Integrace REST API na validaci závěrečných prací oproti standardu PDF/A

Bc. Vojtěch Vagunda

Diplomová práce

2021

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav informatiky a umělé inteligence

Akademický rok: 2020/2021

ZADÁNÍ DIPLOMOVÉ PRÁCE
(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Vojtěch Vagunda**
Osobní číslo:	**A20650**
Studijní program:	**N3902 Inženýrská informatika**
Studijní obor:	**Informační technologie**
Forma studia:	**Kombinovaná**
Téma práce:	**Integrace REST API na validaci závěrečných prací oproti standardu PDF/A**
Téma práce anglicky:	**The Integration of a REST API for the Validation of Theses Against the PDF/A Standard**

Zásady pro vypracování

1. Proveďte literární rešerší na dané téma.
2. Shromážděte požadavky na rozhraní.
3. Navrhněte technické řešení ve formě webové aplikace.
4. Zdůvodněte výběr jednotlivých komponentů technického řešení.
5. Realizujte a otestujte výsledné technického řešení ve spolupráci s uživatelem.
6. Věnujte pozornost zabezpečení aplikace.

Forma zpracování diplomové práce: **Tištěná/elektronická**

Seznam doporučené literatury:

1. Architectural Styles and the Design of Network-based Software Architectures. ics.uci.edu [online], 2004 [cit. 2020-11-16], Dostupné z: https://www.ics.uci.edu/ fielding/pubs/dissertation/top.htm
2. Web Services Architecture. w3c.org [online], 2004 [cit. 2020-11-16], Dostupné z: https://www.w3.org/TR/2004/NOTE-ws-arch-20040211
3. Standard ECMA-404: The JSON Data Interchange Syntax. Ecma International [online]. 2017, December 2017 [cit. 2020-11-16]. Dostupné z: https://ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf
4. INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. Document management — Electronic document file format for long-term preservation: Part 1: Use of PDF 1.4 (PDF/A-1). 2005, 29 s. 1. ISO 19005-1:2005. Dostupné také z: https://www.iso.org/standard/38920.html
5. VeraPDF. Open Preservation Foundation [online]. 2020, May, 2020 [cit. 2020-11-16]. Dostupné z: https://openpreservation.org/products/verapdf/

Vedoucí diplomové práce: **doc. Ing. Jiří Vojtěšek, Ph.D.**
Ústav řízení procesů

Konzultant diplomové práce: **Ing. Ivan Masár**
Ústav informatiky a umělé inteligence

Datum zadání diplomové práce: **15. ledna 2021**
Termín odevzdání diplomové práce: **17. května 2021**

doc. Mgr. Milan Adámek, Ph.D. v.r.
děkan

L.S.

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky
DĚKAN

prof. Mgr. Roman Jašek, Ph.D. v.r.
ředitel ústavu

Ve Zlíně dne 15. ledna 2021

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně;
- byl/a jsem seznámen/a s tím, že na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové práce využít ke komerčním účelům;

  beru na vědomí, že pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne …….

podpis studenta

ABSTRAKT

Tato práce se zabývá výběrem vhodné aplikace umožňující validaci PDF souborů na normu PDF/A pro studijní systém IS/STAG. Dále pak vývojem dodatečných funkcionalit, které souvisejí s validaci PDF/A požadovaných Univerzitou Tomáše Bati ve Zlíně. Teoretická část je věnována přehledu zákonů a vyhlášek, které řeší skartační lhůtu, archivaci a skartaci digitálních dokumentů a závěrečných vysokoškolských prací. Jsou také probrány obecně vhodné formáty souborů pro archivaci textových, obrázkových a audio souborů, rozdíly mezi PDF a PDF/A, probrána základní struktura PDF 1.4 a popsány odlišnosti mezi archivními formáty PDF/A-1, PDF/A-2, PDF/A-3, PDF/A-4. Poslední část je věnována přehledu existujících nástrojů pro validaci PDF/A souborů na trhu. Výstupem praktické části je REST API zprostředkovávající validaci PDF souborů na normu PDF/A pro studijní systém IS/STAG.

Klíčová slova: VŠKP, PDF, PDF/A, ISO-19005, veraPDF, veraPDF-rest, IS/STAG, stag-pdfa,

ABSTRACT

This work deals with the selection of a suitable application enabling the validation of PDF files to the PDF / A standard for the IS / STAG study system. Furthermore, the development of additional functionalities related to PDF / A validation which are required by Tomas Bata University in Zlín. The theoretical part is devoted to an overview of laws and decrees that address: shredding deadline, archiving and shredding of digital documents and final university theses. Generally suitable file formats for archiving text, images and audio files are also discussed, differences between PDF and PDF / A, then is discussed the basic structure of PDF 1.4 and differences between archive formats PDF / A-1, PDF / A-2, PDF / A-3, PDF / A-4. The last part is devoted to an overview of existing tools for validation of PDF / A files on the market. The output of the practical part is the REST API providing the validation of PDF files to the PDF / A standard for the IS / STAG study system.

Klíčová slova: VŠKP, PDF, PDF/A, ISO-19005, veraPDF, veraPDF-rest, IS/STAG, stag-pdfa,

Chtěl bych poděkovat vedoucímu mé diplomové práce doc. Ing. Jiřímu Vojtěškovi, Ph. D. za odborné vedení, poskytnuté rady, trpělivost, ochotu a čas, které mi v průběhu zpracování diplomové práce věnoval. Dále bych chtěl poděkovat svému konzultantovi Ing. Ivanu Masárovi, který dohlížel na proces celého vývoje a byl ochoten odborně poradit a nasměrovat při jakýchkoliv technických nejasnostech.

Prohlašuji, že odevzdaná verze bakalářské/diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

OBSAH

ÚVOD **10**

I TEORETICKÁ ČÁST **12**

1 ARCHIVACE VYSOKOŠKOLSKÝCH ZÁVĚREČNÝCH PRACÍ Z POHLEDU ZÁKONA A VYHLÁŠEK **13**

1.1 ZÁKON O ARCHIVNICTVÍ 13

1.2 ZÁKON Č. 111/1998 SB. O VYSOKÝCH ŠKOLÁCH 14

1.2.1 Institucionární repozitáře 14

1.2.2 Studijní informační systémy 14

1.2.3 Databáze na stránkách vysokých škol 15

1.3 VYHLÁŠKA O PODROBNOSTECH VÝKONU SPISOVÉ SLUŽBY 15

1.4 UKLÁDÁNÍ A ARCHIVACE DIGITÁLNÍCH DOKUMENTŮ 16

1.5 VYŘAZOVÁNÍ DOKUMENTŮ V DIGITÁLNÍ PODOBĚ 16

1.6 SKARTAČNÍ LHŮTA A ARCHIVACE VŠKP NA UTB 17

2 VHODNÉ FORMÁTY SOUBORŮ PRO ARCHIVACI **19**

2.1 PROPRIETÁRNÍ FORMÁT 19

2.2 OTEVŘENÉ FORMÁTY 19

2.3 ZTRÁTOVÉ, BEZZTRÁTOVÉ A VEKTOROVÉ FORMÁTY 20

2.4 VHODNÉ FORMÁTY SOUBORŮ PRO ARCHIVACI U VYBRANÝCH TYPŮ DAT 21

3 DIGITÁLNÍ PŘENOSITELNÉ FORMÁTY PDF VS PDF/A **25**

4 PDF/A-1, ISO 19500-1 ..... **26**

5 STRUKTURA PDF 1.4 ..... **28**

5.1 OBJEKTY 28

5.2 STRUKTURA PDF SOUBORU 31

5.2.1 Hlavička (Header): 32

5.2.2 Tělo (Body): 32

5.2.3 Tabulka křížových odkazů (Cross-reference table) 32

5.2.4 Patička (Trailer) ... 33
5.3 Struktura PDF dokumentu ... 34
5.4 Proud obsahu (Content stream) ... 36
6 PDF/A-2 ISO 19005-2 ... 38
7 PDF/A-3 ISO 19005-3 ... 39
8 PDF/A-4, ISO 19005-4 ... 40
8.1 PDF/A-4f ... 41
8.2 PDF/A-4e ... 41
9 PRŮZKUM EXISTUJÍCÍCH MOŽNOSTÍ VALIDACE PDF/A ... 42
9.1 Placené ... 42
9.1.1 3-Heights(TM) PDF Validator ... 43
9.1.2 Callas PDFa Pilot ... 43
9.1.3 jPDFPreflight (Qoppa Software) ... 45
9.1.4 PDF Automation Server (Qoppa Software) ... 45
9.1.5 PDFTron SDK, knihovna PDF/A ... 46
9.2 Neplacené ... 47
9.2.1 Služba validace PDF/A na stránkách Národního archivu ... 47
9.2.2 Online testovací verze validátoru Heights(TM) PDF Validator ... 48
9.2.3 Verify PDF/A ... 49
9.2.4 PDF to PDF/A Converter PDFTron ... 49
9.2.5 Apache PDFBox ... 50
9.2.6 VeraPDF ... 50
9.3 Výběr vhodného řešení ... 52
II PRAKTICKÁ ČÁST ... 54
10 AKTUÁLNÍ STAV ... 55
10.1 Současný stav validace PDF/A na univerzitách používající IS/STAG ... 55
10.2 Historický a současný stav validace PDF/A na UTB ... 55

10.3 AKTUÁLNÍ POŽADAVKY NA VALIDACI PDF/A NA UTB 56
11 NÁVRH ŘEŠENÍ 58
11.1 POPIS KONKRÉTNÍHO NÁVRHU 59
12 EXISTUJÍCÍ APLIKACE 61
12.1 IS/STAG 61
12.2 VERAPDF-REST 63
12.2.1 Pravidla kontroly PDF/A 64
12.2.2 Požité technologie 66
12.2.3 Endpointy veraPDF-rest 67
12.3 STAG-PDFA 69
12.3.1 Funkce stag-pdfa 69
12.3.2 Spuštění stag-pdfa na localhost 72
12.3.3 Konfigurační soubor *config.yml* 73
12.3.4 Api stag-pdfa 75
12.3.5 Třídy stag-pdfa 77
12.3.6 Nasazení stag-pdfa v ostrém provozu 79
12.3.7 Zabezpečení 79
12.3.8 Maximální možnosti stag-pdfa 80
ZÁVĚR 82
SEZNAM POUŽITÉ LITERATURY 84
SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK 89
SEZNAM OBRÁZKŮ 90
SEZNAM TABULEK 91
SEZNAM PŘÍLOH 92

ÚVOD

V dnešní době nabízí službu exportu dokumentů do formátu PDF/A celá řada aplikací pro práci s textovými dokumenty a obrázky. Nicméně je velmi žádoucí takto exportované soubory ještě dodatečně validovat, zda neobsahují možné skryté chyby ve své souborové struktuře, které by mohly v budoucnu způsobit, že dojde k nečitelnosti části obsahu dokumentu, nebo v horším případě nebude možné přečíst dokument vůbec. Právě kvůli požadavkům na validaci formátu PDF/A vzešlo téma této práce.

Teoretická část práce objasní, proč je důležité řešit validaci závěrečných vysokoškolských prací proti standardu PDF/A při jejich uložení do elektronického systému spisové služby, zejména z pohledu zákona č. 499/2004 Sb., o archivnictví a spisové službě a vyhlášky č. 259/2012 Sb., o podrobnostech výkonu spisové služby. Dále pak bude probrán obecně výběr vhodných datových formátů pro archivaci textových, obrázkových a audio dokumentů, objasnění rozdílů mezi PDF, PDF/A a také odlišnosti jednotlivých standardů PDF/A-1. PDF/A-2, PDF/A-3, PDF/A-4. Teoretická část přinese také přehled placených a neplacených softwarových řešení, které je možno použít, buď pouze pro validaci PDF/A, nebo pro opravu PDF/A souborů prohlášených validátorem za chybné.

Praktická část reflektuje přímý požadavek Univerzity Tomáše Bati ve Zlíně, kde se místní IT správci snaží hledat již od roku 2017 stabilní, efektivní a cenově dostupný nástroj pro kontrolu závěrečných vysokoškolských prací na formát PDF/A. Z jejich pohledu by bylo ideální, aby službu validace prováděl přímo informační systém IS/STAG ve stavu, kdy student nahrává závěrečnou vysokoškolskou práci.

Vývojovým týmem IS/STAG byl v minulosti proveden jeden pokus o implementaci validátoru veraPDF přímo do IS/STAG, nicméně pokus se nezdařil. Důvody, proč byl neúspěšný, budou popsány dále v práci. Díky tomuto pokusu vykrystalizoval požadavek na možnost definovat seznam pravidel pro formát PDF/A, která nebudou validátorem kontrolována. Vzhledem k tomu, že UTB byla pouze jediným zákazníkem, který validaci odevzdávaných závěrečných vysokoškolských prací po IS/STAG požadoval, vývojový tým upustil od dalšího vývoje této funkcionality. Problém byl tehdy uzavřen s tím, že pokud v budoucnu UTB poskytne nezávisle běžící validační službu, na kterou by IS/STAG mohl posílat požadavky, pak vývojáři IS/STAG zařadí validaci PDF/A do IS/STAG tak, aby při odevzdání každé závěrečné práce oznámil IS/STAG studentovi, zda je odevzdaná práce validní s formátem PDF/A.

Hlavní cíle praktické části jsou tedy tři. Prvním je výběr vhodné aplikace pro validaci PDF/A souborů, která bude nejvíce vyhovovat požadavkům UTB. V případě, že některé požadavky nebude aplikace splňovat, tak rozhodnout, zda zasáhnout do zdrojového kódu této aplikace a modifikovat ji, nebo navrhnout a vyvinout vlastní aplikační mezivrstvu, která bude reflektovat požadavky UTB a bude sloužit jako prostředník při komunikaci mezi IS/STAG a vybranou aplikací provádějící validaci PDF/A.

Druhým cílem je vyvinout a začlenit funkcionalitu, která by umožňovala definovat seznam výjimek pravidel na formát PDF/A, jež budou vyloučena z kontroly.

Třetím cílem je dohodnout s vývojáři IS/STAG formát POST požadavků, které bude IS/STAG zasílat na validační službu a formát jednoduchých odpovědí, které bude validační služba vracet do IS/STAG.

I. TEORETICKÁ ČÁST

1 ARCHIVACE VYSOKOŠKOLSKÝCH ZÁVĚREČNÝCH PRACÍ Z POHLEDU ZÁKONA A VYHLÁŠEK

Archivace vysokoškolských závěrečných prací (VŠKP) je důležitá zejména ze dvou důvodů. První důvod je zajištění možnosti navazovat na předchozí vysokoškolské práce, například v situaci, kdy vysokoškolská práce rozvinula zajímavé téma, které o několik let později někoho zaujme a objeví se snaha toto téma dále rozvinout. Druhý důvod je zajištění možnosti zpětné kontroly u pochybně získaných vysokoškolských titulů, kdy je potřeba ověřit, zda VŠKP díky které dotyčná osoba získala vysokoškolský titul, není plagiátem. Proto je důležité, aby bylo nakládání s VŠKP jednotně regulováno. Následující podkapitoly se zaměří na části zákonů a vyhlášek, které stanovují vysokým školám povinnosti, jak má být se závěrečnými VŠKP nakládáno.

1.1 Zákon o archivnictví

Vysoké školy mají podle § 3 odst. 1 písm. i) zákona č. 499/2004 Sb., o archivnictví a spisové službě povinnost uchovávat dokumenty a umožnit výběr archiválií. To, kým budou práce spravovány ve skartační lhůtě, je v kompetenci dané vysoké školy. Skartační lhůta je lhůta od vyřízení písemnosti (od dne obhajoby) do začátku skartačního řízení. Skartační lhůta je na každé škole různě dlouhá a je stanovena v jejím spisovém a skartačním řádu. Po uplynutí skartační lhůty by měla škola navázat kontakt s příslušným archivem a dohodnout se, kde budou práce archivovány. Můžou nastat dvě následující možnosti.

a) Vysoké školy bez vlastního univerzitního archivu

Školy, které nemají vlastní univerzitní archiv zřízen, se po uplynutí skartační lhůty řídí svým skartačním a spisovým řádem. Dále musí navázat kontakt s příslušným oblastním archivem, se kterým se dohodnou, kde budou práce archivovány.

b) Vysoké školy s vlastním univerzitním archivem

Podle § 51 odst. 1 zákona č. 499/2004 Sb., o archivnictví a spisové službě můžou zřizovat specializované archivy také přímo vysoké školy. Pokud vysoká škola má vlastní specializovaný univerzitní archiv, tak správu skartačního řízení má na starosti tento archiv. Po uplynutí skartační lhůty je na archivu dané školy jak se s VŠKP naloží.

Pokud vysoká škola chce mít práce uloženy k provozní potřebě déle než je skartační lhůta, tak se může vysoká školy dohodnout s daným archivem na tom, že práce budou drženy v rámci vysoké školy (organizace) i déle, než je skartační lhůta. [1], [2]

Univerzita Tomáše Bati ve Zlíně (UTB) spadá do kategorie a), tedy vysoká škola bez vlastního univerzitního archivu. Skartační lhůta a proces zpracování VŠKP na UTB bude podrobněji popsán v kapitole 1.7.

1.2 Zákon č. 111/1998 Sb. o vysokých školách

Dle §47b podle odst. (1) zákona č. 111/1998 Sb., o vysokých školách, je vysoká škola povinna nevýdělečně zveřejňovat diplomové, bakalářské, rigorózní a disertační práce u kterých proběhla obhajoba, zároveň s přiloženými posudky oponentů a záznamem o průběhu včetně výsledků obhajoby. Vysoká škola je povinna zveřejnit výše uvedené kvalifikační práce pomocí databáze s kvalifikačními pracemi, kterou spravuje. Způsob zveřejnění si stanoví každá vysoká škola svým vnitřním předpisem. Na UTB je způsob zveřejnění VŠKP stanoven Směrnicí rektora č. 33/2019 v článku 4. [3]

Pro zveřejňování kvalifikačních prací existuje řada řešení lišících se hlavně mírou otevřenosti.

1.2.1 Institucionární repozitáře

Některé vysoké školy během posledních let zřídily svoje vlastní digitální institucionární repozitáře, kde ukládají VŠKP. Jedním z nejčastěji převažujících je DSpace. Institucionární repozitář DSpace provozuje například Univerzita Pardubice, Univerzita Tomáše Bati ve Zlíně a další.

1.2.2 Studijní informační systémy

Další možností pro zveřejňování VŠKP nabízejí některé studijní informační systémy. Hlavní výhoda je v tom, že systém má veškeré informace o závěrečné práci (posudky, plné texty…) zanesené v sobě, kvůli úkonům spojeným se studiem. Toto řešení využívá například Mendelova univerzita v Brně nebo Jihočeská univerzita v Českých Budějovicích. Jihočeská univerzita v Českých Budějovicích používá ke zveřejňování VŠKP informační

systém IS/STAG. Mendelova univerzita v Brně používá ke zveřejňování VŠKP vlastní informační systém UIS MENDELU [1]

1.2.3 Databáze na stránkách vysokých škol

Existují i školy, které zákonnou povinnost řeší pomocí jednoduché databázové aplikace na stránkách vysoké školy nebo její knihovny. Vhodným příkladem je například Vysoká škola polytechnická Jihlava, která řeší zákonnou povinnost pomocí databázové aplikace na stránkách vysoké školy na odkazu zde[2] [4].

1.3 Vyhláška o podrobnostech výkonu spisové služby

Při snaze nalézt, zda zákony v České republice nějak stanovují, v jakém digitálním datovém formátu by měla být závěrečná VŠKP uložena v systému elektronické spisové služby, byla objevena nejjasnější odpověď ve vyhlášce č. 259/2012 Sb., o podrobnostech výkonu spisové služby. Vyhláška pracuje s pojmem **výstupní datový formát dokumentu v digitální podobě**. Tímto pojmem se rozumí:

a) datový formát výstupu z elektronického systému spisové služby,

b) datový formát dokumentu ukládaného ve spisovně, která je součástí elektronického systému spisové služby,

c) datový formát pro předávání do digitálního archivu

Podle § 23 odst. 2 vyhlášky č. 259/2012 Sb., o podrobnostech výkonu spisové služby je výstupním datovým formátem statických textových dokumentů, statických kombinovaných textových a obrázkových dokumentů datový formát Portable Document Format for the Long-term Archiving (PDF/A), který je definován normou ISO 19005. [5]

Hlavním zájmem této práce bude tedy validace závěrečných VŠKP v digitální podobě ve formátu PDF/A.

[1] Dostupné z http: https://is.mendelu.cz/zp/?lang=cz

[2] Dostupné z http: https://knihovna.vspj.cz/bakalarske-prace/obhajene/nazev/polytech/studijni-obor/6501R001

1.4 Ukládání a archivace digitálních dokumentů

Jako základní pomůcka pro evidenci digitálních dokumentů v elektronické podobě je elektronický systém spisové služby. Elektronický systém spisové služby (eSSL) musí umožňovat správu dokumentů v digitální a analogové podobě. Dokumenty jsou uchovávány ve spisové službě po dobu skartační lhůty. Podle § 3 odst. 5 zákona č. 499/2004 Sb., o archivnictví a spisové službě se uchovávání rozumí zajištění věrohodnosti původu dokumentů, neporušitelnosti jejich obsahu a čitelnosti, tvorbu a správu metadat náležejících k těmto dokumentům v souladu s tímto zákonem a připojení údajů prokazujících existenci dokumentu v čase. Elektronický systém spisové služby by měl zajistit tuto správu hlavně prostřednictvím transakčního protokolu. Transakční protokol obsahuje otisky všech digitální dokumentů, které byly v daný okamžik zaevidovány do spisové služby nebo byly v daném okamžiku změněny. Na konci každého dne bývá obsah transakčního protokolu zafixován do elektronického dokumentu ve formátu PDF/A nebo Extensible Markup Language (XML) [1], [7], [6].

Po skončení skartační lhůty následuje skartační řízení, kde se rozhodne, zda budou dokumenty prohlášeny za archiválie a dále archivovány v archivu příslušejícímu dané organizaci, nebo vyřazeny a řádně skartovány ve skartačním řízení. Skartační řízení v eSSL má být prováděno tak, aby v ideálním případě systém zkontroloval skartační lhůty a připravil seznam dokumentů a spisů, kterým již skartační lhůty uplynuly. Výběr dokumentů, které budou prohlášeny za archiválie, provádí podle § 3 odst. 6 archiv podle své působnosti [1], [6].

