

Věcné zadání projektu SDAT

A - Obecné požadavky

Obsah

| | | |
|-------|---|----|
| 1 | Úvod..... | 4 |
| 2 | Obecná pravidla a funkcionality | 4 |
| 2.1 | Provozní prostředí..... | 4 |
| 2.1.1 | Produkční prostředí | 4 |
| 2.1.2 | Testovací prostředí | 5 |
| 2.1.3 | Cvičné prostředí | 8 |
| 2.1.4 | Školící prostředí | 9 |
| 2.1.5 | Akceptační prostředí | 10 |
| 2.1.6 | Vývojové prostředí | 10 |
| 2.2 | Sledování historie instancí objektů..... | 10 |
| 2.2.1 | Číslo verze a varianty instance objektu | 11 |
| 2.2.2 | Objektový model pro sledování historie instancí objektů | 12 |
| 2.2.3 | Přístup „Bez sledování historie“ | 14 |
| 2.2.4 | Přístup „Bez sledování historie – stavy“ | 14 |
| 2.2.5 | Přístup „Sledování historie – časová platnost“ | 15 |
| 2.2.6 | Přístup „Sledování historie – časová platnost + stavy“ | 15 |
| 2.2.7 | Přístup „Sledování historie – časová platnost na každém atributu“ | 24 |
| 2.3 | Komunikační modul | 32 |
| 2.4 | Administrační modul | 39 |
| 2.4.1 | Monitorování procesů | 39 |
| 2.4.2 | Systémové proměnné | 39 |
| 2.4.3 | Monitorování aktivity uživatelů | 39 |
| 2.4.4 | Referenční informace | 40 |
| 2.4.5 | Monitor systému..... | 40 |
| 2.5 | Systémové číselníky | 40 |
| 3 | Sběr Informačních Povinností od Regulovaných Subjektů (SIPRES) | 40 |
| 3.1 | Specifická funkcionality SIPRES | 40 |
| 3.2 | Řešení specifické funkcionality SIPRES v SDAT | 42 |
| 3.2.1 | Volná datová oblast typu formulář..... | 42 |
| 3.2.2 | Přílohy údajů | 43 |
| 3.2.3 | Jednovýkazové kontroly pro dopočítání hodnoty údaje v rámci formuláře | 43 |
| 3.2.4 | Dynamické zobrazování datových oblastí..... | 43 |

| | | |
|-------|---|----|
| 3.2.5 | Autentizace uživatelů | 43 |
| 3.2.6 | Vizuální kontrola příloh Údajů | 43 |
| 4 | Katalog nefunkčních požadavků | 45 |
| 4.1 | Obecné nefunkční požadavky | 45 |
| 4.2 | Obecné nefunkční požadavky – Bezpečnost | 50 |
| 4.3 | Obecné nefunkční požadavky – Audit | 53 |
| 4.4 | Obecné nefunkční požadavky – Provoz systému | 58 |
| 4.5 | Obecné nefunkční požadavky - Provozní prostředí | 60 |
| 4.6 | Obecné nefunkční požadavky - Uživatelské rozhraní | 62 |
| 4.6.1 | Komponenta Tabulka dat (grid) | 67 |
| 4.6.2 | Komponenta Rozbalovací seznam (combobox) | 71 |
| 5 | Katalog funkčních požadavků | 73 |
| 5.1 | Katalog funkčních požadavků pro Komunikační modul | 73 |
| 5.2 | Administrační modul | 75 |
| 5.3 | SIPRES | 77 |

1 Úvod

Účelem dokumentu je definovat obecné a společné funkcionality systému SDAT. Tyto funkcionality se tak prolínají celým systémem a všemi jeho tzv. vrcholovými oblastmi. Dokumenty, které popisují jednotlivé vrcholové oblasti se tak zaměřují na popis specifických funkcionalit systému, které předpokládají existenci společných pravidel a funkcionalit. 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](#)).

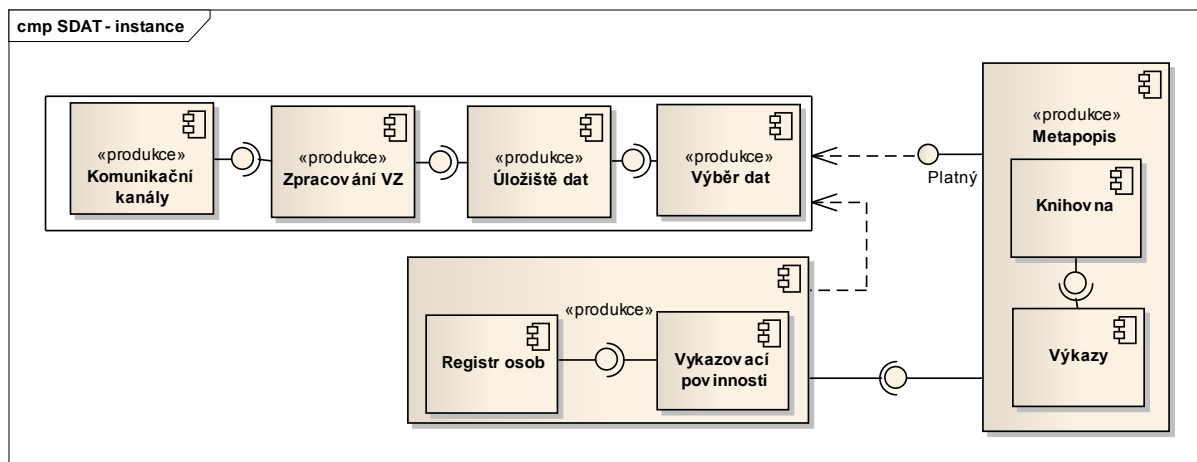
V následujících kapitolách jsou vyjmenována a charakterizována jednotlivá provozní prostředí systému SDAT.

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 je na testovacím prostředí 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:

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

Testovací prostředí je plněno jednosměrným procesem přenosu instancí objektů¹ z produkčního prostředí, dále označovaným jako synchronizace.

¹ Obecně veškerá data zajišťující chod systému (tj. uživatelé, osoby, přístupová práva, metapopis, výkazy...) s výjimkou dat získávaných sběrem výkazů a systémového nastavení, podrobněji dále v textu.

Synchronizace bude řízena z produkčního prostředí příznakem na úrovni objektu Výkaz (resp. jeho verze), který se bude jmenovat „Synchronizovat na test“ a bude typu boolean. V případě vzniku nového výkazu (nebo nové verze již existujícího výkazu) bude tento příznak defaultně nastaven na „false“. Tím bude zajištěno, že se na testovací prostředí nebudou synchronizovat výkazy, které jsou v takovém stavu projektování, ve kterém nemá žádný význam, aby byly testovány. Dále práce s tímto příznakem závisí na stavu Výkazu a bude vypadat takto:

- a) Příznak je uživatelsky měnitelný pouze na produkčním prostředí pro výkazy ve stavu Projektovaný (při založení Výkazu je výchozí hodnota „false“, uživatel mění na „true“, pokud potřebuje zajistit synchronizaci daného výkazu na test). Změna příznaku nevyvolává vznik nové verze nebo varianty výkazu.
- b) V případě, že uživatel změní tento příznak z hodnoty „false“ na „true“ (zapne synchronizaci), systém před samotnou změnou hodnoty provede kontrolu konzistence dané verze výkazu. V případě, že kontrola konzistence skončí s negativním výsledkem, bude tato skutečnost oznámena uživateli a systém změnu příznaku neprovede.
- c) Každý cyklus synchronizace provede automatické shození příznaku na „false“ u těch výkazů, které byly předmětem synchronizace.
- d) Jakákoli změna výkazů ve stavech „Schválený“ nebo „Platný“ automaticky způsobí nastavení příznaku na hodnotu „true“. Ve stavech „Schválený“ nebo „Platný“ nelze provádět změny, které mohou způsobit nekonzistenci výkazu.
- e) V případě, že uživatel provádí tzv. „zplatnění výkazu“, tedy přesouvá verzi výkazu ze stavu schválený do stavu platný (toto může provést jedině v případě, že kontrola konzistence neodhalí žádné problémy), pak systém příznak automaticky nastaví na hodnotu „true“.
- f) Během automatické synchronizace se na testovací prostředí přenášejí pouze verze výkazů, které mají nastaven příznak „synchronizovat na test“ na hodnotu „true“.
- g) V případě, že je verze výkazu synchronizována na test, pak z hlediska uchování dat (výskytů výkazu), která jsou k výkazu navázána, platí, že data (výskyty výkazů) synchronizovaného výkazu jsou vždy na testu odstraněna. To je vynuceno tím, že ve všech stavech (i ve stavu „platný“) existuje nějaká možnost, jak změnit nějaké atributy výkazu. I když se jedná o „drobné změny“ (například popis nebo změna oboru hodnot ukazatele nebo změna Úrovně závažnosti kontroly), mohou tyto změny znamenat, že data zaslaná na předcházející verzi výkazu nejsou v souladu s novou verzí výkazu.

Je požadováno, aby se aplikace na testu chovala odlišně proti tomu, jak se bude chovat na produkci a to v těchto bodech:

- a) Na testu je zakázáno měnit instance objektů Knihovny bez ohledu na to, v jakém jsou stavu.
- b) Na testu je zakázáno projektování výkazů s těmito výjimkami:
 - i. Je povolena úprava definice kontrol + úrovně závažnosti kontroly a to ve stavu projektovaný i schválený, nikoli ve stavu platný.
 - ii. Je povolena úprava definice ukazatele zařazeného do datové oblasti na úrovni definice tzv. oboru hodnot. Oborem hodnot se zde rozumí například změna datového typu (místo původního N11 (číslo na jedenáct míst) lze říci, že daný ukazatel v dané datové oblasti je datového typu N13 (číslo na třináct míst) nebo změna domény číselníkůvých položek/hierarchie). Povolení úprav platí pro stavy projektovaný a schválený, nikoliv pro stav platný – stejné pravidlo jako předchozí bod [i](#).

- c) Na testu nelze měnit příznak „synchronizovat na test“.
- d) Osoby mohou na testu vytvářet výskyty výkazů svými silami a to i na výkazy, na které na produkci výskyty vytvářet nesmí a to ve všech stavech výkazu.
- e) Systém na testovacím prostředí negeneruje avíza a upomínky.

