Dokumentácia systému

Nikola Horníková, Filip Kerák, Iveta Balintová, Erik Bíly

TIS 2019

Časť 1: Katalóg požiadaviek

1. Úvod

1.1 Účel katalógu požiadaviek

Tento dokument predstavuje katalóg požiadaviek vytvorený pri tvorbe informačného systému pre automatické získavanie ohlasov na publikácie zo systému CREPČ2, ich konvertovanie a zapísanie do formátu MARC21/ISO2709. Dokument je napísaný tvorcami systému na základe analýzy požiadaviek, ktoré boli spísané po stretnutí so zadávateľom. Dokument je určený všetkým osobám zapájajúcim sa do tvorby, správy a prevádzky systému. Dokument zároveň slúži ako záväzná dohoda medzi zadávateľom a tvorcami systému popisujúca cieľovú funkcionalitu. Katalóg je písaný zrozumiteľným jazykom pre: zadávateľa, vývojový tím a pre všetky ostatné zapojené osoby.

1.2 Rozsah informačného systému

Primárnym cieľom systému je automatické kontrolovanie novoevidovaných ohlasov na publikácie v systéme CREPČ2, ich následná konverzia a zapísanie do knižničného systému Univerzity Komenského. Konkrétne do formátu MARC21/ISO2709, pričom na uchovávanie záznamov o citáciách slúži pole 591. Úlohou systému nie je kontrolovanie ani synchronizácia samotných publikácií. Systém nekontroluje a nezodpovedá za správnosť údajov získaných zo systému CREPČ2, iba v prípade nekompletnosti povinných údajov pre vygenerovanie poľa 591 bude používateľ upozornený výpisom v konzole. Systém nijako neupravuje štruktúry ani formáty ukladania používané univerzitným knižničným systémom.

1.3 Slovník pojmov

CREPČ2 - Centrálny register evidencie publikačnej činnosti SR (http://app.crepc.sk)

Záznam o publikácií - súhrn bibliografických údajov/metadát jednoznačne identifikujúci publikáciu

Ohlas - informácia o publikácií, ktorá cituje, respektíve recenzuje inú publikáciu (ak autor cituje sám seba, ide o autocitáciu, ktorá sa do KIS neeviduje)

XML - rozšíriteľný značkovací jazyk, umožňuje jednoduché vytváranie konkrétnych značkovacích jazykov na rôzne účely a široké spektrum rôznych typov údajov

MARC21 - bibliografický výmenný formát
(existuje MARC21-XML, ale pre load dát do KIS potrebujeme MARC21-ISO2709)

Knižnično-informačný systém- Virtua je komplexný knižničný systém používaný univerzitou.

1.4 Referencie

Dokumentácia k jazyku Python
https://www.python.org/doc/

Dokumentácia k štruktúre záznamov CREPČ2
http://cms.crepc.sk/crep%C4%8D-2-%C5%A1kolenia-a-pokyny.aspx

Dokumentácia k MARC21
https://www.loc.gov/marc/bibliographic/

Ohlasy v CREPČ2
http://cms.crepc.sk/Data/Sites/1/crepc2/dokumentaciaxml/oai-crepc2_v201805.pdf

Extensible markup language - XML
https://www.w3.org/XML/

Databáza publikačnej činnosti UK
https://alis.uniba.sk:8444/search/query?theme=EPC

1.5 Prehľad ďalších kapitol

V druhej kapitole sú podrobnejšie prebraté dôvody vzniku systému a jeho funkcionalita. Druhá kapitola tiež popisuje používateľov systému. Posledná časť druhej kapitoly hovorí o obmedzeniach a závislostiach systému. V kapitole tri sú podrobne rozpísané všetky primárne aj sekundárne požiadavky na systém. Posledná kapitola je zbierkou príloh a odkazov týkajúcich sa systému.

2. Popis systému

2.1 Perspektíva systému

Systém má slúžiť na synchronizáciu ohlasov na publikácie medzi univerzitným knižničným systémom(Virtua) a Centrálnym registrom publikačnej činnosti 2 (CREPČ2). V súčasnosti bežia oba systémy paralelne avšak nedisponujú automatickým synchronizovaním ohlasov. Ohlasy na publikácie musia byť zadávané ručne a pri množstve publikácií Univerzity Komenského to nie je možné robiť do dvoch systémov. Tento problém by mal vyriešiť vytváraný informačný systém, ktorý bude v pravidelných intervaloch kontrolovať pridané ohlasy v systéme CREPČ2 a zapisovať ich do knižničného systému Univerzity Komenského.