1.5 Vyřazování dokumentů v digitální podobě

Vyřazované dokumenty a spisy vedené ve spisové službě nesmějí být zničeny bez řádného skartačního řízení. Skartační řízení začíná stavem, kdy systém elektronické spisové služby navrhne dokumenty nebo spisy v digitální podobě. Následně pracovník spisovny sestaví seznam z daných navržených dokumentů. Seznam je tvořen podle Národního standardu pro elektronické systémy spisové služby (NSESSS) na základě příloh 2 a 3 uvedených na webových stránkách NSESSS[3]. Výsledkem práce pracovníka spisovny

[3] Dostupné z http: https://www.mvcr.cz/clanek/narodni-standard-pro-elektronicke-systemy-spisove-sluzby.aspx ve formátu XML

je vytvoření datového balíčku Submission Information Balíčku (SIP). SIP balíčky obsahují zpravidla pouze metadata bez datových souborů.

K vytvořenému seznamu je připojen skartační návrh podepsán vedoucím pracovníkem. Vytvořený seznam je následně zaslán se skartačním návrhem přes eSSL do příslušného archivu. Archivář na základě skartačního návrhu vytvoří protokol o skartačním řízení. K protokolu o skartačním řízení přidá dále archivář seznam dokumentů vybraných za archiválie a seznam dokumentů určených k vymazání. Seznam dokumentů je ve formátu PDF/A a zároveň ve strojově čitelném souboru podle přílohy 4 NSESS.

Pracovník spisovny načte soubor do eSSL, systém následně automaticky označí dokumenty vybrané za archiválie a dokumenty určené ke zničení. Z dokumentů označených za archiválie jsou vytvořeny jejich repliky a k nim náležející metadata, tyto jsou následně předány Národnímu digitálnímu archivu. Poté jsou tyto dokumenty vymazány z elektronického systému spisové služby. Digitální dokumenty určené ke zničení jsou taktéž zničeny vymazáním z eSSL. [8], [9]

1.6 Skartační lhůta a archivace VŠKP na UTB

UTB má definovanou skartační lhůtu závěrečných VŠKP v dokumentu s názvem „Spisový a skartační plán“. Závěrečné VŠKP jsou v dokumentu uvedené pod spisovým znakem K III. 3. a jsou řazeny do skartačního režimu V 20. Skartační lhůta závěrečných VŠKP na UTB je tedy 20 let, po tuto dobu leží ve spisovně UTB. Pro splnění požadavku, že práce musí být zveřejněná, jsou práce nahrávány z IS/STAG do systému DSpace. Nahrání závěrečné VŠKP z IS/STAG do knihovního uložiště DSpace mají na starosti jednotlivá studijní oddělení.

UTB vznikla 1. 1. 2001, první závěrečné VŠKP pocházejí z roku 2004, takže ke skartačnímu řízení se budou předkládat až v r. 2025. Co se týče závěrečných VŠKP z roku 2004, tak ty jsou uloženy v listinné formě, jeden výtisk je umístěn v knihovně a jeden výtisk je uložen ve spisovně. Od roku 2005 se na UTB začalo používat digitální knihovní úložiště DSpace. Mezi roky 2005 - 2017 v něm **nebyly** VŠKP ukládány ve formátu PDF/A. To se změnilo od roku 2017, kdy se do DSpace začaly ukládat závěrečné VŠKP ve formátu PDF/A.

Do **spisovny** jsou v současnosti ukládány buď tištěné verze VŠKP nebo digitální verze na CD. Neexistuje jednotný přístup v rámci celé univerzity, protože pravidla si dělá každá fakulta své. Časové období, ve kterém jsou závěrečné VŠKP předávány z fakult do

spisovny, je u každé faktulty individuální. Co se týče časového rozmezí, v jakém jsou závěrečné VŠKP předávány do spisovny, některé fakulty předají závěrečné VŠKP do spisovny přímo v roce obhajoby, jiné si u sebe nechávají závěrečné VŠKP i několik let a do spisovny je dodávají až s několikaletým zpožděním.

Co se týče elektronického systému spisové služby (eSSL), spisovna UTB používá eSSL, ale závěrečné VŠKP do něj v současnosti žádná fakulta nenahrává. Existují dva hlavní důvody, jedním je absence jakéhokoliv automatizovaného propojení eSSL s IS/STAG a druhým je fakt, že zvláště závěrečné VŠKP z Fakulty multimediálních komunikací a Fakulty aplikované informatiky obsahují velmi často přílohy audiovizuálních děl nebo softwarových aplikací. Tyto přílohy by bylo obtížné nahrávat do eSSL.

Co se týče závěrečných VŠKP již uložených ve spisovně, po uplynutí skartační lhůty budou VŠKP zařazeny na skartační návrh, který bude zaslán k posouzení Moravskému zemskému archivu. Před zasláním skartačního návrhu se k seznamu dokumentů vyjádří tajemníci fakult, kde byly závěrečné VŠKP napsány. Tajemníci mohou doporučit, které VŠKP mají trvalou archivní hodnotu. Moravský zemský archiv by v takovém případě s velkou pravděpodobností respektoval toto doporučení a VŠKP by označil jako archiválii. Následně by s velkou pravděpodobností Moravský zemský archiv nechal závěrečnou VŠKP prohlášenou za archiválii uložit z kapacitních důvodů ve spisovně UTB.

Aktuálně UTB řeší integraci IS/STAG s elektronickým systémem spisové služby přes oboustranné rozhraní podle NSESSS. Výsledkem bude, že všechny dokumenty, které budou v budoucnu nahrány v IS/STAG (včetně závěrečných prací), budou zároveň evidovány v eSSL a bude možné u nich automatiky sledovat skartační lhůtu, dělat řádné skartační řízení a jednoduché předání Národnímu digitálnímu archivu. UTB plánuje, že toto propojení začne být používáno od roku 2022.

2 VHODNÉ FORMÁTY SOUBORŮ PRO ARCHIVACI

Mediální technologie se mění velmi rychle, media, u kterých je požadována dlouhá životnost, jsou neustále ohrožena zastaráváním. Například s příchodem Windows 3.1 byl pro ukládání textových dokumentů velmi oblíbený formát „WRI“. V dnešní době dokáže tento formát přečíst už jen velmi malé množství programů a jednoho dne se pro průměrného uživatele může stát úplně nepřístupným. Podobný osud může potkat i dnešní populární formáty (např. DOCX). Aby se zabránilo tomu, že v budoucnu nebude možné přečíst historické elektronické dokumenty a multimédia v zastaralém souborovém formátu, je dobré moudře zvolit vhodný souborový formát pro archivaci. Další podkapitoly budou pojednávat o výběru vhodných archivních formátů. Formáty souborů lze obecně rozdělit na dvě skupiny, proprietární a otevřené formáty.

2.1 Proprietární formát

Proprietární formát souboru je formát, vyvinutý jednou určitou společností. Pokud nemáme software dané společnosti, často nejsme schopni proprietární formáty nijak číst. Proprietární formáty jsou často typické tím, že neexistuje jejich veřejně dostupná dokumentace, takže nikdo nemůže snadno psát software, který dokáže tyto soubory číst. Příkladem jsou formáty: archivní soubor RAR, soubory Corel CDR nebo Microsoft WMA audio formát, nebo Adobe Flash video (SWF) [10].

Speciálním případem jsou formáty dokumentů MicrosoftOffice (DOCX, XLSX…), které sice splňují pravidlo, že jsou vyvíjeny a spravovány jednou určitou společností, ale na druhou byla jejich dokumentace již zanesena do veřejně dostupného standardu ISO/IEC 29500-1:2016 [11].

2.2 Otevřené formáty

S ohledem na výše uvedené nedostatky proprietárních formátů byly vytvořeny otevřené formáty. Tyto formáty jsou plně zdokumentované a dokumentace je veřejně dostupná. Neobsahují omezení na autorská práva ani omezující licence. O dalším vývoji formátu rozhoduje organizace pro standardy nezávislá na jakékoliv společnosti nebo komunita vývojářů, jejichž cílem je vytvořit formát co nejobecnější a promyšlený. Příkladem jsou formát ODT pro dokumenty, HTML, Scalable Vector Graphics (SVG) pro web, dokumentované formáty Portable Document Format (PDF) a Portable Network Graphics (PNG) nebo a archivní formát ZIP. Obecně jsou pro archivaci vhodnější otevřené formáty.

Před výběrem vhodného formátu pro archivaci, je také dobré se ujistit, jak zatím formát obstál ve zkoušce času, jak je **zavedený** na trhu. Indikátory mohou být:

- Jak dlouho již existuje daný formát a zda je podporován více dodavateli (ne jen jednou společností)

- Zda je nezávislý na platformách operačních systému [10].

2.3 Ztrátové, bezztrátové a vektorové formáty

Při práci s multimediálními soubory je možné se setkat se třemi formáty: ztrátový, bezztrátový a vektorový. Každý formát má své určité výhody a omezení. V odstavcích níže budou jednotlivé formáty probrány z pohledu jejich vhodnosti pro archivaci.

Bezztrátové souborové formáty ukládají data přesně tak, jak byly původně vyrobené, nebo získané. Uchovávají každý detail dat, i když jsou detaily pro člověka nepostřehnutelné. Bezztrátové souborové formáty jsou pro archivaci zajímavé zejména ze dvou důvodů. Prvním je možnost je v budoucnu použít i v aplikacích, pro které nebyly původně plánovány. Například u zvukového souboru, může DJ chtít uměle zpomalit starý zvukový záznam, nebo jej smíchat s jiným zvukovým souborem bez ztráty kvality. Kdyby měl dostupný pouze zvukový záznam ve ztrátovém formátu, ztráty dat ve zdroji, které byly původně nepostřehnutelné, by se ve výsledně vzniklém souboru mnohem více projevily. Druhým důvodem je možnost v budoucnu převést bezztrátový soubor do novějšího bezztrátového formátu. Pokud by zdrojový formát i novější formát byly ztrátové, pak se malé ztráty začnou při převodu u každé další generace sčítat, což by v konečném důsledku zhoršovalo kvalitu souboru. Příkladem bezztrátového formátu pro obrázky je PNG, pro audio data například formát Free Lossless Audio Codec (FLAC).

Ztrátové formáty ztrácí podrobnosti o datech, které nejsou člověkem vnímatelné. Například u audio formátů jsou některé kombinace frekvencí pro lidské ucho neslyšitelné, takže pokud dojde k jejich odstranění, soubor se může stát několikrát menší. Příkladem obrázkových ztrátových formátů je JPG, příkladem ztrátového audio formátu je MP3. Ztrátové formáty souborů jsou obecně méně vhodné pro archivaci.

Vektorové formáty ve srovnání s bezztrátovými nabízejí při archivaci dvě hlavní výhody. První je, že jsou schopné uchovat všechny detaily, obrázky je možno nekonečně zvětšovat bez toho, aniž by se objevily viditelné pixely. Druhou výhodou je, že obvykle produkují značně menší soubory než bezztrátové formáty, například formát SVG. Nicméně

při jejich používání je uživatel omezen rozsahem jejich použitelnosti. Například jsou vhodné pro uchovávání obrázků, které je možné kreslit ze základních geometrických objektů jako třeba kruh. Vektorové formáty jsou ale nevhodné třeba v případech, kdy je vyžadováno, zachytit co nejrealističtěji obrazy reálného světa kolem nás, protože ty vyžadují zobrazení složité barevné plochy s velkým množstvím barev, kterou vektorový formát není schopen poskytnout. Dalším příkladem může být hudební vektorový audio formát Musical Instrument Digital Interface (MIDI), který využívá toho, že klavír hraje určitou sekvenci not, kdy není potřeba digitalizovat tvar analogové vlny, ale stačí zachytit konkrétní výšku a sekvenci tónů.

Obecně lze tedy říct, že ze ztrátových, bezztrátových a vektorových formátů jsou pro archivaci nejvhodnější vektorové formáty, protože zrcadlí přesně data, nicméně jsou použitelné pouze, pokud jsou podkladová data vektorizovatelná. Pokud tomu tak není, jsou, nejlepší další cestou bezeztrátové formáty [10].

Níže budou probrány vhodné formáty souborů pro archivaci u vybraných typů dat.

2.4 Vhodné formáty souborů pro archivaci u vybraných typů dat

Následující odstavce se budou zabývat výběrem vhodných souborových formátů pro archivaci. Posuzovány budou formáty pro dokumenty, audio, obraz a archivaci.

Dokumentové formáty

Pro vytváření nového dokumentu, u kterého je kladen důraz na archivaci, ale není vyžadováno formátování textu, je dostačující zvolit formát čistého textového souboru **TXT**. Je to nejrozšířenější a nejotevřenější formát, který existuje, pro jejich čtení není třeba kromě textového editoru žádný speciální software.

Pro vytváření nového dokumentu, u kterého je kladen důraz na archivaci a je vyžadováno, aby mohl být text v budoucnu formátován, jsou vhodnou volbou buď dokumenty **LibreOffice**, nebo text vysázen v systému **LaTeX**. V obou případech jde o zavedené a otevřené standardy.

Pro vytváření nového dokumentu, u kterého je důraz na archivaci a zároveň je vyžadováno formátování textu, ale do budoucna není vyžadována jeho editace, je vhodnou volbou formát **PDF**. Jde o dobře zavedený standard s otevřenou (placenou) dokumentací [10].

Audio formáty

V případě výběru vhodného audio formátu, kdy je zdrojová zvuková stopa vytvářena počítačem nebo elektrickým hudebním nástrojem podporujícím formát MIDI, je nejvhodnější volbou pro archivaci bezztrátový audio formát **MIDI**. Tento formát definuje zvukovou stoupu jako sekvenci instrukcí. Zjednodušeně je možné si ho představit jako notovou osnovu společně s informacemi, který nástroj hraje který řádek. Formát MIDI nemůže být použit pro nahrávání živých audio dat. Nemůže nahradit třeba záznam hrajícího orchestru. Je to proto, že MIDI není schopno vyjádřit změny síly, délky a hlasitosti, které charakterizují hudební skladbu hranou lidmi. MIDI soubory jsou bezeztrátové, velmi malé a s otevřenou dokumentací formátu.

V případě že zdrojový zvuk je dostupný pouze v analogové formě, je nejvhodnější pro archivaci zvuku zvolit bezztrátový audio formát **FLAC**. Tento formát je možno nativně přehrávat ve Windows a všech majoritních webových prohlížečích. Produkty od společnosti Apple (iOS, Mac, Safari) nativně FLAC nepodporují, protože Apple má svůj vlastní bezztrátový zvukový formát ALAC. Přehrávače pro FLAC je však snadno možné najít také pro systém Apple. Formát FLAC umožňuje kódovat různé vzorkovací frekvence („rozlišení“). Větší vzorkovací frekvence jsou věrnější originálu, ale produkují větší velikosti souborů. Na základě toho, co lidé běžně slyší, je používána standardní vzorkovací frekvence 44100 Hz. Profesionálové používají někdy i vyšší vzorkovací frekvence, ale pouze v případech, kdy plánují později zvukový materiál nějak upravovat (např. zpomalovat). Formát FLAC komprimuje zvuková data bezztrátovou kompresí a je plně otevřený.

Dalším bezpečným formátem pro archivování zvuku může být formát MP3. Jedná se sice o ztrátový formát, ale jeho předností však je to, že odřezává malé podrobnosti zvukového materiálu, které lidské ucho není schopno slyšet, díky čemuž MP3 zabírá oproti formátu FLAC mnohem méně místa. MP3 stejně jako FLAC také umožňuje kódovat data v různých vzorkovacích frekvencích. Z pohledu obecných pravidel pro archivaci není MP3 zcela ideální, ale splňuje podmínky, že je velmi dlouho zavedený na trhu a je platformě nezávislý [10].

Obrázkové formáty

Jakékoliv obrázky ve vektorové grafice je nejvhodnější archivovat ve formátu **SVG**. Tento formát je bezztrátový a plně otevřený.

Pro archivování bitmapových obrázků je vhodný formát **PNG**. Tento formát je bezztrátový, otevřený a široce rozšířený. Je podporovaný všemi majoritními internetovými prohlížeči, může být prohlížen a upravován na všech majoritních operačních systémech.

Dalším bezpečným formátem pro archivování bitmapových obrázků je formát **JPG**. Před archivací je vhodné se ujistit, že rozlišení obrázku je dobré a kompresní poměr je nízký. JPG formát je sice ztrátový, ale na druhou stranu otevřený a hojně rozšířený na trhu.

Dále je ještě možné použít formát Tag Image File Format (**TIFF)**. Jedná se o kontejnerový formát pro obrázky. Může obsahovat obrázky kódované bezztrátově i ztrátově. Nejčastěji je však používán jako bezztrátový formát mezi grafickými umělci a vydavatelským průmyslem. Důvod je ten, že TIFF podporuje barevný model CMYK. Tento model je používán pro tisk. Určité barvy není možné zobrazit v RGB modelu na obrazovce, ale pouze vytisknout tiskárnou. Jedná se o bezztrátový formát, často ukládaný v nekomprimované podobě, což může zachování stejné kvality obrázku dělat obrázky uložené v TIFF formátu téměř dvakrát větší než PNG [10].

Archivní a kompresní formáty

Ve chvíli, kdy jsou zdrojová data připravena ve vhodném cílovém formátu pro archivaci, je vhodné je uložit do některého archivního formátu. Existují formáty, buď pouze archivní, kompresní, nebo jsou kompresní i archivní zároveň. Pokud bude bráno v úvahu kritérium, aby archivované soubory bylo možné rozbalit na kterémkoliv z majoritních operačních systémů (iOS, Windows, Linux), je jedním z nejlepších řešení zvolit pro archivaci formát ZIP.

ZIP je kompresní archivní formát. Byl zaveden v roce 1989, díky jeho dlouhé existenci na trhu je nativně podporován všemi hlavními operačními systémy (Linux, iOS, Windows). Z technického hlediska byl ZIP při jeho vydání proprietárním formátem souboru, protože byl vyvinutý společností PKWARE. Dnes však neexistují žádné problémy s licencí. Formát je tak všudypřítomný, že jej lze považovat za otevřený.

Novější alternativou pro ZIP je formát **7Z**. Jedná se o otevřený, archivní a kompresní formát. Není však již tak všudypřítomný a dobře podporovaný. Oproti formátu ZIP, ale nabízí lepší kompresní poměr a možnost šifrovat soubor pomocí standardu AES-256.

V Linuxových a Unixových systémech je velmi populární archivní formát TAR. Jedná se o otevřený archivní formát, umožňuje uložení několika souborů do jednoho souboru bez komprese. TAR je následně komprimován pomocí gzip nebo xz, čímž vznikají formáty tar.gz a tar.xz. Formát TAR je na Windows možné rozbalit například pomocí programu 7-Zip [10], [12].

3 DIGITÁLNÍ PŘENOSITELNÉ FORMÁTY PDF VS PDF/A

Portable Document Format (PDF) je digitální formát pro reprezentaci dokumentů. PDF soubory lze nativně vytvářet ve formátu PDF převodem z jiných elektronických formátů nebo digitalizováním analogových papírových předloh.

Důvodem vzniku PDF bylo vytvořit souborový formát pro ukládání dokumentů nezávislých na softwaru a hardwaru, na kterém byly pořízeny. Díky své nezávislosti formát PDF zajišťuje, že se dokument zobrazí stejně na různých zařízeních. Například dva uživatelé používající operační systém Windows a iOS mohou mezi sebou sdílet textové dokumenty bez nutnosti konvertování, což velmi usnadňuje spolupráci.

Podniky, vlády, knihovny, archivy, jednotlivci a různé instituce z celého světa používají PDF k reprezentování značného množství důležitých informací. Velká část těchto informací musí být uchována po značně dlouhou dobu, některé musí být uchovány trvale. Takové PDF soubory musí zůstat použitelné a přístupné napříč několika generacemi technologií. Budoucí využití a přístup k těmto souborům závisí na zachování jejich vzhledu a vlastností jako je logická organizace stránek, sekcí, odstavců, strojově obnovitelný textový proud v přirozeném pořadí čtení a různých popisných metadat. Pro zajištění zachování vizuální podoby a vlastností elektronických dokumentů PDF v čase byla vytvořena standardizovaná verze PDF nazývaná PDF/A. PDF/A se tedy jednoduše liší od PDF hlavně tím, že zakazuje funkce PDF nevhodné pro dlouhodobou archivaci. Prvotní formát souboru PDF/A byl publikován v roce 2005 standardem ISO 19500-1.

PDF/A soubor je tedy v jádru standardní PDF soubor, jehož struktura je přizpůsobena předepsanému seznamu pravidel definovaných normou ISO 19500-1 [13], [14].

4 PDF/A-1, ISO 19500-1

Standard ISO 19500-1 je označován zkratkou PDF/A-1. Standard definuje mechanismus pro reprezentování elektronických dokumentů způsobem, který zachovává jejich vizuální podobu v čase. Vizuální podoba dokumentů je nezávislá na strojích a systémech používaných k vytváření, ukládání nebo vykreslování souborů. Sekundárním účelem tohoto standardu bylo poskytnout framework pro zaznamenávání kontextu a historie pomocí metadat. Dalším důvodem bylo definování frameworku pro reprezentaci logické struktury a dalších sémantických informací. Požadovaných cílů bylo dosaženo identifikováním sady komponent, které lze použít, a omezením na formu jejich použití v rámci vyhovujících souborů PDF/A. Samotné PDF/A samo o sobě nezaručuje, že vizuální vzhled obsahu bude vždy přesně odrážet původní zdrojový materiál použitý k vytvoření vyhovujícího souboru. Proces použitý k vytvoření vyhovujícího souboru, může nahradit písma, přeformátovat text, převzorkovat obrázky nebo použít ztrátovou kompresi. Organizace, které potřebují zajistit, aby byl vytvořený PDF/A soubor a přesně odpovídal původnímu zdrojovému materiálu, musí rozšířit požadavky na procesy nad rámec požadavků ISO-19005, které generují vyhovující soubor. [13]

Standard PDF/A-1 se dále dělí na následující 2 úrovně shody (conformance levels). Úroveň shody je množina omezení a požadavků, které musí vytvořené PDF/A-1 soubory a softwarové aplikace pro prohlížení PDF souborů splňovat.

