ZAŁĄCZNIK NR 1 DO SIWZ

OPIS PRZEDMIOTU ZAMÓWIENIA

Przedmiotem zamówienia jest wykonanie usługi aktualizacji i ulepszeń funkcjonowania oprogramowania Centrum Infrastruktury Badawczej Indukowanej Sejsmiczności (CIBIS).

I. Wstęp

Oprogramowanie Centrum Infrastruktury Badawczej Indukowanej Sejsmiczności (CIBIS) zostało stworzone w ramach projektu pn. **„Cyfrowa przestrzeń badawcza sejsmiczności indukowanej dla celów EPOS”,** finansowanego w ramach Programu Operacyjnego Innowacyjna Gospodarka **Działania 2.3 *Inwestycje związane z rozwojem infrastruktury informatycznej nauki.*** Jest ono obecnie wykorzystywane przez eNode Kraków do gromadzenia, opisywania metadanymi oraz udostępnienia danych do platformy IS-EPOS (tcs-ah.epos.eu).

Na wykonaną usługę aktualizacji i ulepszeń funkcjonowania oprogramowania Centrum Infrastruktury Badawczej Indukowanej Sejsmiczności (CIBIS), Wykonawca udziela gwarancji na wszystkie elementy przedmiotu zamówienia na okres co najmniej 12 miesięcy, na warunkach określonych we wzorze umowy, stanowiącym załącznik nr 2 do SIWZ.

II. Opis oprogramowania i jego podstawowych funkcjonalności.

Oprogramowanie CIBIS składa się z następujących części:

1. Panel zarządzania użytkownikami i uprawnieniami – w tej części oprogramowania możliwy jest podgląd ustawień użytkownika, grup użytkowników i ich uprawnień oraz zarządzania nimi.
2. Epizody – Moduł służy do grupowania folderów nie zależnie od ich magazynu danych. Epizod jest jednym z elementów publikacji danych do publikacji w IDAP. System oferuje możliwość definiowania wielu epizodów. Obiekty te mają charakter informacyjny i agregują dane zgromadzone w wielu katalogach danych (FilesDirectory). Do każdego epizodu można przypisać wartości na zasadzie pól klucz-wartość.
3. Magazyny Danych – Moduł służy do fizycznego grupowania folderów. Określa miejsce zapisu wszystkich folderów podpiętych pod dany magazyn danych. Jest to jedna z encji stworzona na potrzebę przechowywania danych - FilesStorage – obiekty tej klasy definiują przestrzeń fizyczną na dysku w obrębie której tworzone będą katalogi na pliki danych. Katalogi te reprezentowane będą w systemie jako obiekty klasy FilesDirectory.
4. Foldery - Moduł służący do operacji na plikach (*File*) i folderach. Pozwala na tworzenie folderów głównych, do których można dodać podrzędne foldery i pliki. Podrzędna struktura folderów może być utworzona na podstawie schematu. Oprócz dodawania plików i folderów, możliwe jest też m.in. kopiowanie, pobieranie, pakowanie do archiwum, rozpakowywanie, zmiana nazwy,

przenoszenie lub podgląd zawartości. Pliki są wersjonowanie (*FileVersionInfo*). Jest to także jedna z encji stworzona na potrzebę przechowywania danych- *FilesDirectory* - instancja tej klasy reprezentuje główny katalog w którym system może tworzyć struktury podkatalogów oraz zapisywać pliki danych. Instancja klasy *FilesDirectory* nie może istnieć bez powiązania z instancją klasy FilesStorage. W przypadku folderów można zdefiniować alternatywne źródło danych – *DataSource* – Datasource PUT, SeisComP.

5. Schematy - Moduł służy do importowania schematów LDAP, które są wykorzystywane do tworzenia drzewa katalogów w folderach. Moduł pozwala również na określenie reguł metadanych Danych do publikacji. Podstawową klasą jest cibisDataItem która opisuje poszczególne node'y w strukturze drzewiastej danych, określone jako STRUCTURAL i dziedziczące po klasie top. System będzie pozwalał na dostarczenie schematów LDAP w postaci plików. Schemat po dostarczeniu do systemu poprzez stworzony w tym celu formularz zostanie przekonwertowany na format XML i w takiej postaci zostanie zapisany w bazie danych systemu CIBIS. Na potrzeby przechowywania schematów i ich wersjonowania zostaną wyodrębnione następujące klasy:
   - Scheme – klasa reprezentująca schemat. Instancja tej klasy nie może istnieć bez powiązania do co najmniej jednego obiektu klasy SchemeVersion.
   - SchemeVersion – klasa reprezentująca wersję schematu, instancja tej klasy nie może istnieć bez relacji do obiektu klasy Scheme.