2.2 Funkcie systému

Systém sa bude pripájať na centrálny register publikačnej činnosti buď v pravidelných intervaloch alebo po ručnom spustení oprávneného používateľa, odkiaľ prevezme nové ohlasy na publikácie. Ku každému ohlasu načíta všetky dáta potrebné pre konverziu do MARC21/ISO2709. Systém sa stará o získanie dát a ich zapísanie v správnom formáte do súboru. Výstupom je aj log file o priebehu behu systému.

2.3 Charakteristika používateľov

Jedná sa o jednopoužívateľský systém, ktorý bude spravovaný a používaný administrátorom knižničného systému. Keďže výsledný systém bude dodaný ako konzolová aplikácia komunikujúca s knižničným systémom, tak u každého používateľa sa predpokladá, že je oboznámený s prácou v konzole a prístupom do knižničného systému. Použitie systému nepredpokladá veľké zásahy od používateľa a malo by byť možné nastaviť jeho automatické spúšťanie napr. pomocou task scheduler.

2.4 Všeobecné obmedzenia

- Výsledný formát ukladania dát je určený a dáta budú uložené v MARC21/ISO2709 konkrétne v poli 591. (vychádza zo zavedenej praxe)
- Dáta prevzaté zo systému CREPČ2 sú vo formáte XML s presne špecifikovanou štruktúrou
- Každá publikácia/ohlas má presne stanovené polia, ktoré musí obsahovať(vychádza z predpisov)

2.5 Prepojenia a závislosti

- Systém je napojený na CREPČ2 a pre získanie dát je na ňom plne závislý.
- Systém je závislý aj na knižničnom systéme univerzity keďže na vkladanie využíva prostriedky tohto systému.

3. Špecifické požiadavky

Požiadavky sú delené na niekoľko častí. Primárne požiadavky popisujú najdôležitejšie želané funkcie systému. Sekundárne požiadavky popisujú doplnkovú funkcionalitu, ktorá bude implementovaná podľa výsledku implementácie primárnej časti. Každá požiadavka má jasný identifikátor pozostávajúci z 2-znakoveho označenia oblasti, znaku 1AP a znaku, ktorý tvorí poradové číslo.

3.1 Primárne požiadavky

PP-1AP-1:Získanie ohlasov zo systému CREPČ2

- Systém získa novo vzniknuté ohlasy na publikácie autorov UK zo systému CREPČ2. Získa ohlasy vzniknuté alebo modifikované od dátumu zadaného podľa PP-1AP-2 po konečný dátum podľa PP- 1AP-3.

PP-1AP-2: Zadanie počiatočného dátumu

- Aplikácia umožňuje zadať dátum od ktorého sa majú získavať ohlasy. Dátum sa zadá ako parameter pri spustení. Zadanie počiatočného dátumu je povinné.

PP-1AP-3: Zadanie konečného dátumu

- Aplikácia umožňuje zadať dátum po ktorý sa majú získavať ohlasy. Dátum sa zadá ako parameter pri spustení. Zadanie konečného dátumu nie je povinné.

PP-1AP-4: Zadanie cesty pre uloženie súboru

- Aplikácia umožňuje zvoliť cestu kam má byť uložený súbor so spracovanými údajmi ako parameter. Ak nie je zadaný uloží sa v priečinku, kde bola aplikácia spustená.

PP-1AP-5: Skonvertovanie ohlasov do formátu ISO2709

- Systém skonvertuje ohlasy z PP-1AP-1 do formátu ISO2709.

PP-1AP-6: Formát výstupu

- Výstupom programu je súbor vo formáte ISO2709 obsahujúci skonvertované ohlasy, ktorého názov bude tvaru data_na_zapis_D_T kde D reprezentuje presný dátum a T čas ku ktorému bol vytvorený.

PP-1AP-7: Vytvorenie logu chýb

- Po skončení sa vytvorí log súbor.
- Na každom riadku logu bude jeden záznam v tomto tvare: na začiatku bude id ohlasu a label tvaru [info] alebo [error] nasledované popisnou informáciou. V prípade úspešného zapísania tam bude vložený reťazec inak tam bude správa opisujúca chybu.

3.2 Sekundárne požiadavky

SP-1AP-1: Aktualizácia už vložených ohlasov

- Stáva sa, že niekedy je nutné opraviť/upraviť niektoré informácie v ohlase. Tieto opravy sa robia v systéme CREPČ2. Tieto zmeny je nutné synchronizovať.
- Aplikácia získa zoznam upravených ohlasov zo systému CREPČ2.
- Táto funkcionalita sa spúšťa zadaním parametra.

3.3 Non-functional požiadavky

NF-A1: Forma dodania

- Systém bude dodaný ako konzolová aplikácia, ktorá dostane parametre na vstupe.

NF-A2: Testovanie

- Dodaný systém bude testovaný na testovacej databáze knižnice.

