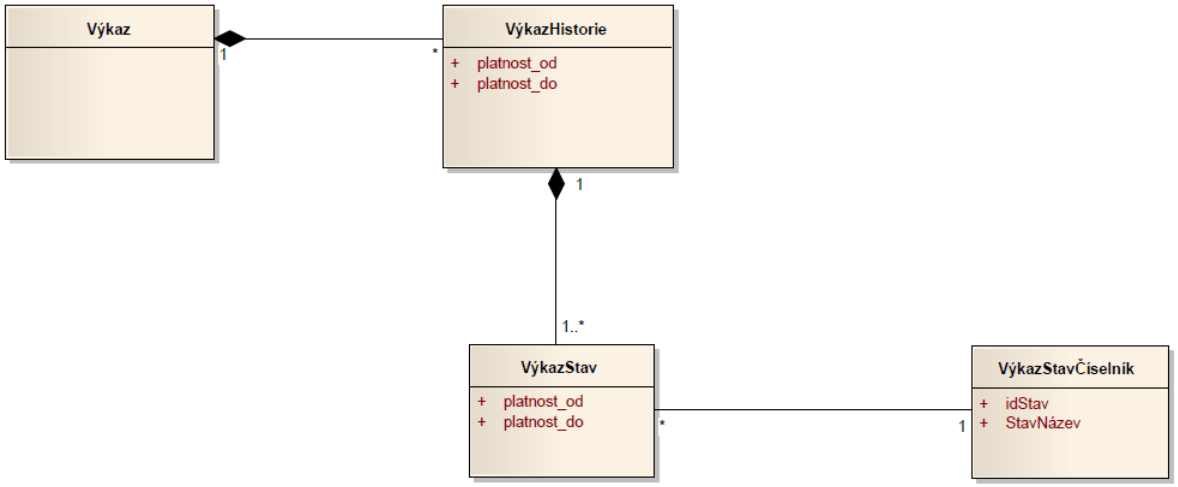
Pro názornost uvádíme, jak se výše uvedený obecný model zhmotní do podoby reálných objektů systému na příkladu objektu Výkaz:

- u objektu Výkaz je definována byznys podmínka, že systém musí evidovat historii všech instancí objektu včetně sledování stavů každé instance. Jedná se tak o přístup „Sledování historie – časová platnost + stavy“,
- v objektovém modelu pro metapopis je zakreslen pouze objekt Výkaz,
- pokud spojíme obě výše uvedené informace dohromady, vyjde nám následující reálný model popisující objekt Výkaz:
    - objekt Výkaz (v obecném modelu zastoupen objektem _Objekt),
    - objekt VýkazHistorie (v obecném modelu zastoupen objektem _ObjektHistorie),
    - objekt VýkazStav (v obecném modelu zastoupen objektem _ObjektStav),
    - objekt VýkazStavČíselník (v obecném modelu zastoupen objektem _ObjektStavČíselník),
- obecný model pro přístup „Sledování historie – časová platnost + stavy“ je patrný z obrázku 6,



Obrázek 6 - Obecný model pro sledování historie - časová platnost + stavy

- z tohoto obecného modelu bude v případě objektu Výkaz odvozen reálný objektový model (viz Obrázek 7 - Odvozený model pro sledování historie a stavů objekt Výkaz),



Obrázek 7 - Odvozený model pro sledování historie a stavů objekt Výkaz

2.2.3 Přístup „Bez sledování historie“

V rámci objektu není sledováno ani číslo verze a varianty instance, ani časová platnost instance. Příkladem takového objektu je například objekt Skupina osob**.** Pokud není uvedeno jinak, pak je každý objekt řízen tímto přístupem.

2.2.4 Přístup „Bez sledování historie – stavy“

V rámci objektu není sledováno ani číslo verze a varianty instance objektu, ani časová platnost instance objektu. Nicméně instance objektu prochází stavy. To znamená, že k hlavnímu objektu _Objekt je přímo připojen objekt pro sledování stavů (_ObjektStav). V rámci objektu _ObjektStav jsou sledovány všechny stavy, kterými instance objektu prošly. Platí, že aby mohla vzniknout instance objektu _Objekt, musí vzniknout alespoň jedna související instance objektu _ObjektStav. Důležitý je fakt, že instance objektu _Objekt existuje v čase právě jednou, nijak se nemění a proto není důvod sledovat její historii. Příkladem objektu, který podléhá pouze sledování stavů, je objekt Vstupní zpráva (viz dokument D – Sběr dat) nebo Vydání výskytu výkazu. Obecně je tento přístup zachycuje obrázek 8.

Obrázek 8 - Přístup „Bez sledování historie – stavy“

2.2.5 Přístup „Sledování historie – časová platnost“

K hlavnímu objektu _Objekt je připojen podřízený objekt (_ObjektHistorie), který obsahuje všechny běžné atributy objektu _Objekt (s výjimkou v čase neměnných atributů typu kód) a v rámci kterého je sledována kompletní historie změn pomocí atributů „platnost_od“ a „platnost_do“ a číslo verze a varianty instance.

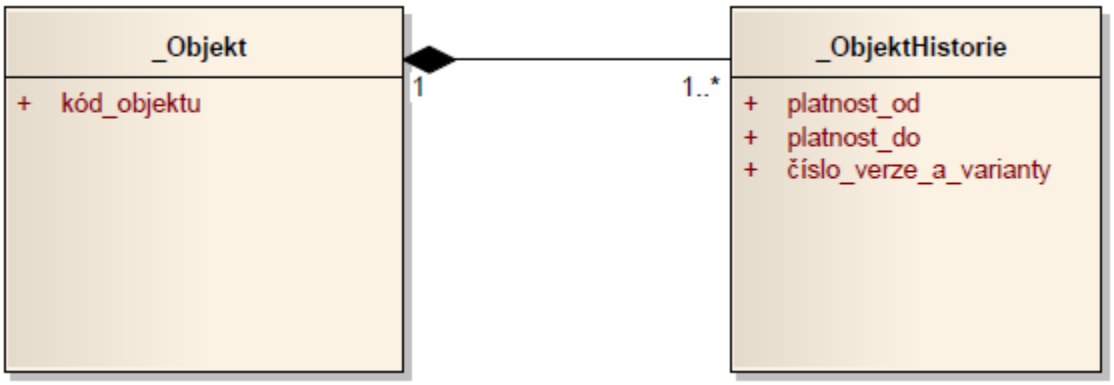
Žádná z instancí souvisejících objektů se neváže na hlavní objekt (_Objekt), ale na objekt _ObjektHistorie.

V případě, že se mění obsah (tj. počet souvisejících instancí podřízených objektů nebo jejich atributy) hlavního objektu _Objekt, nebo jsou měněny atributy hlavního objektu, vzniká nová instance objektu _ObjektHistorie (mění se číslo verze/varianty) a mění se časová platnost instance (atributy „platnost_od“ a „platnost_do“).

Instance třídy _ObjektHistorie se nesmí časově překrývat a nesmí mezi nimi vznikat „časové díry“. To znamená, že záznamy na sebe musejí plynule navazovat, a nesmí existovat žádný časový okamžik, pro který by platilo, že existuje buď žádná nebo dvě a více instancí tohoto objektu.

Časová platnost je zachycována pomocí data a času s přesností na sekundy.

Tento přístup lze použít všude tam, kde stačí řídit platnost objektu pomocí vymezení časové platnosti (platnost_od a platnost_do) a kde není nutné mít v jeden okamžik platné dvě instance jednoho objektu (např. jednu ve Stavu Platný a jednu ve stavu Projektovaný). Typickým příkladem objektu, kde lze použít tento přístup, je objekt Vykazovací povinnost. Obecně tento přístup zachycuje obrázek 9.



Obrázek 9 - Přístup „Sledování historie – časová platnost“

2.2.6 Přístup „Sledování historie – časová platnost + stavy“

Hlavní instance objektu je bez časové platnosti stejně jako u výše uvedeného přístupu. K hlavnímu objektu je připojen podřízený objekt (_ObjektHistorie), v rámci kterého je sledována kompletní historie změn pomocí atributů „platnost_od“ a „platnost_do“ a číslo

verze a varianty instance. Žádná ze souvisejících instancí se neváže na hlavní objekt, ale na objekt _ObjektHistorie stejně jako u předcházejícího přístupu.

Oproti předcházejícímu přístupu je zde navíc k objektu _ObjektHistorie připojen podřízený objekt _ObjektStav a k němu související _ObjektStavCiselnik, který eviduje stavy, kterými může procházet instance daného objektu. Tyto stavy mohou být pro každý objekt různé. V oblasti metapopisu např. platí, že každý historický záznam musí projít základními stavy (Projektovaný, Schválený, Platný), u dalších objektů je seznam možných stavů uveden u popisu každého objektu. Aby bylo možné mít pro každý objekt jiný seznam stavů, kterými musí jeho instance projít, existuje objekt s číselníkovými položkami pro každý objekt samostatně.

Následující text platí pro objekty, které procházejí stavy Projektovaný, Schválený a Platný.

Zcela zásadní rozdíl oproti výše uvedenému přístupu je ten, že instance objektu _ObjektHistorie **se smí časově překrývat** (pro jeden časový okamžik smí existovat více instancí tohoto objektu), ale musí platit, **že právě jedna instance tohoto objektu je pro konkrétní časový okamžik ve stavu Platný.**

Mezi instancemi ve stavu Platný nesmějí vznikat „časové díry“. To znamená, že záznamy na sebe musejí plynule navazovat. Nesmí existovat žádný časový okamžik, pro který by platilo, že existuje buď žádná, nebo dvě a více instancí tohoto objektu (ve stavu Platný). Časová platnost je zachycována pomocí data a času s přesností na sekundy.

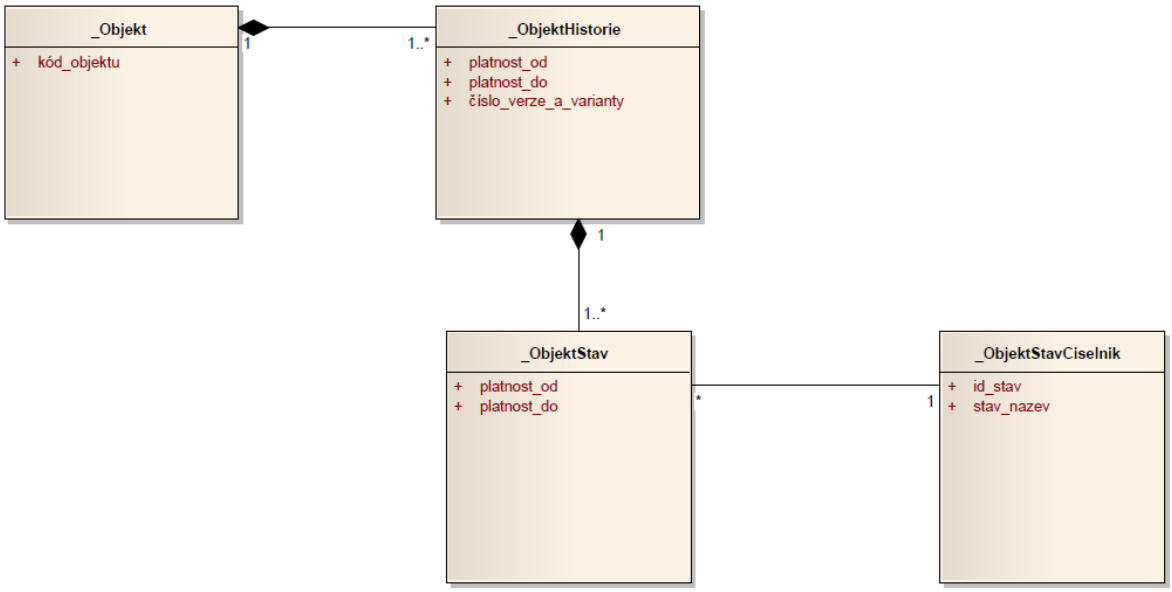
Při manipulaci s časovou a stavovou platností musejí platit tato pravidla (platí pro objekty metapopisu a obecně pro všechny objekty, které procházejí stavy Projektovaný, Schválený a Platný):

- pro vyhodnocení toho, zda instance nějakého objektu je platná, je **primární stav** dané instance, který zachycuje konkrétní životní fázi z pohledu procesu její tvorby. **Sekundární údaj,** nutný pro správné určení rozsahu platnosti, **je časový interval platnosti**, který lze popsat jako dobu účinnosti dané instance. Počáteční datum intervalu (platnost_od) však pouze určuje okamžik, od kterého platnost (účinnost) nastane. Korektní je tak situace, kdy instance objektu má nastaven atribut platnost_od na 1. 1. 2015, ale stav Platný dosáhne tato instance již 1. 12. 2014. To znamená, že dne 1. 12. 2014 vznikne instance objektu _ObjektStav, tzn., že instance objektu je dostupná Osobám a mohou ji číst, nemohou ji ale zatím použít k vykazování dat. K vykazování dat může být tato instance objektu použita nejdříve v okamžiku, kdy reálně nastane datum 1. 1. 2015. Zároveň je korektní i situace, kdy platnost_do je menší než aktuální datum a stav instance objektu má hodnotu Platný. Ani v tomto případě není již instanci možno použít k vykazování, protože uplynul časový rozsah platnosti.

  V případě, že nastane situace, kdy instance nějakého objektu bude v okamžiku, od kdy časově platí, ve stavu jiném než Platný (tedy Projektovaný nebo Schválený; není tak splněna primární podmínka pro platnou instanci), pak tato **instance není Platná a nemůže být použita pro vykazování** (lze zaslat pouze data, jejichž stav ke dni je v intervalu platnosti instance),

- instance každého objektu při svém založení automaticky získává stav Projektovaný. V takovém stavu není instance objektu přístupná externím uživatelům, a to ani tehdy, pokud by platila z hlediska časového intervalu,
- stavem, který následuje po stavu Projektovaný, je stav Schválený. V případě, že je instance ve stavu Schválený, lze ji přesunout zpět do stavu Projektovaný,
- stav Platný je generován uživatelem v rámci tzv. procesu zplatnění. Tento proces se musí odehrát před tím, než fyzicky nastane časový okamžik platnosti. V případě, že se tak nestane a instance objektu nebude přesunuta do stavu Platný před tím, než nastane časový okamžik platnosti, pak daná instance není Platná z pohledu možnosti vykazování. Systém bude uživatele upozorňovat na blížící se začátek platnosti všech instancí, které nejsou ve stavu Platný a hrozí u nich riziko, že časově nastane jejich platnost, ale stavově platné nebudou,
- V okamžiku přechodu instance do stavu Platný je časově předcházející instance, která je ve stavu Platný ukončena k datu D-1 (o jeden den menší datum, než je datum platnost_od nové instance objektu ve stavu Platný). Instance objektu ve stavu Platný nelze přesunout do žádného jiného stavu. Jediná možnost, jak změnit platnou instanci objektu, je vytvořit novou verzi dané instance (nová instance bude vytvořena do stavu Projektovaný),
- z hlediska uživatele zachycují výše uvedené stavy životní cyklus tvorby instance objektu:
    - **Projektovaný:** instance objektu je rozpracována, provádějí se na ní změny a práce není dokončena. V tomto stavu instance objektu není obecně zpřístupněna externím uživatelům s výjimkou možnosti exportu části projektovaného metapopisu k připomínkám,
    - **Schválený:** tento stav nastává akcí uživatele a říká, že práce na instanci je dokončena, ale že existuje možnost, že daná instance bude znovu přesunuta do stavu Projektovaný a její vlastnosti budou změněny. V tomto stavu je instance objektu přístupná externím uživatelům, ale nemohou ji použít pro vykazování (mohou ji pouze číst). Neexistuje definitivní jistota, že daná instance objektu už nedozná žádných změn,
    - **Platný:** tento stav nastává akcí uživatele a říká, že práce na instanci je definitivně dokončena a její vlastnosti už nemohou být změněny. Instance objektu v nezměněné podobě začne platit, tzn. může být použita pro vykazování, od okamžiku, kdy fyzicky nastane čas platnosti instance (datum a čas uvedený v atributu platnost_od).

Obecný model pro tento typ sledování historie je znázorněn v obrázku 10.



Obrázek 10 - Obecný model pro sledování historie se stavy

Následující příklad demonstruje, jak se budou chovat instance jednotlivých objektů pro objekt Výkaz, během životního cyklu jednoho Výkazu, od jeho založení až po jeho zrušení. Příklad je sice popsán pro konkrétní objekt (objekt Výkaz), nicméně lze z něj odvodit pravidla pro chování všech ostatních objektů systému, které se mají z hlediska sledování historie a stavů chovat stejně.