   System podczas wczytywania schematu pozwala użytkownikowi na wybór czy jest to nowy schemat czy tylko nowa wersja istniejącego już w systemie schematu. Jeśli użytkownik wczytuje do systemu nowy schemat system utworzy obiekt klasy Scheme oraz obiekt klasy SchemeVersion i powiąże je wzajemną relacją. Jeśli natomiast użytkownik wczytuje do systemu nową wersję istniejącego schematu, musi wybrać schemat (obiekt klasy Scheme) którego dotyczy wczytywana wersja. System następnie utworzy instancję klasy SchemeVersion i powiąże ją z wybranym przez użytkownika schematem (obiekt klasy Scheme).
   Pliki reguł służą do wskazania w systemie, które metadane mogą być edytowalne oraz jakie wartości mogą przyjmować.
   Schemat pliku reguł:

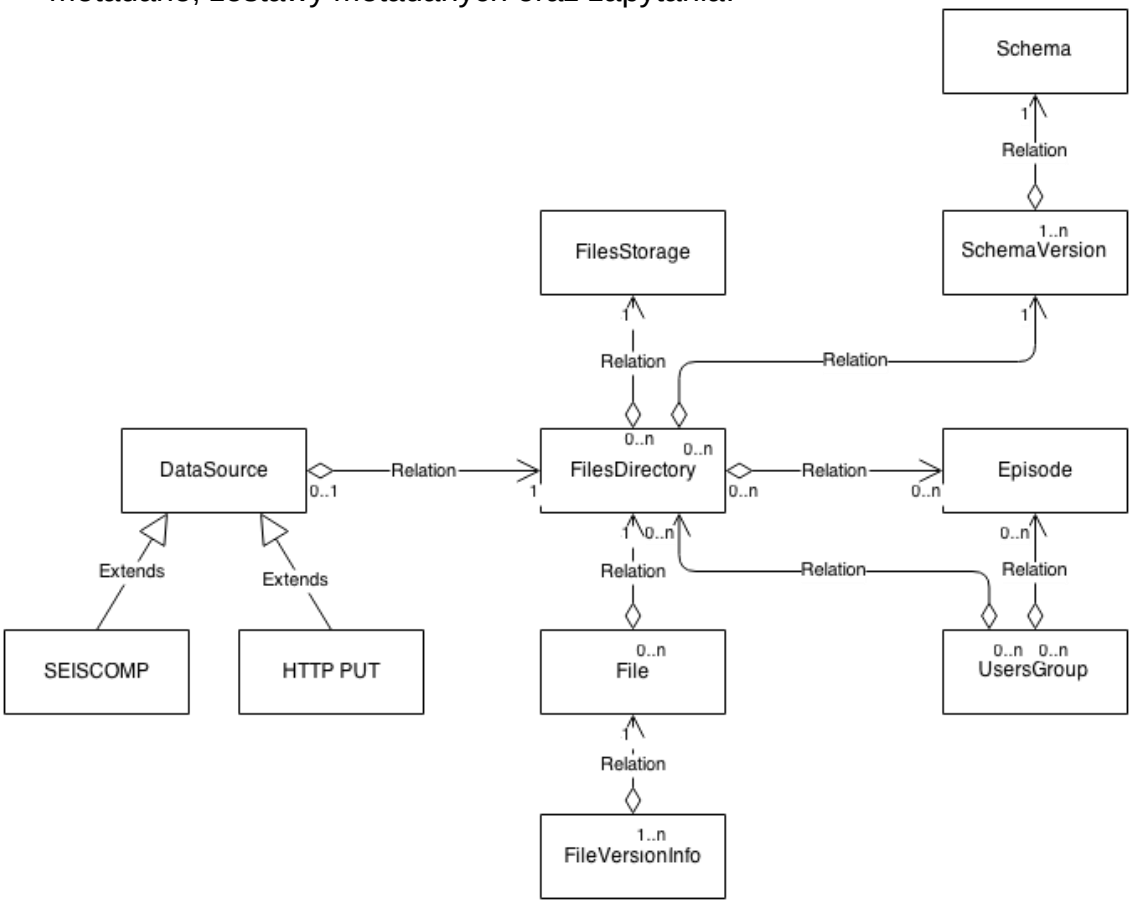
```
<rule>
    <schemaRule objectClass="cibisDataItem">
        <attribute name="name" />
        <attribute name="itemType" />
        <attribute name="path" />
        <attribute name="episode" maxLength="10" isRequired="true"/>
        <attribute name="text" />
        <attribute name="coordinateSystem" enum="['LOCAL:POL001','WGS-84']" />
    </schemaRule>
    <dataTypeRule>
            <item type="episode" required="episode" optional="text" />
            <item type="directory" required="text" optional="coordinateSystem" />
            <item type="file" required="text" optional="coordinateSystem" />
    </dataTypeRule>
    <requirementPolicy>
            <requirement condition="latitude='12' and (episode='ESI-2' or episode='ESI-1')"
            required="coordinateSystem" optional=""/>
    </requirementPolicy>
</rule>
```