NF-A3: Implementačný jazyk

- Pre možnosť budúcej úpravy zadávateľom bude systém vytvorený v jazyku Python

NF-A4: Čistota kódu

- Pri písaní kódu budú dodržiavané zásady čistého kódu aby bolo možné systém neskôr jednoducho upravovať.

NF-A5: Dokumentácia popisujúca vstupné parametre

- K systému bude dodaná dokumentácia, vysvetľujúca jednotlivé parametre zadávané pri spustení programu a ich možné hodnoty.

NF-A6: Vytvorenie mapovania z XML do MARC21

- Aplikácia konvertuje Nodes v XML formáte na polia vo formáte MARC21, toto mapovanie treba vyrobiť.

3.4 Požiadavky na rozhranie

PR-AP-1: Parametrizovateľnosť

- Systém bude nastaviteľný, zadaním parametrov pri spustení.
- Parametre majú jasne určený formát

PR-AP-2: Zaznamenávanie chýb

- Výsledkom každého kroku je aj Log chýb vzniknutých pri vykonávaní

PR-AP-3: Vypisovanie chýb

- Systém informuje užívateľa o vzniknutej chybe aj zjednodušeným výpisom chyby do konzoly.
- Podrobný popis sa nachádza v Logu bod PP-1AP-7

PR-AP-4: Vypisovanie priebehu behu aplikácie

- Systém priebežne informuje o pokroku operácií a to výpisom na konzolu.
- Na konzolu sa priebežne vypisujú už vložené alebo chybné ohlasy.

4. Prílohy

V tejto časti máme odkazy na súbory, ktoré nám poskytla Jana Ilavská, teda zadávateľ projektu.

Knižnično-informačný systém Virtua
https://github.com/TIS2019-FMFI/konverzia-ohlasov/blob/documentation/subory_zadavatel/KIS-record-load_guide.pdf

Záznam v KIS
https://github.com/TIS2019-FMFI/konverzia-ohlasov/blob/documentation/subory_zadavatel/KIS-zaznam_ohlas.png
Štruktúra citácie v KIS
https://github.com/TIS2019-FMFI/konverzia-ohlasov/blob/documentation/subory_zadavatel/KIS-E13-ohlas591.doc

Náčrt mapovania ohlasov
https://github.com/TIS2019-FMFI/konverzia-ohlasov/blob/documentation/subory_zadavatel/nacrt-mapovania-ohlasy.xlsx

Časť 2: Návrh

1. Úvod

1.1. Účel dokumentu

Tento dokument predstavuje podrobný popis návrhu pre systém na konverziu ohlasov. Pomocou diagramov a detailných popisov v tomto dokumente je vysvetlené, akým spôsobom bude systém vyvinutý a ako bude následne fungovať tak, aby splnil všetky požiadavky uvedené v katalógu požiadaviek.

1.2. Zameranie a rozsah návrhu

Predpokladáme, že čitateľ tohto dokumentu má dôkladne prečítaný katalóg požiadaviek a teda má ucelenú predstavu ako má softvér fungovať. V časti návrh implementácie sú popísané všetky triedy z triedneho diagramu. Veľkú časť tohto dokumentu tvoria UML diagramy, v ktorých je dôkladne predstavené, na aké moduly a triedy bude systém rozdelený. V časti UML diagramy sú predstavené všetky diagramy na základe, ktorých je dobre vysvetlený priebeh celého systému.

1.3. Slovník pojmov

CREPČ2 - Centrálny register evidencie publikačnej činnosti SR (http://app.crepc.sk)

Záznam o publikácií - súhrn bibliografických údajov/metadát jednoznačne identifikujúci publikáciu

Ohlas - informácia o publikácií, ktorá cituje, respektíve recenzuje inú publikáciu (ak autor cituje sám seba, ide o autocitáciu, ktorá sa do KIS neeviduje)

XML - rozšíriteľný značkovací jazyk, umožňuje jednoduché vytváranie konkrétnych značkovacích jazykov na rôzne účely a široké spektrum rôznych typov údajov

MARC21 - bibliografický výmenný formát

2. Obsluha aplikácie

2.1. Konzola

```
Usage: ./startup.sh -f DDMMYYYY [-h | --help] [-t <date> | --to=<date> ]
                    [--out-file=<file>] [--out-path=<path>]
                    [--log-file=<file>] [--log-path=<path>]

Povinny argument:
  -f <date>, --from=<date> Datum od ktoreho ziskava ohlasy, date musi byt vo formate
DDMMYYYY
Volitelne argumenty:
  -t <date>, --to=<date>   Datum po ktory ziskava ohlasy, date musi byt vo formate DDMMYYYY,
                    default je dnesny datum
  --out-file=<file>         Nazov suboru do ktoreho sa uklada vystup,
                    default nazov je 'output'. V pripade ze subor neexistuje, vytvori sa novy
  --out-path=<path>      Cesta k vystupnemu suboru, default je priecinok odkial sa aplikacia
spusta
  --log-file=<file>         Nazov suboru do ktoreho sa vypisuje priebeh aplikacie,
                    default nazov je 'log'. V pripade ze subor neexistuje, vytvori sa novy
  --log-path=<path>      Cesta k log suboru, default je priecinok odkial sa aplikacia spusta
  -d, --debug               Spustenie debug modu aby sa vypisovalo aj info a warningy
```