Číslo verze standardu ISO 19005 a úroveň shody souboru PDF/A, by měly být specifikovány pomocí PDF/A identifikačního rozšiřujícího schéma. Identifikační rozšiřující schéma používá 3 vlastnosti:

- *pdfaid:part* (povinná) – číslo verze ISO 19005, kterému soubor odpovídá
- *pdfaid:amd* (nepovinná) - číslo dodatku a rok oddělené dvojtečkou
- *pdfaid:conformance* (povinná) – specifikuje jakou úroveň shody daný PDF/A soubor splňuje

Úroveň schody A

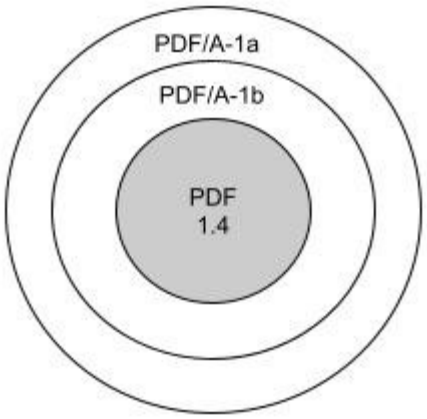
Soubory splňující požadavky úroveň schody A jsou označovány jako PDF/A-1a, nebo ISO 19005-1 Level A Conformance. Soubory s tímto označením musí splňovat všechna pravidla definovaná normou ISO 19005-1 včetně požadavků na mapování do unicode (viz část 6. 3. 8 ve specifikaci International Standard ISO-19005-1) a logickou strukturu (v část

6. 8. ve specifikaci International Standard ISO-19005-1). Unicode mapování je technická norma, která umožňuje přesné mapování znaků, písmen, číslic a symbolů na kódy. Jedná se o znaky, písmena, číslice a symboly používané v nejrůznějších písmech na Zemi. Díky pravidlům na mapování do unicode a logickou strukturu obsahují PDF/A-1a bohaté množství informací umožňujících zachovat podrobnou, logickou strukturu dokumentu a textový proud v přirozeném pořadí čtení, což usnadňuje přístup k souborům pro fyzicky postižené uživatele. Další výhodou je opětovné použití textu, například při exportu z PDF/A souboru do souboru textového editoru např. Wordu. Díky přesně definované struktuře se při exportu předejde problémům, jako například nesprávně zarovnaný text odstavců exportovaného textu.

Úroveň shody B

Soubory vyhovující úrovni B,

musí splňovat všechny požadavky ISO 19005-1, kromě požadavků na mapování do unicode (viz část 6. 3. 8 ve specifikaci International Standard ISO-19005-1) a logickou strukturu (v část 6. 8. ve specifikaci International Standard ISO-19005-1). Takže na PDF/A-1b je možno nahlížet jako na podmnožinu PDF/A-1a. viz Obrázek. 1. Účelem požadavků na úroveň shody B jsou požadavky minimálně nezbytné k zajištění toho, aby vykreslený vizuální vzhled souboru byl dlouhodobě uchovatelný. Soubory splňující požadavky na úroveň shody B jsou označovány jako PDF/A-1b nebo ISO 19005-1 Level A Conformance [13], [15].

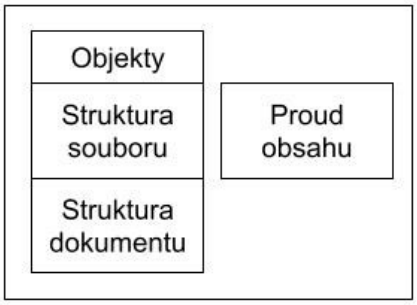


Obrázek 1: PDF/A-1a a PDF/A-1b

5 STRUKTURA PDF 1.4

Struktura PDF/A-1 je založena na struktuře PDF 1.4. Struktura PDF 1.4 byla definována společností Adobe v oficiálním dokumentu s názvem PDF Reference, third edition [14].

PDF syntaxe je definována do 4 komponent viz Obrázek 2:



Obrázek 2: komponenty PDF [14]

5.1 Objekty

PDF dokument je datová struktura v jádru složena z malé sady základních typů datových objektů. PDF rozlišuje 8 základních datových objektů:

1. **Boolean objekty** – Jsou indentifikovány klíčovými slovy true a false a mohou být použity jako hodnoty prvků, nebo záznamů ve slovníku.
2. **Číselné objekty –** PDF poskytuje dva typy číselných objektů (celá čísla a reálná). Rozsah a přesnost čísel je limitována vnitřní reprezentací použitou na stroji, na kterém běží aplikace pro prohlížení PDF. Hodnota celých čísel je interpretována jako celočíselný objekt. Hodnota reálných čísel je interpretována jako reálný objekt. Pokud dojde k překročení limitu pro celá čísla, je celočíselný objekt konvertován do objektu reálných čísel.
3. **Řetězcové objekty -** Pro psaní řetězcových objektů v PDF existují dvě konvence
    a) **Doslovné řetězce –** Doslovný řetězec je psaný jako libovolný počet znaků uzavřený v kulatých závorkách (). V rámci doslovného řetězce je používán jako únikový znak zpětné lomítko. Slouží například pro zadání znaku nového řádku, netisknutelných ASCII znaků, nebo kulatých závorek v textu.
    b) **Hexadecimální řetězce –** Řetězce můžou být také psány v hexadecimálním tvaru. Tento tvar je užitečný pro zahrnutí libovolných binárních dat. Hexadecimální řetězce jsou napsány jako sekvence znaků (0-9 a A-F)

uzavřené ve špičatých závorkách < >. Každý pár hexadecimálních čísel definuje jeden byte řetězce

4. **Jmenné objekty –** Jmenný objekt je atomický symbol jednoznačně definovaný posloupností znaků. Jednoznačně definovaný znamená, že jakékoliv dva jmenné objekty vytvořené ze stejné posloupnosti znaků jsou identicky stejné objekty. Atomický znamená, že jméno nemá žádnou vnitřní strukturu, přestože je definováno sekvencí znaků, znaky nejsou prvky jména. Jméno je definováno pomocí zpětného lomítka viz příklad Kód 1. Zpětné lomítko není součástí samotného jména, ale pouze předponou indikující, že sekvence znaků pokračující za lomítkem je jméno.

```
/ExampleName1
```

Kód 1: Příklad jmenného objektu

5. **Objekt pole –** Pole je definováno jako sekvence objektů uzavřených v hranatých závorkách [ ]. Pole v PDF můžou být heterogenní podobně jako pole v mnoha jiných počítačových jazycích. To znamená, že prvky pole můžou být jakékoliv kombinace čísel, řetězců, slovníků, nebo jiných objektů, včetně dalších polí viz příklad Kód 2.

```
[41 3.12 false (Ralph) /SomeName]
```

Kód 2: Příklad objektu pole

6. **Slovníkový objekt –** Je asociativní tabulka obsahující páry objektů zvané jako slovníkové vstupy. První prvek každého vstupu je **klíč** a druhý prvek je **hodnota**. Klíč musí být typu **jméno**. Hodnota může být jakýkoliv druh objektu včetně dalšího slovníku. Páry klíč a hodnota jsou uvnitř slovníku uzavřeny v dvojitých špičatých závorkách <<…>> viz příklad Kód 3.

```
1  <<
2  /Type /Example
3  /Version 0.03
4  /IntegerItem 15
5  /StringItem (some string)
6  >>
```

Kód 3: Příklad slovníkového objektu

Slovníkové objekty jsou hlavními stavebními bloky PDF dokumentu. Jsou běžně používány pro spojování atributů komplexních objektů, jako jsou fonty nebo stránky PDF dokumentu. Každý vstup ve slovníku specifikuje jméno a hodnotu konkrétního atributu.

7. **Objekty proudů –** Objekt proudu je podobně jako objekt řetězce sekvence bytů. Aplikace zobrazující PDF čte proud postupně, zatímco řetězec musí být čten celý. Proud může být neomezeně dlouhý, zatímco řetězec podléhá určitému implementačnímu limitu. Z toho důvodu objekty s potenciálně velkým množstvím dat (např. obrázky) jsou reprezentovány jako proudy. Proud se skládá ze slovníku obsahujícího vždy klíč `/Lenght`, jehož hodnota indikuje kolik byte z PDF souboru je použito pro data proudu a řádků bytů, které jsou umístěny mezi klíčovými slovy **stream** a **endstream.** Dále můžou být ve slovníku proudu použity klíče: `/Filter`, `/DecodeParams`, `/F`, `/FFilter`, `/FDecodeParams`. Příklad obsahu proudu je uveden Kód 4.

```
endobj
13 0 obj
<<
/Filter /FlateDecode
/N 3
/Length 284>>
stream
…
endstream
endobj
```

Kód 4: Příklad objektu proudu

8. **Nulové objekty –** Nepřímé odkazy na neexistující objekty jsou vyhodnocovány jako nulové objekty.
9. **Nepřímé objekty –** Jakýkoliv objekt v PDF souboru může být označen jako nepřímý objekt. Označení dává objektu unikátní identifikátor, díky kterému se na něj mohou ostatní objekty odkazovat. Identifikátor se skládá ze dvou čísel:
   - Kladné celé číslo, označované jako číslo objektu
   - Záporné celé číslo označované jako generation number. Ve všech nově vytvořených souborech mají všechny nepřímé objekty generation number rovno 0. Nenulové generation numbers mohou být později zavedena, když je soubor updatován.

Příklad definice nepřímého objektu v PDF souboru je uveden v Kód 5:

```
1  3 0 obj
2  <<
3  /Type /Outlines
4  /First 12 0 R
5  /Count 6
6  /Last 12 0 R
7  >>
8  endobj
```

Kód 5: Ukázka odkazování na nepřímý objekt

V příkladu Kód 5 je ukázková definice objektu `3 0 obj`. Tento objekt používá 2 nepřímé odkazy u klíčů `/First` a `/Last` viz řádek 2 a 4 v Kód 5. Hodnota u obou klíčů se nepřímo odkazuje na objekt `12 0 R`. pomocí object number (což je číslo 12), generation number (což je číslo 0) a klíčového slova R.

Nepřímé objekty definují komponenty dokumentu např. fonty, stránky a vzorkované obrázky. Nepřímé objekty jsou vytvářené ze základních objektů 1 – 8 popsaných v kapitole 4. 1. Pro zopakování jsou jimi: boolean objekty, číselné objekty, řetězcové objekty, jmenné objekty, slovníkové objekty, objekty polí, objekty proudů, nulové objekty [14], [16].

5.2 Struktura PDF souboru

Struktura PDF souboru definuje, jak jsou objekty uloženy v PDF souboru, jak je k nim možno přistupovat a aktualizovat je. Skládá se ze 4 částí, viz Obrázek 3:

Obrázek 3: Struktura PDF souboru [13]

5.2.1 Hlavička (Header):

Hlavička je první řádek PDF souboru. Identifikuje číslo verze PDF specifikace, které soubor vyhovuje. Například **%PDF-1.4**. V případě, kdy PDF soubor obsahuje binární data, je doporučeno, aby za hlavičkovým řádkem následoval řádek s komentářem. Tento řádek by měl obsahovat alespoň 4 binární znaky, jejichž kódy jsou větší než 128. Tyto binární znaky zajistí správné chování aplikací přenášejících data, když kontrolují, zda-li zacházet s obsahem souboru jako s binárním nebo s textovým.

5.2.2 Tělo (Body):

Tělo PDF souboru se skládá ze sekvence nepřímých objektů reprezentujících obsah dokumentu. Tyto nepřímé objekty jsou propojené vzájemnými vazbami.

5.2.3 Tabulka křížových odkazů (Cross-reference table)

Tato tabulka je volitelná. Účelem tabulky křížových odkazů je umožnit náhodný přístup k nepřímým objektům v souboru, takže díky ní k vyhledání konkrétního nepřímého objektu není potřeba číst celý PDF soubor. Tabulka křížových odkazů tedy usnadňuje a urychluje přístup k objektům. Tabulka začíná výrazem **`xref`**, následuje řádek se dvěma čísly. První číslo určuje číslo prvního objektu v tabulce křížových odkazů (v příkladu Kód: 2 to je 0), druhé číslo říká, kolik objektů je v tabulce křížových odkazů obsaženo (v příkladu Kód: 2 se jedná o 6 objektů). Dále jsou řádky s konkrétními záznamy. V tabulce křížových odkazů existují 2 druhy obecných záznamů.

a) Obecně ve tvaru **`nnnnnnnn ggggg n`** pro používané objekty.

b) Ve tvaru **`nnnnnnnn ggggg f`** pro volné objekty.

- `nnnnnnnnnn` - Kde desetimístné číslo udává **pozici objektu**, pomocí offsetu v bytech. Offset je počítán od začátku PDF souboru do místa, kde objekt začíná.
- `ggggg` - Kde pětimístné číslo definuje **číslo generace**. Kromě objektu 0, mají při vzniku PDF souboru všechny objekty generační číslo 0. Když je objekt vymazán, objekt je označen pomocí f jako volný objekt a je přidán do spojového seznamu volných záznamů. Generační číslo u tohoto objektu je zvětšeno o 1. Když je v budoucnu volný objekt znovu použit, je vytvořen s generačním číslem, které si držel od posledního zrušení.

```
1   xref
2   0 6
3   0000000000 65535 f
4   0000000010 00000 n
5   0000000079 00000 n
6   0000000173 00000 n
7   0000000301 00000 n
8   0000000380 00000 n
```

Kód 6: Příklad tabulky křížových odkazů [18]

První záznam v tabulce křížových odkazů je vždy volný objekt (označován jako nultý) a má **generační číslo** `65535` (viz Kód 6, řádek 3), je to hlavička spojového seznamu volných objektů, za ním následují odkazy na další nepřímé využité/volné objekty (v příkladu Kód 6 následují na řádcích 4-8 pouze odkazy na využité nepřímé objekty). Pokud by se v tabulce křížových odkazů vyskytovalo více volných objektů, tak poslední volný objekt bude odkazovat zpět na objekt 0. V příkladu Kód 6 se ale žádný další volný objekt již nevyskytuje [17].

5.2.4 Patička (Trailer)

Patička udává umístění různých objektů v těle souboru. Patička PDF souboru umožňuje aplikaci, která čte soubor, rychle najít tabulku křížových odkazů a další speciální objekty. Aplikace čtou PDF soubor od jeho konce. Poslední řádek PDF souboru obsahuje značku `%%EOF`, písmena zkratky jsou odvozeny ze slov „end of file“. Předposlední řádek (viz ukázka Kód 7, řádek 7) obsahuje číslo, toto číslo definuje adresu, kde v souboru začíná tabulka křížových odkazů nebo objekt s proudem křížových odkazů. V příkladu Kód 7 začíná tabulka křížových odkazů na adrese `492`. Řádku s `startxref` předchází slovník, skládající se z klíčového slova `trailer` následovaného páry klíč-hodnota uzavřenými ve špičatých závorkách. Níže je uveden příklad Kód 7 s patičkou PDF souboru, který nebyl zatím nikdy updatován.

```
1   trailer
2   <<
3    /Size 6
4    /Root 1 0 R
5   >>
6   startxref
7   492
8   %%EOF
```

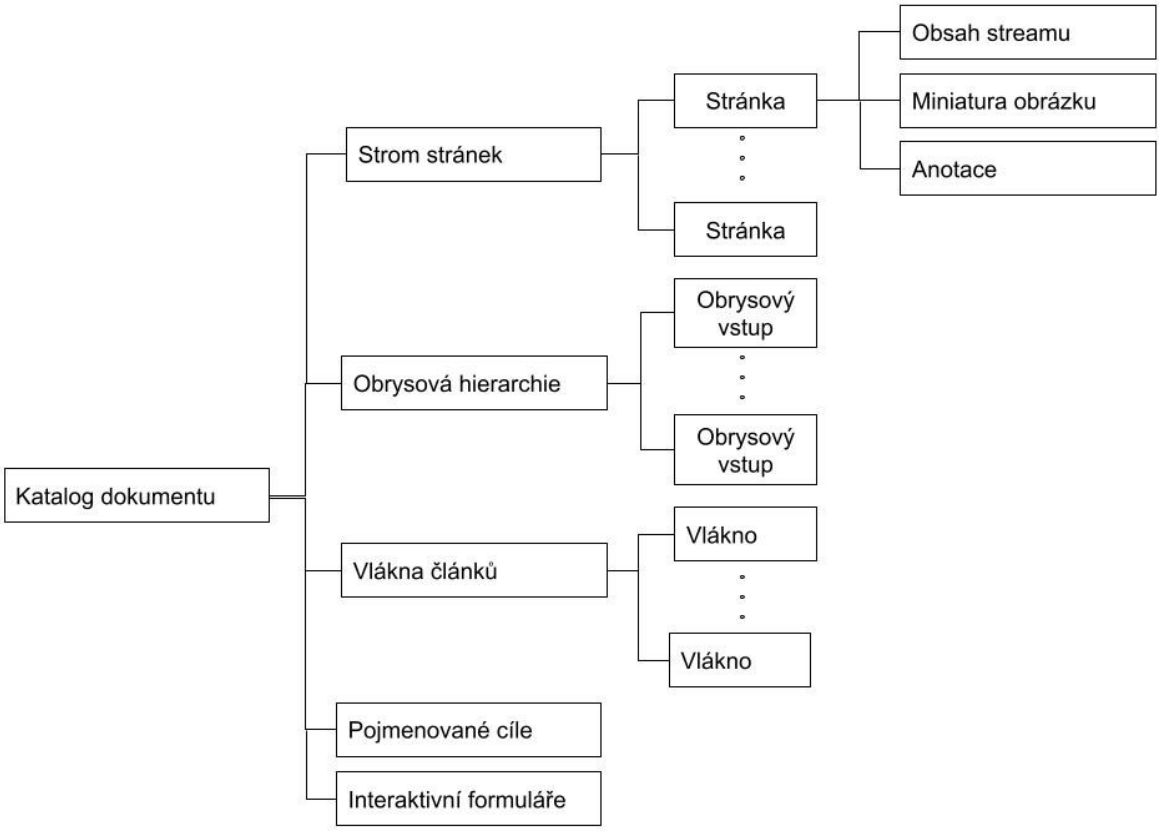
Kód 7: Konkrétní příklad trailer [18]

Sekce tralier může specifikovat následující klíče.

- **Size,** typ: integer – Celkový počet záznamů v tabulce křížových odkazů (včetně objektů v update sekci.)
- **Prev,** typ: integer **–** Specifikuje posunutí od začátku souboru k předcházející sekci tabulky křížových odkazů (cross-reference section), která je použita. Vyskytuje se pouze v případech, kdy má soubor tabulku křížových odkazů s více sekcemi.
- **Root,** typ: slovník **–** Odkaz na nepřímý objekt nazývaný jako katalog (catalog). Toto je kořenový objekt celé struktury PDF dokumentu.
- **Encrypt**, typ: slovník – Specifikuje slovník šifrování dokumentu. (Vyžadováno pokud je dokument šifrován.)
- **Info**, typ: slovník – Informační slovník dokumentu.
- **ID**, typ: pole **–** Pole dvou strignů tvořící identifikátor souboru pro soubor [14], [18].

5.3 Struktura PDF dokumentu

V kapitole **4.2** byla popsána **struktura PDF souboru**. Při větším zobecnění by strukturu PDF souboru bylo možné přirovnat ke struktuře HTML stránky. Kdy HTML stránka má také svoji základní strukturu danou „značkami“ (tagy): (!doctype, HTML, head, title, meta, body), samotný viditelný obsah zobrazovaný uživateli na stránce se nachází v sekci body. Z pohledu struktury PDF dokumentu**,** je na PDF dokument nahlíženo jako na hierarchii objektů obsažených v PDF souboru v sekci body. Struktura PDF dokumentu specifikuje, jak jsou použity základní typy objektů pro reprezentování komponent v PDF dokumentu. Komponenty jsou například stránky, fonty, anotace atd. tedy víceméně to, co se zobrazuje uživateli při prohlížení PDF souboru (viz příklad na Obrázku 4). Většina objektů v hierarchii PDF dokumentu jsou slovníky. Pro příklad každá stránka dokumentu je reprezentována jako objekt stránky (slovník, který zahrnuje odkazy na obsah stránky a další atributy). Jednotlivé objekty stránky jsou spojeny do struktury nazývané strom stránky (page tree), na kterou je umístěn nepřímý odkaz v kořenovém objektu katalog. Vztahy mezi objekty v rámci hierarchie jsou definovány pomocí slovníkových vstupů, jejichž hodnoty jsou nepřímé odkazy na ostatní slovníky.



Obrázek 4: Příklad struktury PDF dokumentu [13]

Katalog dokumentu (Document Catalog)

Objekt katalog dokumentu je kořenem hierarchie objektů v dokumentu. Odkaz na tento objekt je umístěn jako třetí záznam v tabulce křížových odkazů. Katalog obsahuje odkazy na ostatní objekty definující obsah dokumentu např. mód, v jakém bude dokument zobrazen, vlákna článků, přirozený jazyk pro celý text v dokumentu, metadata stream, rozložení stránky po otevření dokumentu.

Strom stránek (Page Tree)

Stránky dokumentu jsou přístupné skrz strukturu stromu stránek, která definuje jejich pořadí v rámci dokumentu. Stromová struktura umožňuje aplikaci zobrazující PDF soubor rychle otevřít dokument obsahující klidně tisíce stránek s použitím omezeného množství paměti. Strom obsahuje uzly dvou typů

- Mezilehlé uzly – nazývané page tree nodes
- Listové uzly – nazývané objekty stránky. Nejjednodušší struktura může být složena z jednoho page tree node, který přímo referencuje všechny objekty stránky dokumentu.

Objekty stránky (Page Objects)

Objekty stránky jsou listové uzly celé struktury stromu stránek, každý z nich je slovník specifikující atributy jedné konkrétní stránky dokumentu. Určité atributy stránky můžou být děděni z uzlů předků ve stromu stránek.

5.4 Proud obsahu (Content stream)

Proudy obsahu jsou primárním prostředkem pro popis vzhledu stránek a dalších grafických prvků. Proud obsahu závisí na informaci obsažené v přidruženém slovníku zdrojů. Kombinací proudu obsahu a slovníku zdrojů je vytvořena samostatná entita.

Proud obsahu je stream objekt PDF souboru, jehož data se skládají ze sekvence instrukcí popisujících grafické elementy vykreslované na stránku. Instrukce jsou representovány formou PDF objektů, používajících stejnou syntaxi objektů jako ve zbytku PDF dokumentu. Nicméně zatímco dokument jako celek je statická random-access struktura dat, objekty v proudu obsahu jsou určeny k tomu, aby byly interpretovány postupně jeden po druhém.

Každá stránka dokumentu je reprezentovaná jedním nebo více proudy obsahu. Při zpracování proudu obsahu, je nejprve proud obsahu dekódován libovolným specifikovaným filtrem a dále interpretován podle syntaxí PDF pravidel pro objekty (viz kapitola 4.1.)

Dekódovaný proud obsahu se skládá z PDF objektů označovaných **operandy** a **operátory**. Operátory konzumují operandy. Všechny operandy, které operátory potřebují, musí ve streamu obsahu předcházet těmto operátorům.

Operand je přímý objekt patřící do některého ze základních datových typů PDF kromě objektu stream. Datový tip slovník je povolen jako operand pouze pro určité specifické operátory. Nepřímé objekty a reference na objekty (příklad reference na objekt by byl: `3 0 R`) nejsou jako operandy povoleny vůbec.

Operátor je v PDF klíčové slovo specifikující nějakou akci, která má být provedena, jako například kreslení grafických tvarů na stránku. Klíčové slovo operátoru se od objektu jméno liší vynecháním zpětného lomítka na začátku řádku. Operátory mají význam pouze uvnitř streamu obsahu.