W pliku reguł wyróżniamy 3 bloki z danymi:

1. Blok schemaRule:
   Zawiera atrybuty z objectClass używanej przez schemat, w bloku tym użytkownik określa kolejność pól wyświetlających się w formularzu edycji metadanej oraz pola które mają się w nim pojawić. Każdy atrybut może zawierać odpowiednie wartości służące do konfiguracji systemu:
   - maxLength: określa maksymalną ilość znaków dla pola,
   - minLength: określa minimalną ilość wymaganych znaków dla pola,
   - maxValue: określa maksymalną wartość liczbową pola,
   - minValue: określa minimalną wartość liczbową pola,
   - regexp: określa patern służący do walidacji pola,
   - enum: wartość numerowana dla pola określa wartości jakie mogą zostać wybrane dla danego pola.
2. Blok dataTypeRule:
   Zawiera atrybuty określające pola wymagane oraz opcjonalne dla pól: epizodu, folderu oraz pliku.
3. Blok requirementPolicy:
   Zawiera konfigurowalne atrybuty służące do określania, pojawienia się wymagalności pola lub jego opcjonalności w przypadku jego spełnienia. Język zawarty w zmiennej condition przybiera formę <nazwa_pola>=<wartość_wymagana> wartości mogą być łączone za pomocą warunków logicznych and oraz or. Można je również zagnieżdżać.
   W pliku reguł pola name, path oraz objectType nie są edytowalne oraz nie mogą przybierać warunków logicznych.
4. Konfiguracje - Moduł służący do tworzenia list zadań, na podstawie istniejących elementów zadań. Elementem zadań jest konwerter. Lista zadań operuje na wskazanych plikach w sposób zależny od rodzaju jej elementów. By ją wykonać należy przypisać ją do folderu. Oprócz edycji oraz tworzenia, konfiguracje (listy zadań) można eksportować i importować.
5. Dane do publikacji Moduł służy do tworzenia metadanych na podstawie zaimportowanych schematów i reguł dla wybranych folderów. Metadane wykorzystywane są później przy publikacji danych do LDAP. Sekcja bazy danych odpowiada za przygotowanie danych i ich opisu metadanowego w celu udostępnienia ich do Data Center. Użytkownik może utworzyć nową bazę danych przypisując ją do epizodu, folderu, schematu i jego wersji. Następnie przeglądając bazę danych system prezentuje użytkownikowi dane dostarczone przez użytkowników oraz wyniki ich konwersji. Do każdego obiektu typu plik i folder system udostępnia formularz pól metadanych wygenerowany na podstawie wersji schematu oraz pliku XML z ograniczeniami. Z poziomu tego formularza użytkownik może podać wartości metadanych wpisując je ręcznie lub zdefiniować reguły dla automatu odczytującego wartości. Ponadto dla każdego obiektu opisanego metadanymi system oferuje możliwość pobrania pliku XML metadanych, importu tego pliku z dysku oraz ponowne odczytanie wartości z tego pliku na dysku serwera. Dla każdego obiektu opisanego metadanymi, lub całej bazy, system oferuje możliwość uruchomienia automatu odczytującego metadane. Pliki metadanych (zapisane w formacie XML) opisują każdy element danych (plik lub folder) i przechowywane są w tym samym

miejscu w którym znajdują się opisywane przez nie dane. Nazwa tego pliku budowana jest zgodnie ze schematem:
{nazwa_pliku_lub_folderu}.{wersja_bazy}.{wersja_schematu}.meta.xml.
W przypadku gdy baza danych jest poprawna system pozwala na jej publikację w serwerze LDAP. Publikowana struktura bazy danych wygląda następująco:
Schemat/Wersja
-- Epizod
---- Baza danych
------ [Struktura bazy danych]
Pliki metadanowe użyte w bazie danych udostępniane są przez LDAP.