Príklad použitia : ./startup.sh -d -f 01012020 -t 08012020
Tento príkaz spustí aplikáciu v debugovacom móde a zbiera ohlasy od prvého januára 2020 po ôsmy január 2020 (vrátane)

3. Používané technológie

3.1. Python

V tomto interpretovanom, interaktívnom programovacom jazyku bude napísaná aplikácia z ktorého sa navrhnutý systém skladá.

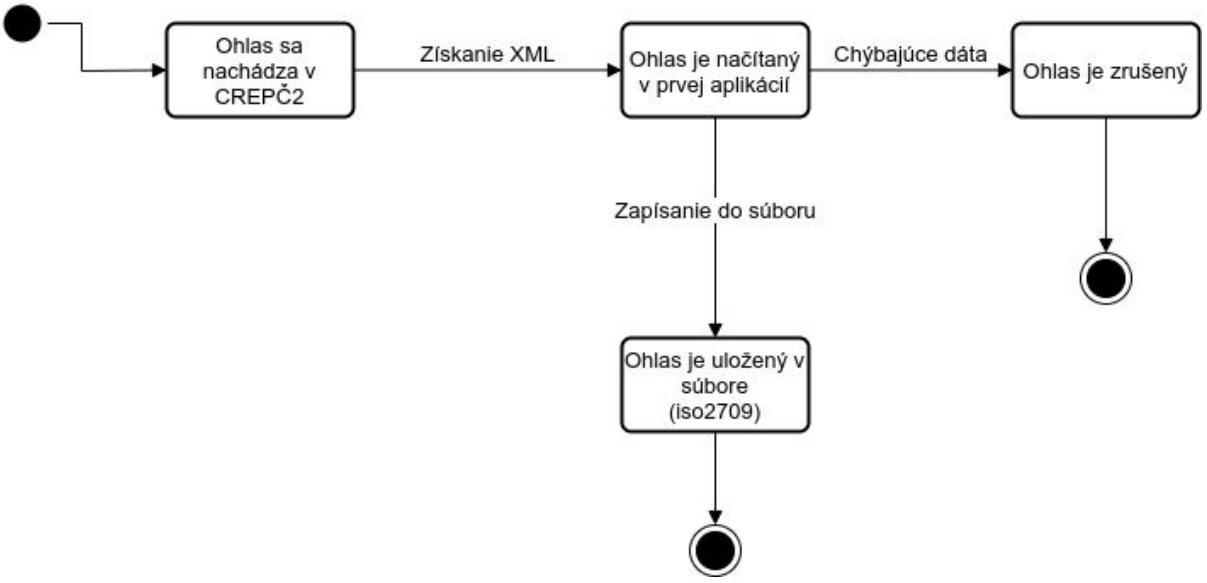
3.2. Terminál

Aplikácia bude ovládana z konzoly, podľa návodu v kapitole 2 časti 2.1.

4. UML Diagramy

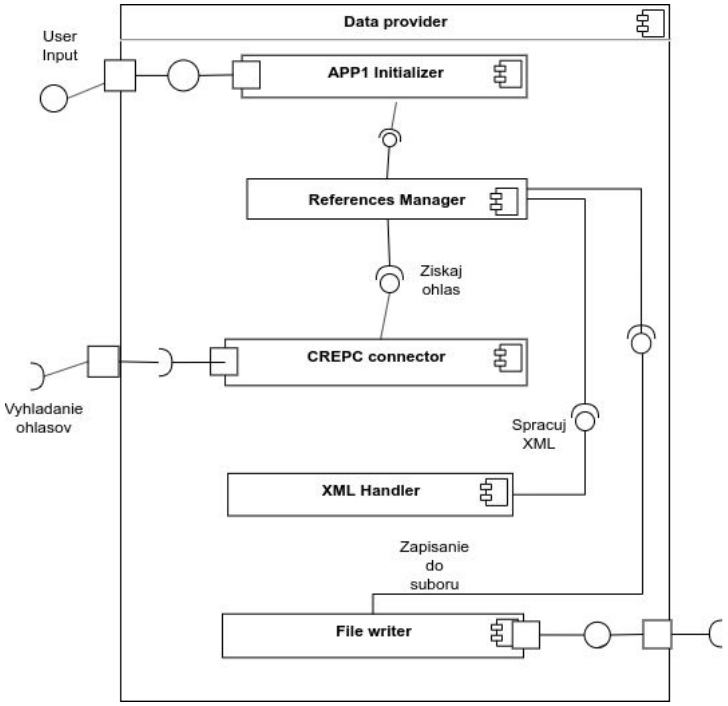
4.1. UML State diagram

Tento diagram znázorňuje všetky možné stavy entity “Ohlas”. Po načítaní Ohlasu z XML súboru existujú dve možnosti: buď v Ohlase chýbajú nejaké dáta a Ohlas sa zruší, alebo je všetko v poriadku a Ohlas sa uloží do výsledného súboru.



