Janusz S. Bień
kierownik projektu
Uniwersytet Warszawski, Katedra Lingwistyki Formalnej

Narzędzia dygitalizacji tekstów na potrzeby badań filologicznych

Sprawozdanie merytoryczne z grantu MNSzWiT nr N N519 384036
13.05.2009 – 12.05.2012
http://bc.klf.uw.edu.pl/297/

27 czerwca 2012 r.

1. Wstęp

Zespół projektu oprócz kierownika (informatyka i lingwisty) składał się z mgr Joanny Bilińskiej (lingwistki, doktorantki Katedry Lingwistyki Formalnej, http://fleksem.klf.uw.edu.pl/~jbilinska/), dr. hab. Krzysztofa Szafrana (informatyka i lingwisty z Instytutu Informatyki UW, http://www.mimuw.edu.pl/~kszafran/) oraz mgr. Jakuba Wilka (informatyka-programisty, jednego z deweloperów systemu Debian, http://jwilk.net). Pierwotne założenie, że mgr Wilk wykona wszystkie niezbędne prace czysto programistyczne, okazało się z różnych względów nierealne. W związku z tym za zgodą władz uczelni pewne istotne prace dla projektu wykonali również inni informatycy-programiści: mgr Grzegorz Chimosz, mgr Tomasz Olejniczak, mgr Michał Rudolf, mgr Piotr Sikora.

Zgodnie z założeniem projektu jego wynikiem jest oprogramowanie swobodne dostępne publicznie na zasadach Powszechnej Licencji Publicznej GNU (http://pl.wikipedia.org/wiki/GNU_General_Public_License), która zapewnia użytkownikom 4 wolności: wolność uruchamiania programu w dowolnym celu (wolność 0); wolność analizowania, jak program działa i dostosowywania go do swoich potrzeb (wolność 1); wolność rozpowszechniania niezmodyfikowanej kopii programu (wolność 2); wolność udoskonalania programu i publicznego rozpowszechniania własnych ulepszeń, dzięki czemu może z nich skorzystać cała społeczność (wolność 3). Licencja ta jest najbardziej właściwa dla prac programistycznych prowadzonych w środowisku akademickim dla tego środowiska.

Podstawową formą udostępnienia wynikowego oprogramowania są publicznie dostępne repozytoria z jego kodem źródłowym, ich wykaz znajduje się na witrynie projektu dostępnej pod adresem https://bitbucket.org/jsbien/ndt oraz w niniejszym sprawozdaniu. W celu ułatwienia pełnego zapoznania się z możliwościami tego oprogramowania stworzono dodatkowo kilka demonstracyjnych maszyn wirtualnych w formacie *Open Virtual Appliance* (systemy GNU//Linux, dystrybucje Debian

i Ubuntu), dostępnych pod adresem http://fleksem.klf.uw.edu.pl/ndt. Te z programów, które przeznaczone są dla szerszego kręgu użytkowników, są dostępne również w inny sposób, w szczególności znajdują się w ważniejszych dystrybucjach systemu Linux.

Część programów znajduje się już w systematycznej eksploatacji, obsługując w szczególności tzw. wirtualną bibliotekę leksykograficzną Katedry Lingwistyki Formalnej Uniwersytetu Warszawskiego dostępną pod adresem http://wbl.klf.uw.edu.pl/ od 2009 r. i sukcesywnie rozbudowywaną.

Dla użytkowników programów przeznaczono dwie listy pocztowe: listę komunikatów (http://lists.mimuw.edu.pl/listinfo/nmpt-ann) i listę dyskusyjną (http://lists.mimuw.edu.pl/listinfo/nmpt-l). Pozwoli to rozwiązywać ewentualne problemy pojawiające się w trakcie eksploatacji. Jednak w przypadku programów znajdujących się w dystrybucjach systemu Linux rekomendowane jest stosowanie w pierwszej kolejności właściwych dla tych dystrybucji systemów zgłaszania i śledzenia błędów. Dla większości programów możliwe jest też stosowanie systemów zgłaszania i śledzenia błędów właściwych dla repozytoriów, w których się one znajdują.