V případě objektu Výkaz tak vzniknou tyto reálné objekty:

- Výkaz (_Objekt)
- VýkazHistorie (_ObjektHistorie)
- VýkazStav (_ObjektStav)
- VýkazStavČíselník (_ObjektStavČíselník), číselník bude obsahovat tyto hodnoty
  - Projektovaný
  - Schválený
  - Platný

Životní cyklus výkazu může být následující:

1. **Dne 1. 1. 2013 je započato projektování zcela nového Výkazu, označeného kódem "V001" a názvem "V1"; cílem je, aby výkaz platil (a bylo na něj možno vykazovat) od 1. 1. 2014**

   Vytvoření nového výkazu znamená z hlediska dopadu na instance výše uvedených objektů:

   i. musí vzniknout instance objektu Výkaz, kde se uvedou v čase neměnné atributy Výkazu (například Kód výkazu). Tento objekt nemá žádné časové platnosti. V tomto případě se jedná o kód výkazu = „V001“,
   ii. vznik nové instance objektu Výkaz automaticky vyvolává vznik nové verze výkazu (verze 001.000); sledování verzí je předmětem objektu VýkazHistorie - musí tak vzniknout nová instance objektu VýkazHistorie. **Platnost každého**

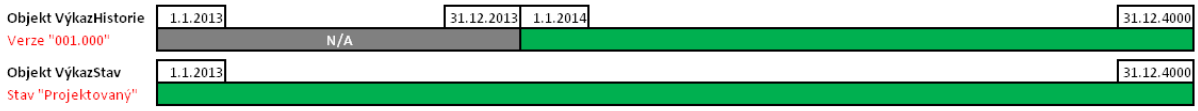
výkazu je sledována v rámci jeho verze, nikoli v rámci hlavní instance objektu Výkaz. Proto instance objektu VýkazHistorie musí vzniknout od 1. 1. 2014 (nikoli od 1. 1. 2013). Tímto datem je řečeno, od kdy bude daná verze platit, pokud ji uživatel označí stavem Platný. Pokud není uvedena hodnota atributu platnost_do, pak se má za to, že bude použito maximální datum,

iii. v rámci instance objektu VýkazHistorie se nadefinují hodnoty všech atributů, které jsou v čase proměnné (například název výkazu apod.),

iv. vznik nové instance objektu VýkazHistorie automaticky vyvolává vznik nové instance objektu VýkazStav, ve stavu Projektovaný. Stav Projektovaný je generován s platností od 1. 1. 2013, ačkoli platnost verze výkazu je nastavena od 1. 1. 2014. Platnost podřízené instance tak může vybočovat z platnosti instance nadřízené. To je možné proto, že tyto platnosti spolu logicky nesouvisí.

Bod 1. se tak projeví v instancích jednotlivých objektů tak, že vzniknou instance objektu Výkaz, VýkazHistiorie a VýkazStav podle obrázku 11 (instance objektu Výkaz neobsahuje časové platnosti, proto není na obrázku zachycena, v reálu však vzniknout musí, protože bez její existence by nemohly vzniknut instance objektu Výkaz, VýkazHistorie a VýkazStav).

Červeně jsou v následujících obrázcích vyznačeny instance objektů, které jsou nějakým způsobem daným krokem zasaženy (vznikají nebo se mění). Černě jsou pak instance, které již existují z předcházejícího kroku, ale nejsou změnou nijak zasaženy.



Obrázek 11 - Vytvoření nového objektu - Dopad na instance objektů

Po vytvoření výkazu podle bodu 1. pak v období od 1. 1. 2013 do 30. 6. 2013 probíhá projektování výkazu. Mění se bloky/datové oblasti/údaje, připravují se parametry/číselníky a kontroly.

2. **Dne 30. 6. 2013 se uživatel rozhodne, že fáze projektování výkazu je u konce, výkaz je dokončený a schválí ho. Akce schválení je "okamžitá" (v reálu platí od okamžiku provedení; v našem příkladu pro jednoduchost ilustrace od dalšího dne).**

Schválení výkazu "V1" ve verzi "001.000" znamená:

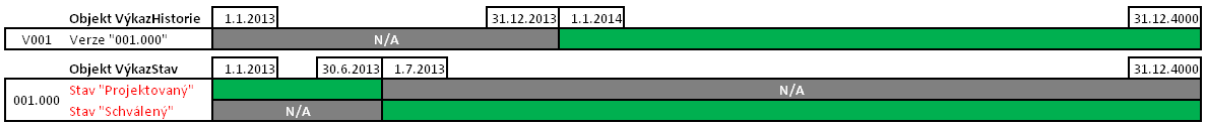
i. instance objektu Výkaz se nemění, výkaz stále existuje,

ii. změna stavu se děje vždy v rámci konkrétní verze/varianty, ale nemění se žádný z atributů objektů VýkazHistorie, takže instance objektu VýkazHistorie zůstane nezměněna,

iii. ke změně musí dojít v rámci objektu VýkazStav, kde musí dojít k ukončení platnosti stavu Projektovaný založení nového stavu Schválený.

Bod 2. se tedy projeví tak, že existující instance objektu VýkazStav (stav Projektovaný) bude změněna (nastaví se hodnota atributu platnost_do na hodnotu aktuálního data) a vznikne nová instance objektu

VýkazStav se stavem Schválený (s hodnotou atributu platnost_od bezprostředně navazující na platnost_do stavu Projektovaný).



Obrázek 12 - Schválení verze výkazu 001.000 - dopad na instance objektů

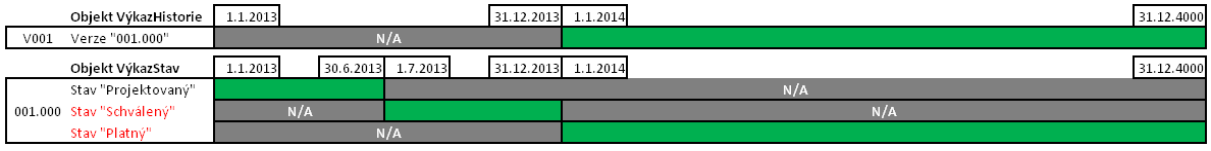
Po schválení verze výkazu pak v období od 1. 7. 2013 do 1. 8. 2013 probíhá testování výkazu.

3. **Dne 15. 8. 2013 se uživatel rozhodne, že výkaz je otestován a rozhodne se ho uvést v platnost (uvedení výkazu v platnost nevyžaduje zadání data platnosti, datum platnosti je atribut objektu** VýkazHistorie**; pokud by bylo potřeba použít jiné datum, je třeba se vrátit do stavu Projektovaný, ve kterém je změna platnosti verze povolena).**

   Uvedení verze výkazu "001.000" v platnost znamená:

   i. instance objektu Výkaz se nemění, výkaz stále existuje,
   ii. změna stavu se děje vždy v rámci konkrétní verze/varianty, ale nemění se žádný z atributů objektů VýkazHistorie, takže instance objektu VýkazHistorie zůstane nezměněna,
   iii. ke změně musí dojít v rámci objektu VýkazStav, kde musí dojít k ukončení platnosti stavu Schválený a založení nového stavu Platný[1].

   Bod 3. se tedy projeví tak, že existující instance objektu VýkazStav (stav Schválený) bude změněna (nastaví se hodnota atributu platnost_do na hodnotu data platnosti_od dané verze výkazu minus 1 sekunda) a vznikne nová instance objektu VýkazHistorie se stavem Platný (s hodnotou atributu platnost_od, která se bude rovnat datu platnosti verze výkazu, v našem případě tedy 1. 1. 2014).



Obrázek 13 - Uvedení verze výkazu 001.000 v platnost - dopad na instance objektů

Bod 3. byl v našem příkladu proveden během srpna 2013 (15. 8. 2013). I když byl Výkaz "V001" uveden v platnost, tak od srpna 2013 ho nelze používat pro vykazování. To bude možno až od data, od kdy platí stav Platný, tedy od 1. 1. 2014. Zároveň to však znamená, že s okamžitou platností již není možné provádět úpravy Výkazu ve verzi 001.000 (verze je již uvedená v platnost, i když do budoucna). **Jakékoli změny, které by bylo nutno provést od srpna 2013 dále, by znamenaly novou verzi výkazu.**

[1] V případě, že by již nějaká verze daného výkazu existovala, pak by bylo nutno ukončit i platnost stavu Platný této verze, ale to není tento případ.

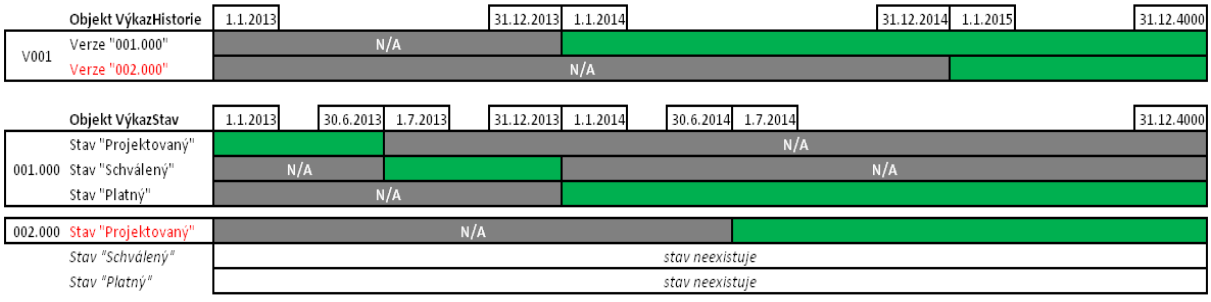
Nyní se po nějakou dobu s výkazem ve verzi 001.000 nic neděje. Jakmile nastane datum, od kdy je daná verze výkazu platná (to nastane 1. 1. 2014), je možno na daný výkaz, verze 001.000 vykazovat (posílat data). Osoby, které mají pro daný výkaz definovanou vykazovací povinnost, vykazují data a probíhá fáze sběru dat. Následně ve 4/2014 vyjdou legislativní úpravy, které znamenají nutnost úpravy Výkazu V1. Do konce 6/2014 probíhá analýza těchto změn a od 7/2014 začne projektování nové verze výkazu.

4. **Dne 1. 7. 2014 se uživatel rozhodne, že je třeba vytvořit novou verzi Výkazu s datem platnosti od 1. 1. 2015. Do 12/2014 je nutno sbírat data s využitím verze 001.000, od 1/2015 se plánuje využití verze 002.00, kterou je potřeba v druhé polovině roku 2014 naprojektovat, schválit a uvést v platnost.**

   Vytvoření nové verze výkazu "002.000" znamená:

   i. instance objektu Výkaz se nemění, výkaz stále existuje,
   ii. vytvoření nové verze Výkazu s číslem verze „002.000“ (je vytvořena nová instance objektu VýkazHistorie). Instance objektu VýkazHistorie pro verzi 001.000 se nijak nemění,
   iii. vznik nové instance objektu VýkazHistorie, verze 002.000 automaticky vyvolává vznik nové instance objektu VýkazStav, ve stavu Projektovaný. Stav Projektovaný je generován s platností od 1. 7. 2014, ačkoli platnost verze výkazu je nastavena od 1. 1. 2015. Platnost podřízené instance tak může vybočovat z platnosti instance nadřízené. To je možné proto, že tyto platnosti spolu logicky nesouvisí.

   Bod 4. se tedy projeví tak, že musí vzniknout nová instance objektu VýkazHistorie. Tato instance reprezentuje novou verzi výkazu „V1“, tj. verzi „002.000“, kde v rámci této instance je nastavena hodnota atributu platnost_od na 1. 1. 2015 (datum, od kdy bude verze platit). Zároveň musí vzniknout související instance objektu VýkazStav. Tato instance však bude mít nastavenou hodnotu atributu platnost_od na aktuální systémové datum (v našem případě na 1. 7. 2013, protože v tento den je pro danou verzi vygenerován stav Projektovaný).



Obrázek 14 - Vytvoření verze 002.000 - dopad na instance objektů

Aktuálně se nacházíme v 7/2014. Existují dvě časově platné verze, ale pouze jedna z nich je platná stavově (verze 001.000). Druhá verze (002.000) je sice platná časově, ale není platná stavově (nachází se ve stavu Projektovaný, nikoli Platný).

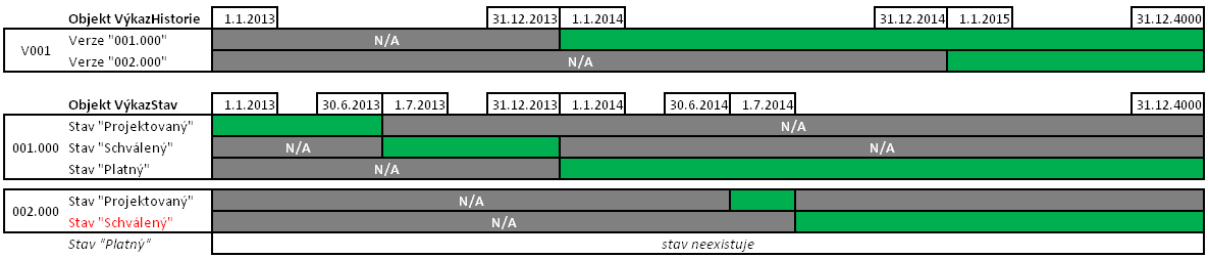
V období od 7/2014 -8/2014 probíhá projektování nové verze Výkazu (002.000). Mění se Bloky/Datové oblasti/Údaje, připravují se Parametry/Číselníky a Kontroly (vše pro verzi 002.000).

5. **Dne 31. 8. 2014 se Uživatel rozhodne, že projektování verze Výkazu 002.000 je dokončené a rozhodne se ji schválit (verze 001.000 je doposud stále platná a jsou na ní vykazována data).**

   Schválení výkazu "V1" ve verzi "002.000" znamená:

   i. instance objektu Výkaz se nemění, výkaz stále existuje,
   ii. změna stavu se děje vždy v rámci konkrétní verze/varianty, taktéž ani instance objektu VýkazHistorie se nemění (ani u jedné verze),
   iii. ke změně musí dojít v rámci objektu VýkazStav, kde musí dojít k ukončení platnosti stavu Projektovaný založení nového stavu Schválený u verze 002.000 (stav verze 001.000 není schválením verze 002.000 nijak dotčen, verze 001.000 je stále platná).

   Bod 5. se tak projeví tak, že existující instance objektu VýkazStav (stav Projektovaný) bude změněna (nastaví se hodnota atributu platnost_do na hodnotu aktuálního data) a vznikne nová instance objektu VýkazStav se stavem Schválený (s hodnotou atributu platnost_od bezprostředně navazující na platnost_do stavu Projektovaný). Toto vše platí pro instance objektu VýkazStav, které mají jako rodiče instanci objektu VýkazHistorie, kde je číslo verze a varianty rovno 002.000.



Obrázek 15 - Schválení verze výkazu 002.000 - Dopad na instance objektů

Aktuálně se nacházíme v 9/2014. Máme dvě časově platné verze, ale pouze jedna z nich je platná stavově (verze 001.000). Druhá verze (002.000) je platná časově, ale není platná stavově (nachází se ve stavu "schválený"). **To umožňuje, aby verze 002.000 byla prezentována Osobám (zatím na ni nelze vykazovat) s tím, že lze očekávat, že od 1. 1. 2015 bude ve stavu Platný.**

V období od 1. 9. 2014 do 30. 10. 2014 probíhá testování výkazu, verze 002.000. Upravují se například kontroly v důsledku chyb nalezených při testování apod.

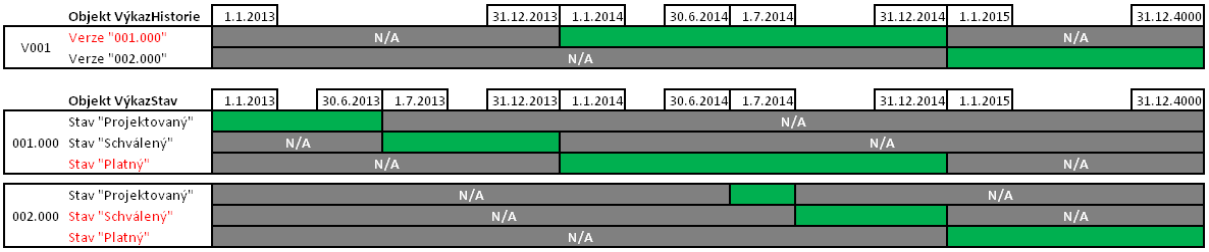
6. **Dne 1. 11. 2014 se Uživatel rozhodne, že výkaz je otestován a rozhodne se ho uvést v platnost (uvedení výkazu v platnost nevyžaduje zadání data platnosti, datum platnosti je atribut objektu** VýkazHistorie**; pokud by bylo potřeba použít jiné datum, je třeba se vrátit do stavu Projektovaný, ve kterém je změna platnosti verze povolena).**

   Uvedení výkazu "V1" ve verzi "002.000" v platnost znamená:

   i. instance objektu Výkaz se nemění, výkaz stále existuje,
   ii. změna stavu se děje vždy v rámci konkrétní verze/varianty, ale nemění se žádný z atributů objektů VýkazHistorie, takže instance objektu VýkazHistorie verze

002.000 zůstane nezměněna. **Musí však dojít ke změně instance objektu VýkazHistorie pro verzi 001.000**. Tím, že uvádíme v platnost verzi 002.000, musí dojít k ukončení platnosti verze předcházející. To znamená, že pro verzi 001.000 se musí nastavit nová hodnota v atributu platnost_do (datum bezprostředně předcházející datu platnosti verze 002.000, tedy 31. 12. 2014).

iii. ke změnám musí dojít v rámci objektu VýkazStav:
   1. pro verzi 001.000 musí dojít k ukončení stavu Platný. Stav Platný verze 001.000 musí být ukončen k 31. 12 2014 (datum bezprostředně předcházející datu platnosti verze 002.000, stejně jako u nadřízené instance objektu),
   2. Pro verzi 002.000 musí dojít k ukončení platnosti stavu Schválený a založení nového stavu Platný.