Synchronizace obsahu produkčního a testovacího prostředí pro instance objektů ve stavu „Platný“ a „Schválený“ probíhá procesem, který je naplánován na okrajové (noční) hodiny, případně je možné spustit synchronizaci ad hoc, tj. mimo pravidelné/plánované časy. 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 (testujícím se mohou změnit podmínky v průběhu testování; předpokládáme, že v nočních hodinách k testování nebude docházet nebo k němu bude docházet v minoritním rozsahu).

Součástí synchronizace jsou dále Vykazovací povinnosti a Uživatelé vč. přístupových práv:

- V případě, že by synchronizací mělo dojít k odstranění vykazovací povinnosti, ke které na testu již vznikl nějaký výskyt výkazu, pak systém zajistí „rozpojení“ rušené vykazovací povinnosti a existujícího výskytu výkazu – danou vykazovací povinnost na testu smaže, ale výskyt výkazu ponechá. Synchronizuje se celá oblast Vykazovacích povinností, uživatel si nemůže vybrat, že chce synchronizovat pouze některé Vykazovací povinnosti.
- Synchronizace Uživatelů a přístupových práv (včetně rolí, aktivit) probíhá pouze na vyžádání. Synchronizuje se celá oblast Uživatelů a přístupových práv včetně rolí a aktivit. Uživatel si nemůže vybrat, že chce synchronizovat pouze některé uživatele, přístupová práva, role či aktivity.

Výskyty výkazů ani vykázaná data nejsou předmětem synchronizace.

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 všechny stavy (Projektovaný, Schválený, Platný). 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).

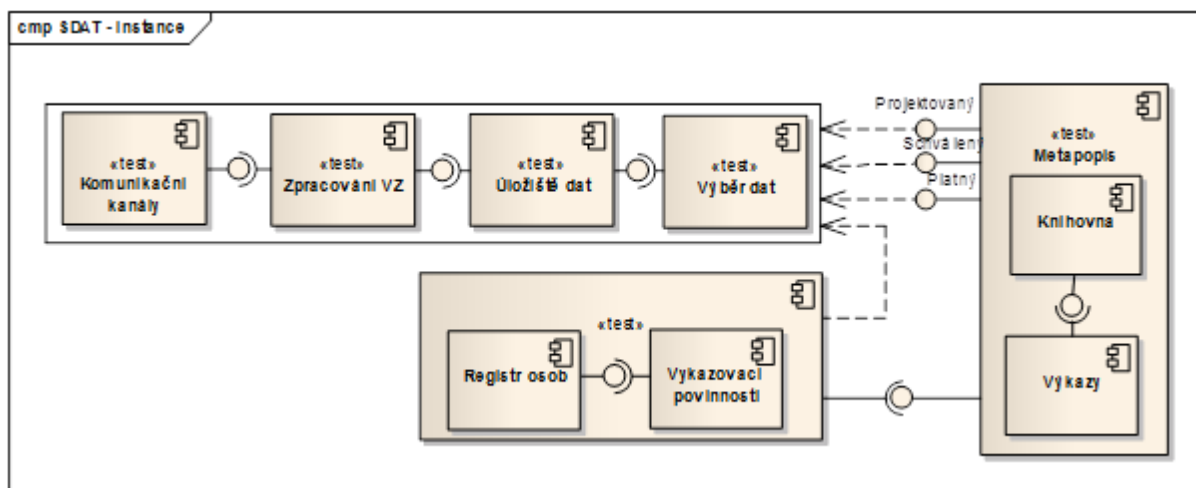
Testovací prostředí odpovídá z pohledu verzí programového vybavení komponent produkčnímu prostředí.

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

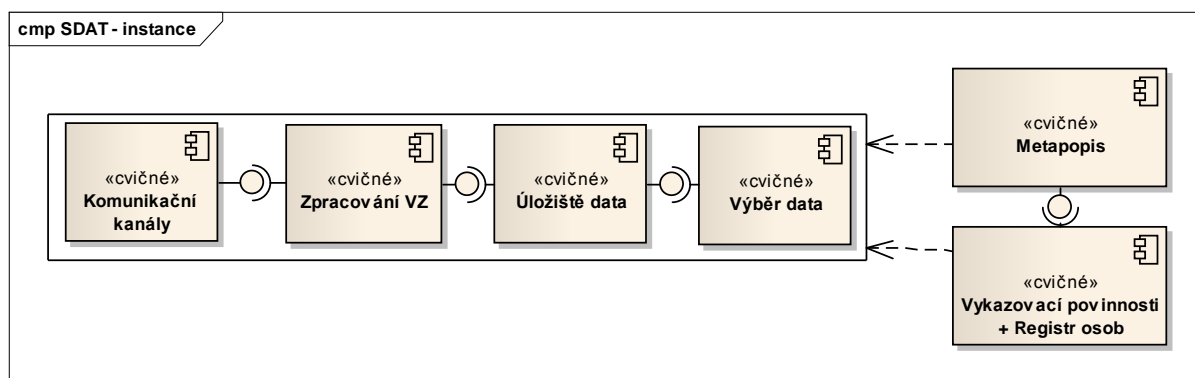
2.1.3 Cvičné prostředí

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. Na tomto prostředí lze zkusit i proces příjmu dat, nicméně se vždy bude jednat o data zasílaná interními uživateli pod „testovací osobou“. Cvičné prostředí tak nebude obsahovat žádná produkční data (hodnoty údajů); tato oblast nebude předmětem synchronizace. Není tak třeba synchronizovat ani externí uživatele a registr osob.

Synchronizovat z produkce se bude pouze oblast „Metapopis“ a to na ad-hoc bázi a stylem „override“ (přepsat vše). V okamžiku synchronizace tak dojde k odstranění všech upravených/nově vzniklých záznamů na cvičném prostředí. 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 Školící prostředí

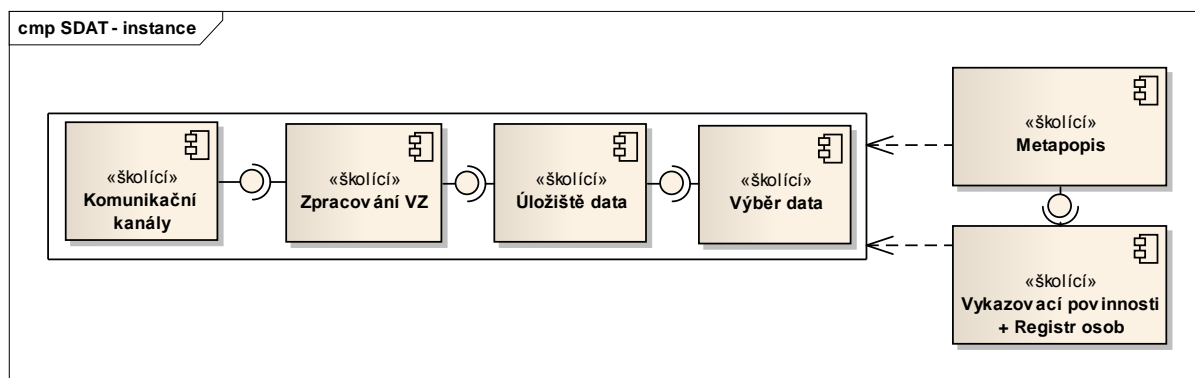
Na školicím prostředí probíhají organizovaná školení uživatelů systému, zejména metodiků výkazů.

Školící prostředí odpovídá z pohledu verzí programového vybavení komponent produkčnímu prostředí. Školící prostředí je primárně určeno pro školení interních uživatelů a nepředpokládá se, že by nad tímto prostředím probíhalo školení externích uživatelů.

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. Školící prostředí nebude obsahovat žádná produkční data (hodnoty údajů); tato oblast nebude předmětem synchronizace. Dále není třeba synchronizovat ani externí uživatele a registr osob.

Synchronizovat z produkce se bude pouze oblast „Metapopis“ a to na ad-hoc bázi a stylem „override“ (přepsat vše). V okamžiku synchronizace tak dojde k odstranění všech upravených/nově vzniklých záznamů na školicím prostředí.

Toto prostředí je dostupné pouze v rámci systémového prostředí ČNB.



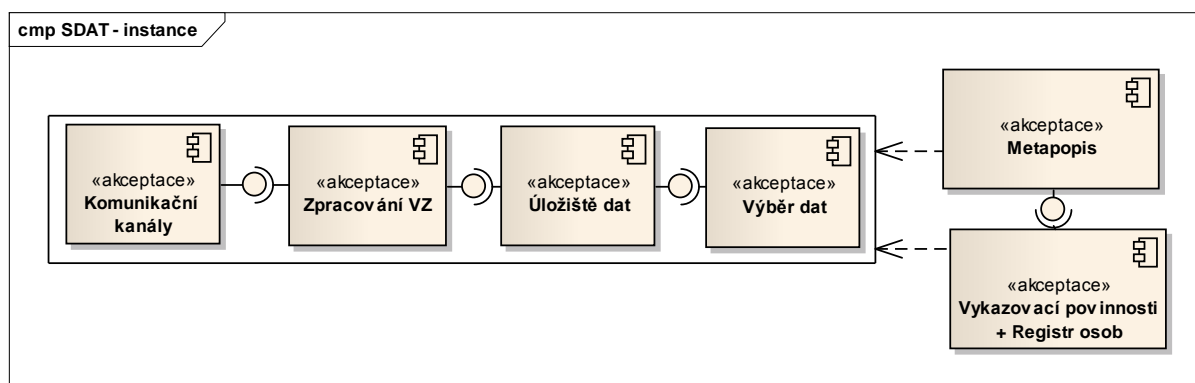
Obrázek 4 - Schéma školicího prostředí

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. deployment).

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:

- Bez sledování historie,
- 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 přechíslování 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.