Zadania projektu polegały na rozwoju narzędzi do obsługi formatu DjVu (http://en.wikipedia.org/wiki/Djvu), który słusznie dominuje w polskich bibliotekach cyfrowych (aktualnie około 80% dokumentów jest w tym formacie, por. http://fbc.pionier.net.pl/owoc/attr-stats). Niektóre z programów zostały już wcześniej zaprojektowane i wstępnie zaimplementowane przez Jakuba Wilka w ramach zrealizowanej pod kierunkiem Janusza S. Bienia pracy magisterskiej (http://bc.klf.uw.edu.pl/28/), niektóre zaprojektował on specjalnie na potrzeby projektu, dla innych mniej lub bardziej szczegółową specyfikację przygotował Janusz S. Bień. Część z nich stanowi alternatywne rozwiązania dla typowych zadań, ale niektóre mają całkowicie oryginalny charakter.

Format DjVu pozwala na zapisywanie skanów czyli obrazów graficznych razem z tekstem stanowiącym najczęściej wynik automatycznego rozpoznawania znaków — taką możliwość ma również np. format PDF, ale w wielu sytuacjach jest on mniej dogodny. Dla badań filologicznych format DjVu jest szczególnie przydatny z kilku powodów. Po pierwsze, w badaniach tych wykorzystuje się często słowniki oraz kartoteki fiszek, które z natury rzeczy nie są czytane strona po stronie — format DjVu pozwala na przechowywanie poszczególnych stron w osobnych plikach, przez co dostęp do dowolnej wybranej strony jest równie szybki. Po drugie, tylko ten format pozwala na swoiste cytowanie fragmentów skanów przez łatwe tworzenie adresów internetowych, których użycie w przeglądarce powoduje wyświetlenie zadanej strony w konkretnym powiększeniu i ustawieniu w oknie z wyraźnie zaznaczonym wskazanym fragmentem — możliwość ta została istotnie wykorzystana w projekcie. Po trzecie, tzw. wspólne słowniki kształtów tworzone na potrzeby kompresji dzięki stworzonym narzędziom mogą być obecnie wykorzystywane do analizy formy i repertuaru znaków np. w dawnych drukach.

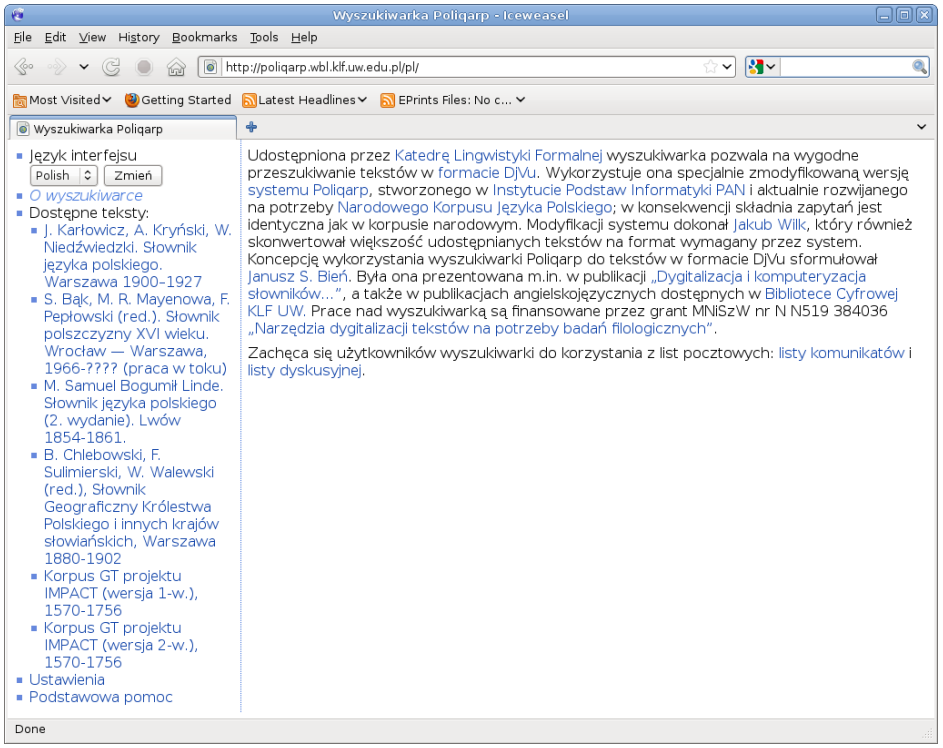
2. Narzędzia programistyczne

W wielu opisanych dalej programach wykorzystywany jest tzw. zestaw dowiązań (ang. *bindings*) do publicznej części biblioteki DjVuLibre dla języka programowania Python. Jego pierwszą wersję stworzył Jakub Wilk w 2008 r. na potrzeby swojej pracy