6. Repozytorium - Moduł służący do wymiany dokumentów. Wszystkie jego części mają na celu jak najtrafniejsze opisanie pliku lub zestawu plików dokumentu, ułatwiając tym samym wyszukiwanie konkretnych dokumentów. Oprócz możliwości wyszukiwania, repozytorium określa jakie osoby mają dostęp do jakich zasobów. Repozytorium składa się z sekcji: kategorie, dokumenty, stany, metadane, zestawy metadanych oraz zapytania.



Rys 1 – schemat zależności pomiędzy głównymi modułami systemu.

Wykaz licencji dostawców trzecich oprogramowania

Lp.	Moduł / Oprogramowanie	Wersja	Licencja	Używane	Link do produktu	Stabilne wersje dostępne na dzień 2019-05-22
1	Python 2.7 + biblioteka standardowa	2.7	PSF https://docs.python.org/3/license.html	Całość projektu	www.python.org	3.6.8
2	Django 1.7.1	1.7.1	BSD https://github.com/django/django/blob/master/LICENSE	Całość projektu	https://www.djangoproject.com/	2.2.1
3	Angular JS	1.3.0	MIT http://opensource.org/licenses/MIT	Widok UI	https://angularjs.org/	1.7.8
4	SB Admin 2 Bootstrap Admin Theme	2.0	APACHE http://www.apache.org/licenses/LICENSE-2.0.	Widok UI	http://startbootstrap.com/template-overviews/sb-admin-2/	4.3.1
5	Fabric	1.10.0	OWN LICENSE https://github.com/fabric/fabric/blob/master/LICENSE	Instalator	http://www.fabfile.org/en/latest/	2.4
6	Django Rest Framework	2.4.3	OWN LICENSE http://www.django-rest-framework.org/	Api REST	http://www.django-rest-framework.org/	3.9.4
7	Django-Countries	3.0.1	OWN LICENSE https://github.com/SmileyChris/django-countries/blob/master/LICENSE	Użytkownicy	https://github.com/SmileyChris/django-countries/	5.3.3
9	Django-Requestlogging	1.0.1	OWN LICENSE https://bitbucket.org/trustcentric/django-requestlogging/src/a391cb7a36a168550563058c95bb6269cc785c05/LICENSE?at=default	Logowanie zdarzeń	https://bitbucket.org/trustcentric/django-requestlogging/	1.0.1 Ostatnie wydanie 2014-01-30 **(wymarły komponent programistyczny?)**
10	Django-Phonenumber-Field	0.6	OWN LICENSE https://github.com/stefanfoulis/django-phonenumber-field/blob/develop/LICENSE	Użytkownicy	https://github.com/stefanfoulis/django-phonenumber-field	2.4.0
11	Sphinx	1.2.3	BSD https://github.com/django/django/blob/master/LICENSE	Generowanie dokumentacji	http://sphinx-doc.org/	2.0.1
12	UI Bootstrap	0.11.2	MIT http://opensource.org/licenses/MIT	Widok UI	http://angular-ui.github.io/bootstrap/	2.5.0
13	Angular-Gettext	0.2.15	MIT http://opensource.org/licenses/MIT	Tłumaczenie	https://angular-gettext.rocketeer.be/	2.4.1
14	Grunt	0.4.5	MIT http://opensource.org/licenses/MIT	Automatyzacja generowania js	http://gruntjs.com/	1.0.3

Opis instalacji oprogramowania.

Użytkownik w celu dokonania instalacji systemu CIBIS musi posiadać zainstalowany system Ubuntu Server 14.04 oraz posiadać konto użytkownika „cibis" z uprawnieniami administratora systemu.