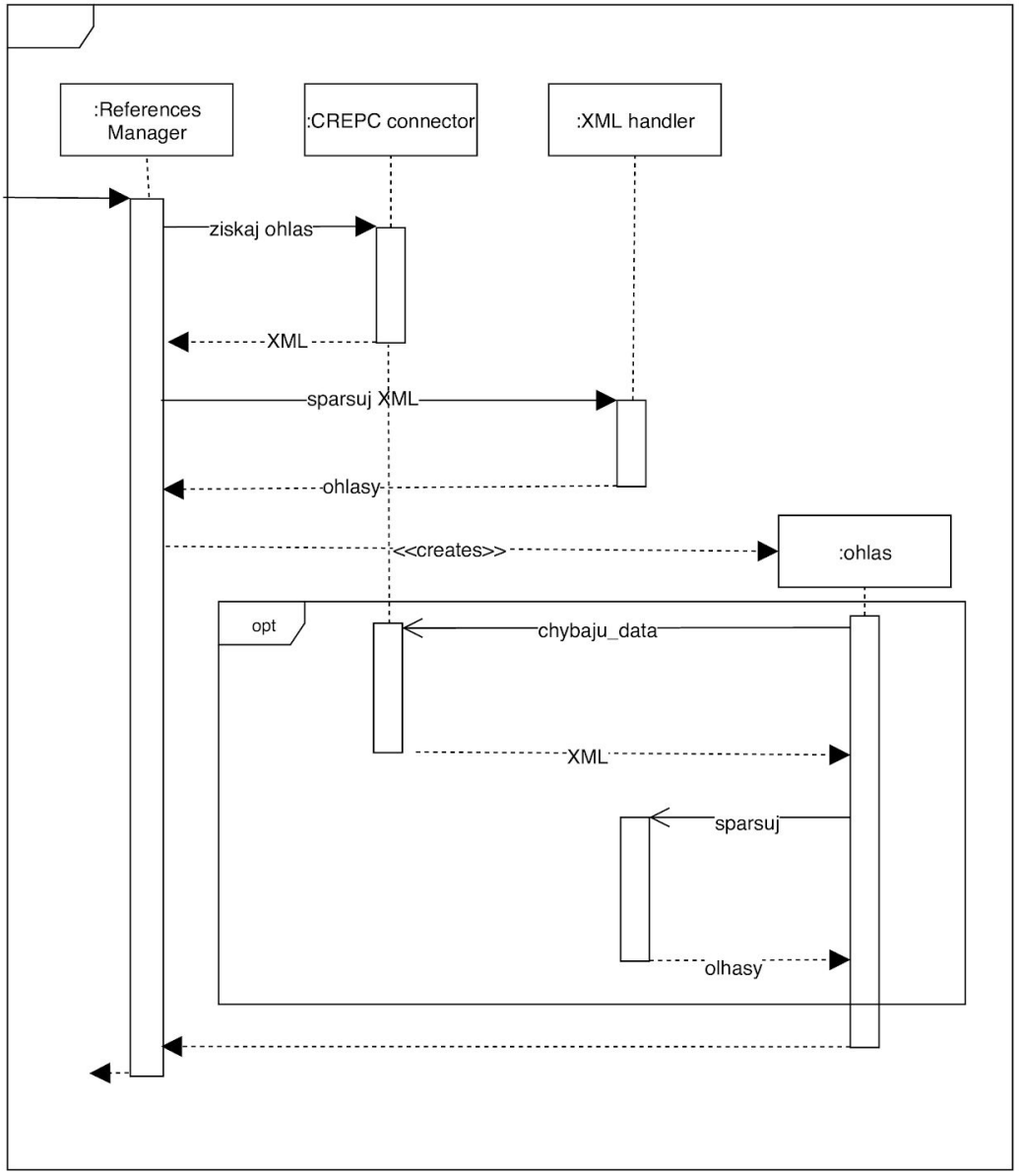
4.2. UML Component diagram

Aplikáciu nemá zmysel rozdeľovať na viacero celkov (komponentov). Skladá sa iba z niekoľkých tried komunikujúcich medzi sebou. Rozhranie na komunikáciu s týmto komponentom je konzola, do ktorej sa pri spustení aplikácie zadávajú parametre.