magisterskiej. W ramach projektu zostało opublikowanych 9 edycji wersji 1 oraz 10 edycji wersji 3 (dostosowanej do wersji 3 języka Python, stąd pominięcie w numeracji wersji 2).

Pełna informacja o programie znajduje się na stronie `http://jwilk.net/software/python-djvulibre`, w formie pakietu jest on dostępny w dystrybucjach systemu GNU/Linux: Debian, Ubuntu i openSUSE. Do sprawozdania załączona jest informacja o popularności pakietu w systemie Debian (zał. nr 10).

3. Korpusy DjVu



Rysunek 1. Przykładowe korpusy DjVu

W lingwistyce przez korpus rozumie się pewien zbiór tekstów przeznaczony do badań statystycznych lub przeszukiwania za pomocą mniej lub bardziej skomplikowanych kwerend; oczywiście w szczególności korpusem może być pojedynczy tekst, zwłaszcza duży — np. wielotomowy słownik. W Polsce narzędziem stosowanym do obsługi korpusów, m.in. Narodowego Korpusu Języka Polskiego, jest system Poliqarp (*Polyinterpretation Indexing Query and Retrieval Procesor — Procesor kwerend i wyszukiwań z indeksowaniem wielointerpretacyjnym*) opracowany w Instytucie Podstaw Informatyki PAN i udostępniony na zasadach Powszechnej Licencji Publicznej GNU (`http://poliqarp.sourceforge.net/`, `https://bitbucket.org/jwilk/poliqarp-core`). Ma on pewne zaawansowane możliwości niedostępne w innych podobnych systemach. System ma architekturę typu klient–serwer, użytkownicy najczęściej korzystają z klienta WWW. Klient WWW aktualnie używany przez Narodowy Korpus Języka Polskiego jest uproszczoną wersją klienta stworzonego na potrzeby niniejszego projektu.

Przez korpus DjVu rozumiemy tutaj korpus tekstów, których podstawową formą jest postać graficzna reprezentowana przez dokumenty w formacie DjVu. Tekst będący przedmiotem wyszukiwania może stanowić tzw. warstwę tekstu ukrytego tych dokumentów, a może być zapisany niezależnie w formacie hOCR — rozszerzeniu formatu HTML przeznaczonym do reprezentacji wyników programów optycznego rozpoznawania znaków (`http://en.wikipedia.org/wiki/Hocr`). Stworzone w ramach projektu rozszerzenie systemu Poliqarp przeznaczone do obsługi takich korpusów nazwaliśmy po prostu Poliqarp for DjVu.

Istotną cechą korpusu DjVu jest możliwość zaznaczenia wyniku wyszukiwania na postaci graficznej odpowiedniego dokumentu. Wyszukiwarki bazujące na innych formatach dokumentów (np. Google Books lub HathiTrust Digital Library) są w stanie w takich sytuacjach pokazać co najwyżej niewielki wycinek tekstu, co nie pozwala np. na obejrzenie większego kontekstu. W przypadku korpusów DjVu możliwości takie daje zarówno klient WWW, jak i klient przeznaczony do instalacji na komputerze użytkownika.

W celu wyczerpującego przetestowania systemu w r. 2009 została stworzona witryna `wbl.klf.uw.edu.pl` nazywana wirtualną biblioteką leksykograficzną Katedry Lingwistyki Formalnej, do której sukcesywnie dodawano wybrane polskiego słowniki historyczne, w tym samodzielnie wskanowane drugie wydanie słownika Lindego (za pomocą skanera zakupionego specjalnie dla projektu). Ostatnio do witryny dodano korpus prawie 5 tysięcy stron dawnych tekstów polskich zdygitalizowanych w ramach projektu IMPACT (por. np. `http://www.digitisation.eu/blog/view/article/polish-impact-ground-truth-added-to-poliqarp-search-engine/`). Planowane jest utrzymywanie witryny również po zakończeniu projektu, najchętniej we współpracy z zainteresowanymi instytucjami — do systematycznych użytkowników witryny należy m.in. Instytut Języka Polskiego PAN, wiadomo też o jej wykorzystywaniu w zajęciach dydaktycznych m.in. na Wydziale Polonistyki UW.