Obrázek 16 - Uvedení v platnost verze výkazu 002.000 - Dopad na instance objektů

Aktuálně se nacházíme v 11/2014. Máme dvě časově platné verze, **dokonce jsou obě ve stavu Platný, nicméně systém zajišťuje to, že se stav Platný u obou verzí navzájem nepřekrývá**. Do 31. 12. 2014 je ve stavu Platný verze 001.000 a od 1. 1. 2015 je to verze 002.000. To umožňuje do konce roku 2014 sbírat data na verzi 001.000 a od 1. 1. 2015 na verzi 002.000, a zároveň to umožňuje, aby Osoby od 1. 11. 2014 (vygenerování stavu Platný verze 002.000) měly k dispozici verzi, která bude platit od dalšího roku. Jakákoli změna, kterou by bylo nutno provést ve verzi 002.000 po jejím uvedení v platnost, znamená, že je nutno vytvořit novou verzi 003.000.

Od 1. 1. 2015 probíhá sběr dat pomocí verze 002.000. Následně, např. dne 7. 8. 2016 vyjde legislativní změna, která daný výkaz od 1. 1. 2017 ruší (daný výkaz nebude nadále používán pro vykazování).

7. Dne 7. 8. 2016 se uživatel rozhodne, že verze Výkazu není od roku 2017 potřeba a je třeba ukončit její platnost (k 31. 12. 2016)

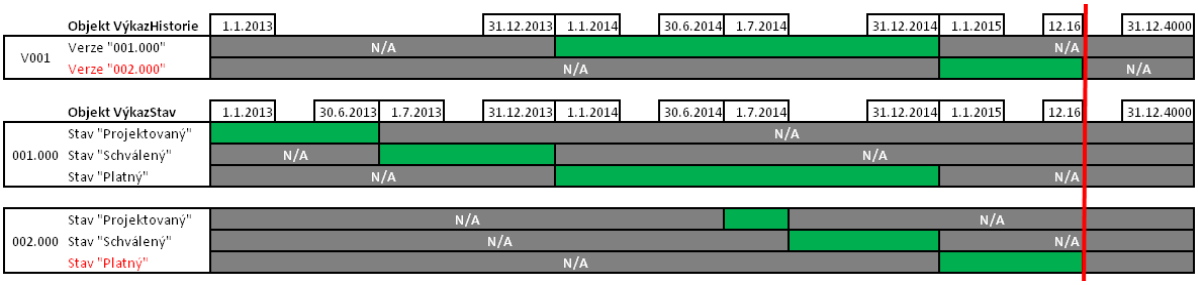
Ukončení Výkazu jako celku znamená:

i. instance objektu Výkaz se nemění; výkaz sice přestává existovat, ale instance objektu nelze smazat, protože by to tím pádem znamenalo odstranění veškerých dat s výkazem souvisejících. Instanci objektu Výkaz nelze ani ukončit, protože tento objekt nemá žádné časové platnosti,

ii. ukončení výkazu znamená ukončení všech aktuálně časově platných verzí (záznamy verze 001.000 a její stavy byly ukončeny k 31. 12. 2014, není tedy třeba

nic měnit); v našem případě je tak třeba ukončit jen verzi 002.000 (v případě, že by byly v daný okamžik rozprojektovány další verze, musely by být ukončeny také),

iii. ukončení všech aktuálně časově platných verzí znamená ukončení aktuálně platného stavu každé verze (v závislosti na tom, v jakém stavu se daná verze nachází). V našem případě to znamená pouze ukončení stavu Platný pro verzi 002.000.



Obrázek 17 - Ukončení výkazu - dopad na instance objektů

Aktuálně se nacházíme v 8/2016. Od 1/2017 je výkaz nepotřebný, ale už v 8/2016 jej ukončuji (s platností od 1. 1. 2017). Všechny související instance Výkazu V1 jsou k 31. 12. 2016 ukončeny - do tohoto data lze Výkaz používat k vykazování, za pozdější data (Stav ke dni) nelze generovat Výskyty Výkazů.

2.2.7 Přístup „Sledování historie – časová platnost na každém atributu“

Zcela speciálním případem sledování historie je situace, kdy je časová platnost sledována na každém jednom atributu daného objektu. Tento přístup však není obecný a je použit pouze ve výjimečných případech. V případě systému SDAT se jedná například o objekt Osoba a Atribut osoby. Tento způsob umožňuje definovat, jaké atributy mají být v rámci objektu Osoba evidovány, a zároveň umožňuje, aby každý atribut měl svoji vlastní časovou platnost.

2.2.7.1 Dynamický model pro oblast atributů Osoby

2.2.7.1.1 Objekt Atribut osoby

Objekt Atribut osoby představuje číselník všech atributů, které jsou u Osoby sledovány, bez ohledu na to, pro kterou kategorii Osoby (fyzická osoba, právnická osoba, jiná osoba) jsou určeny. Pokud má být u osoby sledován jakýkoli atribut, musí být primárně založena instance této třídy.

Objekt Atributy osoby má tyto základní atributy (nemusí se jednat o kompletní výčet):

Název atributu	Účel atributu	Povinný?	Jedinečný?
kód_atributu	Jedinečná, uživatelsky čitelná identifikace atributu.	Ano	Ano

název_atributu	Pojmenování atributu	Ano	Ano
datový_typ	Určuje, jaký typ dat (hodnoty) bude možno k atributu zadat. Předpokládají se tyto základní typy: • STRING • DATE • NUMBER • ENUM (určuje, že hodnota atributu je tvořena číselníkem a může nabývat pouze předem definované hodnoty)	Ano	Ne
je_aktivní	„ano“ (default) / “ne“ – umožňuje zneaktivnit atribut, který již nemá být používán.	Ano	Ne

Tabulka 1 - Atributy objektu Atributy osoby

Výše uvedené atributy představují společnou charakteristiku. Lze předpokládat, že některé atributy nemají smysl pro specifickou kategorii Osoby. Například atribut BANIS nemá smysl vyžadovat po fyzických osobách. Dále lze předpokládat, že pro nějaký atribut bude pro specifickou kategorii Osob povinný, pro jinou nepovinný apod. Z těchto důvodů zavádíme objekt Definice atributu dle kategorie osoby, který umožňuje dodatečný popis atributu v závislosti na kategorii Osoby. Bez existence související instance v tomto objektu nebude atribut použitelný.

2.2.7.1.2 Objekt Validace atributu

Objekt Validace atributu umožňuje definovat k jednomu atributu N (žádnou nebo více) kontrol. Definice kontroly je možná:

- pomocí regulárního výrazu: kontrola je zapsána pomocí regulárního výrazu a umožňuje tak kontrolu základní syntaxe zapsané hodnoty,
- pomocí funkce: kontrola je zapsána pomocí databázové funkce, což umožňuje zapsat složitější kontroly (rozsah hodnot, modulo11 apod).

Kromě samotného formálního zápisu kontroly je ke každé kontrole třeba uvést:

- úroveň hlášení:
  - varování: zadaná hodnota nevyhovuje dané kontrole, ale tato kontrola není fatální a daná hodnota smí být uložena do databáze,
  - chyba: zadaná hodnota nevyhovuje dané kontrole a splnění této kontroly je nutné a daná hodnota musí kontrolu splnit. Údaj nelze vložit do databáze.
- pořadí kontroly: určuje, v jakém pořadí se mají kontroly provádět. Pokud například IČO obsahuje písmena (české IČO nesmí), nemá smysl provádět kontrolu na modulo.

2.2.7.1.3 Objekt Číselníkové hodnoty atributy

Objekt Číselníkové hodnoty atributu umožňuje definovat k jednomu atributu N (žádnou nebo více) číselníkových položek. Tato definice je možná jen v případě, že atribut je typu ENUM. V aplikaci pak pro daný atribut bude uživatel moci vybrat právě jednu položku z nabízeného seznamu položek.

2.2.7.1.4 Objekt Kategorie osoby

Objekt Kategorie osoby umožňuje definici číselníku kategorií Osob. Tento číselník zavádíme proto, aby pro různé osoby bylo možno definovat různé přiřazení atributů a nabízet tak pouze relevantní atributy (po fyzické osobě nepodnikateli nemá smysl vyžadovat IČO, žádné nemá).

V rámci tohoto objektu je sledován atribut možnost autoregistrace (boolean), který určuje, zda se daná kategorie má/nemá nabízet během procesu autoregistrace.

2.2.7.1.5 Objekt Definice atributu dle kategorie osoby

Objekt Definice atributu dle kategorie osoby je vázán jako podřízený k objektu Atribut osoby vazbou 1:N (jedna instance objektu Atribut osoby může mít N souvisejících instancí objektu Definice atributu dle kategorie osoby, naproti tomu jedna Definice atributu dle kategorie osoby musí mít právě jednu související instanci objektu Atribut osoby).

Zároveň platí další byznys omezení, kdy objekt Definice atributu dle kategorie osoby zavádí atributy datum od/do, pomocí kterých je vymezeno zařazení atributu ke konkrétní kategorii. Tím je řečeno, že daný atribut se pro danou kategorii Osoby v daném časovém úseku má zobrazovat a nabízet při zadávání a synchronizování. Musí platit, že v jeden časový okamžik existuje pro konkrétní atribut a pro konkrétní kategorii Osoby maximálně jedna souvisejí instance objektu Definice atributu dle kategorie osoby.

Objekt Atributy osoby má tyto základní atributy (nemusí se jednat o kompletní výčet):

Název atributu	Účel atributu	Povinný?	Jedinečný?
kategorie_osoby	Určuje, k jaké kategorii je Atribut osoby přivázán. Mohou existovat pouze tyto hodnoty: • fyzická osoba, • právnická osoba, • jiná osoba.	Ano	Ano
platnost_od	Datum, od kterého je zařazení atributu ke kategorii osoby platné.	Ano	Ne
platnost_do	Datum, do kterého je zařazení atributu ke kategorii osoby platné.	Ano	Ne
je_povinný	Ano/Ne – určuje, zda pro danou kategorii Osoby je atribut povinný (ano) a musí být zadán při zakládání nové osoby a nebo je	Ano	Ne

Název atributu	Účel atributu	Povinný?	Jedinečný?
	nepovinný a zadán být pouze může.		
je_jedinečný	Ano/Ne – určuje, zda pro danou kategorii Osoby musí být atribut jedinečný (ano) a zadání duplicitní hodnoty znamená nepřekonatelnou chybu a nebo jedinečný není (ne) a pak se hodnoty tohoto atributu mohou shodovat s hodnotami jiného subjektu.	Ano	Ne
je_opakovatelný	Ano/Ne – určuje, zda pro danou kategorii je možno zadat k jednomu subjektu v jeden čas více hodnot daného atributu (ano) a nebo smí být pro daný subjekt a jeden čas zadán právě jednou (ne).	Ano	Ne
pouze_ke_čtení	Pouze pro osoby přebírané z externích zdrojů. Ano/Ne – určuje, zda je daný atribut určen pouze ke čtení (ano) a nebo je možné jej měnit na straně správce SDAT (ne). Pokud není Osoba přebírána z externího zdroje (je založena v SDAT), pak tento atribut je irelevantní.	Ano	Ne
je_schvalován	Pouze pro osoby přebírané z externích zdrojů. Ano/ne – určuje, zda pro přijmutí hodnoty daného atributu je nutné schválení správcem systému (ano) a nebo je jakákoli změna provedená na straně externího systému rovnou přijata do SDAT (bez jakékoli další uživatelské aktivity).	Ano	Ne
úroveň_hlášení	Možné hodnoty ERROR/WARNING. Určuje, jak bude aplikace reagovat v případě, že bude zadána hodnota atributu, která nevyhoví zadanému regulárnímu výrazu: • ERROR – aplikace neumožňuje založení osoby, protože zadaná hodnota je chybná a nelze uložit celou osobu, • WARNING – aplikace umožňuje založení osoby, ale upozorní uživatele, že daný atribut obsahuje nevalidní hodnotu.	Ano	Ne

Tabulka 2 - Atributy dle kategorie Osoby

2.2.7.1.6 Objekt Hodnota atributu osoby

Objekt Hodnota atributu osoby je koncipován jako asociační třída, která spojuje objekty Osoba a Atribut osoby, které jsou propojeny vaznou N:M. To znamená, že jedna Osoba může být popsána více Atributy a jeden Atribut může být popisovat více Osob.

Objekt Hodnota atributu osoby zavádí datumové platnosti, které vymezují, po jakou dobu daná hodnota platí. Názorně si můžeme představit následující situaci:

- mějme osobu „O1“,
- mějme atribut „A1“ (atributem může například „adresa“),
- chceme zachytit situaci, kdy se Osoba „O1“ přestěhovala k 1.1.2014 za adresy „Ulice 1“ na adresu „Ulice 2“,
- vzniknout dvě instance objektu Hodnota Atributu Osoby:
    - O1/A1/1.1.2000/31.12.2013/Ulice 1,
    - O1/A1/1.1.2014/31.12.9999/Ulice 2.

O tom, zda může v jeden okamžik platit více hodnot daného atributu, rozhoduje nastavení daného atributu. Lze tak například uvažovat o tom, že existuje atribut OKEČ (Atribut „A2“), který prohlásíme za opakovatelný, a pak mohou instance objektu Hodnota atributu osoby vzniknout takto:

- O1/A2/1.1.2000/31.12.9999/66 - Pojišťovnictví a penzijní financování kromě povinného sociálního zabezpečení,
- O1/A2/1.1.2000/31.12.9999/67 - Pomocné činnosti související s finančním zprostředkováním.

2.2.7.1.7 Objekt Pracovník osoby

V rámci analýzy byla identifikována potřeba evidovat k Osobám kontaktní pracovníky. Na tuto úlohu je výše zmíněný aparát dynamických atributů nevhodný, protože u pracovníků potřebujeme podchytit více souvisejících atributů najednou (jméno, příjmení, funkci, kontaktní údaje) a navíc potřebujeme evidovat hierarchické vazby mezi pracovníky. S ohledem na fakt, že se jedná o relativně ustálenou oblast, byla pokryta samostatnými objekty.

Objekt Pracovník osoby je podřízený objekt k objektu Osoba ve vazbě 1:N (jedna Osoba může mít N Pracovníků osoby, jeden Pracovník osoby může být zařazen k právě jedné Osobě). V rámci tohoto objektu jsou zavedeny atributy datum od/do, které vymezují období, po které daný pracovník byl přiřazen k odpovědnosti za nějakou oblast.

Název atributu	Účel atributu	Povinný?	Jedinečný?
typ_odpovědnosti	Číselník (editovatelný uživatelem SDAT) umožňující určit typ odpovědnosti, tedy pro jakou oblast je daný zaměstnanec určen jako kontaktní. Bude připojen rozšiřitelný číselník, v základu bude obsahovat:	Ano	Ne

Název atributu	Účel atributu	Povinný?	Jedinečný?
	• Technický správce • Věcný správce • Příjemce upomínek		
platnost_od	Datum, od kterého je pracovník zařazen k odpovědnosti.	Ano	Ne
platnost_do	Datum, do kterého je pracovník zařazen k odpovědnosti.	Ano	Ne
jméno	Jméno pracovníka	Ano	Ne
příjmení	Příjmení pracovníka	Ano	Ne
funkce	Funkce pracovníka (textová hodnota)	Ne	Ne

Tabulka 3 - Atributy Pracovníka osoby

Protože se předpokládá, že jeden Pracovník osoby může mít více různých kontaktů, navíc v různém čase mohou platit různé kontakty, je evidence kontaktů svěřena objektu Kontakt pracovníka.

V rámci objektu Pracovník osoby je zavedena rekurzivní asociační vazba s názvem Nadřízený pracovník. Pomocí této vazby bude možno vytvářet hierarchii pracovníků s tím, že konkrétní Pracovník osoby nemusí mít žádného nebo právě jednoho nadřízeného.

Objekt Pracovník osoby je spojen asociační vazbou s objektem Uživatel. Asociační vazba je na obou svých koncích definována jako 0..1. To znamená následující:

- v systému může existovat instance objektu Pracovník osoby, aniž by k ní existovala vazba na instanci objektu Uživatel. Pokud však tato instance existuje, platí, že k jedné instanci objektu Pracovník osoby může existovat právě jedna instance objektu Uživatel. To je v praxi stav, kdy je třeba k Osobě evidovat Pracovníka osoby, který nemá přístup do aplikace SDAT. To může být například jednatel společnosti/generální ředitel, kterému se budou zasílat zcela zásadní sdělení (upomínky), ale tento jednatel/generální ředitel nebude vybaven právem přístupu do aplikace SDAT. V okamžiku, kdy evidujeme Pracovníka osoby, který má mít přístup do aplikace SDAT, je tento Pracovník osoby spojen právě s jedním Uživatelem (má právě jeden uživatelský účet pomocí kterého přistupuje do aplikace SDAT),
- v systému může existovat instance objektu Uživatel, aniž by k ní existovala vazba na instanci objektu Pracovník osoby. Pokud však tato vazba existuje, platí, že k jedné instanci objektu Uživatel může odpovídat právě jedna instance objektu Pracovník osoby. To je v praxi stav, kdy například zaměstnanec ČNB není pracovníkem žádné Osoby, přesto má uživatelský účet (existuje instance objektu Uživatel, ale ta nemá žádnou související instanci objektu Pracovník osoby), pomocí kterého přistupuje k systému SDAT a vykonává v něm aktivity. V okamžiku, kdy bude třeba zřídit aplikační účet pro pracovníka Osoby, bude tento aplikační účet spojen s právě jedním Pracovníkem osoby.