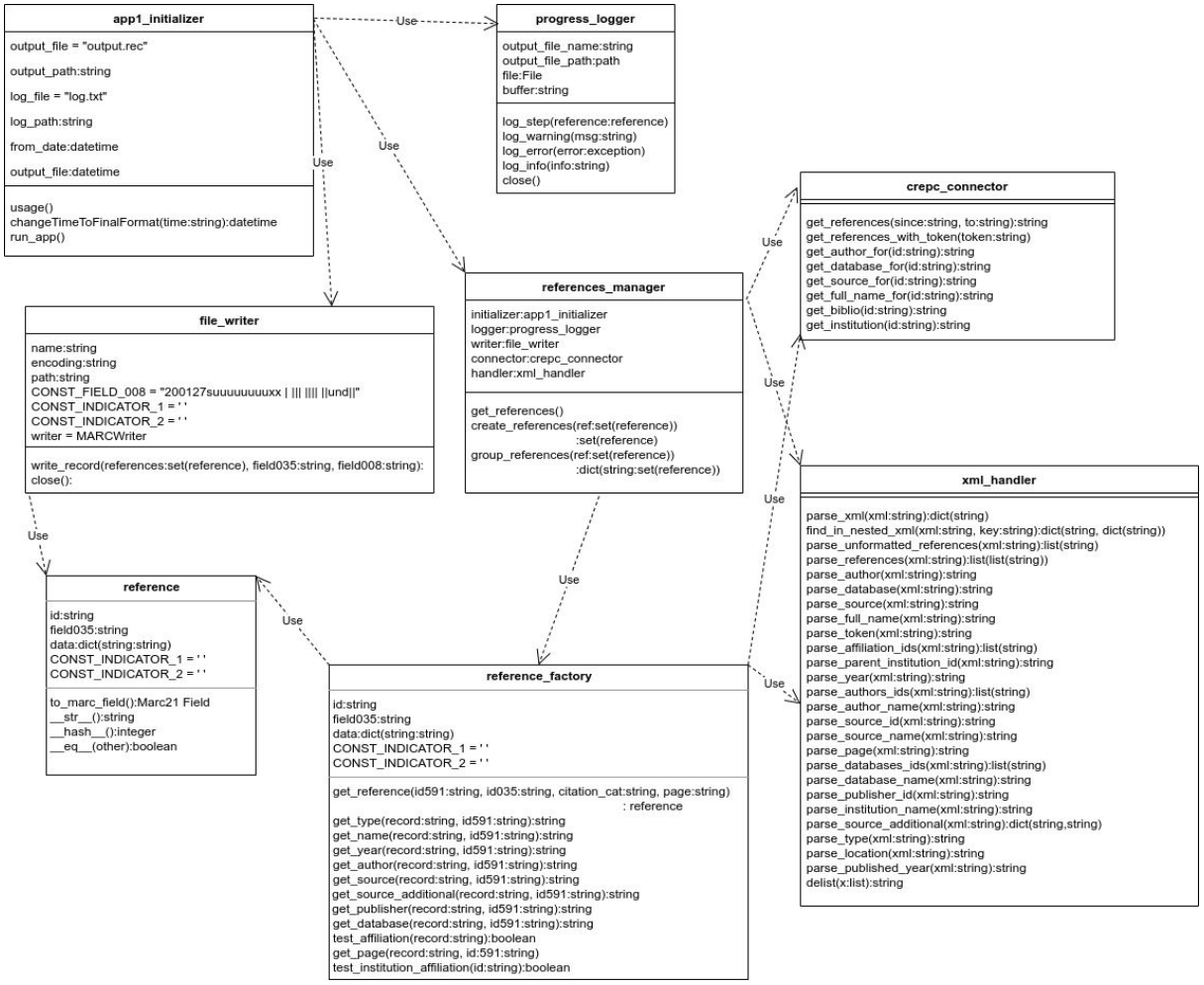
4.3. UML Sequence diagram

Časť prebiehajúcej komunikácie medzi triedami (od vzniku triedy references_manager po jej zánik) je znázornená na nasledovnom sekvenčnom diagrame.



4.4. UML Class diagram

Na obrázku sa nachádza triedny diagram všetkých tried použitých v aplikácii až na 3 výnimky: MissingDataException, CrepConnectionError a WrongXmlDataToParse. Sú to sú jednoduché triedy zdedené od pythonovskej triedy Exception. Dôležité metódy ostatných tried a ich parametre sú popísané v návrhu.



5. Návrh implementácie

Aplikácia sa stará o získanie dát a ich následné zapísanie do súboru v správnom formáte.. Nižšie sú uvedené všetky triedy a metódy, ktoré bude potrebné implementovať na základe class diagramu z kapitoly 4.4 UML Class diagram.