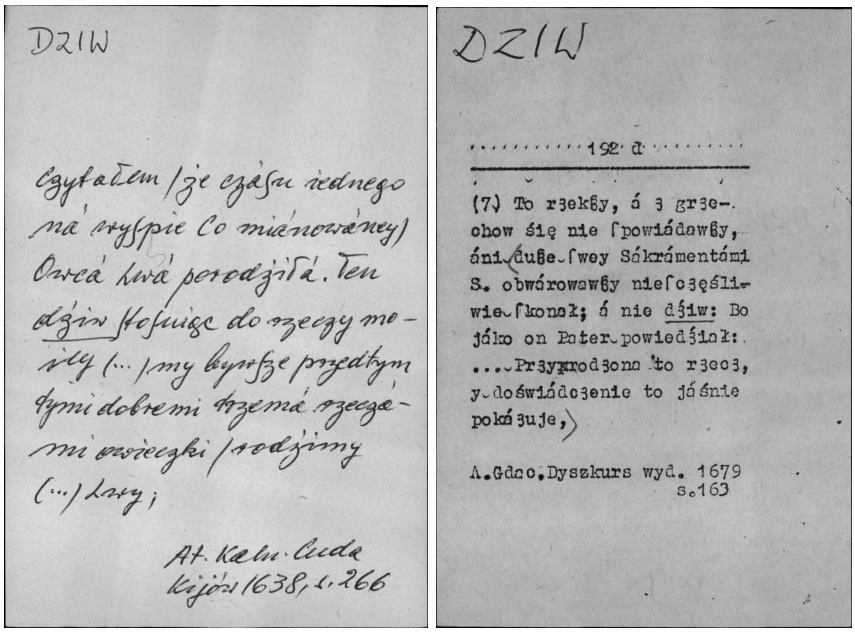
Druga instalacja systemu jest dostępna pod tymczasowym adresem `poliqarp.kanji.klf.uw.edu.pl` i służy do testowania korpusów znajdujących się w przygotowaniu. Można mieć nadzieję, że z czasem pojawią się kolejne instalacje, np. w bibliotekach cyfrowych.

Zasadnicza część systemu, tzn. nowy klient WWW oraz rozszerzenia serwera i klienta o obsługę DjVu, są dostępne w repozytoriach `https://bitbucket.org/jwilk/marasca` i `https://bitbucket.org/jwilk/marasca-wbl`, dodatkowa dokumentacja znajduje się na głównej witrynie projektu. Klient zdalny jest dostępny w repozytorium `https://bitbucket.org/mrudolf/djview-poliqarp`.

Do sprawozdania załączone są trzy artykuły (jeden opublikowany, jeden w druku i jeden złożony do druku) przedstawiające dokładniej założenia systemu i osiągnięte wyniki (zał. nr 1, 2 i 3); pewne aspekty systemu były omawiane również w innych prezentacjach i tekstach wymienionych na witrynie projektu.

4. Kartoteki DjVu

W epoce przedkomputerowej tradycyjnym narzędziem pracy w wielu dziedzinach były fiszki. Obecnie nadal są w użyciu wielomilionowe zbiory fiszek znajdujące się w pracowniach leksykograficznych m.in. Instytutu Języka Polskiego PAN. Kartoteki takie, podobnie jak katalogi kartkowe bibliotek, są coraz częściej dygitalizowane i zapi-



Rysunek 2. Przykłady fiszek z kartoteki IJP PAN

sywane w formacie DjVu. Najczęściej udostępniane są w bardzo prosty i niezaawansowany technicznie sposób — tworzone są dokumenty DjVu odpowiadające poszczególnym szufladkom kartoteki (por. np. `http://rcin.org.pl/publication/20332` i `http://www.bibliotekacyfrowa.pl/publication/185`).