Operátory se dělí na několik skupin, které ale nebudou již v této práci dále rozebírány, protože bychom zbytečně zacházeli příliš hluboko do standardu PDF 1.4. Níže je uveden ilustrativní příklad operátorů, které vykreslují text. Text je vykreslován s použitím grafického znázornění znaků definovaného ve fontech písma.

```
1   2 0 obj
2   <</Length 153>>
3   stream
4   …
5   BT
6    /F 13 12 Tf
7    288 720 Td
8    (ABC) Tj
9   ET
10  …
11  endstream
12  endobj
```

Kód 8: Příklad objektu obsahující stream, který vykresluje text „ABC“ [14]

Kód 8 vykreslí na obrazovku text „ABC“ umístěný 10 palců od dolní části stránky, 4 palce od levého okraje, s použití fontu Helvetica velikosti 12 bodů.

`BT` – definuje začátek textového objektu

```
/F 13 12 Tf
```

- `F 13` – Je zdroj fontu definován externě jako Helvetica. (operand)
- `12` – Nastavuje font na velikost 12. (operand)
- `Tf` – (Operátor)

```
288 720 Td
```

- `288` – (Operand)
- `720` – (operand)
- `Td` – Určí pozici na stránce, kde bude text vykreslen (Operátor)

```
(ABC) Tj
```

- `(ABC)` - Textový řetězec ABC
- `Tj` – Operátor přebere písmena z operandu a vykreslí je na stránku fontem Helvetica a velikostí 12.

`ET` – definuje konec textového objektu

6 PDF/A-2 ISO 19005-2

PDF/A-2 bylo vydáno v roce 2011. PDF/A-2 soubory jsou založeny na struktuře PDF souborů PDF 1.7 jinými slovy na normě ISO32000-1:2008. Účelem PDF/A-2 nebylo nahradit standard PDF/A-1, pouze přinést nové možnosti z PDF 1.7. Hlavní rozdíl mezi PDF/A-2 a PDF/A-1 je přínos nových funkcí z verze PDF 1.7. Přidané funkce byly přidány v souladu s PDF 1.7 a zahrnují podporu pro:

- Vylepšení odkazování v PDF pro zajištění lepší přístupnosti
- Komprimované objekty a XRef streamy (pro zajištění menší velikosti souborů)
- Přílohy souborů kompatibilní s PDF/A
- přenosné kolekce a PDF balíčkus, které umožňují archivaci sad dokumentů jako jednotlivých dokumentů v jednom souboru
- zachování průhlednosti pro grafické prvky
- komprese JPEG 2000
- aplikace digitálních podpisů v souladu se standardem PAdES ((PDF Advanced Electronic Signatures))

PDF/A-2 definuje tři druhy conformance levelů:

Úroveň shody A - PDF/A-3a

Splňuje všechny požadavky uvedené ve specifikaci standardu PDF/A-2.

Úroveň shody B - PDF/A-3b

Stejně jako u PDF/A-1 účelem požadavků na úroveň B jsou požadavky minimálně nezbytné k zajištění toho, aby vykreslený vizuální vzhled souboru byl dlouhodobě uchovatelný. Soubory vyhovující úrovni B musí splňovat všechny požadavky ISO 19005-2, kromě požadavků na strukturální a semantické vlastnosti dokumentu.

Úroveň shody U - PDF/A-3u

PDF/A-2u reprezentuje PDF/A-2b s přídavnými požadavky, aby veškerý text v dokumentu měl mapování Unicode [19], [20].

7 PDF/A-3 ISO 19005-3

PDF/A-3 je omezená forma PDF verze 1.7. ISO32000-1:2008. PDF/A-3 přidává ke svému předchůdci PDF/A-2 pouze jednu vlastnost. Umožňuje vkládat do PDF/A různé formáty souborů (XML, CSV, CAD, textové dokumenty, tabulkové dokumenty a další) jako kompletně archivované objekty.

Úroveň shody A – PDF/A-3a

Splňuje všechny požadavky uvedené ve specifikaci standardu PDF/A-3.

Úroveň shody B - PDF/A-3b

Soubory splňující tuto úroveň nemusí mít dostatečně bohaté interní informace, které by umožňovali zachování logické struktury dokumentu a proudu textového obsahu v přirozeném pořadí čtení, které poskytuje úroveň shody A.

Úroveň shody U- PDF/A-3u

PDF/A-3u reprezentuje PDF/A-2b s přídavnými požadavky, aby veškerý text v dokumentu měl mapování Unicode [21], [22].

8 PDF/A-4, ISO 19005-4

PDF/A-4 je založeno na PDF 2.0 (definovaném v ISO 32000-2:2020), definuje normativní pokyny pro archivaci nových vlastností spojených s PDF 2.0. Jako takové poskytuje archivační cestu pro soubory PDF 2.0 bez ztráty jejich funkcí. Určitý obsah ze specifikací PDF/A-2 a PDF/A-3 byl přidán přímo do PDF 2.0. např. požadavky na vkládání přidružených souborů.

Specifikace PDF / A-4 také přináší některé významné změny oproti předchozím verzím:

- umožňuje nestatický obsah, který může být přítomen v dokumentech PDF, jako jsou pole formulářů a ECMAScript.
- Byly zrušeny speciální požadavky na vkládání XMP Extension Schemas.
- Všechny fonty v PDF/A-4 vyžadují mapování Unicode včetně textu OCR ze skenovaných dokumentů.
- Samostatné conformance levely A, B, U se v PDF/A-4 nepoužívají. Zcela chybí specifikace, která by popisovala požadavky na nejvyšší conformance level A zajišťující obnovu textu v přirozeném pořadí čtení. Je nahrazena povzbuzením k začlenění sementických informací vyšší úrovně do standardu Universal Accessibility PDF (PDF/UA) neboli standardu ISO-14289.
- Lepší podpora pro archivaci vyplněných formulářů.
- Usiluje o zachování více infromací v souboru (tím, že nevyžaduje jejich odstranění během převodu do archivačního formátu)
- V souboru lze zachovat Javascript např. pro uložení informací o hodnotách nebo logice interaktivního formuláře. Javascript musí být uložen ve vloženém proudu souboru (in embedded file stream) a nesmí být proveden aplikací zobrazující soubor bez explicitní akce uživatele.
- V PDF/A-4 nejsou povoleny typy anotací zvuku, obrazu (specifikuje oblast stránky, na které lze přehrávat mediální klipy) a filmu, tyto anotace jsou nahrazeny anotací RichMedia, což je obecnější typ anotace.
- Anotace 3D a RichMedia typů jsou povoleny pouze v PDF/A-4e.

PDF/A-4 má dva pomocné profily definované v přílohách:

8.1 PDF/A-4f

Profil je definovaný v příloze A standardu ISO 19005-4. Umožňuje vkládat soubory v jakémkoliv formátu.

8.2 PDF/A-4e

Profil je definovaný v příloze B standardu ISO 19005-4. Tento profil podporuje Rich Media a 3D anotace. Je určený pro technické dokumenty jako nástupce standardu PDF/E-1 [23], [24].

9 PRŮZKUM EXISTUJÍCÍCH MOŽNOSTÍ VALIDACE PDF/A

Za účelem výběru vhodného nástroje pro validaci PDF/A byla provedena literární rešerše existujících řešení na trhu. Průzkum byl rozdělen do dvou podkapitol na placené a neplacené softwarové nástroje. U jednotlivých nástrojů bylo hodnocení rozděleno na 5 bodů:

- První bod: **Které PDF/A normy umí daný nástroj kontrolovat**. Tento bod bude obsahovat výčet norem PDF/A, které dokáže daný softwarový nástroj kontrolovat. V případě, že nástroj bude mít validaci implementovanou uvnitř a bude zaměřen na konverzi souboru, bude tento bod obsahovat výčet norem PDF/A do kterých daný softwarový nástroj umožnuje konverzi.
- Druhý bod: Zda je **podporován automatický výběr validačního profilu PDF/A**. Validační profil je vždy tvořen dvojicí **PDF/A-{číslo verze}** mezera {**úroveň schody}**. Například PDF/A-1 má pouze 2 úrovně schody proto u něj existují pouze 2 validační profily PDF/A-1a a PDF/A-1b. Když je automatický výběr podporován, tak to znamená, že uživatel pouze nahraje PDF/A soubor a validační nástroj již sám pozná, pod jakým profilem byl PDF/A uložen a provede validaci se sadou pravidel, odpovídajících danému profilu. V případě, že automatický výběr podporován není, musí uživatel po nahrání souboru sám definovat, jaký profil má být pro validaci použit.
- Třetí bod: Zda je možno **definovat seznam výjimek.** Tento bod reflektuje požadavek IT správců z UTB, Aby daný nástroj umožňoval definovat seznam pravidel, které bude validační nástroj při kontrole ignorovat.
- Čtvrtý bod: **Popis** – Se zaměří na formu, v které je produkt dodáván, může to být desktopová aplikace, serverová aplikace, webová aplikace, nebo knihovna v určitém programovacím jazyce. V případě že se bude jednat o knihovnu, tak bude následovat popis pro jaké programovací jazyky je možno ji použít.
- Pátý bod: **Umožňuje** – Tento bod bude obsahovat stručný souhrn všech vlastností, které produkt nabízí.

9.1 Placené

V následujících podkapitolách budou popsány komerční nástroje pro validaci PDF/A.

9.1.1 3-Heights(TM) PDF Validator

Umí kontrolovat následující normy: PDF/A-1a, PDF/A-1b, PDF/A-2a, PDF/A-2b, PDF/A-2u, PDF/A-3a, PDF/A-3b, PDF/A-3u. Dále kontroluje PDF 1.3 – PDF 1.7, PDF 2.0.

Automatický výběr validačního profilu: ne

Definovat seznam výjimek: ano

Popis: Produkt je dodáván jako Application Programming Interface (API), které lze dále použít pro vývoj dalších aplikací vyžadujících validaci PDF/A. Validátor poskytuje interface pro jazyky: C, Java, .NET Framework, .NET Core. Je možné jej použít na operačních systémech: Windows, Linux i MacOS. Pro Windows je dostupný jako Zip Archive obsahující všechny soubory, nebo jako NuGet Balíčku obsahující všechny soubory pro vývoj v .NET.

API funguje ve stručnosti následovně: Nejprve je načten PDF dokument, v tomto místě musí být definována úroveň shody (a, b nebo u) následně je nastavena úroveň hlášení, jaké typy chyb se ohlásí. Možnosti jsou (žádné hlášení, hlášení pouze chyb, chyby + varování, chyby + varování + přídavné informace). Poté je provedena validace dokumentu proti vybrané specifikaci, po dokončení validace je vrácen seznam chyb podle nastavené úrovně hlášení.

Umožňuje: Závislé lexikální, syntaktické a semantické kontroly. Generovat podrobné nebo souhrnné správy do log souboru. Podrobný popis chyby (číslo, typ, popis, objekt PDF, číslo stránky). Klasifikovat chyby podle typu na error a warning. Možnost volitelného zrušení validace při výskytu první chyby. Číst šifrované PDF soubory. Definovat vlastní validační profil na základě vlastních podnikových směrnic [25].

9.1.2 Callas PDFa Pilot

Umožňuje konverzi souborů do norem: PDF/A-1a, PDF/A-1b, PDF/A-2a, PDF/A-2b, PDF/A-2u, PDF/A-3a, PDF/A-3b, PDF/A-3u

Automatický výběr validačního profilu: ne

Definovat seznam výjimek: ne

Popis: Je dostupný ve verzi desktopové aplikace, příkazového řádku, serverové aplikace, nebo samostatné Software development Kit (SDK):

- **Desktopová verze:** Tato aplikace se zaměřuje přímo na konvertování dokumentů a mailů do validních PDF/A souborů. Validace souborů je kompletně přenechána aplikaci pdfaPilot. Uživatel jednoduše přetáhne dokumenty do okna aplikace pdfaPilot a následně dojde ke konvertování do PDF či PDF/A nejlepším možným způsobem. Řešení je dostatečně konfigurovatelné, aby se mohlo přizpůsobit pracovním postupům zákazníka.
- **Serverová verze pdfaPilot Server**: Umožňuje nastavit tolik sledovaných složek, kolik určitá organizace potřebuje pro své pracovní postupy. Všechny soubory umístěné ve sledované složce jsou automaticky zpracovány zcela bezobslužně podle specifikace uživatele. Každá sledovaná složka může být nezávisle konfigurována. Může řešit běžné problémy v PDF dokumentech, nebo provádět generování PDF/A z dokumentů, emailů, nebo jejich příloh. V závislosti na výsledku zpracování umístí pdfaPilot zpracované soubory do konkrétních výstupních složek. V mnoha pracovních automatizovaných postupech je zásadní, aby bylo možné maximální procento dokumentů převést do PDF/A bez dodatečné obsluhy, i když soubory obsahují problémy. K uspokojení této poptávky nabízí server pdfaPilot speciální konfiguraci nazývanou „zaručený převod PDF/A“. Toto nastavení zajistí, že v případě zjištěných problémů v příchozích souborech je použita řada záložních strategií, které pomůžou k úspěšnému konvertování.
- **PdfaPilot CLI:** Je aplikace příkazového řádku dostupná pro Windows (32 a 64 bit), Linux (32 a 64 bit) Max OS, IBM AIX a Sun Solarix. Aplikace byla optimalizovaná pro rychlost a spolehlivost. Je to ideální PDF/A modul pro webové portály, systémy pro správu dokumentů, nebo automatizované archivní systémy. PdfaPilot CLI podporuje konvertování kancelářských dokumentů a obrázků do PDF nebo pouze ověřování, zda jsou PDF/A soubory validní. Extrahování obrázků z PDF souborů, zabezpečování PDF souborů heslem, konvertování tagovaných PDF do HTML.
- **PdfaPilot SDK**: Je sada pro vývoj software, která umožňuje integraci funkcí pdfaPilot do dalších aplikací. Poskytuje nativní knihovnu C, která poskytuje snadnou integraci do kódu v jazyku C, dále prostřednictvím hlavičkových souborů také v C++ projektech. Doplňkové obalení zajišťují aplikovatelnost v prostředí .NET a Java.

Možnosti: Řešení pdfAPilot je určené pro konvertování PDF souborů, emailů nebo jiných nativních dokumentů do PDF nebo PDF/A. Všechny výše zmíněné verze umí konvertovat soubory typu Microsoft Word, Excel, PowerPoint, Project, Publisher and Visio, LibreOffice, OpenOffice, MacO OS X Pages a Keynote do PDF/A1, PDF/A2 a PDF/A3. Callas PDFa

9.1.3 jPDFPreflight (Qoppa Software)

Umí kontrolovat následující normy: PDF/UA, PDF/A-1b, PDF/A-2b, PDF/A-2u, PDF/A-3b, PDF Exchange (PDF/X-3:2002), PDF/X-3:2003

Automatický výběr validačního profilu: ne

Definovat seznam výjimek: ne

Popis: Jedná se o knihovnu v jazyce Java.

Možnosti: Tuto Java knihovnu, je možné použít pro ověřování výše uvedených PDF/A standardů. Dále umožňuje konvertování PDF souboru do všech podporovaných PDF/A formátů zmíněných výše. Umožňuje vytvářet podrobnou zprávu o všech nevyhovujících i vyhovujících položkách. Přidává anotaci do dokumentu pro zvýraznění nevyhovujících položek. Při nasazování knihovny není vyžadována žádná další instalace nebo konfigurace dodatečných driverů a software [26].

9.1.4 PDF Automation Server (Qoppa Software)

Umí kontrolovat následující normy: PDF/A-1a, PDF/A-1b, PDF/A-2b, PDF/A-2u, PDF/A-3b, PDF/X-1a:2001, PDF/X-1a:2003, PDF/X-3:2002, PDF/X-3:2003

Automatický výběr validačního profilu: ano

Definovat seznam výjimek: ne

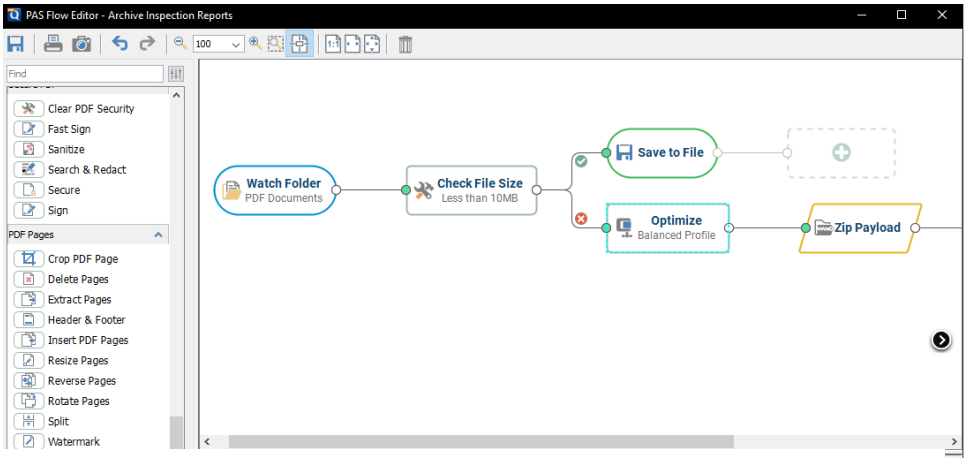
Popis: PDF Automation Server je modulární automatizovaný serverový produkt kompatibilní s operačními systémy Linux, Windows, macOS. PDF Automation Server se skládá ze tří modulů REST API, HTML5 PDF, grafického rozhraní Workflow. V části „Možnosti“ budou podrobněji popsány vlastnosti, které moduly REST API a Workflow nabízejí.

Možnosti:

- **Rest API modul**: Umožňuje přistupovat k funkcím zpracování a konverze PDF prostřednictvím standardních volání REST. Nabízí konvertování PDF to jiných

formátů (obrázků, HTML, SVG, textových souborů), dále konvertování jiných formátů (images, Microsoft Word a Excel, textových souborů) do PDF, upravování dokumentů pro odstranění důvěrných informací, aplikaci a ověřování digitálních podpisů [27], [28].

- **Workflow Module:** Umožňuje automatizovat pracovní toky pomocí jednoduchého grafického rozhraní drag&drop (Obrázek 5), díky čemuž odpadá nutnost psaní kódu. Pokročilé logické nástroje umožňují automatické směrování dokumentů na různé výstupy. Pracovní tok si koncový uživatel skládá pomocí uzlů, které pokrývají jednotlivé procesy práce s PDF. Uzly zahrnují: tvorbu PDF, převod, tisk, digitální podpisy, vyplňování interaktivních formulářů, extrahování textu, kontrolu před výstupem PDF/A [29].



Obrázek 5: Příklad grafického rozhraní drag&drop [29]

9.1.5 PDFTron SDK, knihovna PDF/A

Umí kontrolování + kontrolování následujících norem: PDF/A-1a, PDF/A-1b, PDF/A-2a, PDF/A-2b, PDF/A-2u, PDF/A-3a, PDF/A-3b, PDF/A-3u

Automatický výběr validačního profilu: ne

Definovat seznam výjimek: ne

Popis: Jedná se o cross-platform SDK. Podporované platformy jsou Linux, Windows, iOS, Android, Mac, Web, Xamarin and universal windows platform (UWP). SDK je dostupné pro jazyky C#, C# (.NET Core), C# (Xamarin), C++, Java, Java (Android), Kotlin, objektové-C, PHP, Python, Ruby, Swift, VB, Go. PDFTron SDK je postaven od základu a není závislý na žádném externím open source software třetích stran.

Možnosti: SDK Nabízí několik knihoven pro nejrůznější operace související s PDF dokumenty. Nás zajímá podrobněji z pohledu této diplomové práce knihovna s názvem PDF/A**.** Jedná se o knihovnu umožňující konvertování z více než 20 formátů souborů do PDF/A souborů kompatibilních s ISO standardy. Podporuje přímou konverzi z PDF do XML Paper Specification (XPS), Windows Metafile (EMF/WMF), PNG, JPEG, TIFF, Graphics Interchange Format GIF, BMP, PDF/XPS, SVG, docx. Export ze souborů XPS, HTML, XAML, RichTextBox, Word, Excel, Outlook, PowerPoint do PDF. Dále podporuje velkoobjemové převody PDF do PDF/A z příkazového řádku nebo jako knihovna začleněná do automatizace pracovního postupu u zákazníka. Výrobce garantuje, že po dokončení konverze do souboru formátu PDF/A, projde automaticky každý soubor validací pomocí VeraPDF (více o VeraPDF v kapitole 8. 2. 6) [30], [31].

Nejvyšší úrovní knihovny je třída PDFAComplinace. Nabízí dvě možnosti. Buď pouze validaci PDF/A nebo konvertování existujících PDF souborů do PDF/A.

Pokud je vybrána **možnost konvertování**, je analizován obsah existujících souborů PDF a proveden sled úprav, aby vznikl dokument kompatibilní s PDF/A formátem. Funkce PDF, které nejsou vhodné pro dlouhodobou archivaci (jako je šifrování, zastaralá kompresní schémata, chybějící fonty nebo barvy závislé na zařízení) jsou nahrazeny ekvivalenty kompatibilními s PDF/A. Ztráta informací je minimální, protože proces převodu aplikuje na zdrojový soubor pouze nezbytné změny. Konvertor poskytuje podrobný report o každé změně, takže je snadné zpětně zkontrolovat všechny změny a určit, zda je ztráta převodu přijatelná.

Pokud je vybrána **možnost validace**, je do vstupních parametrů třídy vložen existující PDF soubor. Následně se provede validace, zda PDF soubor vyhovuje PDF/A specifikaci podle mezinárodního standardu ISO 19005. U souborů, které jsou vyhodnoceny jako nekompatibilní, lze vytvořit podrobnou zprávu o porušených předpisech a seznam příslušných chybových objektů [32].

9.2 Neplacené

V následujících podkapitolách budou popsány nekomerční nástroje pro validaci PDF/A.

9.2.1 Služba validace PDF/A na stránkách Národního archivu

Umí kontrolovat následující normy: PDF/A-1a, PDF/A-1b, PDF/A-2a, PDF/A-2b, PDF/A-2u, PDF/A-3a, PDF/A-3b, PDF/A-3u

Automatický výběr validačního profilu: ne

Definovat seznam výjimek: ne

Popis: Jedná se o webové grafické rozhraní[4].

Možnosti: Testované PDF by mělo být postupně ověřováno čtyřmi validátory. Pro považování souboru za validní stačí, když alespoň jeden z validátorů vrátí kladný výsledek. Služba je aktuálně nefunkční. Testováno s validním souborem PDF/A-1a a ani jeden ze čtyř validátorů nevyhodnotil pdf/a jako validní. V dokumentaci je uvedeno, že v jádru jsou použity 4 následující validátory:

- 3-Heights(TM) PDF Validator Shell. Version 2.1.31.0 of Jan 18 2012.
- 3-Heights(TM) PDF Validator Shell. Version 4.4.43.4 of May 8 2015.
- Callas PDFa Pilot
- 3-Heights(TM) PDF Validator Shell. Version 4.10.26.4 of Apr 1 2018 [33].