5.1. Trieda app1_initializer

Funkciou tejto triedy je zabezpečenie inicializácie a následné spustenie aplikácie.

Metódy:

- __init__(self, params)
  - Ako parameter dostane argumenty zadané pri spustení, tie spracuje a uloží do vnútornej reprezentácie.
  - Pri nezadaní alebo nesprávnom zadaní argumentov vypíše možnosti spustenia.
  - params - argumenty pri spustení (typ: list[string])
  - triedne premenné: file_path, since, to, log_path

- run_app(self)
  - Vyrobí inštanciu progress_logger a file_writer, následne aj inštanciu references_manager, na ktorej zavolá požadovanú metódu.

- usage(self)
  - Vypíše na terminál návod na použitie aplikácie

5.2. Trieda file_writer

Funkciou tejto triedy je zapisovanie ohlasov do súboru.

Metódy:

- __init__(self, name, encoding = “utf-8”, path=””)

- Pri inicializácií sa pripraví súbor na zapisovanie.
- name - názov súboru (typ: string)
- encoding - kódovanie súboru (typ: string, default: “utf-8”)
- path - cesta, kde bude súbor uložený (typ: string, default: ””)

- write_record(self, reference, field035, field008):
    - Zapíše do súboru jeden ohlas vo forme MARC21/iso2709
    - field035 - reťazec obsahujúci dáta do poľa 035
    - field008 - reťazec obsahujúci dáta do poľa 008
    - reference - ohlas na zapisanie (typ: reference)

- close(self)
    - Ukončí zápis a zavrie súbor.

5.3. Trieda progress_logger

Trieda zabezpečuje vypisovanie priebehu:

1) Do konzoly postupne po krokoch.
2) Do súboru po zavolaní close.

Metódy:

- __init__(self, output_file_name, output_file_path=””, debug)
    - Pri inicializácií dostane ako parametre meno a umiestnenie výsledného súboru.
    - Vyrobí tento súbor a uloží si ho pre budúci zápis.
    - output_file_name - meno výsledného súboru (typ: string)
    - output_file_path - cesta, kam sa má zapísať výsledný súbor (typ: string, default: “”)
    - debug - boolean hodnota určujúca, či aplikácia beží v debug móde

- log_step(self, reference)
    - Zapíše správu o úspešnom zapísaní ohlasu.
    - reference - ohlas na zapísanie (typ: reference)

- log_warning(self, msg)
    - Ak aplikácia beží v debug móde vypíše varovania vzniknuté pri zápise
    - msg - string obsahujúci warning správu

- log_error(self, error)
    - Zapíše správu o chybe pri spracovaní ohlasu.
    - error - vzniknutá chyba (typ: exception)

- log_warning(self, info)

- Ak aplikácia beží v debug móde vypíše extra informácie o zápise
- string obsahujúci extra informácie

- close(self)
  - Zapíše všetky zostávajúce zmeny do súboru a zavrie ho.

5.4. Trieda reference

Trieda uchováva jeden ohlas. Je abstraktná, všetky jej metódy sú abstraktné a odvodené sú od nej tri podtriedy:

1) reference_in_not_registered_magazine
   - Uchováva ohlas z neregistrovaného časopisu.
2) reference_in_publication
   - Uchováva ohlas z publikácie.
3) reference_in_registered_magazine
   - Uchováva ohlas z registrovaného časopisu.

Metódy:

- __init__(self, data)
  - Nastaví si hodnoty z parametra data.
  - Chýbajúce hodnoty sa pokúsi získať.
  - data - slovník hodnôt pre vytvorenie ohlasu, spresňuje odvodená trieda (typ: dictionary{string:string})

- is_valid(self)
  - vracia bool hodnotu - vráti True, ak je ohlas kompletný a je možné ho zapísať

- to_iso2709_string(self)
  - vracia string - reťazec reprezentujúci ohlas v tvare MARC21/iso2709

- __str__(self)
  - vracia string - reťazec reprezentujúci ohlas v zrozumiteľnom tvare

5.5. Trieda reference_manager

Trieda poskytuje možnosť na získanie ohlasov.

Metódy:

- __init__(self, initializer, logger, writer)
  - initializer - obsahuje argumenty zo spustenia (typ: app1_initializer)

- logger - logger do ktorého sa bude zapisovať priebeh (typ: progress_logger)
- writer - writer, do ktorého sa zapíšu ohlasy (typ: file_writer)

- get_new_references(self)
  - do writeru zadanom pri inicializácií zapíše novovzniknuté ohlasy, pričom sa použijú argumenty z initializer

- get_updated_references(self)
  - do writeru zadanom pri inicializácií sa zapíše upravené ohlasy, pričom sa požijú argumenty z initializer

5.6. Trieda crepc_connector

Trieda zabezpečuje komunikáciu s centrálnym registrom publikačnej činnosti.