1. Przygotowanie do instalacji

   Po instalacji systemu operacyjnego należy dokonać instalacji serwera LDAP oraz skonfigurować serwer map GeoNode. Do katalogu głównego użytkownika należy przekopiować folder „deploy" wraz z jego zawartością:

   Struktura Katalogu deploy:

| - deploy

| - schemes – katalog w którym znajdują się przykładowe schematy z obiektami objectClass

| - templates – katalog w którym znajdują się pliki szablonów służących do konfiguracji systemu

| - config.json – plik konfiguracyjny systemu CIBIS

| - fabfile.py – plik instalatora

| - fabfile_git.py – plik instalatora pobierający repozytorium

2. Tworzenie konfiguracji

   Użytkownik ma możliwość zmiany ustawień systemu CIBIS w pliku config.json. Dokonuje tego poprzez zmianę parametrów pliku json. Parametry zawarte w tym pliku mają następującą formę:

```
{
// główna ścieżka systemu cibis
"www_server_path": "/home/cibis/cibis",
// Ścieżka repozytorium systemu CIBIS
"git_repository": "https://lcaputa@git.infomex.pl/cibis.git",
// Domena, w której pracuje system CIBIS
"domain": "cibistest.infomex.net",
// Administratorzy systemu CIBIS
"admins": [{
"name": "Infomex",
"email": "l.caputa@infomex.pl"
}],
// Ścieżka pliku konfiguracyjnego systemu
"settings": "server.production_settings",
// Katalog zawierający logi systemowe
"log_folder": "server/log",
// Dozwolone adresy IP z których może łączyć się użytkownik
"allowed_hosts": "*",
// Ustawienia baz danych
"databases": {
"default": {
"ENGINE": "django.db.backends.postgresql_psycopg2",
"NAME": "cibis",
"USER": "cibis",
"PASSWORD": "",
"HOST": "",
"PORT": ""
}
}
}
```

3. Uruchamianie instalacji systemu

   W pierwszej kolejności należy wejść w strukturę katalogu deploy następnie z jego poziomu wywołać polecenie:

   sudo pip install fabric

   Po poprawnej instalacji należy wywołać polecenie:

   fab local deploy

Po pojawieniu się prośby o podanie hasła należy wprowadzić hasło użytkownika cibis. Instalator rozpocznie instalację automatyczną systemu po jej poprawnym zakończeniu należy zrestartować system operacyjny.

4. Wgrywanie nowej wersji schematu ldif
   Tworzymy nowy plik schematu ldif poprzez wgranie przykładowego pliku z katalogu schemes lub stworzenie go w następującym formacie:
   // Atrybuty użyte w objectClass
   olcAttributeTypes: {0}(1.3.6.1.4.1.16635.505.100.789.2 NAME 'itemType' SUP name)
   olcAttributeTypes: {1}(1.3.6.1.4.1.16635.505.100.789.5 NAME 'path' SUP name)
   // Obiekt objectClass
   olcObjectClasses: {0}(1.3.6.1.4.1.16635.505.100.78.2 NAME 'cibisDataItem' SUP top STRUCTURAL MUST (name
   $ path $ itemType) MAY description)
   Następnie wywołujemy polecenie służące do wgrania schematu:
   sudo ldapadd -Y EXTERNAL -H ldapi:/// -f <nazwa_pliku>

III. Zakres prac

Aktualizacja i ulepszenia funkcjonalności oprogramowania CIBIS obejmuje następujące zadania.

1. Aktualizacja systemu operacyjnego - Ubuntu Server 18.04 Long-term support (LTS).
2. Aktualizacja frameworków i bibliotek programistycznych oprogramowania. Po aktualizacji oprogramowania przeprowadzenie testu nowej wersji oprogramowania oraz usunięcie ewentualnych usterek. Podniesienie wersji komponentów oprogramowania do aktualnych stabilnych, zmiana technologii w przypadku wymarłej technologii lub zaprzestania rozwoju komponentu programistycznego.
3. Poprawa instalacji i konfiguracji oprogramowania.
   a. Poprawa instalacji oprogramowania oraz instrukcji, tak aby poprawnie można było zainstalować wszystkie niezbędne komponenty oprogramowania.
   b. Dodanie do instalacji domyślnej konfiguracji oprogramowania CIBIS. Zleceniodawca przekaże odpowiednie pliki konfiguracyjne (schema metadanych, plik ldif i rules), które będą stanowiły integralną część instalacji i na ich podstawie zostanie skonfigurowane oprogramowanie.
   c. Poprawa konfiguracji oprogramowania dla innej niż domyślna, nowej schemy metadanych, pliku ldif i rules – usunięcie pojawiających się błędów.
   d. Wymagany jest jasny i spójny instalator, konfigurator oraz instrukcja instalacji z uwzględnieniem możliwości rozdzielenia takich komponentów systemu jak: LDAP, RabbitMQ czy PSQL na różne/osobne serwery.
4. Dodanie nowych funkcjonalności oprogramowania CIBIS obejmuje:
   a. Dodanie opcjonalnego powiadomienia o ukończonych zadaniach – publikowanie epizodu, kopiowanie publikacji, usuwanie publikacji, wczytaj pliki metadanowe z serwera – powiadomienie na email zdefiniowany przez użytkownika.
   b. Dodanie możliwości usuwania użytkownika w panelu zarządzania użytkownikami i uprawnieniami, przez użytkownika o statusie ‘Superuser status’.