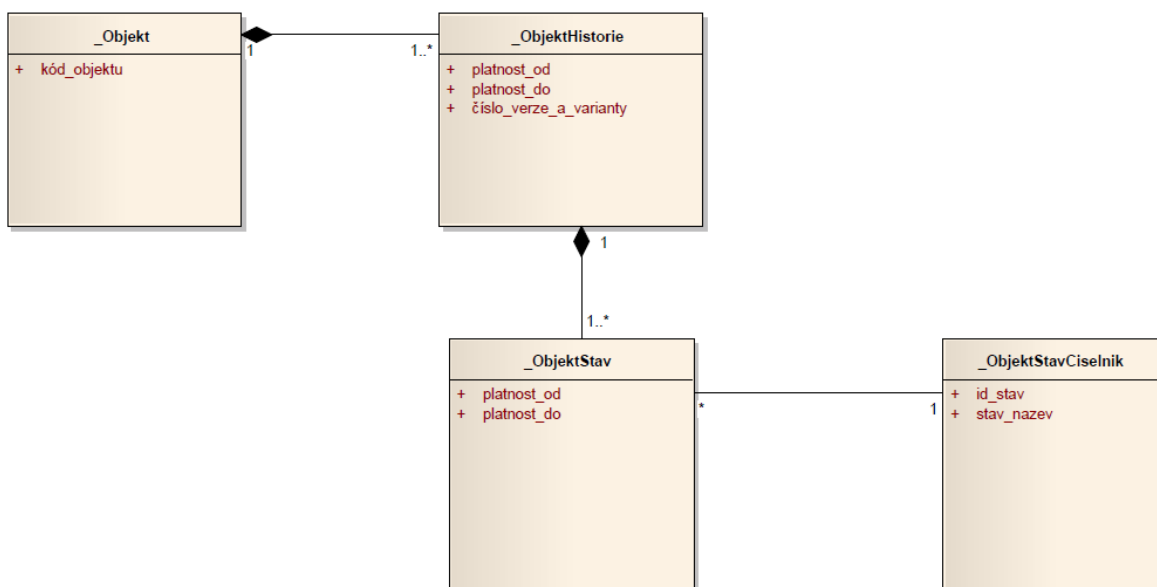
Objekty použité v obecném modelu pro sledování historie (historie a stavů) začínají prefixem „_“, aby bylo patrné, že se jedná o obecné objekty, které v praxi musejí být nahrazeny reálnými objekty toho kterého objektu, ke kterému se váží.

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.

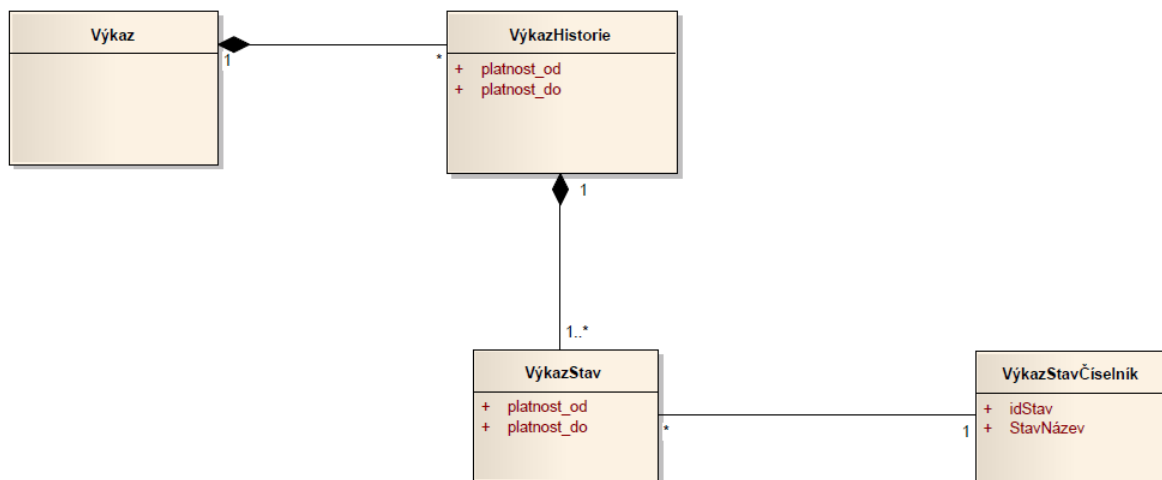
Pro názornost uvádíme, jak se výše uvedený obecný model zhmotní do podoby reálných objektů systému na příkladu objektu Výkaz:

- 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.

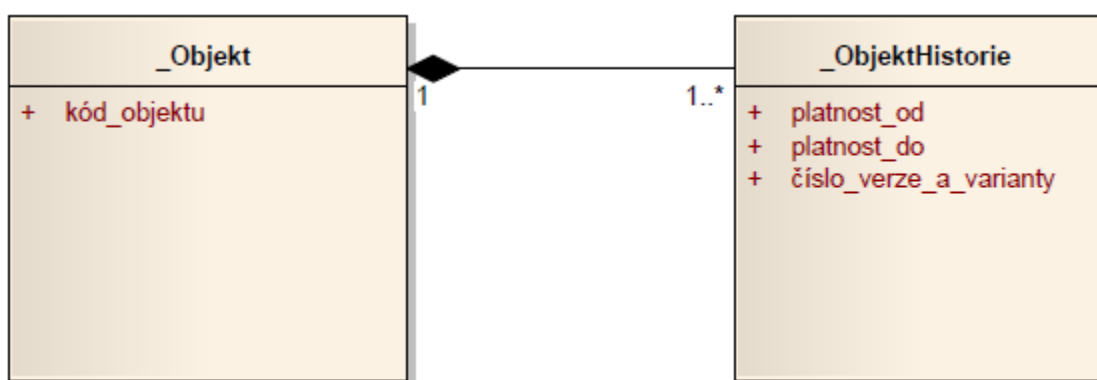
Žádná z instancí souvisejících objektů se neváže na hlavní objekt (`_Objekt`), ale na objekt `_ObjektHistorie`.

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.

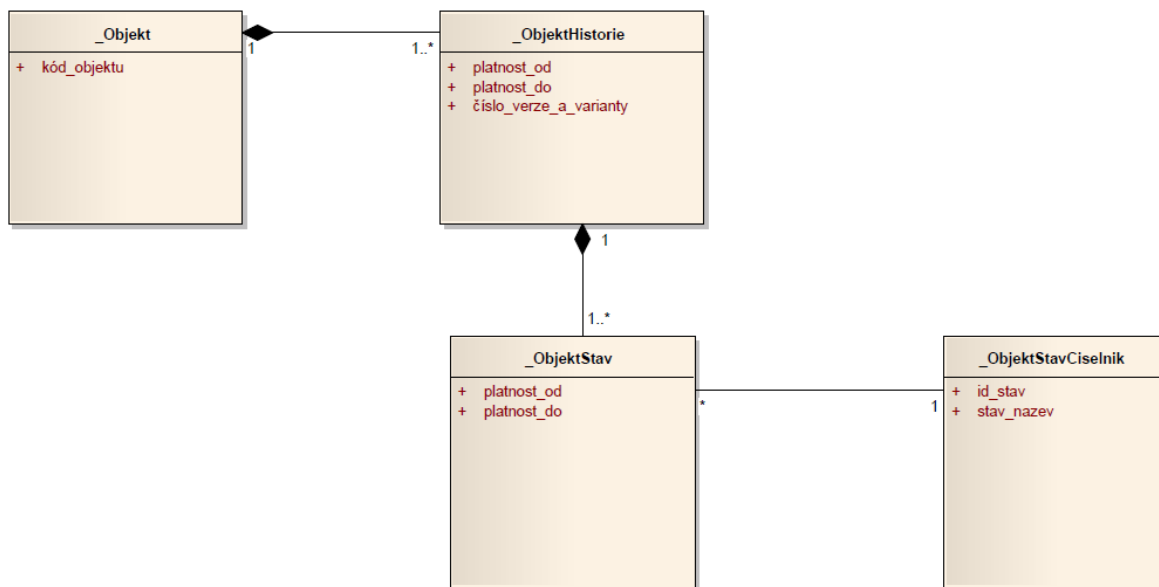
Při manipulaci s časovou a stavovou platností musejí platit tato pravidla (platí pro objekty metapopisu a obecně pro všechny objekty, které procházejí stavy Projektovaný, Schválený a Platný):

- 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 nadřazené. 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, VýkazHistorie 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 vzniknout 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.

| | | | | |
|----------------------|----------|------------|----------|------------|
| Objekt VýkazHistorie | 1.1.2013 | 31.12.2013 | 1.1.2014 | 31.12.4000 |
| Verze "001.000" | N/A | | | |
| Objekt VýkazStav | 1.1.2013 | | | 31.12.4000 |
| Stav "Projektovaný" | | | | |

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).

| | | | | |
|-----------------------------|----------|------------|----------|------------|
| Objekt VýkazHistorie | 1.1.2013 | 31.12.2013 | 1.1.2014 | 31.12.4000 |
| V001 Verze "001.000" | N/A | | | |
| Objekt VýkazStav | 1.1.2013 | 30.6.2013 | 1.7.2013 | 31.12.4000 |
| 001.000 Stav "Projektovaný" | N/A | | | |
| 001.000 Stav "Schválený" | N/A | | | |

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:

- instance objektu Výkaz se nemění, výkaz stále existuje,
- změna stavu se děje vždy v rámci konkrétní verze/varianty, ale nemění se žádný z atributů objektu VýkazHistorie, takže instance objektu VýkazHistorie zůstane nezměněna,
- ke změně musí dojít v rámci objektu VýkazStav, kde musí dojít k ukončení platnosti stavu Schválený a založení nového stavu Platný².

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).

| | | | | |
|-----------------------------|----------|------------|----------|------------|
| Objekt VýkazHistorie | 1.1.2013 | 31.12.2013 | 1.1.2014 | 31.12.4000 |
| V001 Verze "001.000" | N/A | | | |
| Objekt VýkazStav | 1.1.2013 | 30.6.2013 | 1.7.2013 | 31.12.4000 |
| 001.000 Stav "Projektovaný" | N/A | | | |
| 001.000 Stav "Schválený" | N/A | | N/A | N/A |
| 001.000 Stav "Platný" | N/A | | | |

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.**

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

² 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.