Potrzeba lepszej obsługi zdygitalizowanych kartotek była widoczna już przy formułowaniu celów projektu — kiedy nie było jeszcze nikomu wiadomo, że powstanie projekt Repozytorium Cyfrowe Instytutów Naukowych (rozpoczęty w styczniu 2011 r., `http://rcin.org.pl`) pozwalający w bliskiej przyszłości wskanować ważniejsze kartoteki leksykograficzne. W konsekwencji w projekcie zaplanowano i zrealizowano zakup skanera z podajnikiem i wskanowano m.in. pewną liczbę wypożyczonych z Instytutu Języka Polskiego PAN fiszek. Kiedy jednak stało się to możliwe, do testów zaczęto wykorzystywać próbkę około 10 tysięcy fiszek wskanowanych w projekcie RCIN zgodnie z przyjętymi w tym projekcie standardami.

Opracowany program został nazwany przeglądarką materiałów leksykograficznych, w skrócie `maleks`, choć może być używany do wszelkiego rodzaju fiszek uporządkowanych alfabetycznie, a więc również np. do bibliotecznych katalogów kartkowych. Przeglądarka ta pozwala szybko odnaleźć potrzebne fiszki metodą zbliżoną do wyszukiwania binarnego. Ponieważ wszystkie wyniki — również pośrednie — każdego wyszukiwania zostają zapamiętane w bazie danych, każda następna kwerenda wymaga mniejszej liczby kroków, w szczególności dla coraz większej liczby fiszek wynik jest uzyskiwany natychmiast.

Podstawową zaletą programu jest możliwość pracy na materiałach, dla których automatyczne rozpoznawanie znaków jest trudne lub nawet niemożliwe — hasła na fiszkach słownikowych praktycznie zawsze są dopisywane ręcznie i typowe programy do OCR sobie z nimi nie radzą. Sprawa nie jest jednak całkowicie beznadziejna i dlatego przewidziano możliwość dostarczenia programowi wyników OCR w formie plików hOCR, które są wykorzystywane do podpowiadania użytkownikowi tekstu

hasła. Warto dodać, że wyniki indeksowania fiszek po przekształceniu do odpowiedniego formatu mogą być wykorzystywane do trenowania programu do OCR.

Przyjęto istotne założenie, że program będzie używany systematycznie przez niewielką grupę zawodowych leksykografów pracujących zespołowo w instytucji zapewniającej odpowiednie wsparcie informatyczne. W konsekwencji program ma charakter klienta serwera bazy danych MySQL, który może być np. wspólny dla redakcji lub instytutu, ale powinien być skonfigurowany przez informatyka. Interfejs użytkowy programu jest bardzo oszczędny i może sprawiać wrażenie mało przyjaznego, ale mimo to pozwala na bardzo efektywną pracę w najbardziej prawdopodobnym scenariuszu tzw. indeksowania okazjonalnego. W naszej ocenie program nadaje się już obecnie do użytku, ale oczywiście może być też potraktowany jako prototyp czy tylko *proof of concept* i dalej rozwijany np. w ramach projektu RCIN, który dysponuje funduszami na oprogramowanie do udostępniania wskanowanych kartotek.

Program dostępny jest w repozytorium `https://bitbucket.org/tomek87/maleks`.

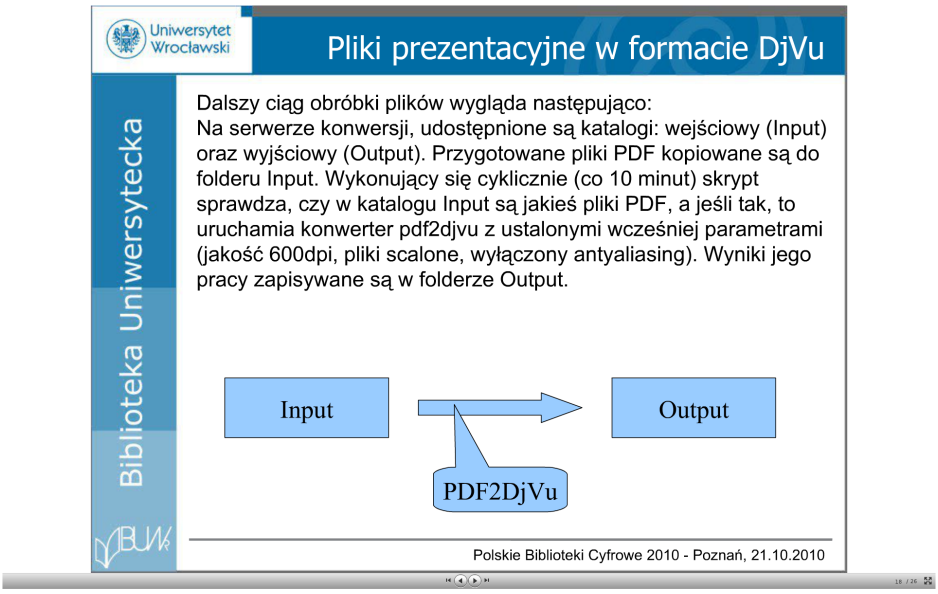
Do sprawozdania załączony jest złożony do druku artykuł przedstawiający dokładniej założenia programu oraz dokumentacja użytkowa (zał. nr 4 i 5).