c. Kopiowanie wartości metadanych epizodu przy kopiowaniu publikacji epizodu z tym samym schematem (Scheme) natomiast inną, wyższą wersją tego samego schematu (SchemeVersion).
d. Funkcjonalności z wiersza poleceń zaimplementować w graficznym interfejsie użytkownika, tak by nie było potrzeby pracować w powłoce systemu operacyjnego.
e. Wymagana jest możliwość klastrowania systemu - wysoka dostępność.
f. Wymagana jest diagnoza i naprawa wycieków pamięci skryptów w Cron'ie.
g. Wymagana integracja z Active Directory. Członkostwo w odpowiedniej grupie domenowej powinno warunkować poziom dostępu do systemu. System powinien posiadać niezależnie własną baza użytkowników, która umożliwi obsługę użytkowników z poza domeny.

5. Poprawa wydajności oprogramowania CIBIS.
   Wymagane jest aby w przypadku dużych zestawów danych - epizdów (tj. powyżej 5 tys. plików) zaczytywanie metadanych z plików na serwerze i publikowanie epizodu z nowymi plikami trwało co najmniej o 20% krócej niż w obecnej wersji oprogramowania.

IV. Opis realizacji projektu

Zakończenie prac potwierdzone będzie protokołem odbioru. Przed zakończeniem Zleceniobiorca ma przekazać pliki instalacyjne zaktualizowanego oprogramowania wraz z uaktualnioną dokumentacją oprogramowania w zakresie instalacji i konfiguracji Zleceniodawcy nie później niż 7 dni przez zakończeniem prac. Odbiór prac odbywać się będzie na podstawie testów oprogramowania trwających nie dłużej niż 5 dni roboczych. Zaktualizowane oprogramowanie musi zachować wszystkie dotychczasowe funkcjonalności oprogramowania. Jednocześnie zostaną przeprowadzone testy według następujących scenariuszy.

Scenariusz 1

1. Użytkownik na podstawie poprawionej instrukcji instalacji oprogramowania CIBIS instaluje je wraz z domyślną konfiguracją.
2. Użytkownik zakłada nowy Magazyn danych i wgrywa dane testowe do Folderów.
3. Użytkownik zakłada nowy Epizod i przypisuje do niego dane testowe.
4. Użytkownik przygotowuje dane do publikacji, opisuje metadane.
5. Użytkownik publikuje nowe dane w związku z czym są one dostępne poprzez LDAP i WEBDAV.
6. Użytkownik podpina nowe oprogramowanie CIBIS do zasobów platformy IS-EPOS – dzięki czemu dane są dostępne poprzez tą platformę.

Scenariusz nr 2

1. Użytkownik konfiguruje oprogramowanie CIBIS przy wykorzystaniu nowego schematu metadanych pliku ldif i rules.
2. Użytkownik zakłada nowy Magazyn danych i wgrywa dane testowe do Folderów.
3. Użytkownik zakłada nowy Epizod i przypisuje do niego dane testowe.
4. Użytkownik przygotowuje dane do publikacji, opisuje metadane.
5. Użytkownik publikuje nowe dane z nowym schematem metadanych, w związku z czym są one dostępne po przez LDAP i WEBDAV.

Scenariusz nr 3 – test wydajności

1. Użytkownik zakłada nowy Epizod i przypisuje do niego dane testowe – przynajmniej 5 tys plików.
2. Użytkownik przygotowuje dane do publikacji, opisuje metadane.
3. Użytkownik zaczytuje metadane z plików metadanych na serwerze.
4. Użytkownik publikuje nowe dane.

Czas realizacji pkt 3 i 4 powinien być co najmniej o 20% krótszy niż w wersji oprogramowania przed aktualizacją i ulepszeniem funkcjonalności oprogramowania CIBIS.