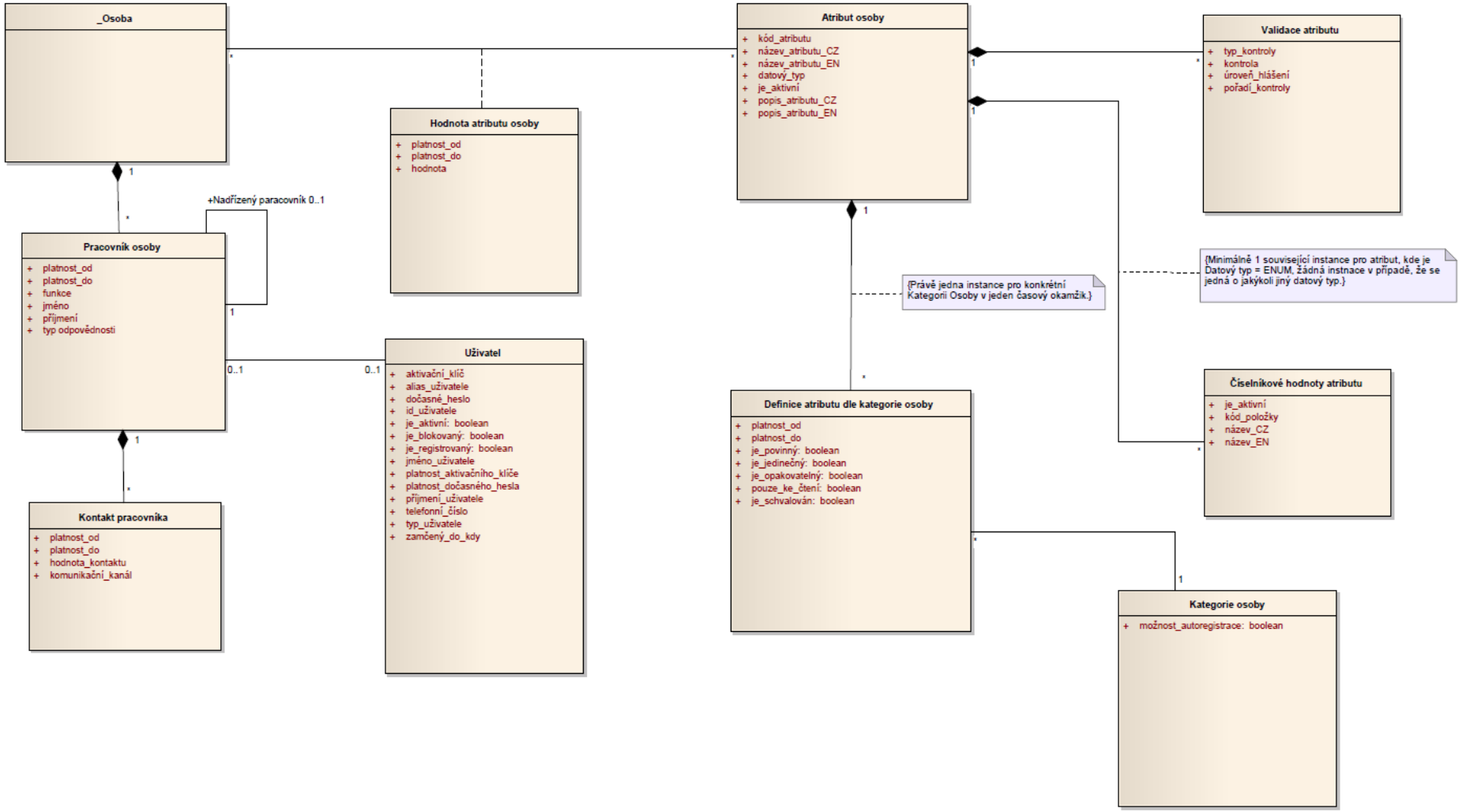
2.2.7.1.8 Objekt Kontakt pracovníka

Účelem objektu Kontakt pracovníka je jednotná evidence všech různých kontaktů, které k pracovníkovi mohou být editovány. Opět je použita dynamická filozofie, kdy je možno evidovat neomezený počet kontaktů k pracovníkovi, dokonce i pro neomezený počet komunikačních kanálů (komunikačním kanálem zde rozumíme prostředek komunikace, v zásadě má smysl udržovat kontakt pro telefon/fax/e-mail, nicméně nic nebrání tomu evidovat jakékoli další komunikační kanály, například SkypeID a jiné).

Název atributu	Účel atributu	Povinný?	Jedinečný?
komunikační_kanál	Číselník (editovatelný uživatelem SDAT) umožňující určit, k jakému komunikačnímu prostředku je hodnota přiřazena, v základu by měla evidence postihovat • telefon • fax • e-mail	Ano	Ne
platnost_od	Datum, od kterého platí daný kontakt.	Ano	Ne
platnost_do	Datum, do kterého platí daný kontakt.	Ano	Ne
hodnota_kontaktu	Vlastní hodnota kontaktu (telefonní číslo, e-mailová adresa, jiný kontakt)	Ano	Ne

Tabulka 4 - Atributy Kontaktu pracovníka

2.2.7.1.9 Objektový model – dynamické atributy pro objekt Osoba



Obrázek 18 - Objektový model pro dynamické atributy objektu Osoba

2.3 Komunikační modul

Komunikační modul je část systému SDAT, pomocí které jsou předávány zprávy nebo požadavky mezi jednotlivými interními uživateli systému. Komunikační modul je tak využíván v případě, kdy uživatel potřebuje aktivitu jiného uživatele, aby mohl jeho proces pokračovat (nebo být dokončen). Nejčastěji bude komunikační modul použit v oblasti správy metadat a projektování (viz dokument B-Metapopis, kapitola 5 Hlavní procesy).

Systém SDAT nemusí obsahovat komunikační modul přímo a je možno požadované funkcionality zajistit s využitím API nějakého externího nástroje (Redmine, JIRA apod.; přičemž napojení na systém Redmine je preferováno, protože tento systém je již ČNB provozován), který se primárně zabývá sledováním úkolů a řízením workflow procesů. Výběr nástroje, který bude použit pro správu úkolů je v kompetenci dodavatele, pouze je nutno vzít v úvahu, že bude-li použito nějaké komerční řešení, je nutno zahrnout náklady na licence do celkové kalkulace ceny dodávaného řešení. V dalším textu bude tato část komunikačního modulu nazývána „task management systém“.

Celkově se tak komunikační modul skládá z:

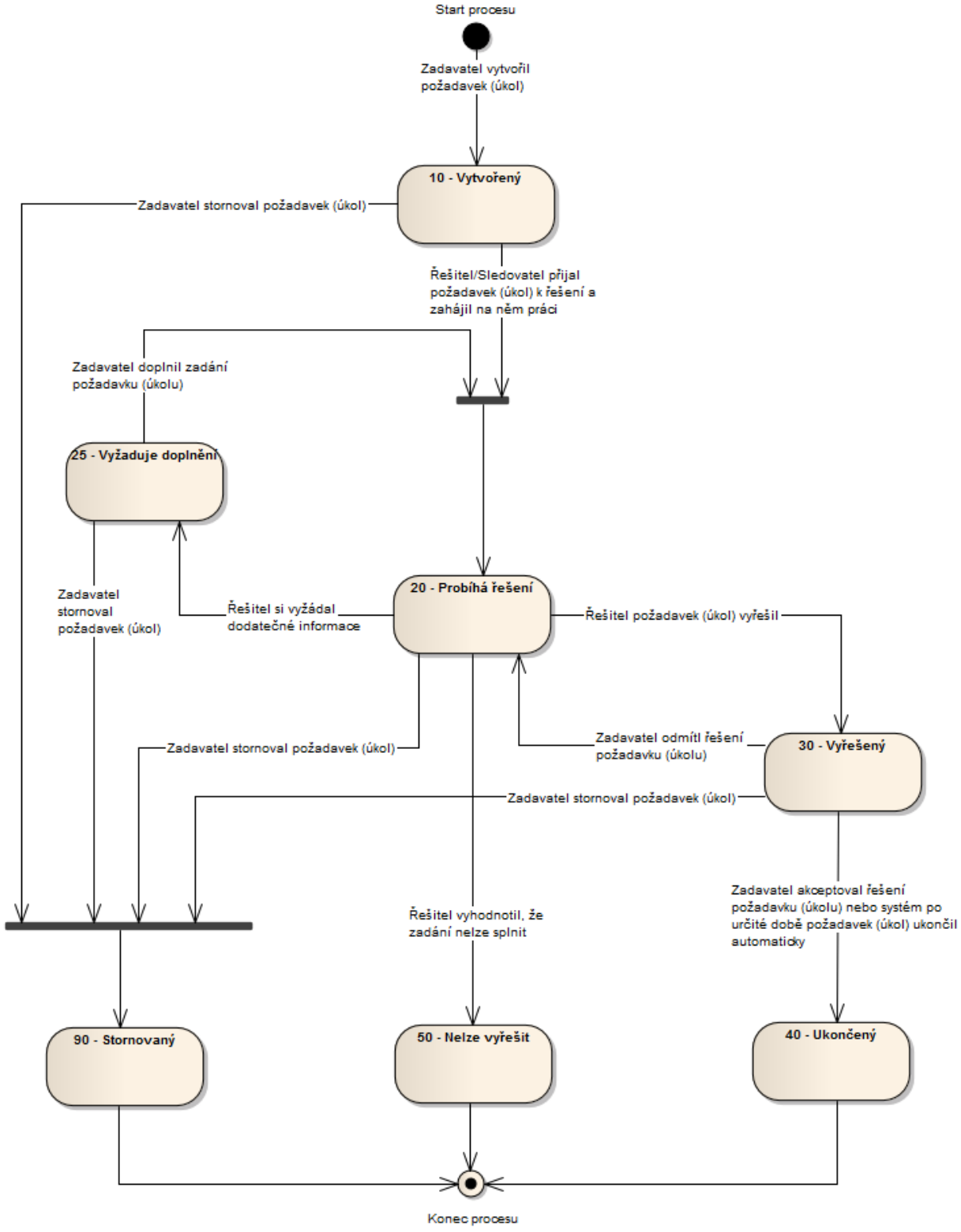
- **task management systému** – jedná o systém, který je primárně určen ke správě úkolů a řízení workflow s těmito úkoly souvisejícími. Tato část komunikačního modulu je realizována mimo systém SDAT,
- **rozhraní SDAT** – jedná se o sadu funkcionalit a formulářů, které jsou dostupné přímo v systému SDAT a zprostředkovávají uživatelům SDAT komunikaci s task management systémem. Uživateli tak umožňují správu dat (zobrazení, výběr, modifikaci) uložených v task management systému přímo z prostředí aplikace SDAT.

Je požadováno, aby komunikační modul jako celek zajišťoval minimálně tyto funkcionality (rozdělení mezi systém SDAT a task management systém není pevně dané):

- možnost vytvořit požadavek (úkol), přiřadit jej konkrétnímu řešiteli (právě jednomu) a uvést termín dodání požadovaného řešení přímo z prostředí aplikace SDAT a následně uložit do task management systému,
- možnost určit tzv. sledovatele úkolu, tedy možnost uvést jednu a více dalších osob, které budou mít k danému požadavku (úkolu) přístup (budou jej moci modifikovat), ale nebudou vystupovat v roli řešitele. Řešitel zároveň nesmí být v množině sledovatelů. Sledovatelé mohou kdykoli převzít řešení problému, nicméně ten uživatel, který toto udělá, se automaticky stává řešitelem (a řešitel sledovatelem),
- možnost provést změnu stavu úkolu (v souladu s níže uvedeným stavovým diagramem) a spolu s každou změnou uvést komentář, který s danou změnou souvisí,
- možnost stornovat požadavek (úkol) zadavatel v jakékoli fázi procesu,
- možnost definovat (nastavovat a měnit) workflow proces, který bude zajišťovat životní cyklus požadavku (základní nastavení a změnu workflow není nutné řešit přes rozhraní SDAT; definice workflow procesu se předpokládá nativními prostředky task management systému),
- možnost prohlížet seznam požadavků (úkolů) v členění na „uživatelem vytvořené“, „uživatelem sledované“ a „uživatelem řešené“,
- možnost zasílat e-mailové notifikace zúčastněným osobám (zadavateli, řešiteli, sledovatelům) v případě jakékoli změny stavu daného požadavku (úkolu),

- možnost přiložit neomezený počet binárních souborů k požadavku úkolu ve všech stavech životního cyklu požadavku (úkolu),
- možnost automaticky ukončit takový požadavek (úkol), který je vyřešen a není zadavatelem akceptován do určité doby (například do 10 dní od vyřešení; počet dní je parametrem systému),
- možnost generovat a zasílat notifikační e-maily před blížícím se termínem požadovaného vyřešení v případě, že požadavek (úkol) nebyl doposud vyřešen,
- možnost generovat a zasílat notifikační e-maily v případě, že došlo k překročení požadovaného data vyřešení úkolu,
- možnost nastavovat příjemce (skupiny příjemců) notifikačních a eskalačních e-mailů (není nutné řešit přes rozhraní SDAT).

Základní nastavení workflow pro zpracování požadavku (úkolu) je patrné z následujícího stavového diagramu a níže uvedené tabulky s popisem jednotlivých stavů.



Obrázek 19 - Stavový diagram workflow pro řízení zpracování požadavku (úkolu)

Tabulka č. 1 Přehled stavů zpracování požadavku (úkolu)

Kód stavu	Název stavu	Definice a účel stavu	Předcházející stav	Návazný stav a událost, která jej generuje	Dodatečná informace
10	**Vytvořený**	**Definice**: Byl vytvořen nový požadavek (úkol) **Účel**: Odlišit požadavky (úkoly), které vznikly a ještě se jim nikdo nezačal věnovat od požadavků (úkolů) kterými se již zabývá nějaký konkrétní řešitel.	N/A	**20 – Probíhá řešení** Buď řešitel, anebo jeden z tzv. sledovatelů, přijal požadavek (úkol) k řešení a zahájil tak práci na požadavku (úkolu). **90 – Stornovaný** Zadavatel rozhodl, že daný požadavek (úkol) není nadále aktuální a není třeba, aby se mu kdokoli dále věnoval.	
20	**Probíhá řešení**	**Definice**: Požadavek (úkol) je přiřazen právě jednomu řešiteli a tento řešitel zahájil práce vedoucí k vyřešení požadavku (úkolu). **Účel**: Odlišit požadavky (úkoly), které vznikly a ještě se jim nikdo nezačal věnovat od požadavků (úkolů) kterými se již zabývá nějaký konkrétní řešitel.	10 – Vytvořený	**25 – Vyžaduje doplnění** Řešitel potřebuje od zadavatele získat další podrobnosti, které nejsou v zadání obsažené. **30 – Vyřešený** Řešitel vyřešil (realizoval požadovanou akci v SDAT) požadavek (úkol) a předává zadavateli řešení k akceptaci. **50 – Nelze vyřešit** Řešitel v průběhu řešení zjistil, že požadavek (úkol) nelze vyřešit.	