Metódy:

- __init__(self)
- get_new_references(self, since, to="")
  - Získa novovzniknuté ohlasy.
  - since - dátum od ktorého sa budú získavať ohlasy (typ: string)
  - to - dátum po ktorý sa budú získavať ohlasy (typ: string, default: "")
  - vracia string - reťazec obsahujúci XML so zoznamom ohlasov

- get_updated_references(self, since, to="")
  - Získa upravené ohlasy.
  - since - dátum od ktorého sa budú získavať ohlasy (typ: string)
  - to - dátum po ktorý sa budú získavať ohlasy (typ: string, default: "")
  - vracia string - reťazec obsahujúci XML so zoznamom ohlasov

- get_author_for(self, id)
  - Získa celé meno autora pre zadané id.
  - id - id autora (typ: string)
  - vracia string - celé meno autora

- get_database_for(self, id)
  - Získa názov databázy pre zadané id.
  - id - id databázy (typ: string)
  - vracia string - názov databázy

- get_source_for(self, id)
    - Získa celý názov zdroja pre zadané id.
    - id - id zdroja (typ: string)
    - vracia string - celý názov zdroja

- get_full_name_for(self, id)
    - Získa celý názov publikácie pre zadané id.
    - id - id publikácie (typ: string)
    - vracia string - celý názov publikácie

5.7. Trieda xml_handler

Trieda poskytuje funkcie na spracovanie rôznych xml vstupov.

Metódy:

- __init__(self)
- parse_new_references(self,xml)
    - Spracuje novovzniknuté ohlasy.
    - xml - reťazec s XML na spracovanie (typ: string)
    - vracia list[dictionary{string:string}] - zoznam slovníkov, jeden slovník reprezentuje dáta z jedného ohlasu

- parse_updated_references(self, xml)
    - Spracuje aktualizované ohlasy.
    - xml - reťazec s XML na spracovanie (typ: string)
    - vracia list[dictionary{string:string}] - zoznam slovníkov, jeden slovník reprezentuje dáta z jedného ohlasu

- parse_author(self, xml)
    - Spracuje xml obsahujúce meno autora.
    - xml - reťazec s XML na spracovanie (typ: string)
    - vracia string - meno autora

- parse_database(self, xml)
    - Spracuje xml obsahujúce názov databázy.
    - xml - reťazec s XML na spracovanie (typ: string)
    - vracia string - názov databázy

- parse_source(self, xml)
    - Spracuje xml obsahujúce zdroj ohlasu.
    - xml - reťazec s XML na spracovanie (typ: string)

    - vracia string - názov zdroja

- parse_full_name(self, xml)
    - Spracuje xml obsahujúce celý názov publikácie.
    - xml - reťazec s XML na spracovanie (typ: string)
    - vracia string - názov publikácie

Časť 3: Testovacie scenáre

Úvod

Keďže testovanie funkčnosti systému je pomerne priamočiare, vytvorili sme 4 testovacie scenáre, ktoré budú zahŕňať a kontrolovať celú funkcionalitu systému. Jednotlivé testovacie scenáre sa líšia iba v parametroch, ktoré budú zadávané na vstupe pri spustení. Každý zo scenárov sa zopakuje dvakrát, pričom prvýkrát bude rozmedzie dátumov (since až to) jeden deň, a druhýkrát to bude sedem dní.

Testovací scenár číslo 1

V tomto prípade bude testovaná požiadavka PP-1AP-2 podľa katalógu požiadaviek o zadaní počiatočného dátumu. Aplikácia je spustená len s parametrom since.

Testovací scenár číslo 2

Aplikáciu spúšťame s parametrami since aj to, a teda je už kontrolovaná aj požiadavka PP-1AP-3 z katalógu požiadaviek, ktorá hovorí o zadaní konečného dátumu.

Testovací scenár číslo 3

Kontroluje požiadavku PP-1AP-4 z katalógu požiadaviek. Ide o zadanie cesty pre uloženie výstupného súboru, ktorý má daný východiskový názov output. Tento súbor sa pri zadaní cesty uloží na dané miesto.

Testovací scenár číslo 4

Pri poslednom scenári je kontrolovaná požiadavka PP-1AP-7 v katalógu požiadaviek, kedy sa zadá cesta k log súboru

Vo výsledku nám aplikácia pri každom scenári vytvorí súbor. Správnosť súboru, v ktorom sú vygenerované záznamy v MARC21/iso2709, môžeme skontrolovať pomocou klienta knižničného programu Virtua, ktorý nám bol poskytnutý zadávateľkou.