9.2.2 Online testovací verze validátoru Heights(TM) PDF Validator

Umí kontrolovat následující normy: PDF/A-1a, PDF/A-1b, PDF/A-2a, PDF/A-2b, PDF/A-2u, PDF/A-3a, PDF/A-3b, PDF/A-3u. Dále kontroluje PDF 1.3 – PDF 1.7, PDF 2.0.

Automatický výběr validačního profilu: ano

Definovat seznam výjimek: ne

Popis: Jedná se o webové grafické rozhraní[5].

Možnosti: Jedná se bezplatnou zkušební verzi komerčního validátoru. V případě, že soubor byl validován bez chyb, je vrácena uživateli stručná odpověď s výpisem, jaký validační profil byl rozeznán. V případě, že validovaný soubor byl vyhodnocen jako chybný, je vrácena uživateli zpráva o chybách [34].

---

[4] Dostupné z http: https://www.nacr.cz/verejnost/2-predarchivni-pece/verejnopravni-puvodci/nastroje-narodniho-digitalniho-archivu

[5] Dostupné z http: https://www.pdf-online.com/osa/validate.aspx

9.2.3 Verify PDF/A

Umí kontrolovat následující normy: PDF/A-1a, PDF/A-1b, PDF/A-2b, PDF/A-2u, PDF/A-3b, PDF/X-1a:2001, PDF/X-1a:2003, PDF/X-3:2002, PDF/X-3:2003

Automatický výběr validačního profilu: ne

Definovat seznam výjimek: ne

Popis: Jedná se o webové grafické rozhraní[6].

Možnosti: V kapitole 3. 1. 4 byl popsán PDF Automation Server od firmy Qoppa Software. Tato firma nabízí bezplatnou možnost vyzkoušet validace PDF/A. Uživatel nahraje soubor a vybere validační profil, se mu objeví možnost stáhnout si výsledek validace uložený v PDF souboru. V případě, že byl soubor vyhodnocen jako chybný, jsou v PDF souboru vypsány chyby. Hlášení chyb je velmi stručné, takže uživatel o zdroji chyb nedozví žádné podronosti [35].

9.2.4 PDF to PDF/A Converter PDFTron

Umožňuje konvertování do následujících norem: PDF/A-1a, PDF/A-1b, PDF/A-2a, PDF/A-2b, PDF/A-2u, PDF/A-3a, PDF/A-3b, PDF/A-3u

Automatický výběr validačního profilu: ne

Definovat seznam výjimek: ne

Popis: Jedná se o webové grafické rozhraní[7].

Možnosti: V podkapitole 3. 1. 5 byla popsána knihovna PDF/A, která je zakomponovaná v rámci komerční SDK PDFTron. Na níže uvedené adrese poskytuje firma PDFTRON zdarma k vyzkoušení konvertor do PDF/A. Protože se jedná o zkušební nástroj, výrobce nikde neuvádí, jaké typy vstupních souborů jsou podporovány pro konverzi. Při psaní této práce byla otestována konverze ze vstupních souborů formátu: .jpeg, .docx, .pdf, .xlsx, .pptx do formátu PDF/A. Jak bylo popsáno v kapitole 3. 1. 5, výrobce garantuje, že po konvertování projdou automaticky všechny vytvořené PDF/A soubory přes validátor VeraPDF. Converter PDFTron je velmi užitečný v případech, kdy uživatel vygeneruje PDF/A soubor například z Wordu. Následně při testování neprojde soubor validací pomocí

[6] Dostupné z http: https://freepdfonline.com/verifypdfa/

[7] Dostupné z http: https://www.pdftron.com/pdf-tools/pdfa-converter/

VeraPDF, protože obsahuje chyby v objektech struktury PDF/A. V takovém případě se nabízí jednoduchá varianta opravy a to nahrát nevalidní PDF/A soubor do online convertoru, který se nám pokusí chyby opravit a vytvořit validní PDF/A soubor [36].

9.2.5 Apache PDFBox

Umí kontrolovat následující normy: PDF/A-1b

Automatický výběr validačního profilu: ne

Definovat seznam výjimek: ne

Popis: Jedná se o open source knihovnu Java.

Možnosti: Knihovna je zaměřena na práci s PDF dokumenty. Umožňuje vytváření nových PDF dokumentů, manipulaci se stávajícími dokumenty a schopnost extrahovat obsah z existujících dokumentů. Knihovna poskytuje třídu PreflightParser. Pomocí této třídy je možno validovat soubory, proti normě ISO-19005 PDF/A-1. Pokud je soubor z nějakého důvodu vyhodnocen jako nevalidní, je vrácen objekt obsahující příčiny selhání [37], [38].

9.2.6 VeraPDF

Umí kontrolovat následující normy: PDF/A-1a, PDF/A-1b, PDF/A-2a, PDF/A-2b, PDF/A-2u, PDF/A-3a, PDF/A-3b, PDF/A-3u

Automatický výběr validačního profilu: ano

Definovat seznam výjimek: ne

Popis: VeraPDF je implementován jako aplikace Java. S příslušnou instalací Javy je použitelná na platformách Windows, Linux a OS X. VeraPDF je možné ji použít třemi možnými způsoby:

- Jako desktopovou aplikaci pro Windows, Linux nebo OS X. Aplikaci je možné stáhnout na adrese[8].
- Dále je možné použít VeraPDF jako samostatnou knihovnu Java. Naklonovat ji jako git repozitář[9].

[8] Dostupné z http: https://verapdf.org/software/

[9] Dostupné z http: https://github.com/verapdfv

- Poslední existující možností, je použít nádstavbu nad VeraPDF s názvem **veraPDF-rest**. Což je serverová Java aplikace postavená na frameworku Dropwizard. Tato aplikace obsahuje VeraPDF jako knihovnu jar. Prostřednictvím REST API zpřístupňuje validační funkce této knihovny. VeraPDF-rest je možné naklonovat jako git repozitář z adresy.[10]

Možnosti: VeraPDF je průmyslově podporovaný open source validátor pro PDF/A. Software byl vyvinut pod dohledem PDF Association PDF Validation Technical Group, která nadále nastavuje zásady pro testovací sadu. Součástí projektu VeraPDF bylo vytvoření komplexního testovacího korpusu, který je tvořen strukturou souborů, kde každý soubor pokrývá určitou kapitolu specifikace PDF/A. Testovací korpus je volně dostupný na adrese[11]. VeraPDF byl financovaný z projektu EU s názvem PREFORMA, jehož hlavním cílem bylo řešit problém implementace standardizovaných formátů souborů pro dlouhodobé uchovávání digitálních objektů. VeraPDF je open source software s dvojitou licencí (MPL v2+ a GPL v3+) pro zajištění udržitelnosti a opětovného použití. Testovací korpusy a dokumentace jsou vydávány pod licencí Creative Commons.

VeraPDF se skládá ze čtyř složek:

- **Implementation Checker** – Implementation Checker analyzuje PDF dokument. Výstupem jsou dva typy zpráv:
    - zpráva popisující PDF a jeho metadata
    - zpráva o ověření popisující shodu s danou úrovní shody (Conformance level)
- **Metadata Fixe**r – Metadata Fixer provádí omezenou sadu oprav metadat v dokumentech PDF. Například odstranění příznaku PDF/A v případě nevyhovujícího dokumentu, nebo opravu rozbitých metadat XMP. Výstupem z Metadat Fixer vytvoří opravenou verzi kontrolovaného dokumentu a zprávu o opravě metadat, která popisuje pokusy o opravy a jejich úspěch či neúspěch.
- **Policy Checker** – Policy Checker analyzuje funkce použité v PDF souboru. Generuje zprávu o tom, zda PDF dokument splňuje všechny zásady výjádřené

[10] Dostupné z http: https://openpreservation.org/products/verapdf/

[11] Dostupné z http: https://github.com/veraPDF/veraPDF-corpus

v Profilu zásad. Policy Checker lze použít ke kontrole téměř jakékoliv kvality souboru PDF. Například použití anotací bez ohledu na to zda se jedná o PDF, nebo PDF/A.

- **Reporter** – Reporter transformuje strojově čitelné zprávy generované nástroji Implementation Checer, Metadata Fixer a Policy Checer na jiné formy pro následné použití.
- **Shell** – Shell zajišťuje pomocí dalších komponent interakci uživatele s VeraPDF. Uživatelé můžou pomocí Shell interagovat s VeraPDF pomocí příkazového řádku, grafického uživatelského rozhraní pro stolní počítače, nebo webového uživatelského rozhraní [39], [40].

9.3 Výběr vhodného řešení

V této kapitole budou zhodnoceny všechny placené i neplacené nástroje pro validaci PDF/A, bude z nich vybrán jeden, se kterým bude následně pracováno v praktické části.

Požadavky UTB byly následující:

a) Vybrat vhodný softwarový nástroj umožňující validaci všech aktuálně používaných norem PDF/A: PDF/A-1a, PDF/A-1b, PDF/A-2a, PDF/A-2b, PDF/A-2u, PDF/A-3a, PDF/A-3b, PDF/A-3u

b) Aby tento softwarový nástroj nabízel HTTP server s rozhraním REST API a mohl běžet jako HTTP serverová služba na adrese HTTP://pdfa.k.utb.cz:8080/.

c) Vhodné by bylo, aby tento nástroj spadal svou licencí do open source, aby za něj UTB nemusela zbytečně platit a mohla si v případě potřeby upravit zdrojový kód.

d) Dále by bylo výhodné, kdyby softwarový nástroj umožňoval definovat **seznam výjimek** na pravidla standardu ISO 19005, která budou během validace ignorovány.

e) Na tento softwarový nástroj budou z IS/STAG posílány požadavky na validaci PDF/A souborů. Velkou výhodou by bylo, aby nástroj uměl automaticky vybírat validační profil. To znamená automatický výběr normy (PDF/A-1, PDF/A-2, PDF/A-3) a úrovně shody (a, b, u) pod kterou testovaný soubor spadá.

   Z placených softwarových nástrojů uvedených v podkapitole 8.1 splňuje kompletně výše uvedené požadavky „3-Heights(TM) PDF Validator“. Jediný parametr, který

nesplňuje, je požadavek na open source, ale ten z podstaty věci splňovat nemůže, protože se jedná o komerční produkt. Na druhém místě se umístil PDF Automation Server od společnosti (Qoppa Software), tento produkt bohužel neumožňuje definovat seznam výjimek a chybí mu validace na standardy: PDF/A-1a, PDF/A-2a, PDF/A-3a. Stručný přehled srovnání je uveden v Tabulce 1.

Tabulka 1: přehled placených SW nástrojů největší shody s požadavky UTB

	a)	b)	c)	d)	e)
3-Heights(TM) PDF Validator	✓	✓	X	✓	✓
Callas PDFa Pilot	✓	✓	X	X	X
jPDFPreflight (Qoppa Software)	X	X	X	X	X
PDFTron SDK	✓	X	X	X	X
PDF Automation Server	✓	✓	X	X	✓

Z neplacených softwarových nástrojů uvedených v podkapitole 8.2 zvítězilo VeraPDF-rest, které kromě definování seznamu výjimek splňuje všechny ostatní požadavky.

Tabulka 2: přehled neplacených SW nástrojů srovnání s požadavky UTB

	a)	b)	c)	d)	e)
validátor Národního archivu	✓	X	X	X	✓
testovací Heights(TM) PDF Validator	✓	✓	X	X	✓
Verify PDF/A	✓	X	X	X	X
Converter PDFTron	✓	X	X	X	X
Apache PDFBox	X	X	✓	X	X
VeraPDF-rest	✓	✓	✓	X	✓

Celkově z komerčních a nekomerční ch řešení bylo pro praktickou část práce vybráno **VeraPDF-rest**.

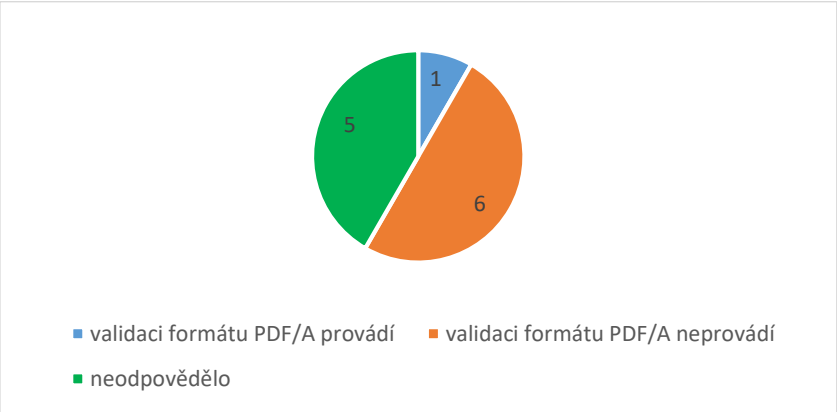
II. PRAKTICKÁ ČÁST

10 AKTUÁLNÍ STAV

Následující podkapitoly se zaměří na zmapování toho, jak se v současném stavu řeší validace odevzdaných závěrečných VŠKP na univerzitách používající IS/STAG, dále pak na to, co už bylo ve směru validace na formát PDF/A v minulosti na UTB podniknuto a následně podrobnému popisu aktuálních požadavků na validaci formátu PDF/A na UTB.

10.1 Současný stav validace PDF/A na univerzitách používající IS/STAG

V rámci praktické části byl udělán drobný průzkum, kolik univerzit používajících IS/STAG řeší nějakým způsobem validaci odevzdaných závěrečný VŠKP. Bylo osloveno celkem 12 univerzit. Odpovědělo 7 z nich, z toho kontrolu na formát PDF/A řeší pouze 1. Ta kontrolu provádí v univerzitním archivu při katalogizaci práce.



Graf 1: Kontrola formátu PDF/A u závěrečných VŠKP na univerzitách používajících IS/STAG

10.2 Historický a současný stav validace PDF/A na UTB

Do roku 2016 se závěrečné VŠKP ukládaly do knihovního systému DSpace pouze ve formátu PDF. Od roku 2017 se začaly ukládat VŠKP ve formátu PDF/A. Tím vznikl požadavek na správce informačního systému na UTB, aby byla nějak zavedena kontrola závěrečných VŠKP na formát PDF/A. Rozhodli se vydat cestou, kdy bude kontrola prováděna při samotném odevzdávání závěrečných prací. Technicky to bylo provedeno tak, že Java knihovna veraPDF byla přidána přímo do IS/STAG. Bohužel se tento přístup během prvního roku používání příliš neosvědčil, protože při nahrání některých závěrečných VŠKP docházelo k pádu veraPDF a ten následně způsoboval pád celého

IS/STAG. Po pádu bylo nutné IS/STAG ručně restartovat. V tomto nestabilním režimu byl dokončen semestr 2017.

Do dalšího semestru byla nasazena klasická verze bez kontroly PDF/A. Správci informačních systémů na UTB směřovali své další úvahy k řešení, kde by se použila Java aplikace veraPDF-rest, což je Representational State Transfer API (REST API), zprostředkovávající validační funkce z veraPDF. VeraPDF-rest by běžela jako služba na školním serveru a prováděla by validaci PDF/A vzdáleně. Případný pád této služby, by tak již neovlivňoval chod IS/STAG.

Pro studenty existuje návod, jak vygenerovat PDF/A z Wordu, OpenOffice a LaTeX. Nicméně se stává, že studenti někdy odevzdají obyčejné PDF, nebo zejména vygenerované závěrečné VŠKP z Wordu občas poruší nějaké pravidlo a proto můžou být validátorem veraPDF-rest označeny jako chybné.

10.3 Aktuální požadavky na validaci PDF/A na UTB

Cílem IT správců není komplikovat studentům odevzdávání závěrečných VŠKP zbytečnými technickými požadavky tzn. chtít po nich za každou cenu 100% validní soubor ve formátu PDF/A, ale na druhou stranu chtějí mít kontrolu, že soubor z větší části vyhovuje standardu PDF/A, že se nejedná například o obyčejný PDF soubor, který je z pohledu vyhlášky (č. 259/2012 Sb., o podrobnostech výkonu spisové služby) pro archivaci nedostačující. Správci informačního oddělení na UTB se rozhodli, že raději přijmou závěrečnou VŠKP, která je validní se standardem PDF/A z 99% než požadovat po studentech za každou cenu 100% validní PDF/A soubor.

1) Z toho důvodu vzešel požadavek na to, aby měli možnost definovat seznam výjimek pravidel, která nebudou validátorem hodnocena. Základní seznam výjimek si IT správci zjistí z korpusu již odevzdaných závěrečný VŠKP, které mají v metadatech informaci, že byly vygenerované z Wordu jako PDF/A, ale přitom neprošly validátorem. Jakmile se narazí na nějaké nové výjimky, množina výjimek bude aktualizována.

2) Dále bylo požadováno, aby bylo možno všechny validované soubory ukládat na školní server, jako název bude sloužit jejich SHA-1 hash.

3) Dalším požadavkem bylo logovat do databáze všechny otisky zpracovaných souborů sha1hex, na stejný řádek k nim také ukládat informace: zda byl soubor

validní/nevalidní, časové razítko, délka zpracování požadavku, status kód odpovědi, v případě selhání aplikace logovat všechny výjimky, které byly daným souborem vyvolány.

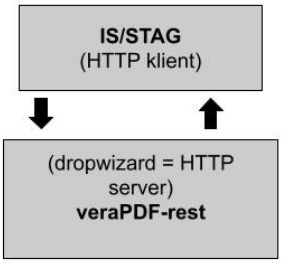
4) V případě, že zpracovávaný požadavek povede k vyvolání jakékoliv výjimky, poslat IT správcům informační email.
5) VeraPDF-rest dokáže vracet tělo odpovědi ve tvarech HTML, JavaScript Object Notation (JSON), XML. Požadavkem bylo, aby tělo odpovědi přicházející do **IS/STAG** bylo pouze ve formátu JSON.

11 NÁVRH ŘEŠENÍ

Z vhodných návrhů uvedených v teoretické části v kapitole 9.3 Výběr vhodného řešení byla vybrána jako nejvhodnější aplikace pro validaci souborů na formát PDF/A Java aplikace **veraPDF-rest**.

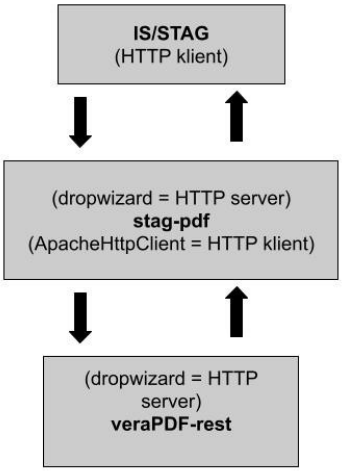
V návrhu řešení tedy figurovaly od začátku 2 části: **IS/STAG** který bude vytvářet požadavky na **veraPDF-rest** a **veraPDF-rest**, který bude validaci provádět. To jak splnit požadavky definované v části **Chyba! Nenalezen zdroj odkazů. Chyba! Nenalezen zdroj odkazů.** bylo možné provést dvěma možnostmi.

První možností bylo upravit **veraPDF-rest** a implementovat do něj všechny požadavky definované v 10.3, viz Obrázek 6.



Obrázek 6: Návrh řešení možnost 1

Druhou možností bylo vytvořit samostatnou aplikaci, která by fungovala jako komunikační mezivrstva. Pro tuto mezivrstvu bude dále používán název **stag-pdf** viz Obrázek 7. V této mezivrstvě by byly aplikovány všechny požadavky z části **Chyba! Nenalezen zdroj odkazů. Chyba! Nenalezen zdroj odkazů.**, mezivrstva by poskytovala HTTP server a HTTP klienta.



Obrázek 7: Návrh řešení - možnost 2

Jednodušším řešením pro správu by byla první možnost, nicméně tento přístup byl vyhodnocen jako časově náročnější. Protože by bylo nutné se kompletně zorientovat v aplikacích **veraPDF-rest** a **veraPDF.** Provést úpravy a následně provádět rozsáhlé testování, zda modifikace **veraPDF** neovlivnily nějak negativně některou jinou její část. Nakonec bylo rozhodnuto, že se práce realizuje podle druhé možnosti. V takovém případě bylo nutno z **veraPDF-rest** nastudovat pouze na jakých endpointech naslouchá, jakou strukturu POST požadavků očekává pro vrácení odpovědi ve formátu JSON, jaká je struktura vrácené JSON odpovědi:

a) Když je testovaný soubor validní,

b) když je soubor nevalidní.

11.1 Popis konkrétního návrhu

Na základě možností, které veraPDF-rest nabízí, byl vytvořen následující návrh řešení, viz Obrázek 6.

IS/STAG bude vytvářet POST požadavky na validaci závěrečných VŠKP tyto požadavky budou posílány na konkrétní endpoint na adrese HTTP://pdfa.k.utb.cz:8080/api/validate/auto. **Stag-pdfa** bude přeposílat požadavky na **VeraPDF-rest** na endpoint HTTP://pdfa.k.utb.cz:7070/api/validate/auto. **VeraPDF-rest** provede validaci a pošle rozsáhlou JSON odpověď na **stag-pdfa**. **Stag-pdfa** rozparsuje přijatou odpověď a dále bude pracovat pouze s informací, zda je soubor validní a se seznamem porušených pravidel pro formát PDF/A.

V případě, že validovaný PDF/A soubor byl vyhodnocen jako validní, pošle **stag-pdfa** jednoduchou kladnou odpověď do **IS/STAG**.

```
{"compliant":"true"}
```

V případě, že validovaný PDF/A soubor byl vyhodnocen, jako nevalidní se bude postupovat podle následujícího pseudokódu Kód 9.

```
1  Načti z konfiguračního souboru seznam výjimek do pole
   RuleViolationExceptions
2  pokud je soubor vyhodnocen jako validní
3        vrať compliant=true
4  pokud je soubor vyhodnocen jako nevalidní
5        foreach všechny vrácená porušené pravidla
6              přidej porušené pravidlo do pole RuleViolations
7        Proveď rozdíl polí RuleViolations a
   RuleViolationExceptions
8        pokud je RuleViolations prázdné
9              vrať compliant=true
10       jinak
11             vrať compliant=false
```

Kód 9: Pseudokód porovnávání výjimek

Stag-pdfa se podívá do seznamu výjimek pravidel. Udělá rozdíl množiny porušených pravidel a rozdíl množiny seznamu výjimek. Pokud bude výsledkem rozdílu prázdná množina, pošle **stag-pdfa** kladnou odpověď `{"compliant":"true"}` do **IS/STAG**, pokud výsledkem rozdílu bude neprázdná množina, pošle **stag-pdfa** jednoduchou zápornou odpověď `{"compliant":"false"}`.