Kód stavu	Název stavu	Definice a účel stavu	Předcházející stav	Návazný stav a událost, která jej generuje	Dodatečná informace
				90 – Stornovaný Zadavatel rozhodl, že daný požadavek (úkol) není nadále aktuální a není třeba, aby se mu kdokoli dále věnoval.	
25	**Vyžaduje doplnění**	**Definice**: Požadavek (úkol) je přiřazen právě jednomu řešiteli a tento řešitel potřebuje k dokončení své práce získat další podrobnosti, které nejsou v zadání obsaženy. **Účel**: Odlišit požadavky (úkoly), které není možno vyřešit, pokud nebude doplněno jejich zadání.	20 – Probíhá řešení	**20 – Probíhá řešení** Zadavatel doplnil zadání dle pokynu řešitele a předává mu požadavek (úkol) zpět k řešení. **90 – Stornovaný** Zadavatel rozhodl, že daný požadavek (úkol) není nadále aktuální a není třeba, aby se mu kdokoli dále věnoval.	V tuto chvíli již znovu neprobíhá aktivita přijetí požadavku (úkolu) řešitelem; má se za to, že když řešitel již jednou požadavek přijal a požádal o doplňující informace, je stále řešitelem tohoto požadavku (úkolu).
30	**Vyřešený**	**Definice**: Požadavek (úkol) byl řešitelem vyřešen a výsledek je předáván zadavateli k akceptaci. **Účel**: Sdělit zadavateli, že jeho požadavek (úkol) je vyřešen a může ověřit	20 - Probíhá řešení	**20 – Probíhá řešení** Zadavatel se seznámil s řešením a zjistil, že řešení nevyhovuje jeho zadání a požaduje od řešitele dodání jiného řešení. **40 – Ukončený**	

Kód stavu	Název stavu	Definice a účel stavu	Předcházející stav	Návazný stav a událost, která jej generuje	Dodatečná informace
		výsledek řešení, případně pokračovat v práci, kterou musel přerušit do doby, než bude jeho požadavek (úkol) vyřešen.		Zadavatel se seznámil s řešením a toto řešení akceptoval. **90 – Stornovaný** Zadavatel rozhodl, že daný požadavek (úkol) není nadále aktuální a není třeba, aby se mu kdokoli dále věnoval.	
40	**Ukončený**	**Definice**: Požadavek (úkol) byl řešitelem vyřešen a zadavatelem akceptován. **Účel**: Sdělit řešiteli, že jeho řešení bylo akceptováno a odlišit požadavky (úkoly), které byly vyřešeny, ale nejsou akceptovány, od požadavků (úkolů), které jsou pouze vyřešené, ale neakceptované.	30 - Vyřešený	**N/A**	Jedná se o definitivní stav v rámci workflow pro sledování požadavků (úkolů), životní cyklus požadavku (úkolu) končí.
50	**Nelze vyřešit**	**Definice**: Požadavek (úkol) byl řešitelem vyhodnocen jako nesplnitelný **Účel**: Sdělit zadavateli, že jeho zadání odporuje	20 – Probíhá řešení	**N/A**	Jedná se o definitivní stav v rámci workflow pro sledování požadavků (úkolů), životní cyklus požadavku (úkolu) končí.

Kód stavu	Název stavu	Definice a účel stavu	Předcházející stav	Návazný stav a událost, která jej generuje	Dodatečná informace
		nějakému pravidlu nebo něco dalšího brání splnění jeho požadavku (úkolu).			
90	**Stornovaný**	**Definice**: Požadavek (úkol) byl zadavatelem stornován a není třeba se jím dále zabývat **Účel**: Umožnit stornovat požadavek (úkol), který není nadále aktuální (nebo byl zadán omylem)	10 – Vytvořený 20 – Probíhá řešení 25 – Vyžaduje doplnění 30 - Vyřešený	**N/A**	Jedná se o definitivní stav v rámci workflow pro sledování požadavků (úkolů), životní cyklus požadavku (úkolu) končí.

2.4 Administrační modul

Systém SDAT obsahuje část, do které jsou koncentrovány informace o běhu celého systému napříč všemi jeho komponentami a dále funkce a nastavení, které souvisí s chodem systému jako takového. Způsob realizace níže uvedených požadavků (zejména část pro monitorování provozu a správa systémových proměnných) záleží na celkovém technickém řešení systému. Z tohoto důvodu je obsah této kapitoly spíše popisem koncepčního požadavku bez definice konkrétnějšího konceptuálního objektového modelu jako u jiných oblastí.

2.4.1 Monitorování procesů

Monitorování provozu systému je prováděno na základě evidence procesů, které v systému v daný okamžik probíhají. Procesem se rozumí jedna nebo zpravidla sekvence více programových procedur, které systém vykonává pro zajištění některé z jeho funkcí. Příkladem procesu je v oblasti sběru dat proces zpracování Vstupní zprávy. Jednotlivými procedurami jsou pak syntaktická kontrola Vstupní zprávy, logické kontroly těla Vstupní zprávy atd. Základem je katalog procesů, který obsahuje textový popis procesu, způsob jakým je proces defaultně spouštěn (např. manuálně, automaticky – událostí, automaticky - plánem), oblast do které proces patří (např. Sběr dat) a seznam programových procedur, které proces tvoří. Samotné monitorování procesů, které umožňuje sledovat běh systému, je realizováno jako přehled všech evidovaných procesů, včetně informace o aktuálním stavu procesu, začátku posledního běhu procesu a skutečném způsobu jeho spuštění (např. proces odeslání upomínek spouštěný defaultně automaticky plánovačem byl spuštěn manuálně uživatelem). V seznamu je možné řadit a filtrovat na základě kritérií jako je čas spuštění posledního běhu procesu, oblast procesu, stav procesu. Z přehledu procesů lze přejít na detail každého procesu v podobě historie všech jeho běhů (výsledný stav, začátek a konec běhu procesu) a dále na detailní log obsahující informace o běhu jednotlivých programových procedur.

2.4.2 Systémové proměnné

Nastavení proměnných jednotlivých komponent systému a jejich procesů, které se nastavují na úrovni jeho správy, je koncentrováno do administračního modulu systému. Kromě hodnoty proměnné, je evidována rovněž její defaultní hodnota a její textový popis.

2.4.3 Monitorování aktivity uživatelů

Tato část představuje jednotný bod ke sledování aktivit uživatelů systému. Základním parametrem pro procházení logů systému je skutečnost, zda se jedná o interní (zaměstnanec ČNB) nebo externí uživatele. Aktivity lze sledovat až na úroveň jednoho konkrétního uživatele. Dostupné jsou vzájemně provázané přehledy aktivit uživatele, seznam aktuálně přihlášených uživatelů, v případě interních uživatelů seznam uživatelských zámků nad projektovanými částmi Výkazů s možností zámek uvolnit.

2.4.4 Referenční informace

Modul poskytuje jednotný bod přístupu k obsahu a správě systémovým číselníků.

2.4.5 Monitor systému

Jednotné místo pro sledování stavu systémového prostředí, které uživatelsky prezentuje informace o aktuálních transakčních zámcích databázových objektů, zaplnění databázových prostorů, zaplnění aplikačních disků apod.

2.5 Systémové číselníky

Systémové číselníky představují obory hodnot atributů objektů různých komponent systému. Všechny tyto číselníky, lze ve fyzickém datovém modelu řešit jednotně, ale zároveň musí být odděleny od číselníků metapopisu (objekt Číselník), viz dokument B – Metapopis, kapitola Objekt Číselník. Příkladem systémových číselníků je číselník standardizovaných způsobů zpracování, viz dokument D – Sběr dat, kapitola Objekt Způsob zpracování.

3 Katalog nefunkčních požadavků

Následující kapitola popisuje nefunkční požadavky za jednotlivé oblasti systému. Do samostatné kapitoly jsou odděleny požadavky na uživatelské rozhraní, které jsou definovány až na úroveň jednotlivých ovládacích prvků, které budou v systému použity a tvoří tak základ definice ergonomie uživatelského rozhraní.

3.1 Obecné nefunkční požadavky

Obecné nefunkční požadavky na systém popisují dostupnost systému, řešení závad, školení a další aspekty provozu systému. U těchto požadavků nemá smysl definovat, v jaké fázi vývoje budou realizovány, protože se týkají buď architektury systému, nebo jeho provozu. Proto je u těchto požadavků uvedena jako Kategorie hodnota „N/A“, tedy „není určeno“. Pokud má nějaký nefunkční požadavek definovánu kategorii tímto způsobem, má se za to, že musí být splněn k okamžiku zahájení ověřovacího provozu na produkčním prostředí, což však znamená, že jeho splnění bude testováno před zahájením nasazením systému k produkčnímu užívání. Bez splnění takto definovaných nefunkčních požadavků nebude možné provést finální akceptaci systému a nasazení systému k ověřovacímu provozu na produkčním prostředí.

Tam, kde je jako Kategorie uvedeno číslo, tam je nutno zajistit splnění tohoto nefunkčního požadavku v předepsané kategorii plnění.

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
NFP_1.0	SLA - Dostupnost systému	Systém je dostupný 99,9% disponibilního času za rok. Systém tak smí být nedostupný pouze 8,76 hodin v roce. Do disponibilního času se nezapočítává čas určený na údržbu systému (aktualizace operačního systému).	Závazný	N/A
NFP_2.0	SLA – Řešení závad systému	Závady systému budou vyřešeny v takových dobách, jaké jsou uvedeny v příloze č. 7 Smlouvy, Provozní podpora.	Závazný	N/A
NFP_3.0	Zaškolení interních uživatelů	Zaškolení uživatelů, kteří na straně ČNB budou připravovat výkazy, musí proběhnout jednorázově a nesmí překročit 3 dny.	Závazný	N/A
NFP_4.0	Zaškolení externích	Zaškolení uživatelů z řad externích subjektů musí být možná pouze	Závazný	N/A

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
	uživatelů	s využitím dokumentace, případně jiných, k tomu určených materiálů (videotutoriály apod.).		
NFP_5.0	Výkon – běžný provoz	95% všech standardních (mimo hromadné, dlouhotrvající akce) operací bude mít odezvu do 2 sekund, zbývajících 5% standardních operací bude mít odezvu maximálně do 8 sekund. Po dobu provádění akcí, které nebude možno vyřídit okamžitě (s dobou trvání pod 1 sekundu), bude uživateli systém indikovat svoji činnost (viz URO_1.0 a URO_2.0).	Závazný	N/A
NFP_6.0	Výkon - špička	Systém zajistí takové odezvy systému, jako je popsáno v NFP_5.0, v případě, že k systému přistupuje 500 současných uživatelů. V případě, že je tento počet současně pracujících uživatelů překročen, platí dvojnásobné limity.	Závazný	N/A
NFP_8.0	Synchronizace času	Systém umožňuje (jak na aplikační, tak databázově vrstvě) synchronizaci se zdrojem přesného času.	Závazný	1
NFP_9.0	Autentifikace - metody	Viz dokument F – Uživatelé a přístupová práva a funkční požadavky UMU_9.0, UMU_10.0 a UMU_14.0.	Závazný	1
NFP_10.0	HW a SW nároky – systémové prostředí	Pro práci se systémem stačí běžné kancelářské PC s operačním systémem Windows. Instalace produktů třetích stran na klientské stanice je povolena pouze v případě, že takový produkt odpovídá systémovému prostředí ČNB.	Závazný	N/A
NFP_11.0	HW a SW nároky – externí uživatelé	V případě, že systém nebo jeho část bude dodána jako webová aplikace, pak funkčnost aplikace není vázána na použití konkrétního webového prohlížeče, aplikace plně podporuje minimálně tyto prohlížeče: - Internet Explorer, - Mozilla Firefox, - Google Chrome,	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
		ve verzích, které jsou aktuální v době realizace zakázky. Pokud během času trvání podpory dojde ze strany výrobce prohlížeče k ukončení podpory verze prohlížeče, na které byl systém odladěn, je dodavatel povinen upravit na své náklady systém tak, aby byl kompatibilní s novou verzí prohlížeče.		
NFP_13.0	Audit – zjednodušený	Každý záznam uložený v databázi obsahuje následující informace: - informace o uživateli, který záznam vytvořil, - časové razítko vytvoření záznamu, - informace o uživateli, který záznam naposledy změnil, - časové razítko poslední změny záznamu. Výše uvedené údaje jsou viditelné v aplikaci v místě, kde se s daným záznamem pracuje, každému uživateli, který má oprávnění daný záznam vidět.	Závazný	1
NFP_14.0	Kompletní aplikační audit	Systém umožňuje kompletní aplikační audit, tj. uchovává informace o tom, jaký uživatel, k jakým údajům a kdy přistupoval, případně co s nimi dělal za operace. Oproti zjednodušenému auditu je schopen zaznamenat i operace čtení a mazání záznamu a v případě modifikace záznamu je schopen zaznamenat hodnoty záznamu před a po modifikaci. Systém poskytuje aplikační rozhraní, které umožňuje s daty aplikačního auditu pracovat přímo v aplikaci. Stejně tak poskytuje možnost nastavit, po jakou dobu se mají auditní data uchovávat, než budou smazána/archivována.	Závazný	1
NFP_16.0	Autorizace	Systém disponuje nástrojem pro řízení přístupových práv. Lze pomocí něj řídit přístup vybraného uživatele k vybraným datům. Autorizaci je věnován samostatný dokument F – Uživatelé a přístupová práva a funkční požadavky v něm uvedené.	Závazný	1
NFP_17.0	Automatické	Systém umožňuje použít automatické postupy zálohování a obnovy,	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
	zálohování	a to jak samotného SW řešení, tak i jeho konfigurace i dat.		
NFP_18.0	Plánování záloh	Systém umožňuje naplánování programů zálohování stanovením frekvence záloh.	Závazný	1
NFP_19.0	Obnova ze zálohy – aplikační část	Systém umožňuje při zachování stejného operačního systému obnovu ze zálohy nezávisle na použitém hardware.	Závazný	1
NFP_20.0	Obnova ze zálohy – databáze	Systém umožňuje na úrovni databáze on-line vytvářet tzv. transakční log, který umožňuje ex-post nastavit databázi do libovolného stavu v minulosti (v rámci doby, po kterou jsou archivovány body obnovy a transakční logy).	Závazný	1
NFP_21.0	Obnovitelnost v záložní lokalitě	Architektura aplikace je zvolena tak, aby při výpadku hlavních serverů mohl objednatel vlastními zdroji a bez účastni dodavatele přesunout zpracování do záložního střediska během 1 hodiny a obnovit kompletní funkčnost aplikace do 8 hodin.	Závazný	1
NFP_22.0	Instalace části systému na klientské stanice	Instalace aplikace na klientské stanice nevyžaduje administrátorská oprávnění.	Závazný	1
NFP_23.0	Licence	Licence je nezávislá na počtu Osob a uživatelů (tzv. „multilicence“).	Závazný	1
NFP_24.0	Kódová stránka	Systém umožňuje zpracovávat znaky v kódování UTF8, a to jak na databázové, tak na aplikační vrstvě.	Závazný	1
NFP_25.0	Šíře a kvalita vývojářské dokumentace	Vývojářská dokumentace musí být psána buď v českém, nebo anglickém jazyce a musí obsahovat popis všech aplikačních rozhraní IS, včetně jejich příkladů použití.	Závazný	1
NFP_26.0	Nápověda (kontextový help) – koncový uživatel	Systém v rámci uživatelského rozhraní nabízí možnost zobrazení kontextové nápovědy (kontextem se myslí to, že nápověda se uživateli zobrazuje s ohledem na právě prováděnou aktivitu). Tato nápověda je lokalizovaná do českého jazyka.	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
NFP_27.0	Nápověda – administrátor	K systému je dodána kompletní administrátorská dokumentace. Tato dokumentace je lokalizována do českého jazyka a musí být v elektronické formě. Není požadováno, aby tato část dokumentace měla podobu kontextového helpu jako je tomu v případě nápovědy pro koncového uživatele (viz NFP_26.0).	Závazný	1
NFP_28.0	Lokalizace aplikace	Systém je lokalizován do dvou jazyků/národních prostředí – český jazyk a anglický jazyk. Každý jazyk je spojen s právě jedním národním prostředím/znakovou sadou. Jeden jazyk/národní prostředí/znaková sada je označen jako hlavní (český jazyk). Systém umožňuje uživatelským způsobem přidat lokalizaci do neomezeného počtu dalších jazyků/národních prostředí/znakových sad. Lokalizací se rozumí: • překlady prostředí aplikace (popisky formulářových polí (labely), informační/varovná/chybová hlášení (messages) a další prvky uživatelského rozhraní), • překlady metadat, tj. číselníky a jiné objekty metapopisu.. • formát dat, tj. čísla (desetinná tečka nebo čárka, seskupování číslic, oddělovač skupiny číslic, znak měny a jeho umístění), datumové a časové údaje (formát data a času, definice prvního dne v týdnu). Národní prostředí v systému SDAT je koncipováno shodně s národním prostředím operačního systému Windows; pro jazyk „Angličtina“ je použito národní prostředí „Spojené Království“. Lokalizace pro hlavní jazyk musí být úplná, tzn. nesmí existovat žádný text v rámci uživatelského rozhraní, který by nebyl lokalizován do hlavního jazyka.	Závazný	1
NFP_28.1	Lokalizace aplikace – defaultní jazyk	Systém uživateli nastaví takový jazyk/znakovou sadu, který odpovídá nastavením jeho jazykových preferencí ve	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
		webovém prohlížeči, kterým přistupuje k aplikaci SDAT (defaultní jazyk/znaková sada). Toto pravidlo platí pro tu část aplikace, kde se pracuje s neověřeným uživatelem (veřejná část systému a neveřejná část systému do doby, než se uživatel autentifikuje, tj. prokáže svoji identitu). V okamžiku, kdy systém zná identitu uživatele, nastaví jazyk/národní prostředí/znakovou sadu aplikace na základě jeho preferencí uložených v uživatelské konfiguraci v systému.		
NFP_28.2	Lokalizace aplikace – změna jazyka	Systém umožňuje uživateli kdykoli během práce s aplikací změnit jazyk/národní prostředí/znakovou sadu na jakýkoli jiný, systémem podporovaný jazyk. Systém při změně jazyka provede: • změnu preferovaného jazyka/národního prostředí v uživatelské konfiguraci (preferencích) daného uživatele, • změnu uživatelského rozhraní, tj. změnu všech překladů aplikace/dat a jejich formátu (viz výše), bez toho, aniž by se uživatel musel znovu přihlašovat. Při změně jazyka systém provede změnu tak, že neopustí aktuálně zobrazený formulář. V případě, že po změně jazyka neexistuje lokalizační text pro nějaký prvek/datový objekt, systém zobrazí související text z lokalizační sady hlavního jazyka.	Závazný	1
NFP_29.0	Anonymizace dat	Systém disponuje nástrojem na anonymizaci dat. Tento nástroj bude použit při synchronizaci dat mezi různými prostředími, zejména mezi produkčním a školicím prostředím. Po anonymizaci nesmí být možno poznat, jaké reálné Osobě (či za jakou Osobu) patří vykázaná data (Hodnoty údajů). Dále z anonymizovaných dat nesmí být zjistitelné jakékoli reálné osobní údaje reálných uživatelů systému.	Závazný	1
NFP_30.0	Profylaxe	Dodavatel připraví a dodá objednateli všechny procedury nezbytné pro provádění optimalizace uložení dat a indexů v relační databázi.	Závazný	1