5. Tworzenie dokumentów DjVu

5.1. Konwersja z formatu PDF

Autorem koncepcji programu pdf2djvu jest Jakub Wilk, który pierwszą wersję programu stworzył w 2007 r. na potrzeby swojej pracy magisterskiej. Jak wskazuje nazwa, program służy do konwersji dokumentów w formacie PDF na format DjVu. W przeciwieństwie do wcześniej dostępnych rozwiązań, program zachowuje praktycznie wszystkie te własności dokumentu, które dają się reprezentować również w DjVu. Choć program ten może być stosowany do dokumentów PDF „urodzonych cyfrowo", jego najczęstsze zastosowanie to konwertowanie zapisanych w formacie PDF wyników optycznego rozpoznawania znaków. W taki sposób zapisuje wyniki między innymi bardzo popularny i mający bardzo dobrą opinię program FineReader firmy ABBY. Do czasu pojawienia się w roku 2011 wersji 11 tego programu, która może zapisywać wyniki również w formacie DjVu, wykorzystanie pdf2djvu było praktycznie jedyną metodą wykorzystania rozpoznanego przez FineReader tekstu. W przypadku wersji 11 wyniki w formacie PDF zawierają więcej informacji niż w formacie DjVu, w konsekwencji stosowanie pdf2djvu w niektórych przypadkach jest nadal uzasadnione. Swoją drogą ponieważ wariant FineReader stosowany w bibliotekach cyfrowych jest dość kosztowny, nie śpieszą się one z jego uaktualnianiem i często nadal stosują wersję 10 lub nawet wcześniejszą.

Z przedstawionych wyżej powodów pdf2djvu wzbudził zainteresowanie niektórych bibliotek cyfrowych. Na przykład w referacie T. Kaloty, R. Raczyńskiego i P. Rękara z Biblioteki Uniwersyteckiej we Wrocławiu prezentowanym w 2010 r. na konferencji *Polskie Biblioteki Cyfrowe* (`http://lib.psnc.pl/publication/365`) można przeczytać *pdf2djvu wydaje się być najkorzystniejszym rozwiązaniem do zrealizowania celów postawionych przy digitalizacji czasopism drukowanych gotykiem*. Wiadomo też, że program jest od pewnego czasu w systematycznym użyciu w Bibliotece Uniwersyteckiej w Warszawie.



Rysunek 3. T. Kalota, R. Raczyński i P. Rękar o wykorzystaniu pdf2djvu (`http://lib.psnc.pl/publication/366`)

Przez cały czas trwania projektu program był systematycznie rozwijany. Pojawiły się w nim pewne nowe funkcje, ale przede wszystkim system był dostosowany do zmieniającego się środowiska (program jest zależny od innych narzędzi takich jak Poppler, por. `http://en.wikipedia.org/wiki/Poppler_(software)`), były też na bieżąco usuwane zgłaszane przez użytkowników błędy. Z inicjatywy użytkowników program otrzymał dodatkowe wersje językowe, w tym niemiecką i rosyjską.