Co se týče návratových stavových kódů: Ve světě webových technologií můžou být používány stavové kódy 100-599. Podle první číslice z čísla stavového kódu je možné stavový kód zařadit do některé z pěti následujících skupin:

- 100-199 Označují, že zpráva obsahuje prozatímní informační odpověď.
- 200-299 Označují, že požadavek byl úspěšný.
- 300-399 Označují, že požadavek musí být přesměrován na jiný zdroj.
- 400-499 označují, že klient udělal chybu, která by se neměla opakovat
- 500-599 Označují, že server narazil na chybu, ale že klient může získat lepší odpověď později [41].

Stag-pdfa bude vracet odpovědi pouze s dvěma stavovými kódy. Odpověď se stavovým kódem 200 v případě, že dojde k úspěšnému provedení validace, tzn. je vrácena buď odpověď `{"compliant":"true"}` nebo odpověď `{"compliant":"false"}`. V případě, že dojde k jakékoliv vnitřní chybě uvnitř **stag-pdfa,** je vrácena odpověď.

`{"Error 500 Internal Server Error"}` se stavovým kódem 500.

12 EXISTUJÍCÍ APLIKACE

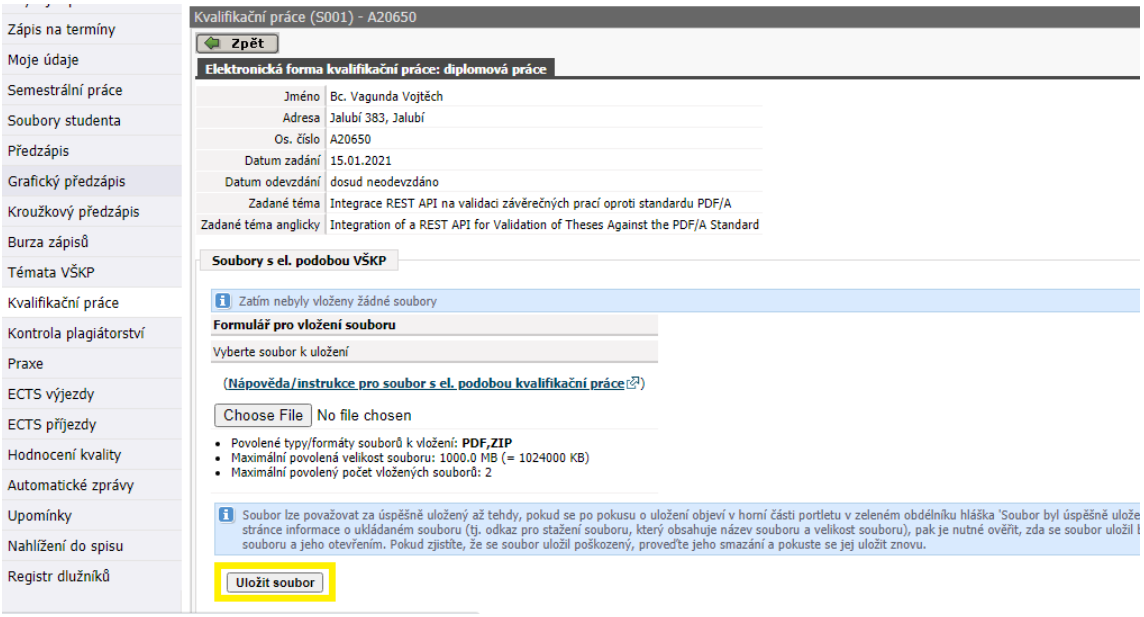
Vývoj **stag-pdfa** byl ovlivněn strukturou existujících aplikací, mezi kterými bude zprostředkovávat komunikaci. Ve dvou následujících podkapitolách budou ve stručnosti popsány **IS/STAG** a **vera-pdf**.

12.1 IS/STAG

IS/STAG je informační systém studijní agendy vysoké školy. Tento informační systém je vyvíjen v Centru informatizace a výpočetní techniky na Středisku informačních systémů na Západočeské univerzitě v Plzni [42].

Na IS/STAG je běžně nahlíženo jako na server, kdy klient (běžný uživatel) volá jeho služby. V této práci bude na IS/STAG nahlíženo jako na klienta, který bude posílat požadavky na validační službu. Jak bylo zmíněno v kapitole **Chyba! Nenalezen zdroj odkazů. Chyba! Nenalezen zdroj odkazů.**, v roce 2017 byl ze strany UTB vytvořen pokus o validaci PDF/A přímým vložením knihovny veraPDF do IS/STAG. Časté pády celého IS/STAG nakonec vedly k návratu k původní staré verzi bez validace PDF/A. Vzhledem k tomu, že UTB byla jediným zákazníkem, který tuto funkcionalitu požadoval, nebylo již dále ze strany dodavatele IS/STAG na této službě nijak pracováno. To vedlo k tomu, že se IT správci na UTB dohodli, že bude vhodné minimalizovat množství úprav na straně IS/STAG a vyvinout samostatně stojící řešení na které se bude moci IS/STAG posílat požadavky.

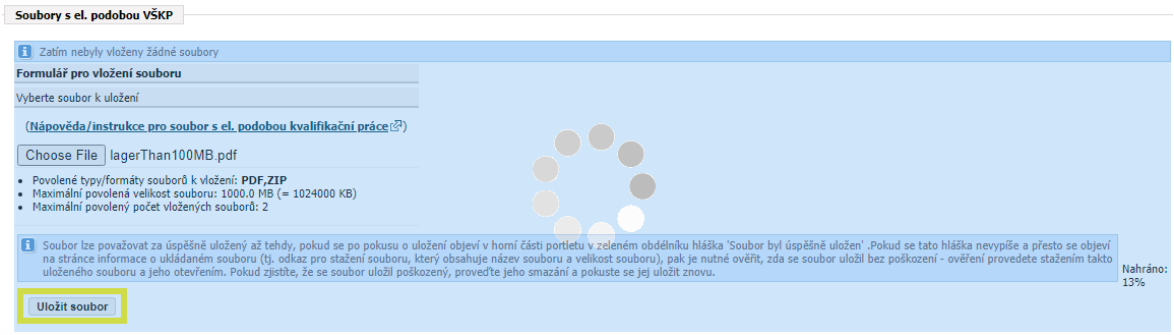
Odevzdávání závěrečných VŠKP se v IS/STAG nachází: Moje studium > Kvalifikační práce > Doplnit údaje o diplomové práci (popř. odevzdat el. podobu práce) > Formulář pro odevzdání souborů s elektronickou podobou VŠKP. Formulář je zachycen na Obrázku 8.



Obrázek 8: Formulář pro odevzdání souborů s elektronickou podobou VŠKP

Při běžném procesu nahrávání závěrečných VŠKP vybere uživatel soubor a klikne na tlačítko „Uložit soubor“, soubor je následně nahrán.

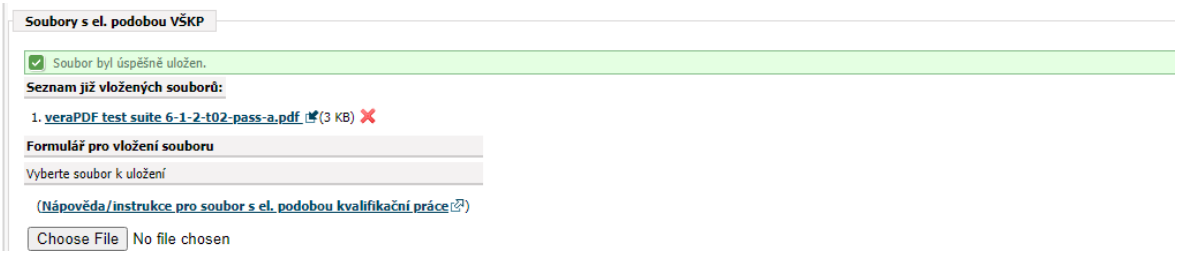
Po přidání funkcionality validace se z pohledu uživatele v postupu nahrávání práce nic nezmění. K provedení validace na formát PDF/A, dojde ihned po nahrání souboru na server IS/STAG, viz Obrázek 9.



Obrázek 9: Načítání souboru na formuláři Obrázek 8

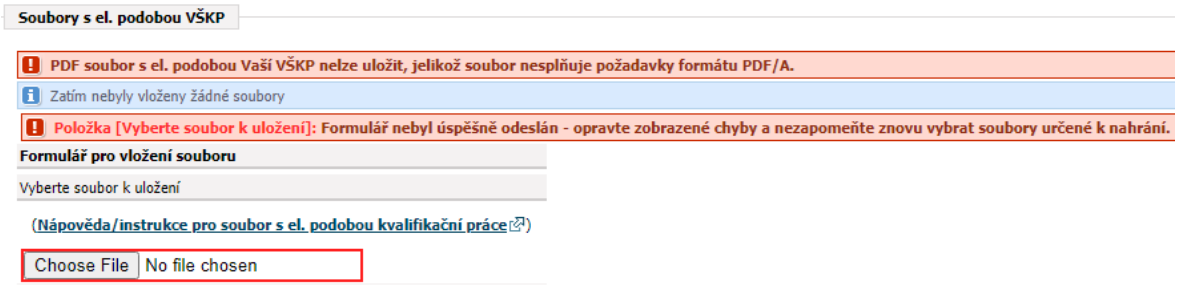
Provedení validace může skončit třemi běžnými stavy:

a) Soubor byl ohodnocen jako validní s formátem PDF/A, nebo na něj byly aplikovány výjimky pravidel (viz Kód 10) a díky nim byl soubor také ohodnocen jako validní. V takovém případě, je IS/STAG vrácena odpověď ze **stag-pdfa** s tělem `{"compliant":"true"}` a v IS/STAG zobrazena hláška Obrázek 10:



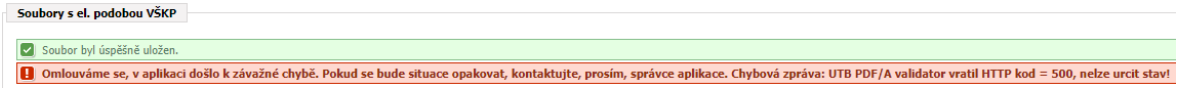
Obrázek 10: Soubor prošel validací PDF/A

b) Soubor nebyl validní s formátem PDF/A, v takovém případě je IS/STAG vrácena ze **stag-pdfa** odpověď s tělem `{"compliant":"false"}` a v IS/STAG uživateli zobrazeno informační okno Obrázek 11. Nahraný soubor je v takovém případě zahozen.



Obrázek 11: Soubor neprošel validací PDF/A

V případě jakýchkoliv problému uvnitř **stag-pdfa** je IS/STAG vrácena odpověď `{"Error 500 Internal Server Error"}`, uživateli zobrazeno informační okno Obrázek 12 a validovaná práce uložena na server IS/STAG. Důvod je ten, že v případě jakýchkoliv chyb na straně **stag-pdfa** nechtějí IT správci blokovat odevzdávání závěrečných VŠKP.



Obrázek 12: Chyba **stag-pdfa**

12.2 VeraPDF-rest

VeraPDF-rest je Java aplikace zpřístupňující pomocí REST API endpointů funkce pro validaci PDF/A souborů z knihovny VeraPDF. VeraPDF je průmyslově podporovaný open source validátor pro kontrolu struktury souborů na standard PDF/A. Více o veraPDF z obecného pohledu je v podkapitole 9.2.6 VeraPDF.

12.2.1 Pravidla kontroly PDF/A

Pravidla, pomocí kterých knihovna **veraPDF** provádí automatickou kontrolu na formát PDF/A jsou definována v repozitáři veraPDF-validation-profiles[12], který byl vytvářen souběžně s projektem **veraPDF**. Na příkladu níže bude ukázáno, jakým způsobem jsou pravidla vytvářena. Pro ukázku budou vzata pravidla ze standardu PDF/A-1.

Jak bylo zmíněno v kapitole 4, pravidla standardu PDF/A-1 jsou definována v dokumentu ISO 19005-1, oficiálně je nutno tento dokument zakoupit přímo na stránkách organizace ISO, neoficiálně je možné dokument stáhnout zdarma na webové stránce[13]. V tomto dokumentu se přesuneme na část 6 Technical requirements do podkapitoly 6. 1 File structure. Zde je definováno pravidlo 6. 1. 2 pro hlavičku souboru. Pravidlo v orginálu zní:

„The % character of the file header shall occur at byte offest 0 to the file. The file header line shall be immediatelly followed by a comment consisting of a % character followed by at least four characters, each of whose encoced byte values shall have a decimal value greater than 127.“ [13]

Podle svého rozsahu můžou být pravidla rozdělena na několik částí. Pravidlo 6. 1. 2 je rozděleno na 2 části:

1) ***The % character of the file header shall occur at byte offset 0 of the file. The first line of a PDF file is a header identifying the version of the PDF specification to which the file conforms***.
2) *The file header line shall be immediately followed by a comment consisting of a % character followed by at least four characters, each of whose encoded byte values shall have a decimal value greater than 127.*

Pro každou část je definován testovací profil. Testovací profily jsou umístěné v repozitáři veraPDF-validation profiles[14] a jsou zapsány ve formátu XML. V případě 6. 1. 2 se jedná o dva testovací profily, ukázka prvního, viz Kód 10.

---

[12] Dostupné z: https://github.com/veraPDF/veraPDF-validation-profiles

[13] Dostupné z: http://www.csygen.com/pdf/iso_19005.PDF

[14] Dostupné z: https://github.com/veraPDF/veraPDF-validation-profiles.git

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<profile xmlns="HTTP://www.verapdf.org/ValidationProfile" flavour="PDFA_1_A">
 <details creator="veraPDF Consortium" created="2016-02-15T10:58:04.766+03:00">
        <name>ISO 19005-1:2005 - 6.1.2 File header - PDF header</name>
        <description>The % character of the file header shall occur at byte offset 0 of the file. The first line of
        a PDF file is a header identifying the version of the PDF specification to which the file conforms
        </description>
 </details>
 <hash></hash>
 <rules>
 <rule object="CosDocument">
        <id specification="ISO_19005_1" clause="6.1.2" testNumber="1"/>
        <description>The % character of the file header shall occur at byte offset 0 of the file. The first line of
        a PDF file is a header identifying the version of the PDF specification to which the file conforms
        </description>
        <test>headerOffset == 0 &amp;amp;&amp;amp; /%PDF-\d\.\d/.test(header)</test>
        <error>
        <message>File header does not start at byte offset 0 or does not correctly identify the version of the PDF
        document</message>
 <arguments/>
 </error>
 <references>
 <reference specification="PDF 1.4 Reference" clause="3.4.1"/>
 </references>
 </rule>
 </rules>
 <variables/>
</profile>
```

Kód 10: Ukázka validačního profilu 6. 1. 2, test 1

Popis nejdůležitějších atributů a elementů, viz Tabulka 3:

- **<name>** Obsahuje ISO PDF/A standard, číslo a název pravidla které testuje daný profil.
- <**description**> Obsahuje textovou část, pravidla na které se vztahuje daný testovací profil. V našem příkladu se jedná o část **1)**. Description je použit v případě, že validátor nalezne v testovaném souboru chybu. V takovém případě, je v odpovědi použito description jako popisek chyby.
- <**id**> Tento element obsahuje 3 atributy:
    - specification – Definuje číslo PDF/A standardu, ke kterému byl daný test vytvořen.

  - clause – Číslo pravidla, které odpovídá stejnému číslu, použitém v souboru ISO 19005-1.
  - testNumber – Je číslo části pravidla. V našem případě bylo pravidlo 6. 1. 2 rozděleno na 2 části.

- <**test**> Tento element obsahuje konkrétní test zapsaný pomocí JavaScript regular expression (regex).
- <**reference**> Tento element obsahuje 2 atributy:
  - specification – Definuje název PDF normy, ze které bylo pravidlo 6. 2. 1 odvozeno.
  - clause - Definuje podkapitolu PDF standardu, z které bylo pravidlo 6. 2. 1 odvozeno. V našem příkladu je hodnota atributu číslo „3.4.1“ což odpovídá stejnojmenné podkapitole ve zdroji [13].

12.2.2 Požité technologie

Aplikace **veraPDF-rest** je založena na frameworku DropWizard. DropWizard spojuje spolehlivé Java knihovny do jednoduchého balíčku, který umožňuje soustředit se na vývoj konkrétní aplikace a šetří čas s vymýšlením infrastruktury. V DropWizard jsou nejvíce zastoupené následující knihovny:

Jetty – Tato knihovna zprostředkovává HTTP server. Místo předání aplikace složitému aplikačnímu serveru, obsahují projekty DropWizard main metodu, která přímo spouští Jetty HTTP server. Jetty je pod licencí Apache 2.0 [43]. Projekt Apache usiluje o vývoj softwaru založený na spolupráci, zaměřuje se na vytvoření robustní, plně funkční, plně vybavené open source softwarové implementaci HTTP serveru. Projekt je společně řízen skupinou dobrovolníků z celého světa, kteří používají internet a web ke komunikaci, plánování a vývoji serveru a související dokumentace. Apache se stal známý jak pro své robustní chování v reakci na různé požadavky internetových služeb, tak pro svou důslednou implementaci standardů protokolu HTTP [44].

Jersey – Velmi osvědčený REST Framework, který poskytuje implementaci JAX-RS (JSR-370, JSR 339, JSR 311). Zjednodušuje vývoj webových služeb REST a jejich klientů v Javě pomocí rozhraní JAX-RS API [45].

Jackson – Poskytuje sadu nástrojů pro zpracování dat pro Javu. Základní komponenty tohoto projektu, žijí pod vlastními projekty, projekt funguje jako centrální rozbočovač pro propojení všech částí dohromady. Základ tvoří 3 moduly:

- **Streaming** – Definuje nízko úrovňové streamovací API a zahrnuje implementace specifické pro JSON.
- **Anotace** – Obsahuje standardní anotace pro Jackson (@JsonCreator, @JsonProperty, @JsonIgnore a další).
- **Databind** – Implementuje datové vazby a serializaci objektů.

Na těchto třech základních modulech staví ostatní rozšiřující moduly [46].

V DropWizard jsou dále implementované knihovny:

- Logback
- Hibernate Validator
- Apache HTTPCleint a Jersey client
- Liquidbase
- Freemaker a Mustache
- Joda Time

12.2.3 Endpointy veraPDF-rest

VeraPDF-rest nabízí několik endpointů. V přístupu, jaký byl pro tuto práci nakonec zvolen, si vystačíme se dvěma z nich. První endpoint je vhodný pro zjištění, zda je aplikace v běžícím stavu. Jedná se o endpoint vyřizující požadavek GET: HTTP://pdfa.k.utb.cz:7070/api/info. Jako odpověď vrátí XML s informacemi o verzi serveru, na kterém běží, a o verzi Javy, která je na serveru nahrána, viz ukázka Kód 11.

```
<Environment>
<os>
<name>Linux</name>
<version>4.19.0-16-amd64</version>
<architecture>amd64</architecture>
</os>
<Java>
<vendor>Debian</vendor>
<version>11.0.9.1</version>
<architecture>x64</architecture>
<home>/usr/lib/jvm/Java-11-openjdk-amd64</home>
```

```
12 </Java>
13 <server>
14 <ipAddress>195.178.95.130</ipAddress>
15 <hostName>pdfa</hostName>
16 <machAddress>00-15-5D-8D-03-6D</machAddress>
17 </server>
18 </Environment>
```

Kód 11: Ukázka odpovědi na dotaz na endpoint /api/info

Druhý pro nás důležitý endpoint přijímá POST požadavky na adrese HTTP://pdfa.k.utb.cz:7070/api/validate/{profileId}. Definici můžeme najít v balíčku resources ve třídě ValidateResource.Java, viz Kód 12.

```
1  @POST
2  @Path("/{profileId}")
3  @Consumes(MediaType.MULTIPART_FORM_DATA)
4  @Produces({ MediaType.APPLICATION_JSON, MediaType.APPLICATION_XML
   })
5  public static ValidationResult validatePost(
6  @PathParam("profileId") String profileId,
7  @FormDataParam("sha1Hex") String sha1Hex, @FormDataParam("file")
   InputStream uploadedInputStream,          @FormDataParam("file")
   final FormDataContentDisposition contentDispositionHeader)throws
   VeraPDFException {
8         return validate(profileId, sha1Hex, uploadedInputStream);
9  }
```

Kód 12: metoda validatePost na adrese /api/validate/{profileID}

Parametr {profileID} může nabývat hodnot: "1b", "1a", "2b", "2a", "2u", "3b", "3a", "3u" nebo "auto". Hodnoty parametru definují, jakým validačním profilem má být daný PDF/A soubor zkontrolován. Používají se v případě, že je dopředu známo jaká norma PDF/A byla použita při vytváření PDF/A souboru. V případě že si uživatel není jistý, jaká norma PDF/A byla při vytváření souboru použita, nabízí se poslední volba "auto", při tomto parametru je výběr nejvhodnějšího validačního profilu ponechán na aplikaci **veraPDF-rest**. Aplikace se podle obsahu validovaného souboru rozhodne, jakým validačním profilem bude soubor testovat. Parametry `sha1Hex` a `contentDispositionHeader` jsou nepovinné. Metoda validatePost umí vracet tělo odpovědi ve formátu XML a JSON.

V Kód 13 je příklad POST požadavku pomocí curl. Je poslán post požadavek na validování souboru „test.pdf“. Parametr "Accept: application/json" znamená, že požadujeme odpověď ve formátu JSON.