3.2 Obecné nefunkční požadavky – Bezpečnost

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
BEZ_1.0	Přístup interních uživatelů na základě nastavení Řídící databáze	Systém umožňuje SSO autentifikaci pouze těm uživatelům, kteří jsou zařazeni alespoň v jedné aplikační skupině určené pro SDAT. V případě, že se k aplikaci pokusí přistoupit uživatel, který není členem žádné aplikační skupiny SDAT, systém odmítne uživatele ověřit a oznámí mu důvod odmítnutí.	Závazný	1
BEZ_2.0	Zabezpečení síťové komunikace	V případě, že systém nebo jeho část bude dodána jako webová aplikace, je tato webová aplikace provozována přes zabezpečený protokol TLS. V případě, že dochází ke komunikaci „aplikace vs. SDAT“ (například při využití aplikace třetí strany na straně Osoby a použití kanálu „webová služba“) proběhne při vytvoření SSL kanálu vzájemné ověření certifikátů serverů (tzv. handshake).	Závazný	1
BEZ_3.0	Hosting/Outsourcing	Řešení nesmí využívat žádné outsourcing a hosting služby mimo infrastrukturu ČNB.	Závazný	1
BEZ_4.0	Zpracovávání informací mimo ČNB	SW nesmí zpracovávat data, informace ani jejich části mimo systémové prostředí ČNB a nesmí obsahovat jakoukoli SW/HW část provozovanou mimo infrastrukturu ČNB.	Závazný	1
BEZ_5.0	Licenční ujednání	Součástí zdrojových kódů aplikace nejsou části, které by dodavatel z licenčních nebo jiných důvodů nemohl předat ve zdrojové formě objednateli za účelem následné modifikace, rozšíření nebo údržby systému.	Závazný	1
BEZ_6.0	Ukončení spojení při nečinnosti uživatele	V případě, že systém nebo jeho část je dodávána jako webová aplikace, pak systém zajišťuje, že po určité době nečinnosti (tato doba je konfigurovatelná objednatelem) dojde k vypršení tzv. session a je nutné se znovu přihlásit.	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
BEZ_6.1	Ukončení spojení při nečinnosti uživatele – uchování neuložených dat	V případě, že dojde k ukončení spojení kvůli nečinnosti uživatele (viz BEZ_6.0), pak systém umožňuje uchovat neuložené změny a nabídnout je při příštím přihlášení uživatele k dopracování. Tímto požadavkem **není myšleno**, že má tyto změny před vypršením tzv. session uložit do databáze způsobem, jakým by to učinil uživatel.	Vítaný	3
BEZ_7.0	Uložení dat v externím úložišti	Pokud bude použita taková architektura systému, která vyžaduje uložení dat mimo databázi (například uložení binárních souborů přímo v souborovém systému), pak systém zajistí zabezpečení tohoto externího úložiště pomocí šifrování.	Závazný	1
BEZ_8.0	Šifrovací algoritmy	Systém používá (pokud není konkrétním funkčním požadavkem uvedeno jinak) kryptografické prostředky v souladu s přílohou 3 vyhlášky č. 316/2014 Sb. - Minimální požadavky na kryptografické algoritmy.	Závazný	1
BEZ_9.0	Odolnost proti známým hrozbám	Systém neobsahuje známé zranitelnosti (dle seznamu OWASP TOP10 a CWE/SANS TOP 25). Testy na zranitelnosti budou prováděny pravidelně i po nasazení systému a zhotovitel bezodkladně odstraní všechny nalezené zranitelnosti od kategorie mírné výše na své náklady.	Závazný	1
BEZ_10.0	Extended Validation certifikáty	Pro produkční i testovací prostředí jsou použity Extended Value certifikáty (certifikáty s tzv. „zeleným pruhem“).	Závazný	1
BEZ_11.0	Soulad se Zákonem o ochraně osobních údajů (Compliance – ZoOOÚ)	Systém SDAT splňuje bezpečnostní standardy pro data spadající pod zákon č.101/2000 Sb., o ochraně osobních údajů, ve znění pozdějších předpisů.	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
BEZ_12.0	Soulad se Zákonem o kybernetické bezpečnosti (Compliance – ZoKB)	Systém SDAT splňuje bezpečnostní požadavky pro významné informační systémy dané zákonem č. 181/2014 Sb., o kybernetické bezpečnosti, ve znění pozdějších předpisů.	Závazný	1
BEZ_13.0	Soulad se Zákonem o elektronickém podpisu (Compliance - ZoEP)	Systém SDAT umožňuje pracovat s uznávaným elektronickým popisem/uznávanou elektronickou značkou ve smyslu zákona č. 227/2000 Sb., ve znění pozdějších předpisů.	Závazný	1

3.3 Obecné nefunkční požadavky – Audit

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
AUD_1.0	Obsah auditního logu	Systém musí udržovat nezměnitelný auditní log, schopný automaticky zachytit a uložit údaje o: • všech operacích provedených v systému souborů a dat, • uživateli, který operaci iniciuje nebo provádí, • datu a času této události.	Závazný	1
AUD_2.0	Délka životnosti auditního logu	Systém musí udržovat auditní log bez časového omezení.	Závazný	1
AUD_3.0	Čitelnost auditního logu	Systém musí zajistit, aby údaje z auditního logu byly na požádání dostupné pro kontrolu uživatelům, kteří se systémem nejsou obeznámeni vůbec nebo jen málo.	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
AUD_4.0	Tisk auditních logů	Systém musí správci umožnit vytištění auditního logu.	Závazný	1
AUD_5.0	Manuální export auditních logů	Systém musí správci umožnit export auditního logu v čitelné podobě např. csv (comma-separated values).	Závazný	1
AUD_7.0	Auditní události (přiřazení přístupu)	Systém musí zaznamenávat auditní události řízení přístupu: • přidání oprávnění (např. zařazení uživatele na uživatelské místo), • odebrání oprávnění (např. odstraňování uživatele z uřivatelského místa nebo odebírání role od uživatelského místa a další podobné akce).	Závazný	1

AUD_8.0	Auditní události (řízení přístupu)	Systém musí zaznamenávat auditní události související s řízením přístupu:
		• logování přístupu funkcionalitě a datům, včetně generování reportů pro následný monitoring,
		• přístup k datům:
		  ○ o povolení přístupu (včetně toho, jaká akce byla provedena u daného objektu),
		  ○ o zamítnutí přístupu (včetně toho, jaká akce byla zamítnuta u daného objektu a z jakého důvodu).
			Závazný	1

AUD_9.0	Automatický export auditních logů	Systém musí umožnit automatický export strojově čitelných auditních logů v konfigurovatelný čas a do konfigurovatelného výstupního adresáře.	Závazný	1
AUD_10.0	Audit hromadných operací	V případě provedení hromadných operací musí být v auditním logu uchována informace i o této hromadné operaci s vazbou na jednotlivé dílčí akce.	Závazný	1
AUD_11.0	Přístupová práva	Systém bude evidovat práva uživatelů/skupin provádět určité operace	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
	– logování změn v oprávněních	(aktivity) a bude logovat změny v tomto nastavení a bude umožňovat prezentaci tohoto nastavení (včetně nastavení platného v minulosti).		
AUD_12.0	Přístupová práva – logování přístupu k citlivým údajům	Systém loguje všechny přístupy uživatelů k citlivým datům včetně operace čtení. V případě čtení je logován dotaz (včetně parametrů tohoto dotazu), ne výsledná data.	Závazný	1
AUD_13.0	Ochrana auditních záznamů	Auditní záznamy jsou chráněny před neoprávněným čtením, zápisem, nebo změnou.	Závazný	1
AUD_14.0	Přístupová práva – logování změn autentizačních údajů	Systém zaznamenává změnu údajů sloužících k autentizaci.	Závazný	1
AUD_15.0	Přístupová práva - logování	Systém zaznamenává pokusy o manipulaci s logovými záznamy a pokusy o změnu nastavení logování.	Závazný	1
AUD_16.0	Přístupová práva – logování neprovedených akcí	U všech auditovaných akcí se zaznamenávají i neúspěšné pokusy, např. z důvodu nedostatku oprávnění.	Závazný	1
AUD_17.0	Autentifikace – historie přihlášení	Systém sbírá a uchovává informace o historii přihlášení a odhlášení, zaznamenává i neúspěšné pokusy o přihlášení. V rámci tohoto požadavku sbírá následující informace: • časové razítko pokusu o přihlášení, • výsledek akce (úspěch/neúspěch), • IP adresu serveru stroje, ze kterého proběhl pokus o přihlášení, • informace o uživatelském účtu, pomocí kterého proběhl pokus o	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
		přihlášení.		
AUD_18.0	Audit – zjednodušený	Každý záznam uložený v databázi obsahuje následující informace: • informace o uživateli, který záznam vytvořil, • časové razítko vytvoření záznamu, • informace o uživateli, který záznam naposledy změnil, • časové razítko poslední změny záznamu. Výše uvedené údaje jsou viditelné v aplikaci v místě, kde se s daným záznamem pracuje, a jsou přístupné každému uživateli, který má oprávněním číst daný záznam.	Závazný	1
AUD_22.0	Logování – aplikační vrstva	Systém umožňuje logování činnosti systému s možnosti nastavit si úroveň logování s tím, že na nejvyšší úrovni (default pro produkci) budou logovány všechny zprávy typu FATAL, zatímco na nejnižší budou zaznamenány zprávy typu TRACE, což umožňuje krokovat činnost aplikaci. Součástí logování musí být výpis prováděných SQL příkazů, včetně hodnot parametrů, které byly v SQL příkazech použity. Systém umožňuje aplikačním rozhraním uživateli nastavit/změnit úroveň logování, případně ho úplně vypnout. Systém umožňuje přístup k logům přes aplikační rozhraní. Systém umožňuje definovat vyhrazené místo na disku, které mohou logy zabírat a definovat, co se má stát v případě, že toto místo bude zaplněno.	Závazný	1
AUD_23.0	Auditní log – legislativní správnost	Auditní záznam musí být v souladu se záznamovou povinností § 13 odst. 3 písm. c) zákona č. 101/2000 Sb.	Závazný	1
AUD_24.0	Auditní log – import do	Vybrané auditní záznamy musí být možné průběžně v reálném čase importovat do systému SIEM.	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
	systému SIEM			
AUD_25.0	Auditní log – legislativní správnost	Auditní záznam musí být v souladu s bezpečnostními požadavky pro významné informační systémy dané zákonem č. 181/2014 Sb., o kybernetické bezpečnosti, ve znění pozdějších předpisů.	Závazný	1
AUD_26.0	Auditní log - vyhledávání, filtrování, třídění.	Log musí umožňovat vyhledávání, filtrování, třídění záznamů.	Závazný	1

3.4 Obecné nefunkční požadavky – Provoz systému

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
PRV_1.0	Virtualizace	Všechny části systému musí být možné nasadit a provozovat ve virtualizovaném prostředí.	Závazný	1
PRV_2.0	Instalace klientských stanic	Pokud část systému SDAT vyžaduje instalaci na klientskou stanici, pak je možné instalaci této části systému provést vzdáleně bez nutnosti asistence koncového uživatele	Závazný	1
PRV_3.0	Provoz na klientské stanici	Systém musí být dostupný koncovému uživateli ze všech konfigurací klientské stanice standardního systémového prostředí ČNB.	Závazný	1
PRV_4.0	Provozní prostředí – záruka	V případě, že ČNB bude zajišťovat běžnou údržbu provozu aplikace, nebude to mít vliv na podmínky záruky dodávky.	Závazný	1
PRV_5.0	Slučitelnost	Systém musí zajistit slučitelnost s běžně dostupnými informačními technologiemi, například pomocí Webových služeb.	Závazný	1
PRV_6.0	Klientské virtuální	Desktop aplikace musí pracovat na terminálovém serveru Windows	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
	prostředí	s nadstavbou Citrix XAserver.		
PRV_7.0	Architektura – více prostředí na jednom serveru	Použitá architektura řešení nesmí zabraňovat možnosti instalovat na jeden fyzický aplikační server více různých prostředí (například musí být umožněno, aby na jednom fyzickém aplikačním serveru běželo jak cvičné, tak školicí prostředí).	Závazný	1
PRV_8.0	Výkon systému – příjem dat od Osob	Systém je schopen zpracovat Vstupní zprávu dle následujících provozních parametrů: • v době špičky (pracovní dny, období mezi 7:45 a 17:30) umožňuje systém zpracovávat 10 Vstupních zpráv paralelně (v jednom okamžiku) s tím, že se předpokládá, že velikost žádné ze zpracovávaných zpráv nepřekročí 500 MB. V takovém případě bude zpracování všech 10 Vstupních zpráv dokončeno nejpozději do 15 minut, • v době mimo špičku (nepracovní dny a pracovní dny, období mezi 17:30 a 7:45) umožňuje systém: o zpracovávat 10 Vstupních zpráv paralelně (v jednom okamžiku) s tím, že se předpokládá, že velikost žádné ze zpracovávaných zpráv nepřekročí 500 MB. V takovém případě bude zpracování všech 10 Vstupních zpráv dokončeno nejpozději do 5 minut, o zpracovávat 1 vstupní zprávu o maximální velikosti 5 GB s tím, že taková Vstupní zpráva bude zpracována nejpozději do 30 minut. Zprávy jsou bez ohledu na svoji velikost přijímány kdykoli, tedy bez časového omezení. Zpracování zpráv, které jsou větší než 500 MB, je automaticky odsunuto na dobu mimo špičku.	Závazný	1

3.5 Obecné nefunkční požadavky - Integrace s ostatními IS

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
INT_1.0	API	Systém musí být vybaven API (Application Programming Interface) pro programovací jazyky Java (verze 7) a/nebo C# (verze 4). API musí být vypracováno jako množina volatelných metod (umístěných ve třídách a případně balících/jednotkách) dostupných z vývojových prostředí uvedených jazyků. Rozsah API bude definován v Realizační studii.	Závazný	3
INT_2.0	Logický detail API funkcionalit	API musí především zapouzdřovat jednotlivé věcné funkcionality. Je požadováno maximální odstínění od technických procesů zpracování. Názvosloví metod API je voleno tak, aby bylo srozumitelné i pro uživatele z věcné oblasti.	Závazný	3
INT_3.0	Dokumentace API	Dokumentaci API požadujeme pro programovací jazyk Java ve formátu JavaDoc a pro C# ve formátu HTML Help. Požaduje se generovaná dokumentace dle zdrojového kódu. Pro každou dokumentovanou entitu (objekt) je požadováno zdokumentovat: • výstižný popis dokumentované entity (objektu), • pro vlastnosti povinně popis významu hodnot, kterých mohou nabývat (konstanty, rozsahy), • pro metody volitelně, pro třídy a balíky/jednotky pak povinně podrobný popis užití a příklady implementace.	Závazný	3

3.6 Obecné nefunkční požadavky – Migrace dat

Předmětem migrace jsou data uložená v databázi stávajícího systému MTS-ISL-SÚD-SDNS (dále jen „MTS“) a vstupní zprávy (soubory) zasílané vykazujícími osobami do ČNB k následnému zpracování.

Pod pojmem „migrace dat“ jsou zahrnuty tyto akce a procesy:

a) analýza zdrojové (MTS) a cílové struktury (SDAT), transformační mapování mezi strukturami - analýzu a přípravu procesu zajišťuje dodavatel systému SDAT ve spolupráci s dodavatelem stávajícího systému MTS-ISL-SÚD a objednatelem. Objednatel zajistí součinnost dodavatele stávajícího systému,
b) export objednatelem určených dat a metadat z IS MTS pro SDAT,
c) import vyexportovaných dat a metadat z MTS do SDAT - zhotovitel SDAT musí poskytnout SW nástroje pro import vyexportovaných dat a metadat z MTS. Po importu musí být proveden test komplexnosti migrovaných data a metadat, jejich správnosti a splnění integritních pravidel struktur nového SDAT,
d) post-migrační polo-automatizované nebo manuální customizace dat a nastavení SDAT - lze předpokládat, že některé informace, které bude vyžadovat datový model SDAT, se v MTS nevyskytují, a bude je nutné pro SDAT doplnit default hodnotami, případně realizovat určité post-migrační postupy.