W ramach projektu zostały opublikowane 4 wydania wersji 0.5 (od 0.5.8 do 0.5.11), 3 wydania wersji 0.6, w której po raz pierwszy pojawiła się obsługa metadanych w standardzie XMP (*Extended Metadata Platform*), oraz 13 wydań wersji 0.7, która ma możliwość przyśpieszania konwersji drogą równoległego przetwarzania na wskazanej liczbie procesorów. Pomimo zakończenia projektu autor w miarę możliwości czasowych dalej rozwija program — aktualna wersja to 0.7.13 z 3.06.2012 r.

Pełna informacja o programie znajduje się na stronie `http://jwilk.net/software/pdf2djvu`. W formie pakietu program jest dostępny w dystrybucjach systemu GNU/Linux: Debian, Ubuntu i openSuse, dostępna jest również wersja dla FreeBSD i dla MS Windows. Dla MS Windows istnieje dodatkowo stworzony przez jednego z użytkowników interfejs graficzny.

Do sprawozdania załączona jest strona podręcznikowa programu oraz informacja o popularności programu w systemie Debian (zał. nr 5 i 10).

5.2. Konwersja plików graficznych

Źródłem plików graficznych w procesie dygitalizacji jest oczywiście skanowanie. Na potrzeby przygotowania danych testowych dla projektu został opracowany program scanhelper (`https://bitbucket.org/jwilk/scanhelper`). Pozwala on na skanowanie z automatycznego podajnika z maksymalną prędkością osiąganą przez skaner (eksperymenty były przeprowadzane ze skanerem Fujitsu fi-6130 w systemie Debian GNU/Linux na dwurdzeniowym komputerze); jest to możliwe dzięki podziałowi zadań między procesory. Dodatkowo skrypt dokumentuje przebieg skanowania

zapisując parametry procesu w formie plików XMP. Eksperymenty pokazały, że skanowanie tego samego materiału pod MS Windows z wykorzystaniem standardowego oprogramowania jest wielokrotnie wolniejsze.

Jedną z najważniejszych zalet formatu DjVu jest możliwość rozwarstwienia obrazu strony na właściwy tekst pisany lub drukowany (zapis/zadruk, ang. *foreground*) oraz tło (ang. *background*), co z kolei pozwala wykorzystać fakt występowania w tekście powtarzających się znaków w celu stworzenia słowników kształtów wspólnych dla pojedynczych stron lub ich większej liczby — poprawia to znacznie stopień kompresji. Komercyjne programy do tworzenia dokumentów DjVu wykorzystują te możliwości w istotny sposób, jednak do niedawna jedynym swobodnym narzędziem o podobnej funkcji był program minidjvu (`http://minidjvu.sourceforge.net/`). Program ten jest niestety mało efektywny, wymagający dużej pamięci operacyjnej, słabo udokumentowany, praktycznie nieutrzymywany i nierozwijany.

Na potrzeby projektu Jakub Wilk zaprojektował program didjvu, który do rozwarstwiania stosuje algorytmy dostępne w zestawie narzędziowym do rozpoznawania znaków i analizy dokumentów Gamera (`http://gamera.informatik.hsnr.de/`). Program tworzy słowniki kształtów wspólnych tylko dla pojedynczych stron, może jednak współpracować z programem minidjvu w celu tworzenia słowników wspólnych dla zadanej liczby stron.

Wydanie 0.1 programu didjvu zostało opublikowane w grudniu 2009 r., następnie ukazało się 5 wydań wersji 2; od wersji 0.2.3 program obsługuje metadane w standardzie XMP (*Extended Matadata Platform*). Pomimo zakończenia projektu autor w miarę możliwości czasowych dalej rozwija program — aktualna wersja to 0.2.6 z 15.05.2012 r.

Pełna informacja o programie znajduje się na stronie `http://jwilk.net/software/didjvu`. W formie pakietu program jest dostępny w dystrybucjach systemu GNU/Linux: Debian i Ubuntu.

Do sprawozdania załączona jest strona podręcznikowa programu oraz informacja o popularności programu w systemie Debian (zał. nr 7 i 10).