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:

- instance objektu Výkaz se nemění, výkaz stále existuje,
- 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,
- 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 nadřazené. 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).

| Objekt VýkazHistorie | | 1.1.2013 | 31.12.2013 | | 1.1.2014 | 31.12.2014 | | 1.1.2015 | 31.12.4000 | | |
|----------------------|---------------------|-----------------|------------|----------|------------|------------|-----------|----------|------------|--|--|
| V001 | Verze "001.000" | N/A | | | | | | | | | |
| | Verze "002.000" | N/A | | | | | | | | | |
| Objekt VýkazStav | | 1.1.2013 | 30.6.2013 | 1.7.2013 | 31.12.2013 | 1.1.2014 | 30.6.2014 | 1.7.2014 | 31.12.4000 | | |
| 001.000 | Stav "Projektovaný" | | | N/A | | | | | | | |
| | Stav "Schválený" | N/A | | | | N/A | | | | | |
| | Stav "Platný" | N/A | | | | | | | | | |
| 002.000 | Stav "Projektovaný" | N/A | | | | | | | | | |
| | Stav "Schválený" | stav neexistuje | | | | | | | | | |
| | Stav "Platný" | stav neexistuje | | | | | | | | | |

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.

| Objekt VýkazHistorie | | 1.1.2013 | 31.12.2013 | | 1.1.2014 | 31.12.2014 | | 1.1.2015 | 31.12.4000 | |
|----------------------|---------------------|-----------------|------------|----------|------------|------------|-----------|----------|------------|--|
| V001 | Verze "001.000" | N/A | | | | | | | | |
| | Verze "002.000" | N/A | | | | | | | | |
| Objekt VýkazStav | | 1.1.2013 | 30.6.2013 | 1.7.2013 | 31.12.2013 | 1.1.2014 | 30.6.2014 | 1.7.2014 | 31.12.4000 | |
| 001.000 | Stav "Projektovaný" | | | | N/A | | | | | |
| | Stav "Schválený" | N/A | | | | | N/A | | | |
| | Stav "Platný" | N/A | | | | | | | | |
| 002.000 | Stav "Projektovaný" | N/A | | | | | | | | |
| | Stav "Schválený" | N/A | | | | | | | | |
| | Stav "Platný" | stav neexistuje | | | | | | | | |

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 nadřazené instance objektu),
2. Pro verzi 002.000 musí dojít k ukončení platnosti stavu Schválený a založení nového stavu Platný.

| Objekt VýkazHistorie | | 1.1.2013 | 31.12.2013 | 1.1.2014 | 30.6.2014 | 1.7.2014 | 31.12.2014 | 1.1.2015 | 31.12.4000 | | |
|----------------------|---------------------|----------|------------|----------|------------|----------|------------|----------|------------|----------|------------|
| V001 | Verze "001.000" | N/A | | | | | | | N/A | | |
| | Verze "002.000" | | | | | | | | | | |
| | | | | | | | | | | | |
| Objekt VýkazStav | | 1.1.2013 | 30.6.2013 | 1.7.2013 | 31.12.2013 | 1.1.2014 | 30.6.2014 | 1.7.2014 | 31.12.2014 | 1.1.2015 | 31.12.4000 |
| 001.000 | Stav "Projektovaný" | | | N/A | | | | | | | |
| | Stav "Schválený" | N/A | | | | | N/A | | | | |
| | Stav "Platný" | N/A | | | | | | | N/A | | |
| 002.000 | Stav "Projektovaný" | N/A | | | | | | N/A | | | |
| | Stav "Schválený" | N/A | | | | | | N/A | | | |
| | Stav "Platný" | N/A | | | | | | | | | |

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.

| Objekt VýkazHistorie | | 1.1.2013 | 31.12.2013 | | | 1.1.2014 | 30.6.2014 | 1.7.2014 | 31.12.2014 | | 1.1.2015 | 12.16 | 31.12.4000 | |
|----------------------|---------------------|----------|------------|----------|------------|----------|-----------|----------|------------|--|----------|-------|------------|--|
| V001 | Verze "001.000" | N/A | | | | | | | | | | N/A | | |
| | Verze "002.000" | | | | | | | | | | | | N/A | |
| Objekt VýkazStav | | 1.1.2013 | 30.6.2013 | 1.7.2013 | 31.12.2013 | 1.1.2014 | 30.6.2014 | 1.7.2014 | 31.12.2014 | | 1.1.2015 | 12.16 | 31.12.4000 | |
| 001.000 | Stav "Projektovaný" | | | N/A | | | | | | | | | | |
| | Stav "Schválený" | N/A | | | | | | | N/A | | | | | |
| | Stav "Platný" | N/A | | | | | | | | | N/A | | | |
| 002.000 | Stav "Projektovaný" | N/A | | | | | | | N/A | | | | | |
| | Stav "Schválený" | N/A | | | | | | N/A | | | | | | |
| | Stav "Platný" | N/A | | | | | | N/A | | | | | | |

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: <ul style="list-style-type: none"> • STRING • DATE • NUMBER • ENUM (určuje, že hodnota atributu je tvořena číselníkem a může nabývat pouze předem definované | Ano | Ne |

| | hodnoty) | | |
|------------|---|-----|----|
| 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 související 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: <ul style="list-style-type: none"> • 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 anebo je nepovinný a zadán být pouze může. | Ano | Ne |
| je_jedinečný | Ano/Ne – určuje, zda pro danou kategorii Osoby musí být atribut jedinečný (ano) a zadání duplicitní hodnoty znamená nepřekonatelnou chybu anebo 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) | Ano | Ne |

| Název atributu | Účel atributu | Povinný? | Jedinečný? |
|----------------|---|----------|------------|
| | anebo smí být pro daný subjekt a jeden čas zadán právě jednou (ne). | | |
| pouze_ke_čtení | Pouze pro osoby přebírané z externích zdrojů. Ano/Ne – určuje, zda je daný atribut určen pouze ke čtení (ano) anebo je možné jej měnit na straně správce SDAT (ne). Pokud není Osoba přebírána z externího zdroje (je založena v SDAT), pak tento atribut je irrelevantní. | Ano | Ne |
| je_schvalován | Pouze pro osoby přebírané z externích zdrojů. Ano/ne – určuje, zda pro přijetí hodnoty daného atributu je nutné schválení správcem systému (ano) anebo 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: <ul style="list-style-type: none"> • 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 - Pojišťovníctví 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_odpovednosti | 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: <ul style="list-style-type: none"> • Technický správce • Věcný správce • Příjemce upomínek | Ano | Ne |
| 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 |

| Název atributu | Účel atributu | Povinný? | Jedinečný? |
|----------------|-------------------------------------|----------|------------|
| 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.

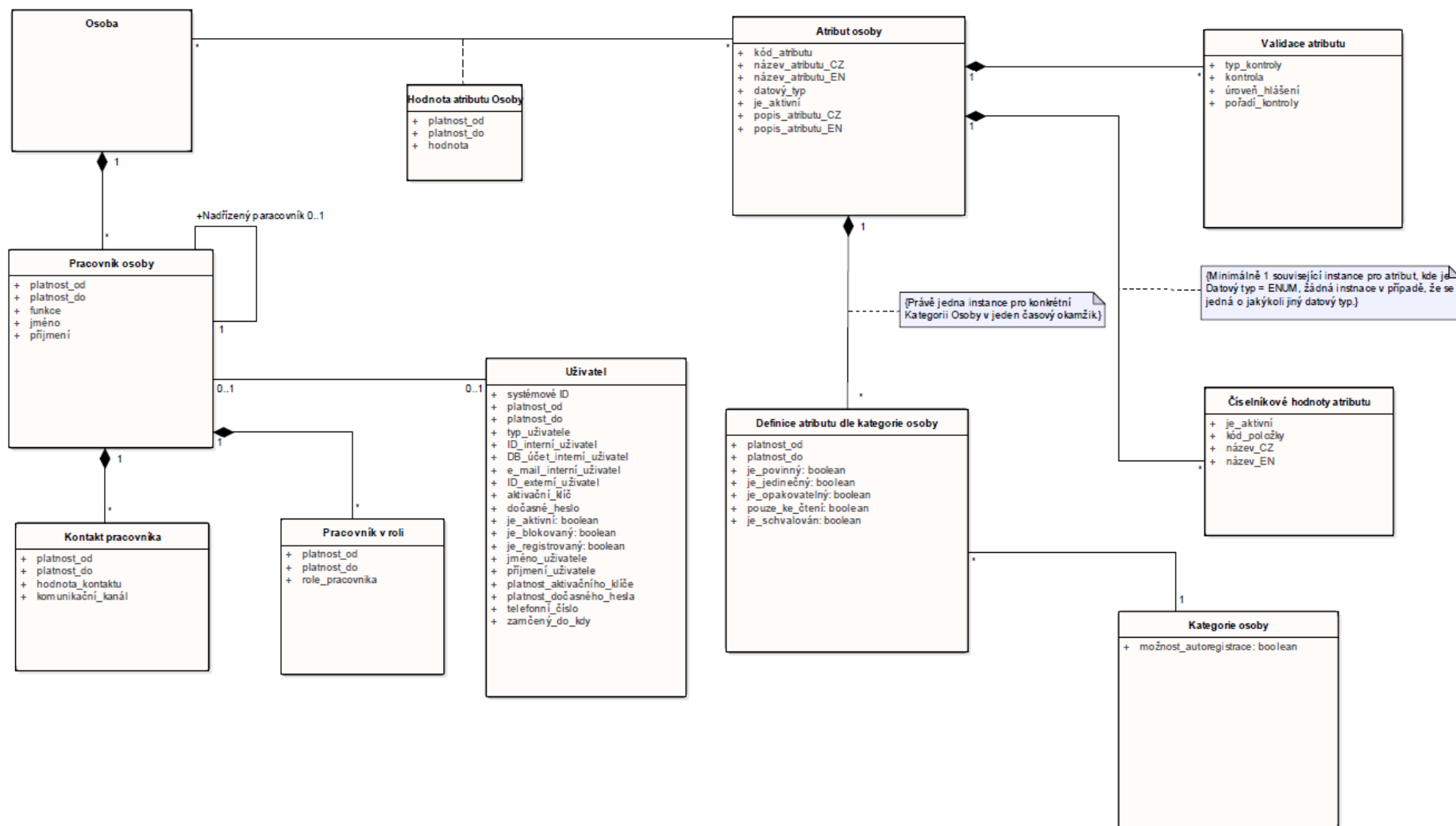
2.2.7.1.8 Objekt Kontakt pracovníka

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 <ul style="list-style-type: none"> • 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 Kontaktů 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:

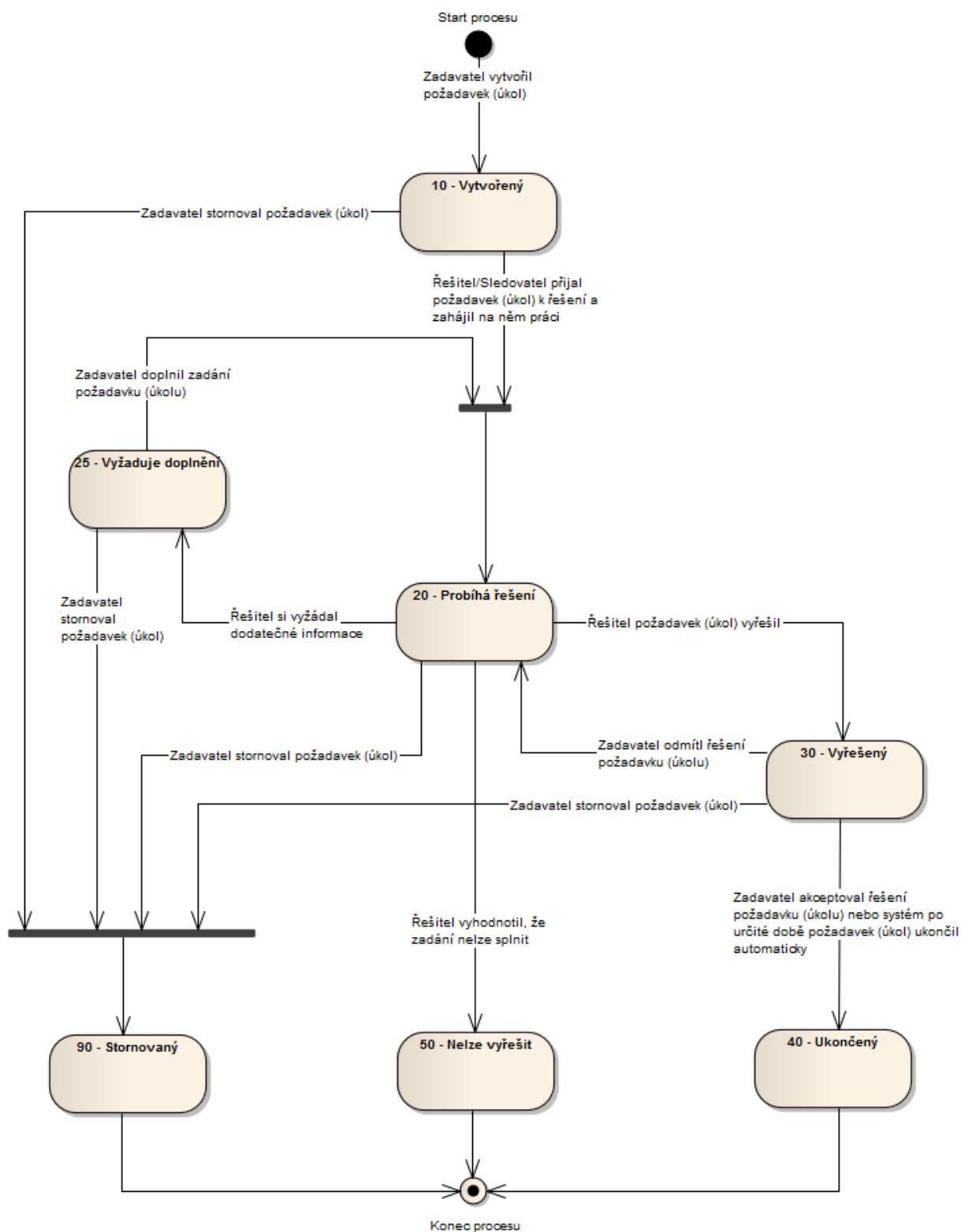
- **task management systému** – jedná o systém, který je primárně určen ke správě úkolů a řízení workflow s těmito úkoly souvisejícími. Tato část komunikačního modulu je realizována mimo systém SDAT,
- **rozhraní SDAT** – jedná se o sadu funkcionalit a formulářů, které jsou dostupné přímo v systému SDAT a zprostředkovávají uživatelům SDAT komunikaci s task management systémem. Uživatelé 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 (úkol) 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 „uživatelé vytvořené“, „uživatelé sledované“ a „uživatelé řešené“,
- možnost zasílat e-mailové notifikace zúčastněným osobám (zadavateli, řešiteli, sledovatelům) v případě jakékoli změny stavu daného požadavku (úkol),
- možnost přiložit neomezený počet binárních souborů k požadavku úkolu ve všech stavech životního cyklu požadavku (úkol),

- 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 (úkolů) je patrné z následujícího stavového diagramu a níže uvedené tabulky s popisem jednotlivých stavů.



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

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

| 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 (úkolů) řešitelem; má se za to, že když řešitel již jednou požadavek přijal a požádal o doplňující informace, je stále řešitelem tohoto požadavku (úkolů). |
| 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 (úkolů) 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 (úkolů) 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 (úkolů). | | | |
| 90 | Stornovaný | Definice: Požadavek (úkol) byl zadavatelem stornován a není třeba se jím dále zabývat Účel: Umožňovat 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 (úkolů) končí. |

Tabulka 5 - Přehled stavů zpracování požadavku (úkolů)

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 zámků uvolnit.

2.4.4 Referenční informace

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

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](#). Příklady systémových číselníků jsou:

- číselník rolí osob viz dokument [C – Vykazovací povinnosti a Registr osob](#) (ROS_1.2),
- číselník periodicit viz dokument [C – Vykazovací povinnosti a Registr osob](#) (VPO_1.0 a VPO_7.0).

3 Sběr Informačních Povinností od Regulovaných Subjektů (SIPRES)

ČNB provozuje systém SIPRES za účelem sběru, uchování a dalšího zpracování informací od subjektů působících na finančním trhu, které jsou regulovány právě ČNB. Systém tvoří 4 moduly:

1. Externí aplikace pro sběr informací od subjektů.
2. Interní aplikace (správa informačních povinností, vkládání informací za vykazující osoby, zpřístupnění informací uživatelům v ČNB, administrace systému).
3. Prezentace informací na webu ČNB.
4. Reporting informací do agentury ESMA.

Záměrem ČNB je nahrazení modulu pro sběr informací (bod [1](#) výše) systémem SDAT a funkcionalit zajišťujících správu informačních povinností a vkládání informací za vykazující osoby (část bodu [2](#) výše). Ostatní moduly a funkcionality zůstanou zachovány jako systém návazného zpracování SIPRES2. Předávání dat z SDAT do SIPRES2 bude řešeno standardně přes Aplikační datové rozhraní SDAT.

3.1 Specifická funkcionality SIPRES

Z pohledu datového obsahu se informace sbírané systémem SIPRES od klasického statisticko-dohledového výkaznictví liší v těchto charakteristikách:

1. **Informace mají charakter, který vede na použití** vstupních struktur výkazů v podobě **formulářů**, spíše než datových oblastí v podobě tabulek. Některé výkazy jsou kombinací obou typů informací. Jedna část má charakter formuláře, kde Osoba vyplní údaje identifikačního charakteru. Další část výkazu pak může tvořit dynamická datová

oblast pro vykazání sledované skutečnosti. V současné době jsou formuláře v systému SIPRES vytvářeny programátorem pro každý jednotlivý výkaz. Tato skutečnost umožňuje maximální přizpůsobení vstupní struktury formuláře potřebám uživatelů. Uživatelský komfort výkazů je velmi důležitý, protože kromě výkazů sbíraných periodicky, existují i výkazy vyplňované převážně manuálně a mají charakter hlášení, tj. ad-hoc vykazovací povinnosti, která nastává na základě události (např. provedení určitého typu obchodní transakce), což znamená, že uživatelé (vykazující osoby) nepracují se systémem pravidelně a nelze spoléhat na znalost prostředí systému. Vzory jednotlivých výkazů jsou dostupné on-line přímo na adrese aplikace

http://www.cnb.cz/cnb/sipres_cz (veřejně jsou dostupné pouze výkazy pro neregistrované subjekty) nebo jsou rovněž součástí přílohy č. 1 Smlouvy – Věcné zadání, v adresáři „05-SIPRESPřílohy“, název archívu „SDAT-SIPRES-Přílohy.zip“.

2. **Předmětem sběru jsou informace v podobě binárních souborů.** Typicky je výkaz tvořen formulářem, kam se vkládají pouze průvodní informace. Samotný věcný obsah je vykazován prostřednictvím připojeného binárního souboru. Z pohledu struktury lze rozdělit obsah souborů rozdělit na tyto skupiny:
 - a. **Nestrukturovaná data** ve formátu pdf, doc, jpg, gif (jedná se o výroční zprávy apod.).
 - b. **Strukturovaná data** ve formátu MS Excel, XML. Touto formou jsou data sbírána z těchto důvodů:
 - i. Pro vykazující osoby je z pohledu pořizování dat praktičtější vytvoření souboru MS Excel.
 - ii. V rámci MS Excel lze pomocí vzorců realizovat funkcionalitu pro dopočítání hodnoty pole na základě pole jiného.

Kromě datového obsahu existují další odlišnosti výkazů SIPRES.

3. **Dynamika formulářů**, kdy na základě vyplnění jednoho údaje je nabídnuta odpovídající datová oblast k dalšímu vyplnění.
4. **Autentizace uživatelů** probíhá několika způsoby, které je navázaná na konkrétní výkazy:
 - a. **Registrované osoby** jsou do systému registrovány pracovníkem ČNB. Pro vykazování je požadován kvalifikovaný certifikát.
 - b. **Neregistrovaný přístup** je situace, kdy vykazující osoba není v systému předem registrována a v prvním kroku každého vykazování uvede své jméno, příjmení a email. Toto přihlášení je chráněno pomocí CAPTCHA. U tohoto způsobu autentizace není zajištěna jednoznačná identita osoby na úrovni příjmu dat, ta je součástí až dat samotných.
 - c. **Evidované osoby** jsou předem v systému registrovány a před vykazováním dostávají emailem autorizační kód, na základě kterého jsou výkazy identifikovány a přiřazeny konkrétnímu subjektu (kromě kódu uvádějí jméno, příjmení, email).

Vizuální kontrola obsahu binárních souborů není prováděna automaticky systémem při příjmu dat, protože zejména nad nestrukturovanými daty ji nelze

vykonat strojově. Kontrolu provádí uživatel v ČNB, který obsah vizuálně zkontroluje (soubor skutečně obsahuje výroční zprávu).