```
curl -s -F"file=@D:\dokumenty\test.pdf"
HTTP://pdfa.k.utb.cz:7070/api/validate/auto -H "Accept:
application/json"
```

Kód 13: Post požadavek pomocí curl

12.3 Stag-pdfa

Pro splnění požadavků v návrhu **Chyba! Nenalezen zdroj odkazů.** bylo nutno zajistit, aby aplikace **stag-pdfa** poskytovala REST API pro **IS/STAG** a HTTP klienta pro přeposílání požadavků dál na **veraPDF-rest**. Při prvotním návrhu se uvažovalo nad možnostmi vytvoření REST API pomocí:

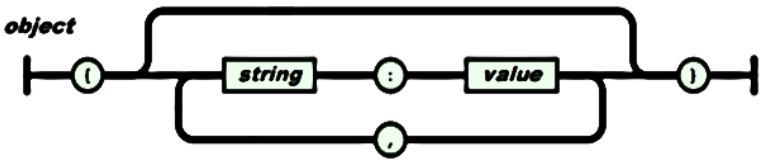
- ASP.NET MVC 4 Web Application

- Java Frameworku Spring

- Java framoworku DropWizard

Nakonec byl vybrán Java framework DropWizard. Hlavní důvod pro jeho výběr byl, že aplikace **veraPDF-rest** je také vytvořena pomocí tohoto Java frameworku. Tudíž pokud se osvědčí aplikování seznamu výjimek na výsledek validace, mohla by se tato funkcionalita přidat v budoucnu přímo do **veraPDF-rest**. Případné modifikace **veraPDF-rest** v budoucnu, by tak byly mnohem snadnější a intuitivnější díky použití stejného frameworku.

Co se týče formátu odpovědí, ze stag-pdfa budou všechny odpovědi vraceny ve formátu JSON. Syntaxe JSON je definována ve standardu ECMA-404. JSON nabízí hodnoty: objekt, pole, číslo, řetězec, pravda, nepravda, null. Stag-pdfa používá pouze objekty. Struktura objektu je reprezentována jako dvojice složených závorek obklopujících jeden nebo více párů **jméno/hodnota**. **Jméno** je vždy typu řetězec, za ním následuje znak dvojtečky, který odděluje jméno od hodnoty. Pokud objekt obsahuje více párů, jsou odděleny pomocí čárky [47]. Viz Obrázek 13 níže.



Obrázek 13: Struktura JSON objektu [46]

12.3.1 Funkce stag-pdfa

Stag-pdfa poskytuje všechny funkce, které byly definovány v požadavcích ze strany IT správců z UTB (viz kapitola 10.3). Funkce jsou následující:

- Je možné definovat seznam výjimek - Seznam výjimek pro pravidla PDF/A je možné definovat v konfiguračním souboru *config.yml*. Pravidla jsou uložena v

repozitáři **veraPDF-validation-profiles**[15]. Každé pravidlo je definováno, pomocí atributu **clause** a **testNumber. TestNumber** je podmnožinou **clause,** viz kapitola 12. 2. 1. Například: Pravidlo 6. 1. 2 se skládá z pravidla **6. 1. 2-testNumer=1** a pravidla **6. 1. 2-testNumer=2.** Z tohoto důvodu jsou u seznamu výjimek možné 2 konfigurace:

1. Výjimky budou definovány na celé pravidlo včetně jeho podmnožin. Například 6. 1. 2.
2. Výjimky budou definovány pro konkrétní podmnožinu pravidla, použitím čísla **testNumber**. Tedy například 6. 1. 2-1, nebo 6. 1. 2-2.

- Dále je možné ukládat testované soubory na server, jako název je používán jejich hash otisk.
- Odesílání emailu v případě, výskytu chyby v aplikaci – Pro odesílání emailů jsou možné dvě konfigurace:
    - První: Je odesílání emailů bez ověřování uživatele vůči emailovému serveru. Toto použití je možné například v případě, že aplikace stag-pdfa běží lokálně uvnitř UTB a využívá emailový server „smtp.utb.cz“. V tomto případě je nutno v konfiguračním souboru nastavit na řádku #authentization hodnotu „false“ a dále jen nastavit řádky: `#from, #to a #port`, “, viz Kód 14.
    - Druhý: Je odesílání emailů s ověřováním uživatele vůči emailovému serveru. V tomto případě je nutno na řádku `#authentization` nastavit hodnotu na „true“ a dále nastavit všechny ostatní řádky: `#user, #pass, #from, #to, #host` a `#port`.

      Poznámka: Při používání emailového serveru od google je nutné u emailu uvedém v #to povolit přístup pro méně zabezpečené aplikace [48].
- Logování do databáze – Existoval požadavek, aby byl každý testovaný soubor, který projde přes **stag-pdfa** zalogován. Nakonec bylo rozhodnuto, že se budou logovat o každém zpracovaném souboru následující informace, viz Tabulka 3.

[15] Dostupné z: https://github.com/veraPDF/veraPDF-validation-profiles

Tabulka 3: Návrh logovací tabulky stagpdfa_logs

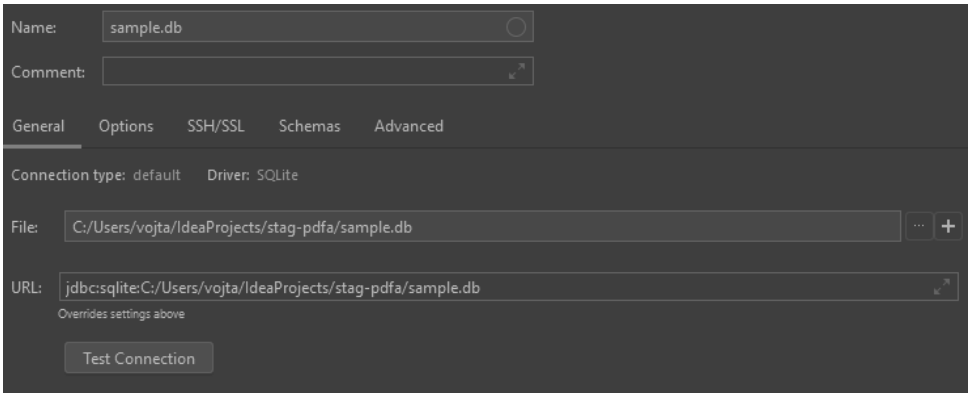
stagpdfa_logs	
sha1	text
verapdf_rest_response	text
request_time	integer
verapdf_rest_request_time	integer
status_code	integer
error_message	text
request_timestamp	TIMESTAMPTZ

Vysvětlení významu jednotlivých sloupců:

- Sha1 – Je čtyřiceti znakový hexadecimální otisk kontrolovaného daného souboru.
- verapdf_rest_response – Je výsledek testování souboru z **veraPDF-rest**, zda je validní, nebo není.
- request_time – Je délka zpracování požadavku na **stag-pdfa**, počítáno od přijetí požadavku do odeslání dopovědi, jednotka [milisekundy].
- verapdf_rest_request_time – Je délka zpracování požadavku na **veraPDF-rest**, počítáno od odeslání požadavku ze **stag-pdfa** do přijetí dopovědi, jednotka [milisekundy].
- status_code – Je stavový kód odpovědi, která byla vrácena z **veraPDF-rest**.
- error_message – Stack trace všech výjimek v případě, že dojde k vnitřní chybě ve **stag-pdfa**.
- request_timestamp – Je konkrétní čas, kdy byl požadavek přijatý na **stag-pdfa**.

Co se týče konkrétní databáze, na produkčním linuxovém serveru je použita databáze PostgreSQL. Na lokálním stroji lze pro testovací účely použít například databáze SQLite. Výchozí název pro databázový soubor je uveden *config.yml* na řádku 25 (viz Kód 14), defaultní název je `sample.db`. Při testování na lokálním stroji je nejjednodušší se připojit k SQLite například pomocí vývojového prostředí IntelliJ. Postup pro připojení pomocí IntelliJ IDEA Ultimate 2020.3.3 je následující:

1) Otevřeme Database window: View>Tool Windows > Databases
2) Pomocí tlačítka „+“ přidáme nový Datový zdroj „SQLite“
3) Vyplníme konfigurační okno, viz příklad Obrázek 14.



Obrázek 14: Připojení k databázovému souboru sample.db v IntelliJ IDEA Ultimate

12.3.2 Spuštění stag-pdfa na localhost

Postup při spuštění stag-pdfa na lokálním počítači je následující:

1) V příkazovém řádku zkontrolovat pomocí příkazu `mvn -version`, zda je na lokálním PC nainstalovaný Maven. Pokud nainstalovaný je, vypíšou se informace o jeho verzi, pokud nainstalovaný není, je vypsána hláška „příkaz nerozeznán“, v takovém případě je nutno Maven nainstalovat[16].
2) Stáhnout na lokální počítač zdrojový kód **stag-pdfa**, který je součásti přílohy na CD.
3) Přejmenovat soubor *config.yml.example* na soubor *config.yml.*
4) Nastavit parametry v souboru *config.yml* tak, aby vyhovovaly operačnímu systému počítače, kde je aplikace spouštěna. Konkrétní parametry jsou popsány v souboru *README.md*
5) V příkazovém řádku přejít do kořenové složky **stag-pdfa,** zadat příkaz `mvn clean balíčku` pro zkompilování a zabalení do jar souboru.
6) Spuštění aplikace pomocí: `Java -jar target/dropwizard-1.0-SNAPSHOT.jar server config.yml`.

[16] Dostupné z: https://mkyong.com/maven/how-to-install-maven-in-windows/

12.3.3 Konfigurační soubor *config.yml*

Při spuštění aplikace stag-pdfa je zadáván v parametrech název konfiguračního souboru. V aplikaci je používán konfigurační soubor s názvem *config.yml*. Při změně parametrů konfiguračního souboru, je nutné běžící aplikaci zastavit a spustit znovu. Není nutné ji znovu kompilovat pro daný PC. V konfiguračním souboru se vyskytují následující parametry:

```
1  logging:
2   level: INFO
3   loggers:
4   dropwizard: DEBUG
5  server:
6   applicationConnectors:
7   - type: HTTP
8   port: 8080
9   adminConnectors:
10  - type: HTTP
11  port: 8081
12 stagpdfa:
13  exceptions:
14       - 6.1.2-1
15       - 6.1.3
16       - 6.3.2
17  pathToSentFilesFolder:
18       #- logs/files/
19       - D:\\tmp\\
20  urlToVeraPDFrest:
21       #- HTTP://HOST:PORT/api/validate/auto
22       - HTTP://pdfa.k.utb.cz:7070/api/validate/auto
23  databaseUrlJdbc:
24       #- jdbc:postgresql://HOST/DB?user=USER&password=PASS
25       - jdbc:sqlite:sample.db
26  cleanDatabaseTableAtStart:
27       # parameters: true or false
28       - false
29  delayProcessingTheRequest:
30       #parameters: true or false
31       - false
32  testSwitch:
33       #parameters: deff, f31, f32, f4, f5, f6,
34       - deff
35  inputStramProcessor:
36       #parameters:  InputStreamProcessor1,  InputStreamProcessor2,
    switchoff
37       - InputStreamProcessor1
38  JavaMail:
39       - stag.pdfa@gmail.com     #user
40       - Stag.Pdfa1J*            #pass
41       - stag.pdfa@gmail.com     #from
42       - vojta@vagunda.eu        #to
43       - smtp.gmail.com          #host
44       - 587                     #port
45       - true                    #authentization
```

Kód 14: Konfigurační soubor *config.yml*

Parametry na řádcích 1-11 jsou defaultně vytvářeny při založení nového DropWizard projektu.

- Parametr `level` na řádku 2 definuje, výchozí parametr pro logování.
- Parametr `dropwizard` na řádku 4 definuje, jaká konkrétní úroveň zpráv je logována.
- Na řádku 7 je definován typ komunikace, zda bude šifrovaná, nebo nešifrovaná.
- Na řádku 8 je definován port, na jakém se aplikace spustí.

Parametry na řádcích 12 - 45 souvisí s aplikací **stag-pdfa**, budou podrobněji probrány níže:

- Na řádcích 14-16 jsou položky seznamu, parametru `exceptions.` Tyto položky definují výjimky pravidel, viz pseudokód Kód 9. Je možno použít dva přístupy definice výjimek: Buď definovat do výjimek **celá pravidla** např. 6.1.2, nebo definovat výjimky více specifičtěji pomocí **pravidla a podmnožiny** např. 6.1.2-1. Konkrétní příklad je v kapitole 12.3.4 příklad c).
- Parametr `pathToSentFilesFolder` na řádku 17, definuje cestu, kam bude uložen testovaný soubor.
- Parametr `urlToVeraPDFrest` na řádku 20 definuje cestu k endpointu běžící aplikace **veraPDF-rest**, kam bude ověřovaný PDF soubor přeposlán k validaci na formát PDF/A.
- Parametr `databaseUrlJdbc` na řádku 23 definuje přístup k databázi. V databázi je vytvořena 1 tabulka pro logování provozu. Viz více v části 12. 3. 1.
- Parametr `cleanDatabaseTableAtStart` na řádku 26 definuje, zda bude databázová tabulka s logovaným provozem při dalším spuštění aplikace **stag-pdfa** promazána (`true`), nebo ponechána (`false`) a bude se do ní pouze připisovat.
- Parametr `testSwitch` na řádku 32 definuje, zda bude aplikace spuštěna v testovacím režimu (`f31, f32, f4, f5, f6,`), v případě že `testSwitch="deff"` je aplikace spuštěna běžným způsobem.
- Parametr `inputStramProcessor` na řádku 35 definuje, zda bude pro zpracování požadavku použita třída InputStramProcessor1, nebo třída InputStramProcessor2, každá třída zpracovává InputStream z požadavku jiným způsobem. Více o třídách v kapitole 12. 3. 5.

- Na řádcích 39 – 45 se definují informace pro odesílání emailu. Více o odesílání emailů v kapitole 12. 3. 1. Pokud je vyžadováno přihlášení k emailovému serveru, je nutné zkontrolovat řádky `#pass, #from, #to, #host, #port`. Pokud přihlášení k emailovému serveru vyžadováno není, stačí nastavit pouze řádky `#from, #to, #port, #authentization.`
    - Řádek 39 – Definuje uživatelské jméno, pod kterým bude možné se přihlásit k emailovému serveru.
    - Řádek 40 – Definuje heslo pro přihlášení k emailovému serveru.
    - Řádek 41 – Definuje odesílatele.
    - Řádek 42 – Definuje příjemce.
    - Řádek 43 – Definuje emailový server.
    - Řádek 44 – Definuje port.
    - Řádek 45 –(true) Definuje, že bude vyžadována autentizace k emailovému serveru, (false) definuje, že emaily se budou posílat bez přihlášení uživatele k webovému serveru.

12.3.4 Api stag-pdfa

Stag-pdfa poskytuje 2 endpointy. Jsou jimi POST `/api/validate/auto` a GET `/api/ok`.

Endpoint GET `/api/ok` slouží pro jednoduché otestování, zda služba stag-pdfa běží. Tento endpoint je možné otestovat pomocí následujícího curl příkazu:

```
curl HTTP://pdfa.k.utb.cz:8080/api/ok
```

Odpověď na curl příkaz je:

```
{"hello": "This is a JSON response"}
```

Pro otestování endpointu POST `/api/validate/auto` je nejprve nutno zkontrolovat, zda *config.yml* neobsahuje v parametru `exceptions` položky: `- 6.1.3` nebo `- 6.1.3-1`, pokud by se některá z uvedených položek v *config.yml* vyskytovala, je nutné ji odstranit a zpustit aplikaci znovu. Dále je nutno si připravit jeden soubor PDF plně validní s normou PDF/A a dva soubory PDF porušující některé z pravidel pro PDF/A. Pokud nemáme vlastní soubory, je ideální cestou použít testovací soubory, které byly

vytvořené při vývoji **veraPDF**, tyto soubory jsou dostupné v git repozitáři veraPDF-corpus[17]. V ukázkách níže, bude pracováno se soubory (jsou přiložené i na CD):

- `veraPDF_test_suite_6-1-3-t01-pass-a.pdf`,
- `veraPDF_test_suite_6-1-3-t01-fail-a.pdf`,
- `veraPDF_test_suite_6-1-3-t02-fail-a.pdf`

Jakmile máme soubory připraveny lokálně na PC, je možné otestovat funkcionalitu endpointu pomocí následujících curl příkazů:

a)

```
curl -F"file=@D:\dokumenty\veraPDF_test_suite_6-1-3-t01-pass-a.pdf" HTTP://pdfa.k.utb.cz:8080/api/validate/auto
```

Je vrácena odpověď `{"compliant":"true"}`. Soubor `veraPDF_test_suite_6-1-3-t01-pass-a.pdf` odpovídá všem požadavkům na normou PDF/A, proto je v tomto případě vrácena odpověď true.

b)

```
curl -F"file=@D:\dokumenty\veraPDF_test_suite_6-1-3-t01-fail-a.pdf" HTTP://pdfa.k.utb.cz:8080/api/validate/auto
```

Je vrácen odpověď `{"compliant":"false"}`. Soubor `veraPDF_test_suite_6-1-3-t01-fail-a.pdf` porušuje pravidlo 6. 1. 3 ze standardu PDF/A, proto v tomto případě je vrácen odpověď false.

c) V této ukázce je nutno nejprve upravit konfigurační soubor *config.yml* tak, že do parametru `exceptions` je přidána položka `- 6.1.3-1,` což značí, že do výjimky má být zahrnuta pouze podmnožina 1 z pravidla 6. 1. 3. Po změně *config.yml* je nutné znovu zpustit aplikaci **stag-pdf**. Nyní může být zadán následující ukázkový curl příkaz:

```
curl -F"file=@D:\dokumenty\veraPDF_test_suite_6-1-3-t01-fail-a.pdf" HTTP://pdfa.k.utb.cz:8080/api/validate/auto
```

Je vrácena odpověď: `{"compliant":"true"}`. Soubor `veraPDF_test_suite_6-1-3-t01-fail-a.pdf` sice porušuje podmnožinu 1 pravidla 6. 1. 3 ze standardu PDF/A, ale v konfiguračním souboru *config.yml* byla podmnožina 1 pravidla 6. 1. 3 vložena do výjimek, proto je odpověď true.

Zadáme curl příkaz s testovacím souborem `veraPDF_test_suite_6-1-3-t02-fail-a.pdf`.

[17] Dostupné z: https://github.com/veraPDF/veraPDF-corpus

```
curl  -F"file=@D:\dokumenty\veraPDF_test_suite_6-1-3-t02-fail-a.pdf
" HTTP://pdfa.k.utb.cz:8080/api/validate/auto
```

Je vrácena odpověď: `{"compliant":"false"}`. Soubor `veraPDF_test_suite_6-1-3-t02-fail-a.pdf` porušuje podmnožinu 2 pravidla 6. 1. 3 ze standardu PDF/A, ale v konfiguračním souboru *config.yml* byla do výjimky vložena pouze podmnožina 1 pravidla 6. 1. 3, proto je odpověď false.

d) V této ukázce je nutno nejprve upravit konfigurační soubor *config.yml* tak, že do parametru `exceptions` je přidána položka `- 6.1.3` poté zpustit aplikaci stag-pdfa znovu. Nyní může být zadán následující ukázkový curl příkaz:

```
curl  -F"file=@D:\dokumenty\veraPDF_test_suite_6-1-3-t01-fail-a.pdf
" HTTP://pdfa.k.utb.cz:8080/api/validate/auto
```

Je vrácena odpověď: `{"compliant":"true"}`. Soubor `veraPDF_test_suite_6-1-3-t01-fail-a.pdf` sice porušuje pravidlo 6. 1. 3 ze standardu PDF/A, ale v konfiguračním souboru *config.yml* bylo pravidlo vloženo do výjimek, proto je odpověď true.

```
Dále použijeme curl příkaz:
```

```
curl  -F"file=@D:\dokumenty\veraPDF_test_suite_6-1-3-t02-fail-a.pdf
" HTTP://pdfa.k.utb.cz:8080/api/validate/auto
```

Je vrácena odpověď: `{"compliant":"true"}`. Testovaný soubor `veraPDF_test_suite_6-1-3-t02-fail-a.pdf` sice porušuje podmnožinu 2 z pravidla 6. 1. 3, nicméně použitím `- 6.1.3` jsou zahrnuty do této výjimky všechny části, tzn. `- 6.1.3-1` i `-6.1.3-2.`

12.3.5 Třídy stag-pdfa

Veškeré funkce, které vytvořená Java aplikace stag-pdfa poskytuje, jsou definovány pomocí následujících tříd.

java.org.api.CustomHTTPClient

Popis: Tato třída slouží pro vytvoření HTTP klienta a veškerých operací s ním spojených: vytvoření POST požadavku, odeslání požadavku na hosta a zpracování přijaté JSON odpovědi.

java.org.api. CustomJsonDeserializer

Popis: Třída slouží k serializaci přijaté JSON odpovědi, konkrétně získává seznam porušených pravidel na standard PDF/A.

java.org.api.CustomResponse

Popis: Třída vytváří odpověď pro IS/STAG.

java.org.api.CustomRuleEvalutaion

Popis: Třída v sobě drží seznam porušených pravidel vrácených v odpovědi z veraPDF-rest a seznam výjimek pravidel načtených v *config.yml*. Dále obsahuje metodu, která aplikuje seznam výjimek na porušená pravidla.

java.org.api.Email

Popis: Třída se stará o vytvoření spojení s emailovým serverem a odeslání emailů.

java.org.api.InputStreamProcessor2

Popis: Třída zpracovává InputStream ze zaslaného požadavku. Přijatý InputStream si nejdříve uloží do dynamicky vytvářeného byte[] array, poté si počítá hash a následně dojde k uložení souboru z byte[] array. Dále obsahuje metodu, která z byte[] array vytvoří nový InputStream.

Řešení v této třídě se při zpracovávání velkých souborů (cca. 100MB a výš) příliš neosvědčilo, protože díky načítání InputStreamu do dynamicky vytvářeného pole docházelo k zabírání velkého množství paměti RAM. Z toho důvodu byl zvolen jiný přístup aplikovaný ve třídě InputStreamProcessor1.

java.org.api.InputStreamProcessor1

Popis: Třída zpracovává InputStream ze zaslaného požadavku. Čte postupně InputStream po malých pevných blocích bytů, byty ukládá do souboru a zároveň přidává do metody digest.update() pro výpočet hash. Po provedení výpočtu hash, přejmenuje uložený PDF soubor vypočteným hashem. Dále obsahuje metodu, která vytváří nový InputStream z uloženého souboru.

java.org.resources.ApiResource

Popis: Tato třída obsahuje metody, které obsluhují konkrétní endpointy našeho REST API.

java.org.api.SQLite

Popis: Třída využívající driver Java Database Connectivity (JDBC) pro komunikaci s databází. Třída obsahuje metody pro vytvoření databázové tabulky „stagpdfa_logs“, vkládání do tabulky a aktualizaci dat v tabulce.

java.org.DropwizardApplication

Popis: Jedná se o výchozí třídu obsahující metodu main. Tato třída byla automaticky vytvořena při generování nového projektu DropWizard.

java.org. DropwizardConfiguration

Popis: Jedná se o třídu, která byla také vytvořena automaticky při generování nového projektu DropWizard. Třída je použita k načítání dat z konfiguračního souboru *config.yml*.

12.3.6 Nasazení stag-pdfa v ostrém provozu

Stag-pdfa běží na linuxovém serveru serveru UTB, jako služba. Nastavení služby je ukázané v Kód 15. Služba stag-pdf se na linuxovém serveru spustí klasickým příkazem `sudo systemctl start stag-pdfa.service.` Po zadání tohoto příkazu se služba stag-pdfa podívá do složky target a spustí předkompilovanou Java aplikaci `dropwizard-1.0-SNAPSHOT.jar.`