Lze předpokládat, že procesy a) b) a c) d) budou opakovány několikrát v průběhu realizace a to minimálně před každým dílčím plněním a před zahájením ověřovacího provozu.

Z hlediska obsahu migrace lze obsah MTS rozdělit na:

a. Data – hodnoty údajů, kromě jejich iniciální migrace (přenesení obsahu ke zvolenému okamžiku) se předpokládá i proces další inkrementální synchronizace v případě souběhu provozu obou systémů.
b. Metadata – metapopis který definuje jednotlivé výkazy, registr vykazujících osob, vykazovací povinnosti.
c. Provozní a stavové informace – zejména informace o stavu a zpracování vstupních souborů propagované až na úroveň stavů jednotlivých hodnot údajů a plnění vykazovacích povinností. V případě synchronizace dat (hodnot údajů) mezi oběma systémy jsou synchronizována i tato data.

Lze předpokládat, že migraci bude možno z hlediska obsahu provádět buď kompletně, nebo po definovaných věcných celcích v rámci přírůstků dodané funkcionality SW řešení.

Předpokládaným objemem migrovaných dat a metadat jsou v případě MTS desítky GB.

V oblasti migrace vstupních zpráv (souborů) zasílaných vykazujícími osobami je požadována plně automatizovaná migrace bez ručních nebo polo-automatizovaných zásahů.

Pokud zhotovitel bude implementovat i požadavky, které jsou v zadání označeny jako „vítané“, musí také splnit odpovídající požadavek na migraci údajů dané oblasti (daného požadavku).

Samostatnou problematikou v oblasti migrace je přesun dat uložených v systému MKT (Monitoring kapitálového trhu). MKT je aplikace vyvinutá a provozovaná v ČNB a obsahuje data sebraná pomocí systému MTS, ale uložena přímo do dedikovaného úložiště systému MKT, které je strukturou odlišné od úložiště systému MTS. V případě analýzy a realizace migrace těchto dat poskytuje součinnost zadavatel. Předpokládaným objemem migrovaných dat z MKT jsou jednotky TB.

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
MIG_1.0	Migrace – opakovatelnost	Je zajištěna migrace metadat i dat z původního systému a tuto migraci lze provádět opakovatelně. Systém tak disponuje importním modulem, který umožňuje opakované načítání externích dat a metadat, a jejich import do systému SDAT. V případě, že nelze data migrovat automatizovaně (v důsledku změny filozofie nového systému), pak systém umožňuje metadata/data před provedením doplnit/modifikovat (viz MIG_3.0) Importní modul provádí v rámci procesu importu dat předepsané byznys kontroly.	Závazný	1
MIG_2.0	Migrace – Výkazy - per partes	Je umožněno, aby bylo možno migrovat metadata i data (včetně kompletní historie) za vybraný Výkaz.	Závazný	1
MIG_3.0	Migrace – doplnění/úprava hodnot	Proces migrace umožňuje doplňovat chybějící data či modifikaci existujících dat, tak aby mohla být migrace provedena.	Závazný	1

3.7 Obecné nefunkční požadavky - Provozní prostředí

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
PPR_1.0	Produkční prostředí	V rámci architektury systému SDAT existuje produkční prostředí	Závazný	2

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
		dostupné ze sítě Internet a současně přímo ze systémového prostředí ČNB (viz kapitola 2.1.1 Produkční prostředí).		
PPR_1.1	Produkční prostředí – geocluster	Architektura systému SDAT umožňuje provozovat produkční prostředí v rámci tzv. geoclusteru. To znamená, že systém běží ve dvou různých geografických lokalitách s možností přepínání se mezi těmito prostředími bez výpadku dostupnosti systému (přepínání mezi prostředími je zajištěno funkcionalitou standardního systémového řešení).	Závazný	2
PPR_2.0	Testovací prostředí	V rámci architektury systému SDAT existuje testovací prostředí dostupné ze sítě Internet a současně přímo ze systémového prostředí ČNB (viz kapitola 2.1.2 Testovací prostředí).	Závazný	2
PPR_2.1	Testovací prostředí – proces synchronizace metapopisu (stavy Schválený a Platný)	Systém SDAT umožňuje synchronizaci obsahu produkčního a testovacího prostředí pro instance objektů ve stavu Platný a Schválený podle popisu v kapitole 2.1.2 Testovací prostředí body i. a ii. Systém SDAT umožňuje proces synchronizace spouštět automaticky v naplánovaných periodách nebo ad-hoc na pokyn uživatele.	Závazný	2
PPR_2.2	Testovací prostředí - geoclustrer	Architektura systému SDAT umožňuje provozovat testovací prostředí v rámci tzv. geoclusteru. To znamená, že systém běží ve dvou různých geografických lokalitách s možností přepínání se mezi těmito prostředími bez výpadku dostupnosti systému (přepínání mezi prostředími je zajištěno funkcionalitou standardního systémového řešení).	Závazný	2
PPR_2.2	Testovací prostředí -přesun metapopisu (stav Projektovaný)	Systém SDAT umožňuje provádět přesun uživatelem vybraných instancí objektů z produkčního prostředí na testovací prostředí podle popisu v kapitole 2.1.2 Testovací prostředí, bod iii. Systém SDAT umožňuje proces přesunu metapopisu spouštět na ad-hoc	Závazný	2

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
		pokyn uživatele.		
PPR_2.3	Testovací prostředí – synchronizace Registru osob	Systém SDAT umožňuje provádět synchronizaci obsahu Registru osob mezi produkčním a testovacím prostředím podle popisu v kapitole 2.1.2 Testovací prostředí. Systém SDAT umožňuje proces synchronizace spouštět automaticky v naplánovaných periodách nebo na ad-hoc na pokyn uživatele.	Závazný	2
PPR_3.0	Cvičné prostředí	V rámci architektury systému SDAT existuje cvičné prostředí dostupné v rámci systémového prostředí ČNB (viz kapitola 2.1.3 Cvičné prostředí).	Závazný	3
PPR_3.1	Cvičné prostředí – synchronizace	Systém umožňuje provádět synchronizace obsahu cvičných a produkčních instancí komponent. Systém SDAT umožňuje spouštět proces synchronizace na ad-hoc pokyn uživatele.	Závazný	3
PPR_4.0	Školicí prostředí	V rámci architektury systému SDAT existuje školicí prostředí dostupné v rámci systémového prostředí ČNB (viz kapitola 2.1.4 Školicí prostředí).	Závazný	3
PPR_4.1	Školicí prostředí – synchronizace	Systém umožňuje provádět synchronizaci obsahu školicích a produkčních instancí komponent. Systém SDAT umožňuje spouštět proces synchronizace na ad-hoc pokyn uživatele.	Závazný	3
PPR_4.0	Akceptační prostředí	V rámci architektury systému SDAT existuje akceptační prostředí dostupné v rámci systémového prostředí ČNB (viz kapitola 2.1.5 Akceptační prostředí).	Závazný	2
PPR_4.1	Akceptační prostředí -	Pokud charakter provedených programových změn dovolí (např. nedojde k zásadní změně datového modelu databázové části systému),	Závazný	2

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
	synchronizace	systém umožňuje provedení synchronizace obsahu z produkčních do akceptačních instancí komponent a to na ad-hoc bázi.		
PPR_5.0	Vývojové prostředí	V rámci architektury systému SDAT existuje vývojové prostředí dostupné pouze ze systémového prostředí ČNB. Je požadováno, aby v rámci tohoto prostředí bylo možno: • Po dobu fáze realizace systému bude dodavatel na toto prostředí nasazovat tzv. noční sestavení systému (tzv. nightly builds). • Pod dobu fáze realizace systému bude dodavatele na toto prostředí nasazovat tzv. stabilní sestavení (tzv. stable builds). Periodicita nasazování těchto stabilních sestavení by neměla přesáhnout jeden měsíc (výjimkou může být počáteční fáze projektu).	Závazný	1

3.8 Obecné nefunkční požadavky - Uživatelské rozhraní

Obecné nefunkční požadavky na uživatelské rozhraní popisují očekávané chování jednotlivých základních komponent uživatelského rozhraní. Výčet prvků uživatelského rozhraní není kompletní, to však neznamená, že není povoleno použít jiné komponenty uživatelského rozhraní. V případě, že dodavatel použije takový ovládací prvek, který není upraven žádným nefunkčním požadavkem, očekává se od takového prvku chování „obvyklé“, tedy například známé z operačního systému Windows. Pokud se bude jednat o ovládací prvek, který není obvykle použit v žádném z obvykle dostupných systémů, pak je dodavatel povinen chování takového prvku popsat v dokumentaci pro uživatele.

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
URO_1.0	Indikace činnosti aplikace	V případě, že systém provádí akci, jejíž výsledek není okamžitý (akce neskončí do 1 sekundy od jejího započetí), bude svou činnost indikovat stylem, kdy uživatel pozná, že aktuálně systém zpracovává jeho	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
		požadavek.		
URO_2.0	Indikace činnosti aplikace v případě dlouhotrvajících akcí (progress bar)	V případě, že systém provádí akci, která je dlouhotrvající (z principu prováděné operace lze předpokládat, že akce nebude dokončena do 10 sekund od okamžiku, kdy ji uživatel zahájil, typicky upload velkého souboru na server, zpracování velkého objemu dat), bude svou činnost indikovat způsobem, ze kterého bude patrné, jaká část celku již byla provedena a jakou část celé akce je ještě třeba provést, než bude požadavek kompletně vyřízen. Informace o dokončené části operace bude v pravidelných intervalech obnovována až do úplného vyřízení celé operace.	Závazný	1
URO_3.0	Uživatelské nastavení (uživatelská konfigurace)	Aplikace bude trvale (trvale, tj. mimo PC uživatele) ukládat změny v nastavení uživatelského rozhraní, které uživatel provede během práce se systémem. Typicky se tímto myslí např. pořadí sloupců v tabulce dat, velikost jednotlivých sloupců, výchozí řazení, ale i jiné preference uživatele a bude toto nastavení pro přihlášeného uživatele zohledňovat před defaultním nastavením aplikace.	Závazný	1
URO_4.0	Ovládací prvky	Všechny prvky aplikace, které provádějí nějakou akci (tlačítka, hypertextové odkazy, významové ikony), budou po najetí kurzoru nad daný ovládací prvek: • demonstrovat uživateli, že se jedná o ovládací prvek změnou kurzoru myši, • zobrazí nápovědu k danému ovládacímu prvku formou tooltipu.	Závazný	1
URO_5.0	Povinné hodnoty	Pokud bude na formuláři nějaká hodnota povinná (bez ohledu na to, zda se jedná o zadávací pole, rozbalovací seznam či jiný prvek), bude tato skutečnost uživateli jasně prezentována.	Závazný	1
URO_6.0	Klávesové zkratky	Aplikace umožňuje často prováděné akce realizovat pomocí klávesových zkratek.	Závazný	1
URO_7.0	Validace dat –	Systém validuje vstupní data uživatele na správnost zadaného typu dat	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
	základní	bez nutnosti kontaktovat server. Zejména se jedná o tyto validace: • pro textové pole validuje maximální délku zadávaného řetězce, • pro datumové pole validuje správnost zadávaného data (například jako nevalidní označí datum 29. 2. 2013, ale i 5.18.2014), • do číselného pole neumožňuje zadat pro číslo nerelevantní znaky, • do rozbalovacího seznamu neumožňuje zadat jinou hodnotu, než je v seznamu hodnot. Splněním tohoto požadavku nejsou dotčeny další validace, které vyplývají z aplikační logiky dané oblasti aplikace. Pokud je nějaká validace porušena, systém upozorní uživatele konkrétním (nikoli obecným) hlášením. Hlášení tedy je například „Překročena povolená délka pro „Jméno“ [Povoleno: 255, Zadáno: 318]” a nebude znít “Nevyhovující formát pole”.		
URO_8.0	Validace dat – specifické	Kromě základního způsobu validace na formát zadávaných data systém disponuje dalšími specifickými validacemi: • pokud se se v uživatelském rozhraní vyskytují dva datumy, které navzájem vymezují časový úsek od-do (datumy tvoří dohromady logický celek), systém označí za nevalidní takové zadání, kdy je „datum od“ větší než „datum do“, • u polí typu INTREGER systém označí za nevalidní vstup, který sice obsahuje číslo, ale zároveň obsahuje desetinnou část, • u polí typu DECIMAL systém označí za nevalidní vstup, který je mimo rozsah (větší počet míst před nebo za desetinnou čárkou). Pokud budou výše uvedená pravidla porušena, systém takový vstup označí jako chybný a neumožňuje taková data uložit do databáze. U polí typu EMAIL proběhne validace na zadání platné e-mailové adresy pomocí regulárního výrazu. V případě porušení validátoru však bude následovat pouze varování, nikoli chyba.	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
URO_9.0	Zadávací pole pro datum - maska	Zadávací pole pro datum disponuje maskou, která umožňuje zadání data ve formátu DD.MM.RRRR, aniž by uživatel musel psát oddělovací tečky mezi jednotlivými údaji. Zadání „07062014“ je tak správné. Zadání „29022013“ (únor roku 2013 nemá 29 dní) je nevalidní, stejně tak jako „05312014“ (přehozeny měsíce a dny). Zároveň platí, že pokud uživatel chce zadat datum včetně oddělovacích teček, systém mu to umožňuje. V případě, že uživatel zvolí způsob zadání včetně oddělovacích teček, není třeba psát vodící nuly u čísel dní a měsíců. Správné je tedy i zadání „7.6.2014“.	Závazný	1
URO_9.1	Zadávací pole pro datum a čas - maska	Zadávací pole pro datum a čas disponuje maskou, která umožňuje zadání data a času ve formátu DD.MM.RRRR H24:MM:SS, aniž by uživatel musel psát oddělovací tečky a dvojtečky mezi jednotlivými údaji.	Závazný	1
URO_10.0	Zadávací pole pro datum – výběr hodnoty z kalendáře	Zadávací pole pro datum obsahuje možnost vybrat datum z kalendáře. Za tím účelem je k poli připojena komponenta kalendáře, která se zobrazí až po kliknutí uživatele na ovládací prvek k rozbalení kalendáře určený. V případě, že je v poli pro datum nějaké validní datum, je kalendář nastaven na toto datum. Pokud v poli pro datum žádné datum zadáno není, je kalendář nastaven na aktuálně platné datum (pokud není v konkrétním funkčním požadavku řečeno jinak).	Závazný	1
URO_10. 1	Zadávací pole pro datum a čas – výběr hodnoty z kalendáře	Zadávací pole pro datum a čas obsahuje možnost vybrat datum a čas z kalendáře. Za tím účelem je k poli připojena komponenta kalendáře a času, která se zobrazí až po kliknutí uživatele na ovládací prvek k rozbalení kalendáře určený.	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
		V případě, že je v poli pro datum nějaké validní datum a čas, je kalendář nastaven na toto datum a čas. Pokud v poli pro datum žádné datum a čas zadáno není, je kalendář nastaven na aktuální čas a datum (pokud není v konkrétním funkčním požadavku řečeno jinak).		
URO_11.0	Zadávací pole pro heslo	V případě, že bude nutné na nějakém místě aplikace zadat uživatelem heslo, pak systém místo reálných znaků zobrazí v tomto poli znaky zástupné tak, aby nebylo možno heslo přečíst neoprávněným uživatelem.	Závazný	1
URO_12.0	Odesílání nevalidních dat ke zpracování	Systém zamezí odesílání dat ke zpracování v případě, že nebyly splněny všechny validační podmínky, tj. jednak validační podmínky popsané v URO_7.0 a URO_8.0 a jednak validační podmínky vyplývající z aplikační logiky konkrétní oblasti. Systém uživateli jasně sdělí, z jakých důvodů nelze data odeslat. Tento způsob validace dat neznamená, že je možno provádět kontroly pouze ve vrstvě uživatelského rozhraní. Aplikace musí provádět veškeré předepsané kontroly i na úrovni aplikační vrstvy v rámci procesu persistence dat.	Závazný	1
URO_13.0	Potvrzení provedené akce	V případě, že uživatel provede akci, která nějakým způsobem modifikuje data nebo provede jinou akci, která sice data nemodifikuje, ale má smysl o jejím výsledku informovat uživatele, a systém tuto akci provede a úspěšně dokončí, zobrazí systém uživateli informační hlášení, ze kterého je patrné, jaká akce byla provedena a že byla úspěšně dokončena. Jedná se o zobrazení informačního hlášení, kterým je uživateli sděleno, že akce, kterou provedl, byla úspěšně dokončena. Zároveň systém umožňuje uživateli prohlásit, že v budoucnu pro danou akci již nechce podobná hlášení zobrazovat. Pokud uživatel tuto volbu zaškrtne, při další akci už nebude hlášení zobrazeno.	Závazný	1
URO_14.0	Potvrzení provedené akce –	Informace o tom, že se dané hlášení (viz URO_13.0) již nebude pro danou akci v budoucnu zobrazovat, je uložena v uživatelské konfiguraci,	Důležitý	2

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
	možnost měnit preference	která je přístupná uživateli a uživatel může kdykoli svoji preferenci změnit (může si opět nastavit, že potvrzovací dialog zobrazovat pro určitou akci chce).		