3.2 Řešení specifické funkcionality SIPRES v SDAT

3.2.1 Volná datová oblast typu formulář

Systém SDAT umožňuje skloubit výhody poskytované standardním způsobem projektování (metapopis vykazovaných Údajů) s výhodami, které nabízí dnešní autonomní aplikace SIPRES (možnost definovat vlastní layout formuláře, pomocí kterého jsou hodnoty Údajů získávány).

Systém SDAT umožňuje vytvořit strukturu datové oblasti, v níž lze jednotlivé Údaje umísťovat nezávisle na matici danou kombinací Ukazatelů a jejich Parametrů rozmístěných na osy tabulky (běžný přístup při standardním projektování). V datové oblasti typu formulář navíc umožňuje jednotlivé Údaje umísťovat na formulář zcela nezávisle, pouze s dodržением základních ergonomických pravidel. Příkladem je umístění jednotlivých údajů reprezentujících adresu (ulice, město, PSČ, stát) na jeden řádek vedle sebe, zatímco na dalším řádku jsou uvedeny například pouze dva další údaje (telefonní číslo, e-mailová adresa). Toto je realizováno bez toho, aby metodik zaváděl umělé parametry popisující takto seskupené Údaje, kterými tohoto cíle dosáhne ve standardní datové oblasti.

- a) Každý údaj, který je na formulář přidáváný, je popsán podle standardních pravidel, která platí pro popis údaje.
- b) Systém pro každý údaj, který je do datové oblasti zařazen, navrhne defaultní ovládací prvek (pro údaje, které jsou vázány na Číselník/Doménu číselníku/Datový typ) nabídne rozbalovací seznam, pro údaje, které vyžadují manuální vstup uživatele (jméno, příjmení) nabídne klasické zadávací pole (input box). Jsou tak dostupné minimálně tyto standardní formulářové prvky:
 - vstupní pole pro textovou nebo číselnou hodnotu,
 - textové pole pro dlouhý text typu poznámka apod.,
 - vstupní pole pro výběr ze seznamu hodnot,
 - vstupní pole pro upload souboru,
 - vstupní pole pro datum (případně i s časovou složkou) s možností zapsat hodnotu přímo nebo jí vybrat z kalendáře.
- c) Aby bylo možné pracovat s jednotlivými údaji jako s formulářovými prvky, nejsou u nich definovány žádné „viditelné“ parametry.
- d) Ke každému formulářovému prvku údaj je přiřazen prvek popis (label), který má defaultně hodnotu názvu Ukazatele, kterým je Údaj definován. Popis lze:
 - upravit – přepsat jeho hodnotu,
 - odstranit,
 - umístit vlevo nebo nad formulářový prvek, ke kterému se váže.
- e) Pro jednotlivé formulářové prvky je možné stanovit jejich rozměr. Pokud by např. bylo celé řešení pojato jako umísťování prvků na souřadnice řádko-sloupcové

matice je přípustné rozměry řešit určením počtu řádků a sloupců matice, který prvek zabírá.

- f) Jednotlivé formulářové prvky mají dostupnou kontextovou nápovědu (tooltip), která je naplněna popisem Ukazatele, kterým je Údaj definován.
- g) Mezi jednotlivými prvky formuláře, na úrovni řádků je možné přidat libovolný popisný text. Popisný text je možno formátovat běžnými prostředky (barva, velikost, font).

3.2.2 Přílohy údajů

V systému SDAT lze vytvořit Údaj s datovým typem „příloha“ (binární nebo textový soubor), tj. soubory jsou ukládány jako Údaje a jako Údaje je lze z SDAT vybírat.

3.2.3 Jednovýkazové kontroly pro dopočítání hodnoty údaje v rámci formuláře

V systému SDAT lze zapsat kontroly, které se vykonávají, pokud je alespoň jeden člen vzorce vyplněn a dopočítá se hodnota jiného člena vzorce. Takto vytvořenou hodnotu Údaje lze při pořizování dat ve formuláři uživatelem přepsat. Díky existenci této funkce bude možné převést některé z binárních příloh (soubory MS Excel) výkazů SIPRES do podoby standardní datové oblasti SDAT.

3.2.4 Dynamické zobrazování datových oblastí

Systém SDAT umožňuje v dynamických Datových oblastech „vykřížkovat“ Údaje (tj. znemožní vyplnit hodnotu daného Údaje) na základě zvolené kombinace dynamických Parametrů. Vytknuté dynamické Parametry (např. umístěné v comboboxu) platící pro celou Datovou oblast umožňují na základě volby měnit množinu Údajů, do kterých lze při dané kombinaci vytknutých Parametrů vyplnit hodnoty.

3.2.5 Autentizace uživatelů

V systému SDAT budou jednotlivé typy stávající autentizace uživatelů v SIPRES řešeny následovně:

- a) pro SIPRES-Registrované osoby je použit SDAT-Externí registrovaný uživatel
- b) pro SIPRES-Neregistrovaný přístup je použit SDAT-Neregistrovaný uživatel
- c) pro SIPRES-Evidované osoby je dle charakteru výkazu použit SDAT-autoregistrace nebo SDAT-Externí registrovaný uživatel.

3.2.6 Vizuální kontrola příloh Údajů

Systém SDAT obsahuje kontrolu typu „vizuální kontrola“, která může být použita nad Údajem typu „příloha“.

- Při procesu příjmu Vstupní zprávy platí, že Vydání výskytu výkazu pro Výkaz, který obsahuje tento typ kontroly je převedeno v případě úspěšného splnění všech

ostatních kontrol do stavu „Předběžně platné – čeká se na provedení vizuální kontroly“ a současně je Výskyt výkazu převeden do stavu „70 – Nedokončený“.

- Poté co uživatel v ČNB (např. gestor výkazu nebo operátor systému) provede kontrolu souboru, je v případě že soubor obsahuje požadovaný obsah převeden stav Vydání výskytu výkazu do stavu „60 – Platné“ a stav Výskyt výkazu do stavu „60 – Splněný“. V opačném případě je převeden stav Vydání výskytu výkazu do stavu „80 – Závažná chyba“.

Systém SDAT obsahuje automatický proces, který periodicky (např. 1x denně, s možností konfigurace) odesílá gestorům výkazu upozornění o tom, že existuje Vydání výskytu výkazu ve stavu „Předběžně platné – čeká se na provedení vizuální kontroly“.

4 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.

4.1 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 definovanou 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: <ul style="list-style-type: none"> - 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 | <p>Každý záznam uložený v databázi obsahuje následující informace:</p> <ul style="list-style-type: none"> - informace o uživateli, který záznam vytvořil, - časové razítko vytvoření záznamu, - informace o uživateli, který záznam naposledy změnil, - časové razítko poslední změny záznamu. <p>Výše uvedené údaje jsou viditelné v aplikaci v místě, kde se s daným záznamem pracuje, každému uživateli, který má oprávnění daný záznam vidět.</p> | Závazný | 1 |
| NFP_14.0 | Kompletní aplikační audit | <p>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.</p> <p>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.</p> <p>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.</p> | 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 účasti 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 | <p>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). Lokalizací se rozumí:</p> <ul style="list-style-type: none"> • překlady prostředí aplikace (popisky formulářových polí (labels), 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). <p>Národní prostředí v systému SDAT je koncipováno shodně s národním prostředím operačního systému Windows; pro jazyk „Angličtina“ je použito národní prostředí „Spojené Království“.</p> <p>Lokalizace pro hlavní jazyk musí být úplná, tzn. nesmí existovat žádný text v rámci uživatelského rozhraní, který by nebyl lokalizován do hlavního jazyka.</p> | Závazný | 1 |
| NFP_28.1 | Lokalizace aplikace – defaultní jazyk | Systém uživateli nastaví takový jazyk/znakovou sadu, který odpovídá nastavením jeho jazykových preferencí ve webovém prohlížeči, kterým přistupuje k aplikaci SDAT (defaultní jazyk/znaková sada). Toto pravidlo platí pro tu část aplikace, kde se | Závazný | 1 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|-----------------|------------------------------------|---|------------|-----------|
| | | 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 | <p>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:</p> <ul style="list-style-type: none"> • 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. <p>V případě, že po změně jazyka neexistuje lokalizační text pro nějaký prvek/datový objekt, systém zobrazí související text z lokalizační sady hlavního jazyka.</p> | 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 |

4.2 Obecné nefunkční požadavky – Bezpečnost

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|----------------|--|--|------------|-----------|
| BEZ_1.0 | Přístup interních uživatelů na základě | Systém umožňuje SSO autentifikaci pouze těm uživatelům, kteří jsou zařazeni alespoň v jedné aplikační skupině určené pro SDAT. | Závazný | 1 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|----------------|---|---|------------|-----------|
| | nastavení Řídící databáze | 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. | | |
| 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. Aplikace musí podporovat verze 1.0,1.1, 1.2, přičemž defaultními nastavení bude TLS 1.0 vypnuto, TLS 1.1 a 1.2 zapnuto. 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 objednatel) dojde k vypršení tzv. session a je nutné se znovu přihlásit. | Závazný | 1 |
| BEZ_6.1 | Ukončení spojení při | V případě, že dojde k ukončení spojení kvůli nečinnosti uživatele | Vítaný | 3 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|-----------------|---|--|------------|-----------|
| | nečinnosti uživatele – uchování neuložených dat | (viz BEZ_6.0), pak systém umožňuje uchovat neuložené změny a nabídnout je při příštím přihlášení uživatele k dopracování. Tímto požadavkem <u>není myšleno</u> , že má tyto změny před vypršením tzv. session uložit do databáze způsobem, jakým by to učinil uživatel. | | |
| 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 |
| BEZ_12.0 | Soulad se Zákonem o kybernetické | Systém SDAT splňuje bezpečnostní požadavky pro významné informační systémy dané zákonem č. 181/2014 Sb., o kybernetické | Závazný | 1 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|-----------------|---|--|------------|-----------|
| | bezpečnosti (Compliance – ZoKB) | bezpečnosti, ve znění pozdějších předpisů. | | |
| 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 |
| BEZ_14.0 | Hodnoty údaje – fyzické oddělení úložišť dat | Systém je koncipován tak, že umožňuje na úrovni databázové vrstvy uložit odděleně hodnoty údajů pro označené výkazy do separátních fyzických objektů databáze. Důvodem pro oddělení úložišť uvedených údajů od běžně zpracovávaných údajů jsou zvýšené bezpečnostní požadavky, které pro některé výkazy stanovuje legislativa již na úrovni jejich uložení (šifrování, zvláštní režim přístupu apod.). | Závazný | 1 |