```
1  [Unit]
2  Description=STAG PDF/A validation service (frontend for veraPDF-
   rest)
3  After=network.target
4
5  [Service]
6  Type=simple
7  User=ctenar
8  SyslogIdentifier=stag-pdfa
9  LimitNOFILE=65536
10 WorkingDirectory=/home/ctenar/stag-pdfa
11 ExecStart=/usr/bin/Java  -Xmx1500m  -XX:+ExitOnOutOfMemoryError  -
   jar target/dropwizard-1.0-SNAPSHOT.jar server config.yml
12 TimeoutStopSec=5s
13 Restart=on-failure
14 RestartSec=5s
15
16 [Install]
17 WantedBy=multi-user.target
```

Kód 15: Ukázka nastavení služby stag-pdfa

12.3.7 Zabezpečení

Aplikace **stag-pdfa** nepoužívá žádný API klíč, jak je zvykem u veřejných REST API. Důvody jsou tři: První důvod je ten, že služba poběží na privátní adrese UTB. Druhý důvod je, že API klíč nepoužívá ani **vereapdf-rest,** se kterým byla snaha, zůstat co nejvíce kompatibilní. Třetí důvodem je, že si IT správci nastaví omezení na firewall, aby ke službě stag-pdfa byl povolen přístup pouze z IP adresy, na které běží IS/STAG.

Denial of Service - útok

Knihovna **veraPDF-rest** poběží na školním serveru. V teoretickém případě můžou nastat 2 typy problémů:

1) Útočník si projde veřejně dostupný zdrojový kód **veraPDF-rest**, najde slabiny a vygeneruje speciální PDF/A, které pošle na službu **veraPDF-rest**.
2) Nová verze nějakého programu, případně i současné verze některých textových editorů může vygenerovat soubor PDF/A, který svým formátem může způsobit, že služba **veraPDF-rest** zhavaruje.

Snížení rizik: V případě, že by na **stag-pdfa** došlo k přeplnění paměti RAM k vyvolání výjimky OutOfMemoryError. Parametr „`-XX:+ExitOnOutOfMemoryError`“ na řádu 11 v Kód 15 zajistí, že dojde k ukončení aplikace.

Zhavarování stag-pdfa i veraPDF-rest bylo ošetřeno tím způsobem, že služby mají nastaveno, že v případě selhání se po 5 sekundách znovu automaticky sami spustí.

Dalším snížením rizik bylo automatické odesílání emailů o chybách uvnitř stag-pdfa.

12.3.8 Maximální možnosti stag-pdfa

Na závěr praktické části bylo provedeno několik testů, aby se zjistilo, jaké jsou aktuální limity stag-pdfa a veraPDF-rest. Testy byly prováděny tak, že na službu stag-pdfa byl vždy poslán 10x po sobě jeden PDF soubor a zaznamenával se celkový čas od přijetí požadavku na stag-pdfa do odeslání odpovědi, dále pak celkový čas na veraPDF-res který byl potřebný pro zpracování požadavku. Testování se provedlo 2x v různou denní dobu a následně se naměřená data zprůměrovala. Testovací PDF soubory se lišily velikostí a počtem stránek.

Tabulka 4: Testování délky kontroly PDF souboru

velikost PDF souboru [MB]	počet stránek PDF souboru	celkový čas [s]	čas na vera_PDF-rest [s]	čas na stag-pdfa [s]
100	77	3,1	2,4	0,8
160	113	4,3	3,2	1,1
200	154	6,2	4,7	1,5
230	106	6,0	4,2	1,7
230	246	8,9	7,3	1,5
350	76	8,1	5,7	2,4
420	56	9,7	6,5	3,3
550	37	27,4	22,7	4,7

Vývojáři IS/STAG nastavili timeout na validování PDF/A souboru maximálně 5 sekund. Z testování tedy vyplývá, že PDF soubory do velikosti 160MB budou v odevzdávacím formuláři IS/STAG na formát PDF/A bez problému zkontrolovány. Další omezení nastavili vývojáři IS/STAG na maximální velikost souboru, který je možné automaticky validovat, automatickou validaci povolili pro sobory s velikostí mezi 1-100MB. Na soubory větší jak 100MB nebude automatická validace aplikována a tyto soubory bude nutno na validátor poslat zpětně. Vše bylo otestováno na testovací verzi IS/STAG s názvem demo IS/STAG.

ZÁVĚR

Po zhodnocení vybraných komerčních i nekomerčních produktů umožňujících validaci PDF souborů na formát PDF/A, byla pro realizaci praktické části nakonec vybrána Java aplikace veraPDF-rest. Aplikace zprostředkovává přes REST API funkce knihovny veraPDF pro validaci souborů na formát PDF/A.

Po srovnání požadavků UTB a rozsáhlosti aplikace veraPDF-rest a knihovny veraPDF bylo rozhodnuto, že se požadavky UTB nebudou zanášet do zdrojového kódu veraPDF-rest a veraPDF, ale vyvine se pro ně samostatná aplikační mezivrstva s pracovním názvem stag-pdfa. Důvodem byla zejména menší časová náročnost. V tomto případě totiž stačilo z veraPDF-rest pouze nastudovat, jakým způsobem veraPDF-rest poskytuje funkce pomocí REST API, jaký formát požadavků očekává, a jaký formát odpovědí odesílá. Na strukturu veraPDF stačilo nahlížet pouze jako na fungující Java knihovnu, jejíž funkcionality jsou volány pomocí REST API veraPDF-rest. Vzhledem k tomu, že aplikace veraPDF-rest je postavena na frameworku DropWizard. Stag-pdfa byl vyvinut také pomocí tohoto frameworku. Důvodem pro výběr DropWizard byl hlavně ten, že byla snaha zachovat co největší podobnost ve struktuře stag-pdfa a veraPDF-rest pro případ, že by se v budoucnu integrovala funkcionalita stag-pdfa do veraPDF-rest a veraPDF s cílem snížení počtu běžících služeb.

V mezivrstvě stag-pdfa byly aplikovány aktuální požadavky IT správců z UTB. Jako první byl naprogramován hlavní požadavek na možnost definování seznamu výjimek pravidel na formát PDF/A, která budou z kontroly vyloučena. Při vývoji aplikace vyvstaly od IT správců další požadavky na funkcionalitu, kterými byly: možnost ukládání kontrolovaných souborů na server a jako název u nich používat jejich hash otisk, dále na produkčním linuxovém serveru vytvořit v databázi PostgreSQL databázovou tabulku pro logování provozu na stag-pdfa, v případě výskytu jakékoliv výjimky během provozu, je automaticky posílán emailem IT správcům. Díky dodatečným požadavkům od IT správců se ukázal vývoj vlastní mezivrstvy jako velmi praktický. Veškerou konfiguraci stag-pdfa je možné nastavit na jednom místě a to v konfiguračním souboru *config.yml*.

Při tvorbě aplikace byla snaha provádět vývoj v souladu s principy agilního vývoje, kdy byly IT správcem UTB (v roli product ownera) vždy stanoveny prioritní úkoly na následující týden, během týdne se konaly 1 až 2 hovory, kde se probíraly případné

problémy a upřesňovaly nejasnosti. Pro vedení úkolů byla používána aplikace Planner dostupná v rámci balíčku Office 360.

Další vývoj stag-pdfa by mohl být v optimalizaci zpracování požadavků obsahující velké soubory. Aktuální verze již jednou fází optimalizace prošla. Z toho důvodu jsou vytvořené 2 třídy InpustStreamProcessor zakončené čísli 1 a 2. Třída zakončená číslem 2 pracovala s dynamicky alokovaným polem datového typu byte, do kterého si ukládala přijatý soubor z proměnné InputStream, toto se ukázalo jako velmi nevhodné při větších velikostech souborů (160MB a víš) docházelo ke zbytečnému zaplňování paměti RAM a následnému pádu aplikace. Na problém bylo reagováno vytvořením třídy zakončené číslem 1, která zpracovává InputStream po fixních velikostech pole bytů a tím neplýtvá pamětí RAM. IS/STAG má aktuálně definovaný timeout na validaci PDF souboru 5 sekund. Ze závěrečného testování vyplynulo, že soubory do velikosti 160MB budou bez problému zkontrolovány. Mezivrstva stag-pdfa by mohla být do budoucna ještě dále lépe optimalizována, aby zpracování a přeposílání požadavků na veraPDF-rest spotřebovalo ještě méně času. Další vylepšení by bylo vytvořit webové grafické rozhraní pro stag-pdfa, kde by studenti měli možnost validovat práci před nahráváním do IS/STAG.

Práce s databází je řešena pomocí třídy Java.sql.DriverManager, která je základní službou pro správu sady ovladačů JDBC. Při dokončování této práce bylo zjištěno, že framework DropWizard nabízí modul dropwizard-hibernate, který poskytuje řízený přístup k Hybernate, což je objektově relační mapovací framework, který by byl pro práci s databází mnohem ideálnější. Použití modulu dropwizard-hibernate ve stag-pdfa by bylo jedním z dalších možných vylepšení.

Po optimalizování seznamu výjimek pravidel by v budoucnu mohlo být další cestou implementovat funkcionalitu vytváření odpovědi pro IS/STAG přímo ve veraPDF-rest a funkcionalitu definování seznamu výjimek pravidel na formát PDF/A přímo do veraPDF **.**

SEZNAM POUŽITÉ LITERATURY

[1] ČESKO, 2004. Zákon č. 499/2004 Sb., o archivnictví a spisové službě a změně některých zákonu. In: Sbírka zákonů České republiky. Dostupné také z: HTTPs://www.zakonyprolidi.cz/cs/2004-499

[2] GRULICH, Petr, 2007. VYSOKOŠKOLSKÉ KVALIFIKAČNÍ PRÁCE A ZÁSADY NAKLÁDÁNÍ S NIMI Z POHLEDU ARCHIVÁŘŮ. Evskp [online]. [cit. 2021-4-11]. Dostupné z: HTTP://www.evskp.cz/Dokumentyver/archivace-071026141239.pdf

[3] ČESKO, 1998. Zákon č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů. In: Sbírka zákonů České republiky. 39/1998. Dostupné také z: HTTPs://www.zakonyprolidi.cz/cs/1998-111

[4] Praktiky zveřejňování vysokoškolských prací v ČR, © 2013. Wikisofia [online]. [cit. 2021-03-18]. Dostupné z: HTTPs://wikisofia.cz/wiki/Praktiky_zve%C5%99ej%C5%88ov%C3%A1n%C3%AD_vysoko%C5%A1kolsk%C3%BDch_prac%C3%AD_v_%C4%8CR

[5] ČESKO, 2012. Vyhláška č. 259/2012 Sb., Vyhláška o podrobnostech výkonu spisové služby. In: Sbírka zákonů České republiky. 88/2012. Dostupné také z: HTTPs://www.zakonyprolidi.cz/cs/2012-259#p23-2

[6] PICHL, Marek et al., 2015. Metodika dlouhodobého ukládání a archivace digitálních dokumentů. In: MUNI [online]. [cit. 2021-04-06]. Dostupné z: HTTPs://is.muni.cz/publication/1322181/

[7] Funkce a možnosti transakčního protokolu. Munis [online]. [cit. 2021-04-06]. Dostupné z: HTTPs://www.munis.cz/art/969

[8] PICHL, Marek et al., 2015. METODIKA DLOUHODOBÉHO UKLÁDÁNÍ A ARCHIVACE DIGITÁLNÍCH DOKUMENTŮ [online]. Brno [cit. 2021-5-1]. ISBN 978-80-210-8094-2. Dostupné z: HTTPs://is.muni.cz/publication/1322181/Metodika_dlouhodobeho_ukladani_a_archivace_digitalnich_dokumentu.pdf

[9] 2017. Národní standard pro elektronické systémy spisové služby: Informace o NSESSS, schéma XML pro předávání dokumentů a jejich metadat do archivu a příklad datového balíčku SIP. Ministerstvo vnitra České republiky [online]. Ministerstvo vnitra

České republiky [cit. 2021-4-20]. Dostupné z: HTTPs://www.mvcr.cz/clanek/narodni-standard-pro-elektronicke-systemy-spisove-sluzby.aspx

[10] SUCHANEK, Fabian M., 2021. Best File Formats for Archiving. Suchanek.name [online]. [cit. 2021-5-9]. Dostupné z: HTTPs://suchanek.name/texts/archiving/index.html#open

[11] Office Open XML, 2021. Wikipedia [online]. [cit. 2021-5-10]. Dostupné z: https://en.wikipedia.org/wiki/Office_Open_XML#Licensing

[12] ARTHUR, Jamie, 2021. How to Extract tar xz File in Linux. LinOxide [online]. [cit. 2021-5-9]. Dostupné z: HTTPs://linoxide.com/how-to-extract-tar-xz-file-linux/

[13] ISO 19005-1, 2005. Csygen [online]. [cit. 2021-4-1]. Dostupné z: HTTP://www.csygen.com/pdf/iso_19005.PDF

[14] PDF Reference third edition, 2001. Adobe [online]. [cit. 2021-4-5]. Dostupné z: HTTPs://www.adobe.com/content/dam/acom/en/devnet/pdf/pdfs/pdf_reference_archives/PDFReference.pdf

[15] Callas software GmbH, 2011. PDF/A – A Look at the Technical Side. web.archive.org [online]. [cit. 2021-5-9]. Dostupné z: HTTPs://web.archive.org/web/20120618161616/HTTP://www.pdfa.org/2011/08/pdfa-%E2%80%93-a-look-at-the-technical-side/

[16] KRAUS, Vladimír, 1999. Struktura formátu PDF. Katedra informatiky a výpočetní techniky [online]. [cit. 2021-5-9]. Dostupné z: HTTPs://www.kiv.zcu.cz/~herout/html_sbo/pdf/3.htm

[17] 2008. Blog.didierstevens [online]. [cit. 2021-5-9]. Dostupné z: HTTPs://blog.didierstevens.com/2008/04/09/quickpost-about-the-physical-and-logical-structure-of-pdf-files/

[18] Introduction to PDF, 10n. l. Web.archive [online]. [cit. 2021-3-10]. Dostupné z: HTTPs://web.archive.org/web/20141010035745/HTTP://gnupdf.org/Introduction_to_PDF

[19] PDF/A-2, PDF for Long-term Preservation, Use of ISO 32000-1 [online], 2011. [cit. 2021-5-9]. Dostupné z: HTTPs://www.loc.gov/preservation/digital/formats/fdd/fdd000319.shtml

[20] Document management - Portable document format - Part 1: PDF 1.7, 2008. Adobe [online]. [cit. 2021-4-2]. Dostupné z: HTTPs://www.adobe.com/content/dam/acom/en/devnet/pdf/pdfs/PDF32000_2008.pdf

[21] PDF/A-3, PDF for Long-term Preservation, Use of ISO 32000-1, With Embedded Files, 2012. Library Of Congress [online]. [cit. 2021-5-9]. Dostupné z: HTTPs://www.loc.gov/preservation/digital/formats/fdd/fdd000360.shtml

[22] PDF Association Arranges Its First Seminar on PDF/A to Include Standards 1 to 3 [online], 2012. [cit. 2021-4-29]. Dostupné z: HTTPs://web.archive.org/web/20120915003604/HTTP://www.pdfa.org/2012/03/pdf-association-arranges-its-first-seminar-on-pdfa-to-include-standards-1-to-3/

[23] The new PDF 2.0 and subset standards, 2020. PDF Association [online]. [cit. 2021-4-5]. Dostupné z: HTTPs://www.pdfa.org/the-new-pdf-2-0-and-subset-standards/

[24] PDF/A-4, PDF for Long-term Preservation, Use of ISO 32000-2, 2020. Library Of Congress [online]. [cit. 2021-3-23]. Dostupné z: HTTPs://www.loc.gov/preservation/digital/formats/fdd/fdd000532.shtml

[25] 3Heights™ PDF Validator API: Version 6.15.0, 2021. Pdf-tools [online]. [cit. 2021-4-3]. Dostupné z: HTTPs://www.pdf-tools.com/public/downloads/manuals/PdfValidatorAPI.pdf

[26] Java Preflight PDF/X PDF/A, 2020. Qoppa Software [online]. [cit. 2021-4-19]. Dostupné z: HTTPs://www.qoppa.com/pdfpreflight/

[27] PDF Automation Server – REST API Module, 2020. Qoppa Software [online]. [cit. 2021-4-19]. Dostupné z: HTTPs://www.qoppa.com/pdfautomation/restapi/

[28] PDF Automation Server – PDF Processing Server, 2020. Qoppa Software [online]. [cit. 2021-4-19]. Dostupné z: HTTPs://www.qoppa.com/pdfautomation/

[29] NEW Workflow Module, 2020. Qoppa Software [online]. [cit. 2021-4-19]. Dostupné z: HTTPs://www.qoppa.com/pdfautomation/workflow/

[30] Document conversion library for Cross-Platform (Core), 2021. Pdftron [online]. [cit. 2021-4-25]. Dostupné z: HTTPs://www.pdftron.com/documentation/core/guides/features/conversion/

[31] PDF/A Library, 2021. Pdftron [online]. [cit. 2021-4-25]. Dostupné z: HTTPs://www.pdftron.com/pdf-sdk/pdfa-converter-library/

[32] Validate PDF/A documents in Java, 2021. Pdftron [online]. [cit. 2021-4-25]. Dostupné z: HTTPs://www.pdftron.com/documentation/Java/guides/features/pdfa/validate/

[33] Validace PDF/A, 2018. Digi.nacr [online]. [cit. 2021-5-10]. Dostupné z: HTTPs://digi.nacr.cz/validatorPDF/

[34] 3-HEIGHTS™ PDF VALIDATOR ONLINE TOOL, 2021. Pdf-online [online]. [cit. 2021-4-29]. Dostupné z: HTTPs://www.pdf-online.com/osa/validate.aspx

[35] Verify PDF/A Compliance, 2021. Freepdfonline [online]. [cit. 2021-4-29]. Dostupné z: HTTPs://freepdfonline.com/verifypdfa/

[36] PDF to PDF/A Converter, 2021. Pdftron [online]. [cit. 2021-4-25]. Dostupné z: HTTPs://www.pdftron.com/pdf-tools/pdfa-converter/

[37] Apache PDFBox® - A Java PDF Library, 2021. PDFBox [online]. [cit. 2021-5-10]. Dostupné z: HTTPs://pdfbox.apache.org/index.html

[38] PDF/A Validation, 2021. PDFBox [online]. [cit. 2021-5-10]. Dostupné z: HTTPs://pdfbox.apache.org/1.8/cookbook/pdfavalidation.html

[39] Industry Supported PDF/A Validation, 2015. Verapdf [online]. [cit. 2021-5-10]. Dostupné z: HTTPs://verapdf.org/home/

[40] VeraPDF. Open Preservation Foundation [online]. 2020, May, 2020 [cit. 2020-11-16]. Dostupné z: HTTPs://openpreservation.org/products/verapdf/

[41] VeraPDF. Open Preservation Foundation [online]. 2020, May, 2020 [cit. 2020-11-16]. Dostupné z: HTTPs://openpreservation.org/products/verapdf/HTTPs://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

[42] NOVINKY V IS/STAG FORMOU NEWSLETTERU. Is-stag [online]. [cit. 2021-5-10]. Dostupné z: HTTPs://is-stag.zcu.cz/zakaznici/newsletter

[43] Dropwizard Core, 2020. Dropwizard [online]. [cit. 2021-5-10]. Dostupné z: HTTPs://www.dropwizard.io/en/latest/manual/core.html

[44] CHAIR, Professor Richard N. Taylor, Professor Mark S. ACKERMAN a Professor David S., 2000. Architectural Styles and the Design of Network-based Software

Architectures. Ics.uci [online]. [cit. 2021-5-10]. Dostupné z: HTTPs://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

[45] Eclipse Jersey is a REST framework that provides a JAX-RS (JSR-370) implementation and more. Eclipse-ee4j [online]. [cit. 2021-5-10]. Dostupné z: HTTPs://eclipse-ee4j.github.io/jersey/

[46] Jackson Project Home @github, 2020. Github [online]. [cit. 2021-5-10]. Dostupné z: HTTPs://github.com/FasterXML/jackson

[47] Standard ECMA-404: The JSON Data Interchange Syntax. Ecma International [online]. 2017, December 2017 [cit. 2020-11-16]. Dostupné z: HTTPs://ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf

[48] Less secure apps & your Google Account, 2021. Support.google [online]. [cit. 2021-5-10]. Dostupné z: HTTPs://support.google.com/accounts/answer/6010255#zippy=%2Ckdy%C5%BE-je-v-%C3%BA%C4%8Dtu-zapnut%C3%BD-p%C5%99%C3%ADstup-pro-m%C3%A9n%C4%9B-zabezpe%C4%8Den%C3%A9-aplikace

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

PDF/A	Portable Document Format for the Long-term Archiving
PDF	Portable Document Format
API	Application Programming Interface
eSSL	Elektronický systém spisové služby
FLAC	Free Lossless Audio Codec
GIF	Graphics Interchange Format
ISO	International Organization for Standardization
JDBC	Java Database Connectivity
JSON	JavaScript Object Notation
MIDI	Musical Instrument Digital Interface
NSESSS	Národního standardu pro elektronické systémy spisové služby
PDF/UA	PDF Universal Accessibility
PDF/X	PDF Exchange
PNG	Portable Network Graphics
REST API	Representational State Transfer API
SDK	Software development Kit
SIP	Submission Information Balíčku
SVG	Scalable Vector Graphics
TIFF	Tag Image File Format
WMF	Windows Metafile
XML	Extensible Markup Language
XPS	XML Paper Specification
SIP	Submission Information Package
UTB	Univerzita Tomáše Bati ve Zlíně
VŠKP	vysokoškolských závěrečných prací

SEZNAM OBRÁZKŮ

Obrázek 1: PDF/A-1a a PDF/A-1b 27

Obrázek 2: komponenty PDF [14] 28

Obrázek 3: Struktura PDF souboru [13] 31

Obrázek 4: Příklad struktury PDF dokumentu [13] 35

Obrázek 5: Příklad grafického rozhraní drag&drop [29] 46

Obrázek 6: Návrh řešení možnost 1 58

Obrázek 7: Návrh řešení - možnost 2 58

Obrázek 8: Formulář pro odevzdání souborů s elektronickou podobou VŠKP 62

Obrázek 9: Načítání souboru na formuláři Obrázek 8 62

Obrázek 10: Soubor prošel validací PDF/A 63

Obrázek 11: Soubor neprošel validací PDF/A 63

Obrázek 12: Chyba **stag-pdfa** 63

Obrázek 13: Struktura JSON objektu [46] 69

Obrázek 14: Připojení k databázovému souboru sample.db v IntelliJ IDEA Ultimate 72

SEZNAM TABULEK

Tabulka 1: přehled placených SW nástrojů největší shody s požadavky UTB 53

Tabulka 2: přehled neplacených SW nástrojů srovnání s požadavky UTB 53

Tabulka 3: Návrh logovací tabulky stagpdfa_logs..... 71

Tabulka 4: Testování délky kontroly PDF souboru 80

SEZNAM PŘÍLOH

Seznam příloh na CD:

Složka test_files obsahuje testovací PDF soubory, zmíněné v praktické části práce.

Složka stag-pdfa obsahuje komletní git repozitář s aplikací stag-pdfa.