3.8.1 Komponenta Tabulka dat (grid)

Komponenta Tabulka dat (grid) je použita pro prezentaci a manipulaci s daty tabulárního charakteru.

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
GRI_1.0	Zobrazení velkého objemu dat	Pokud počet řádků v tabulce přesahuje místo, které je v uživatelském rozhraní vyhrazeno pro tabulku dat, budou zobrazeny ty záznamy, které se do vyhrazeného místa vejdou a bude zobrazen vertikální posuvník, který umožňuje zobrazit i další záznamy. Záznamy, které se nevejdou, budou do tabulky dat, budou načteny až v okamžiku, kdy si je uživatel vyžádá (například posunutím posuvníku). Během donačítání dat ze serveru bude uživateli zobrazena indikace činnosti aplikace (viz URO_1.0).	Závazný	1
GRI_2.0	Zobrazení počtu záznamů	U každé tabulky dat bude zobrazena informace o celkovém počtu záznamů, které tabulka obsahuje bez ohledu na to, kolik z těchto záznamů je aktuálně viditelných/odfiltrovaných, tj. celkový počet záznamů. V případě, že je tabulka dat uživatelem filtrována (viz GRI_8.0), pak je zobrazena i informace o počtu záznamů, které vyhovují zadaným filtrovacím podmínkám.	Závazný	1
GRI_3.0	Pořadí sloupců	Každá tabulka dat bude zobrazovat defaultní sloupce s tím, že uživatel si	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
		bude moci toto pořadí přetažením myší změnit. Informace o změnách v pořadí sloupců je zaznamenána do uživatelské konfigurace a je upřednostněna před defaultním nastavením (viz URO_3.0).		
GRI_4.0	Skrývání a přidávání sloupců	Pokud tabulka dat obsahuje příliš mnoho sloupců, budou některé sloupce v defaultním zobrazení skryty. Uživatel bude moci kdykoli jakýkoli relevantní sloupec skrýt nebo naopak přidat. Systém bude toto nastavení uživatele uchovávat v tzv. uživatelských nastaveních (viz URO_3.0) a při příštím přihlášení nabídne uživateli zobrazení podle jeho posledního nastavení.	Závazný	1
GRI_5.0	Šířka sloupců	Každá tabulka dat bude zobrazovat sloupce s defaultní šířkou každého sloupce. Uživatel může pomocí myši změnit šířku sloupce dle svých potřeb. Systém bude toto nastavení uživatele uchovávat v tzv. uživatelských nastaveních (viz URO_3.0) a při příštím přihlášení nabídne uživateli zobrazení podle jeho posledního nastavení.	Závazný	1
GRI_6.0	Zobrazení hodnoty sloupce, která je delší než šířka sloupce	Pokud je v tabulce dat v nějakém sloupci prezentována hodnota, která je delší, než je šířka sloupce, bude zobrazena jen ta část hodnoty, pro kterou je místo. Po najetí kurzorem myši na danou hodnotu bude zobrazena hodnota celá s využitím tzv. tooltipu.	Závazný	1
GRI_7.0	Řazení dat	Data zobrazená v tabulce budou defaultně seřazena podle právě jednoho sloupce (pokud sloupec obsahuje data, které lze z jejich povahy řadit). Pokud je předmětem řazení sloupec, který obsahuje textovou hodnotu, pak je řazení vykonáno podle té znakové sady, kterou má uživatel právě zvolenou. Viz NFP_28.0. Pokud je hodnota sloupce, podle kterého je prováděno řazení, rovna NULL, pak je takový záznam zařazen na konec seznamu. Uživatel může kdykoli toto řazení změnit kliknutím na záhlaví	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
		jakéhokoli zobrazeného sloupce. První kliknutí znamená, že hodnoty ve sloupci budou seřazeny vzestupně (A-Z, 0-9), druhé kliknutí na záhlaví sloupce změní řazení na sestupné (Z-A, 9-0), každé další kliknutí změní směr řazení podle výše uvedeného vzoru. Pokud jsou data ve sloupci řazena podle nějakého sloupce, pak je v záhlaví tohoto sloupce vykreslena ikona, která naznačuje, že data v tabulce jsou seřazena podle daného sloupce a navíc indikuje směr (vzestupně/sestupně), kterým jsou data seřazena. Systém bude toto nastavení uživatele uchovávat v tzv. uživatelských nastaveních (viz URO_3.0) a při příštím přihlášení nabídne uživateli zobrazení podle jeho posledního nastavení.		
GRI_8.0	Filtrování dat	Systém umožňuje data v tabulce filtrovat. Filtrovat lze podle jakéhokoli zobrazeného sloupce. Systém umožňuje filtrovat data podle podmínek zadaných nad více sloupci. Pokud uživatel zadá více jak jedno filtrovací kritérium, má se za to, že mezi filtrovacími kritérii je použit logický operátor AND.	Závazný	1
GRI_9.0	Filtrování dat textového charakteru	Pokud uživatel zadává filtr nad sloupcem s daty textového charakteru, pak systém vybere všechny záznamy, které v daném sloupci obsahují řetězec, který odpovídá zadanému filtrovacímu kritériu (ekvivalentní k SQL konstrukci nazev_sloupce like '%<filtrovací kritérium>%'.	Závazný	1
GRI_10.0	Filtrování dat číselného a datumového charakteru	Pokud uživatel zadává filtr nad sloupcem s daty číselného nebo datumového charakteru, je možno použít operátory: • větší než „>“ • menší než „<“ • rovno nebo větší než „>=“ • rovno nebo menší než „<=“ • rovná se „=“ • nerovná se „!=“	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
		V případě, že uživatel žádný operátor nepoužije, má se za to, že použil operátor „rovná se“.		
GRI_11.0	Zobrazení informace o aplikovaném filtru a možnost filtr zrušit	V případě, že uživatel aplikoval filtr a došlo k omezení celkového počtu řádků zobrazovaných v tabulce dat, bude uživateli zobrazena výrazná informace o tom, že data jsou filtrována. Zároveň systém umožňuje z jednoho místa vypnout všechny aplikované filtry.	Závazný	1
GRI_12.0	Prezentace multimediálního obsahu	Tabulka dat umožňuje prezentaci multimediálního obsahu (například binárního souboru) formou grafické ikony, která umožňuje uživateli zahájit proces stahování daného binárního souboru na disk uživatele. Sloupec obsahující binární data, resp. zástupné ikony nepodléhá řazení (GRI_7.0 pro něj neplatí)	Závazný	1
GRI_13.0	Editace dat přímo v tabulce dat	Tabulka dat standardně prezentuje data v režimu pro čtení, nicméně aplikace poskytuje nástroj, který umožňuje uživateli editovat data právě jednoho vybraného řádku přímo v tabulce dat (bez nutnosti spouštět další formulář).	Důležitý	2
GRI_14.0	Export dat	Tabulka dat umožňuje export dat v tabulce zobrazených do následujících formátů: • TXT • CSV • XML • XLS (XLSX) • PDF	Závazný	1
GRI_15.0	Aktualizace dat	Tabulka dat umožňuje provést aktualizaci (znovunačtení) dat uživatelem v případě, že se data změnila způsobem, který není aplikace schopna zaznamenat (například updatem dat přímo v databázi pomocí SQL příkazu, editací záznamu jiným uživatelem).	Závazný	1

3.8.2 Komponenta Rozbalovací seznam (combobox)

Komponenta, která umožňuje uživateli vybrat z omezené množiny hodnot jednu, případně více hodnot (v závislosti na aplikační logice). Většinou je použita pro prezentaci číselníkových dat.

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
CBX_1.0	Možnosti výběru hodnot	Rozbalovací seznam umožňuje: • výběr žádné nebo právě jedné hodnoty ze seznamu dostupných hodnot, • výběr žádné, jedné nebo více hodnot ze seznamu dostupných hodnot.	Závazný	1
CBX_2.0	Režim práce s hodnotami	Rozbalovací seznam pracuje v režimu: • čtení: uživatel nemůže ovlivnit rozsah nabízených hodnot a může pouze vybrat nějakou existující hodnotu, • čtení a zápis: uživatel buď může vybrat nějakou existující hodnotu anebo zapsat hodnotu neexistující, kterou pak systém zařadí do seznamu dostupných hodnot.	Závazný	1
CBX_3.0	Počet sloupců	Rozbalovací seznam umožňuje zobrazit více jak jeden sloupec. Tedy například u číselníku měn může být požadováno prezentovat jak název měny, tak její kód, tj. např.: • EUR – Euro • CZK – Česká koruna	Závazný	1
CBX_4.0	Zobrazované hodnoty	Rozbalovací seznam umožňuje kromě textových informací prezentovat v řádcích seznamu i grafické prvky (například pokud by rozbalovací seznam prezentoval číselník zemí, pak by u každé země byla ikona zobrazující její vlajku).	Důležitý	1
CBX_5.0	Filtrování hodnot	Rozbalovací seznam umožňuje uživateli filtrovat data v seznamu obsažená pomocí zadání řetězce. Vybrány budou ty řádky rozbalovacího seznamu, kde data alespoň v jednom z prezentovaných sloupců odpovídají zadanému filtrovacímu kritériu (ekvivalentní ke konstrukci	Závazný	1

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
		like '%<filtrovací kritérium>%'.		
CBX_6.0	Vymazání vybrané hodnoty	Systém umožňuje uživateli odstranit vybranou hodnotu a nastavit rozbalovací seznam do stavu, kdy nemá žádnou hodnotu přiřazenu (tím není dotčena validace rozbalovacího seznamu; pokud je označeno, že je hodnota pro dané pole povinná, nevyhoví takový ovládací prvek validaci, viz URO_5.0).	Závazný	1
CBX_7.0	Návratová hodnota	Rozbalovací seznam umožňuje vrátit jinou hodnotu, než která je uživateli zobrazena. Typicky v případě výběru z číselníku měn uživatel vybere měnu „Česká koruna", ale rozbalovací seznam vrátí aplikaci ID této číselníkové položky (buď systémové ID a nebo kód měny, tedy „CZK")	Závazný	1

4 Katalog funkčních požadavků

4.1 Katalog funkčních požadavků pro Komunikační modul

ID požadavku	Název Požadavku	Popis požadavku	Důležitost	Kategorie
KOM_1.0	Vytvoření úkolu	Systém umožňuje přímo z aplikace vytvoření nového požadavku (úkolu) minimálně v následujícím rozsahu: • název úkolu, • slovní popis úkolu,	Závazný	3

ID požadavku	Název Požadavku	Popis požadavku	Důležitost	Kategorie
		• priorita úkolu (číselník), • datum požadovaného vyřešení úkolu, • řešitel, • sledovatelé, • přílohy (0..N binárních souborů).		
KOM_1.1	Výběr řešitele/sledovatele	Systém umožňuje při vytváření nového požadavku (úkolu) přímo z aplikace vybrat řešitele požadavku (právě jednoho) a vybrat sledovatele požadavku (žádného, jednoho nebo více) tak, že v kontextu akce, kterou uživatel zrovna provádí, systém nabídne takový seznam uživatelů, kteří mohou být řešiteli/sledovateli pro danou akci. Seznam uživatelů je dán tím, kteří uživatelé mají právo provést danou schvalovací (nebo jinou související) akci. Jiný seznam uživatelů tak bude například nabídnout v případě, že je požadováno schválení nového číselníku a jiný například v případě schválení výkazu.	Závazný	3
KOM_2.0	Seznam úkolů	Systém zobrazuje každému uživateli seznam požadavků (úkolů), které tento uživatel buď vytvořil nebo kde je daný uživatel uvedený jako sledovatel nebo kde je daný uživatel uvedený jako řešitel s tím, že tyto tři skupiny požadavků (úkolů) jsou od sebe vizuálně odděleny. Barevně jsou v seznamu zvýrazněny požadavky (úkoly), které mají být k datu a času zobrazení seznamu vyřešeny (aktuální datum a čas je větší než datum požadovaného vyřešení úkolu) a daný požadavek (úkol) je ve stavu „20 – Probíhá řešení“. Požadavky (úkoly) jsou čerpány přímo z nástroje pro evidenci požadavků (úkolů) přes API.	Závazný	3
KOM_2.1	Seznam úkolů – filtrování	Systém umožňuje filtrovat zobrazené požadavky (úkoly) dle KOM_2.0 minimálně v tomto rozsahu:	Závazný	3

ID požadavku	Název Požadavku	Popis požadavku	Důležitost	Kategorie
		• dle stavu (s tím, že je možné, aby uživatel zadal jeden nebo více stavů; viz tabulka stavů výše), • dle řešitele, • dle priority, • dle data požadovaného vyřešení úkolu, • k aktuálnímu datu nevyřešené úkoly.		
KOM_3.0	Seznam úkolů – akce k provedení	Systém pro každý jeden vybraný požadavek (úkol) umožňuje provést tyto akce: • změnit stav požadavku (úkolu) na jiný v souladu s tabulkou č. 1 Přehled stavů zpracování požadavku (úkolu). Při změně systém povinně vyžaduje zadání komentáře (jedinou výjimkou je přechod ze stavu 30 – Vyřešený do stavu 40 – Ukončený, kdy komentář není třeba uvádět), • předat požadavek (úkol) jinému řešiteli (tato akce je dostupná pouze ve stavech 10 – Vytvořený, 20 – Probíhá řešení a 25 – Vyžaduje doplnění, • stornovat požadavek, který daný uživatel vytvořil v případě, že je tento požadavek ve stavu, který umožňuje stornování	Závazný	3
KOM_4.0	Notifikační e-maily	Systém (task management systém) zajišťuje odesílání notifikačních e-mailů při každé změně stavu konkrétního požadavku – notifikační e-maily jsou odesílány na zadavatele, řešitele a všechny sledovatele.	Závazný	3
KOM_5.0	Eskalační e-maily	Systém (task management systém) zajišťuje odesílání eskalačních e-mailů na adresu řešitele v těchto případech: • blíží se datum požadovaného vyřešení a požadavek není ve stavech 30 – Vyřešený, 40 - Ukončený nebo 90 – Stornovaný. Systém umožňuje definovat různé časové okamžiky, ve kterých	Závazný	3

ID požadavku	Název Požadavku	Popis požadavku	Důležitost	Kategorie
		se budou tyto eskalační e-maily odesílat, • bylo překročeno datum požadovaného vyřešení a požadavek není ve stavech 30 – Vyřešený, 40 - Ukončený nebo 90 – Stornovaný.		
KOM_6.0	Automatické ukončení požadavku (úkolu)	Systém (task management systém) zajišťuje automatické ukončení požadavku úkolu – automaticky bude ukončen takový požadavek (úkol), který je ve stavu 30 - Vyřešený a nebyl do X dní přesunut zadavatelem do stavu 40 – Ukončený. X je v tomto případě počet dní a toto je konfigurovatelné. V případě automatického ukončení požadavku (úkolu) je zaslán e-mail pouze zadavateli.	Závazný	3
KOM_7.0	Notifikace při změně řešitele	Systém (task management systém) odešle notifikační e-mail v případě, že během životního cyklu požadavku (úkolu) dojde ke změně řešitele (viz KOM_3.0).	Závazný	3
KOM_8.0	Nastavení notifikačních a eskalačních e-mailů	Systém (task management systém) umožňuje definovat příjemce/skupiny příjemců pro notifikační a eskalační e-maily zasílané na základě KOM_4.0 a KOM_5.0. Zároveň umožňuje definovat (a v čase měnit) text notifikačního/eskalačního e-mailu.	Závazný	3

4.2 Administrační modul

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
AMS_1.0	Administrační	Systém obsahuje administrační modul aplikace jako centrální bod pro	Závazný	3

ID požadavku	Název požadavku	Popis požadavku	Důležitost	Kategorie
	modul	správu systému.		
AMS_2.0	Monitorování procesů	Administrační modul systému obsahuje uživatelský nástroj pro monitorování procesů systému, viz kapitola 2.4.1 Monitorování procesů.	Závazný	3
AMS_3.0	Správa systémových proměnných	Administrační modul systému obsahuje uživatelský nástroj pro správu systémových proměnných, viz kapitola 2.4.2 Systémové proměnné.	Závazný	3
AMS_4.0	Monitorování aktivity uživatelů	Administrační modul systému obsahuje uživatelský nástroj pro monitorování aktivity uživatelů, viz kapitola 2.4.3 Monitorování aktivity uživatelů.	Závazný	3
AMS_4.1	Monitorování aktivity uživatelů – uvolňování uživatelských zámků Výkazů	Systém umožňuji uživateli v rámci Monitorování aktivit uživatelů uvolňovat uživatelské zámky nad projektovanými částmi Výkazů.	Závazný	3
AMS_5.0	Referenční informace	Administrační modul systému obsahuje jednotný přístupový bod k obsahu a správě systémovým číselníků.	Závazný	3
AMS_6.0	Monitorování stavu systému	Administrační modul obsahuje jednotné místo pro sledování stavu systémového prostředí, které uživatelsky prezentuje informace o aktuálních transakčních zámcích databázových objektů, zaplnění databázových prostorů, zaplnění aplikačních disků apod.	Závazný	3