4.3 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: <ul style="list-style-type: none"> všech operacích provedených v systému souborů a dat, | Závazný | 1 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|--------------|---------------------------------------|---|------------|-----------|
| | | <ul style="list-style-type: none"> • uživateli, který operaci iniciuje nebo provádí, • datu a času této události. | | |
| 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 |
| AUD_4.0 | Tisk auditních logů | Systém musí správci umožňovat vytištění auditního logu. | Závazný | 1 |
| AUD_5.0 | Manuální export auditních logů | Systém musí správci umožňovat export auditního logu v čitelné podobě např. csv (comma-separated values). | Závazný | 1 |
| AUD_7.0 | Auditní události (přiřazení přístupu) | <p>Systém musí zaznamenávat auditní události řízení přístupu:</p> <ul style="list-style-type: none"> • přidání oprávnění (např. zařazení uživatele na uživatelské místo), • odebrání oprávnění (např. odstraňování uživatele z uživatelského místa nebo odebrání role od uživatelského místa a další podobné akce). | Závazný | 1 |
| AUD_8.0 | Auditní události (řízení přístupu) | <p>Systém musí zaznamenávat auditní události související s řízením přístupu:</p> <ul style="list-style-type: none"> • logování přístupu funkcionalitě a datům, včetně generování reportů pro následný monitoring, • přístup k datům: <ul style="list-style-type: none"> ○ 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 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|-----------------|--|---|------------|-----------|
| AUD_9.0 | Automatický export auditních logů | Systém musí umožňovat 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 – logování změn v oprávněních | Systém bude evidovat práva uživatelů/skupin provádět určité operace (aktivity) a bude logovat změny v tomto nastavení a bude umožňovat prezentaci tohoto nastavení (včetně nastavení platného v minulosti). | Závazný | 1 |
| 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 | U všech auditovaných akcí se zaznamenávají i neúspěšné pokusy, např. z důvodu nedostatku oprávnění. | Závazný | 1 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|--------------|-------------------------------------|--|------------|-----------|
| | akcí | | | |
| AUD_17.0 | Autentifikace – historie přihlášení | <p>Systém sbírá a uchovává informace o historii přihlášení a odhlášení, zaznamenává i neúspěšné pokusy o přihlášení. V rámci tohoto požadavku sbírá následující informace:</p> <ul style="list-style-type: none"> 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 přihlášení. | Závazný | 1 |
| AUD_18.0 | Audit zjednodušený – | <p>Každý záznam uložený v databázi obsahuje následující informace:</p> <ul style="list-style-type: none"> informace o uživateli, který záznam vytvořil (ukládáno bude systémové ID uživatele, nikoli username, který se může v čase měnit. Prezentován však bude reálný username platný k okamžiku prohlížení logu.), časové razítko vytvoření záznamu, informace o uživateli, který záznam naposledy změnil, časové razítko poslední změny záznamu. <p>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. Představa o prezentaci v aplikaci je taková, že například tam, kde je záznam (řádek databázové tabulky) zobrazen v UI v komponentě grid, bude možnost výběru daného záznamu a například vyvolání kontextového menu přes pravé tlačítko myši, vybrání volby „Auditní data“ a v pop-up okně zobrazení těchto auditních informací.</p> | Závazný | 1 |
| AUD_22.0 | Logování – aplikační vrstva | Systém (míněno aplikace, nikoli nástroje RDBMS) umožňuje logování činnosti systému s možností nastavit si úroveň logování s tím, že na nejvyšší úrovni (default pro produkci) budou logovány všechny zprávy | Závazný | 1 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|--------------|-----------------|---|------------|-----------|
| | | <p>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.</p> <p>Aplikačně je požadováno logovat tato data:</p> <ul style="list-style-type: none"> • Jaká aktivita byla spuštěna. • Kdy byla aktivita spuštěna. • Jakým uživatelem byla aktivita spuštěna (systémové ID uživatele + v tu dobu platný username). • Jaké měla aktivita vstupní parametry (seznam hodnot parametrů, které byly uživatelem předány aplikaci při provedení dané aktivity). <p>Aplikační log je uživatelsky dostupný přímo v aplikaci SDAT pro prohlížení, vyhledávání, filtrování atd. Aplikační log je dostupný prostřednictvím objektu Aplikačního datového rozhraní. Pokud se v požadavku mluví o různých úrovních logování, má se za to, že SDAT bude umožňovat logovat různou úroveň detailu (standardně budou například logovány aktivity a jejich parametry, při vyšší úrovni logování by se měly logovat i prováděné SQL příkazy).</p> <p>Systém umožňuje aplikačním rozhraním uživateli nastavit/změnit úroveň logování, případně ho úplně vypnout.</p> <p>Systém umožňuje přístup k logům přes aplikační rozhraní.</p> <p>Systém umožňuje definovat vyhrazené místo na disku, které mohou logy zabírat a definovat, co se má stát v případě, že toto místo bude zaplněno.</p> | | |
| AUD_23.0 | Auditní log – | Auditní záznam musí být v souladu se záznamovou povinností § 13 | Závazný | 1 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|-----------------|---|--|------------|-----------|
| | legislativní správnost | odst. 3 písm. c) zákona č. 101/2000 Sb. | | |
| AUD_24.0 | Auditní log – import do systému SIEM | Vybrané auditní záznamy musí být možné průběžně v reálném čase importovat do systému SIEM. | Závazný | 1 |
| 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 |

4.4 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 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|----------------|---|---|------------|-----------|
| 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 prostředí | Desktop aplikace musí pracovat na terminálovém serveru Windows s nadstavbou Citrix XAserver. | Závazný | 1 |
| 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 | <p>Systém je schopen zpracovat Vstupní zprávu dle následujících provozních parametrů:</p> <ul style="list-style-type: none"> • 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: <ul style="list-style-type: none"> ○ 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, ○ zpracovávat 1 vstupní zprávu o maximální velikosti 5 GB s tím, že taková Vstupní zpráva bude zpracována nejpozději | Závazný | 1 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|--------------|-----------------|--|------------|-----------|
| | | 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. | | |

4.5 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 dostupné ze sítě Internet a současně přímo ze systémového prostředí ČNB (viz kapitola 2.1.1 Produkční prostředí). | Závazný | 2 |
| 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 | Systém SDAT umožňuje synchronizaci obsahu produkčního a testovacího prostředí podle popisu v kapitole 2.1.2 Testovací prostředí . Systém SDAT umožňuje proces synchronizace spouštět automaticky | Závazný | 2 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|----------------|---|---|------------|-----------|
| | | v naplánovaných periodách nebo ad-hoc na pokyn uživatele. | | |
| PPR_2.2 | Testovací prostředí - geocluster | 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_2.3 | 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 | Školící prostředí | V rámci architektury systému SDAT existuje školící prostředí dostupné v rámci systémového prostředí ČNB (viz kapitola 2.1.4 Školící prostředí). | Závazný | 3 |
| PPR_4.1 | Školící prostředí – synchronizace | Systém umožňuje provádět synchronizaci obsahu školících a produkčních instancí komponent. Systém SDAT umožňuje spouštět proces synchronizace na ad-hoc | Závazný | 3 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|----------------|--------------------------------------|--|------------|-----------|
| | | pokyn uživatele. | | |
| PPR_4.2 | 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_4.3 | Akceptační prostředí - synchronizace | Pokud charakter provedených programových změn dovolí (např. nedojde k zásadní změně datového modelu databázové části systému), systém umožňuje provedení synchronizace obsahu z produkčních do akceptačních instancí komponent a to na ad-hoc bázi. | Závazný | 2 |
| 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: <ul style="list-style-type: none"> 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 |

4.6 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 požadavek. | Závazný | 1 |
| 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: <ul style="list-style-type: none"> 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 | Závazný | 1 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|----------------|---------------------------|---|------------|-----------|
| | | skutečnost uživateli jasně prezentována. | | |
| URO_6.0 | Klávesové zkratky | Aplikace umožňuje často prováděné akce realizovat pomocí klávesových zkratk. | Závazný | 1 |
| URO_7.0 | Validace dat – základní | <p>Systém validuje vstupní data uživatele na správnost zadaného typu dat bez nutnosti kontaktovat server. Zejména se jedná o tyto validace:</p> <ul style="list-style-type: none"> • 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. <p>Splněním tohoto požadavku nejsou dotčeny další validace, které vyplývají z aplikační logiky dané oblasti aplikace.</p> <p>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”.</p> | Závazný | 1 |
| URO_8.0 | Validace dat – specifické | <p>Kromě základního způsobu validace na formát zadávaných data systém disponuje dalšími specifickými validacemi:</p> <ul style="list-style-type: none"> • pokud se se v uživatelském rozhraní vyskytují dva datумы, 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). <p>Pokud budou výše uvedená pravidla porušena, systém takový vstup</p> | Závazný | 1 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|-----------------|---|--|------------|-----------|
| | | 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. | | |
| 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 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|--------------|---|---|------------|-----------|
| URO_10.1 | Zadávací pole pro datum a čas – výběr hodnoty z kalendáře | <p>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.</p> <p>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).</p> | Závazný | 1 |
| 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 přecíst neoprávněným uživatelem. | Závazný | 1 |
| URO_12.0 | Odesílání nevalidních dat ke zpracování | <p>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.</p> <p>Tento způsob validace dat neznamena, 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.</p> | Závazný | 1 |
| URO_13.0 | Potvrzení provedené akce | <p>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.</p> <p>Zároveň systém umožňuje uživateli prohlásit, že v budoucnu pro danou</p> | Závazný | 1 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|-----------------|---|---|------------|-----------|
| | | akci již nechce podobná hlášení zobrazovat. Pokud uživatel tuto volbu zaškrtně, při další akci už nebude hlášení zobrazeno. | | |
| URO_14.0 | Potvrzení provedené akce – možnost měnit preference | Informace o tom, že se dané hlášení (viz URO_13.0) již nebude pro danou akci v budoucnu zobrazovat, je uložena v uživatelské konfiguraci, která je přístupná uživateli a uživatel může kdykoli svoji preferenci změnit (může si opět nastavit, že potvrzovací dialog zobrazovat pro určitou akci chce). | Důležitý | 2 |
| URO_15.0 | Generátor sestav | <p>Systém disponuje nástrojem, pomocí kterého lze uživatelským způsobem tvořit tiskové sestavy. Nástroj musí umožňovat</p> <ul style="list-style-type: none"> • Vytvořit definici zdroje dat, která budou předmětem zobrazení v tiskové sestavě (SQL příkaz „SELECT“) • Vytvářet takové definice zdroje dat, které obsahují parametry, které bude následně vyplňovat uživatel při spuštění tiskové sestavy • Vytvářet předlohu (layout) tiskové sestavy, která bude použita pro zobrazení dat <p>Výsledné sestavy musejí respektovat přístupová práva, tzn. není přípustné, aby uživatel jejich spuštěním získal jiná data, než ke kterým má definováno oprávnění.</p> | Závazný | 2 |

4.6.1 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 bude moci toto pořadí přetažením myši změnit. Informace o změnách v pořadí sloupců je zaznamenána do uživatelské konfigurace a je upřednostněna před defaultním nastavením (viz URO_3.0). | Závazný | 1 |
| 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 | Závazný | 1 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|----------------|---|---|------------|-----------|
| | | 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. | | |
| 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 | <p>Data zobrazená v tabulce budou defaultně seřazena podle právě jednoho sloupce (pokud sloupec obsahuje data, které lze z jejich povahy řadit).</p> <p>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.</p> <p>Pokud je hodnota sloupce, podle kterého je prováděno řazení, rovna NULL, pak je takový záznam zařazen na konec seznamu.</p> <p>Uživatel může kdykoli toto řazení změnit kliknutím na záhlaví 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.</p> <p>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.</p> | Závazný | 1 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|-----------------|---|---|------------|-----------|
| 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 filtrační kritérium, má se za to, že mezi filtračními 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 filtračnímu kritériu (ekvivalentní k SQL konstrukci <code>nazev_sloupce like '%<filtrační kritérium>%'</code>). | 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: <ul style="list-style-type: none"> větší než „>“ menší než „<“ rovno nebo větší než „>=“ rovno nebo menší než „<=“ rovná se „=“ nerovná se „!=“ V případě, že uživatel žádný operátor nepoužije, má se za to, že použil operátor „rovná se“. | Závazný | 1 |
| 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. | Závazný | 1 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|-----------------|---------------------------------|--|------------|-----------|
| | | Sloupec obsahující binární data, resp. zástupné ikony nepodléhá řazení (GRI_7.0 pro něj neplatí) | | |
| 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: <ul style="list-style-type: none"> • 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 |

4.6.2 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 | Rozbalovací seznam umožňuje: | Závazný | 1 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|----------------|--------------------------|--|------------|-----------|
| | hodnot | <ul style="list-style-type: none"> výběr žádné nebo právě jedné hodnoty ze seznamu dostupných hodnot, výběr žádné, jedné nebo více hodnot ze seznamu dostupných hodnot. | | |
| CBX_2.0 | Režim práce s hodnotami | <p>Rozbalovací seznam pracuje v režimu:</p> <ul style="list-style-type: none"> 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 | <p>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.:</p> <ul style="list-style-type: none"> 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 like '%<filtrovací kritérium>%'). | Závazný | 1 |
| 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 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|----------------|-------------------|--|------------|-----------|
| 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 |

5 Katalog funkčních požadavků

5.1 Katalog funkčních požadavků pro Komunikační modul

| ID požadavku | Název Požadavku | Popis požadavku | Důležitost | Kategorie |
|----------------|-----------------|---|------------|-----------|
| KOM_1.0 | Vytvoření úkolu | <p>Systém umožňuje přímo z aplikace vytvoření nového požadavku (úkolů) minimálně v následujícím rozsahu:</p> <ul style="list-style-type: none"> • název úkolu, • slovní popis úkolu, • priorita úkolu (číselník), • datum požadovaného vyřešení úkolu, • řešitel, • sledovatelé, • přílohy (0..N binárních souborů). | Závazný | 3 |
| KOM_1.1 | Výběr | Systém umožňuje při vytváření nového požadavku (úkolů) přímo | Závazný | 3 |

| ID požadavku | Název Požadavku | Popis požadavku | Důležitost | Kategorie |
|--------------|---------------------------|--|------------|-----------|
| | řešitele/sledovatele | 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. | | |
| KOM_2.0 | Seznam úkolů | <p>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.</p> <p>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“.</p> <p>Požadavky (úkoly) jsou čerpány přímo z nástroje pro evidenci požadavků (úkolů) přes API.</p> | Závazný | 3 |
| KOM_2.1 | Seznam úkolů – filtrování | <p>Systém umožňuje filtrovat zobrazené požadavky (úkoly) dle KOM_2.0 minimálně v tomto rozsahu:</p> <ul style="list-style-type: none"> • 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. | Závazný | 3 |

| ID požadavku | Název Požadavku | Popis požadavku | Důležitost | Kategorie |
|--------------|---------------------------------|---|------------|-----------|
| KOM_3.0 | Seznam úkolů – akce k provedení | <p>Systém pro každý jeden vybraný požadavek (úkol) umožňuje provést tyto akce:</p> <ul style="list-style-type: none"> změnit stav požadavku (úkol) na jiný v souladu s tabulkou č. 1 Přehled stavů zpracování požadavku (úkol). 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_7.0 | Notifikace při změně řešitele | Systém (task management systém) odešle notifikační e-mail v případě, že během životního cyklu požadavku (úkol) dojde ke změně řešitele (viz KOM_3.0). | Závazný | 3 |

5.2 Administrační modul

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

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|--------------|---|--|------------|-----------|
| 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 umožňují 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 číselníkům. | 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 záznamech databázových objektů, zaplnění databázových prostorů, zaplnění aplikačních disků apod. | Závazný | 3 |

5.3 SIPRES

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|----------------|-------------------------------------|---|------------|-----------|
| S2S_1.0 | Volná datová oblast (tzv. formulář) | Systém umožňuje označení datové oblasti atributem „volná“. Údaje v takto označené oblasti lze umísťovat nezávisle na matici danou kombinací Ukazatelů a jejich Parametrů rozmístěných na osy tabulky (běžný přístup při standardním projektování) v souladu s pravidly popsány v kapitole 3.2 Řešení specifické funkcionality SIPRES v SDAT . | Závazný | 1 |
| S2S_1.1 | Ovládací prvky volné DO | Systém pro každý údaj, který je do volné datové oblasti zařazen, navrhne defaultní ovládací prvek, seznam typických prvků je uveden v kapitole 3.2.1 Volná datová oblast typu formulář . | Závazný | 1 |
| S2S_1.2 | Popisy údajů ve volné DO | Systém umožňuje ke každému formulářovému prvku typu údaj přiřadit popis (label), který má defaultně hodnotu názvu Ukazatele, kterým je Údaj definován. Další práce s popisem je uvedena v kapitole 3.2.1 Volná datová oblast typu formulář . | Závazný | 1 |
| S2S_1.3 | Rozměry vstupních polí ve volné DO | Systém umožňuje pro jednotlivé formulářové prvky ve volné DO stanovit jejich rozměr, viz bod e) kapitoly 3.2.1 Volná datová oblast typu formulář . | Závazný | 1 |
| S2S_1.4 | Kontextová nápověda | Jednotlivé formulářové prvky v DO dle S2S_1.0 mají dostupnou kontextovou nápovědu, viz bod f) kapitoly 3.2.1 Volná datová oblast typu formulář . | Závazný | 1 |
| S2S_1.5 | Dodatečné popisy ve volné DO | Systém umožňuje umístit mezi jednotlivé prvky formuláře libovolný popisný text, viz bod g) kapitoly 3.2.1 Volná datová oblast typu formulář . | Závazný | 1 |
| S2S_2.0 | Datový typ „příloha“ | Systém umožňuje projektantovi založit údaj s využitím datového typu „soubor“, viz kapitola 3.2.2 Přílohy údajů . | Závazný | 1 |

| ID požadavku | Název požadavku | Popis požadavku | Důležitost | Kategorie |
|----------------|--|---|------------|-----------|
| S2S_3.0 | Kontrola typu „vizuální kontrola souboru“ | Systém umožňuje zavedení kontroly typu „vizuální kontrola souboru“, která spočívá v zobrazení obsahu příloh dle S2S_2.0 a posouzení jeho validity (tj. zda bylo zasláno to, co mělo být zasláno). | Závazný | 1 |
| S2S_3.1 | Příjem souborů s kontrolou dle S2S_3.0 | Systém provede při přijetí zprávy, která obsahuje údaj s definovanou kontrolou dle S2S_3.0, všechny ostatní kontroly a pokud jsou tyto splněny, nastaví stav zprávy „Předběžně platné – čeká se na provedení vizuální kontroly“. Výskyt výkazu je převeden do stavu 70 – nedokončený, viz kapitola 3.2.6 Vizualní kontrola příloh Údajů . | Závazný | 1 |
| S2S_3.2 | Provedení kontroly S2S_3.0 | Systém umožňuje provedení kontroly S2S_3.0, tj. gestorovi výkazu nebo administrátorovi systému je umožněno zobrazit obsah souboru přiloženého k údaji a na základě posouzení obsahu převést stav vydání výskytu výkazu do 60 – Platné a výskyt výkazu do 60 – splněný, případně naopak nastavit stav vydání výskytu výkazu na 80 – závažná chyba, viz kapitola 3.2.6 Vizualní kontrola příloh Údajů . | Závazný | 1 |
| S2S_4.0 | Dopočítání hodnoty údaje v rámci formuláře | Systém umožňuje zavedení kontrol, které jsou vykonány dynamicky (tj. přímo ve formuláři zobrazeném na straně klienta), podobně jako jsou v MS Excel prováděny výpočty vzorců. Číselný výsledek kontroly je vložen do příslušné buňky údaje. Výpočet je pouze „pomocný“ a vypočtenou hodnotu lze uživatelsky přepsat. V případě výkazů se jedná o součtové řádky a funkcionality bude zajištěna projektováním jako dynamické DO se součtovým řádkem, viz také kapitola 3.2.3 Jednovýkazové kontroly pro dopočítání hodnoty údaje v rámci formuláře . | Závazný | 1 |
| S2S_5.0 | Dynamické zobrazování datových oblastí | Systém umožňuje pro dynamické datové oblasti vytknout parametr, podle kterého lze měnit zobrazení údajů v datové oblasti dle popisu v kapitole 3.2.4 Dynamické zobrazování datových oblastí . | Závazný | 1 |