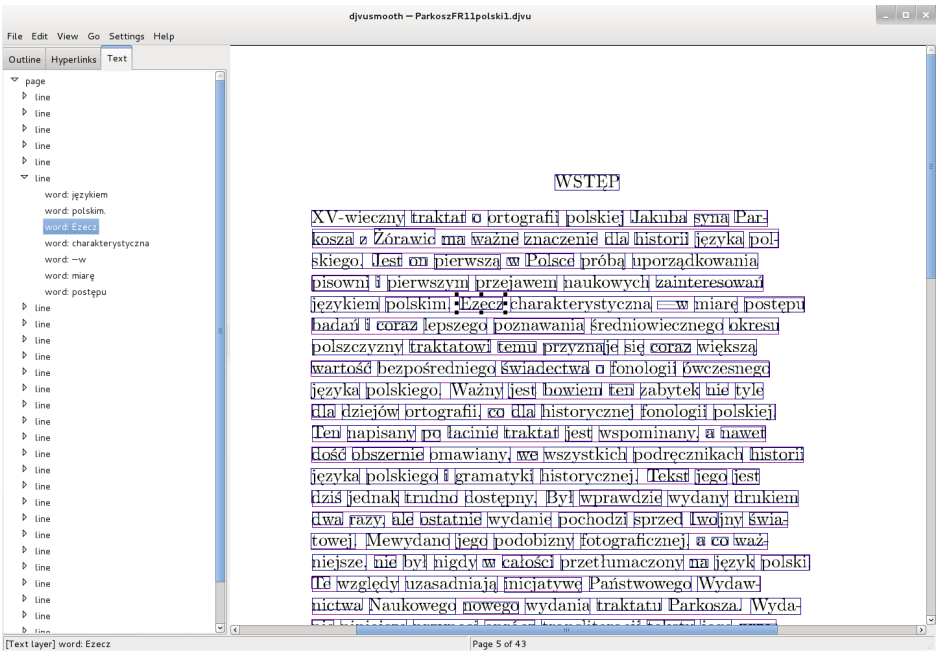
6. Edycja dokumentów DjVu

Do edycji warstwy tekstowej dokumentu DjVu oraz jego adnotacji służy program djvusmooth. Jego pierwszą wersję stworzył Jakub Wilk w 2008 r. na potrzeby swojej pracy magisterskiej, częściowo według sugestii Janusza S. Bienia. Nadal aktualny jest opis użytkowy przedstawiony w tej pracy (`http://bc.klf.uw.edu.pl/28/`).

W ramach projektu zostało opublikowanych 12 wydań wersji 0.2 w celu uwzględnienia uwag użytkowników programu; wersja 0.2.0 była pierwszą wersją z polską wersją językową, od wersji 0.2.7 dostępna jest również wersja rosyjska. Pomimo zakończenia projektu autor w miarę możliwości czasowych dalej rozwija program — aktualna wersja to 0.2.12 z 13.06.2012 zawierająca m.in. hiszpańską wersję językową.

Pełna informacja o programie znajduje się na stronie `http://jwilk.net/software/djvusmooth`. W formie pakietu program jest dostępny w dystrybucjach systemu GNU/Linux: Debian, Ubuntu i openSUSE.

Do sprawozdania załączona jest strona podręcznikowa programu djvusmooth i djvu2hocr oraz informacja o popularności programu w systemie Debian (zał. nr 8 i 11).



Rysunek 4. djvusmooth — przykład zastosowania (poprawianie błędów OCR)

7. DjVu i optyczne rozpoznawanie znaków

7.1. Wykorzystanie dostępnych programów do OCR

Jak było wspomniane wcześniej, w przypadku stosowania komercyjnego programu FineReader, najwygodniej jest zapisać wynik w formacie PDF i użyć konwertera pdf2djvu; procedura ta wydaje się możliwa również w przypadku innych programów komercyjnych. Programy te mają jednak coraz silniejszą konkurencję w postaci programów swobodnych, takich jak OCRopus (`http://en.wikipedia.org/wiki/OCRopus`), Tesseract (`http://en.wikipedia.org/wiki/Tesseract_(software)`), Cuneiform (`http://en.wikipedia.org/wiki/CuneiForm_(software)`), Ocrad (`http://en.wikipedia.org/wiki/Ocrad`) czy Gocr (`http://en.wikipedia.org/wiki/GOCR`).

Możliwość użycia niekomercyjnego oprogramowania do OCR skłoniła Jakuba Wilka do przygotowania w 2008 r. na potrzeby swojej pracy magisterskiej pierwszej wersji programu ocrodjvu, który wykorzystywał wyniki programu OCRopus zapisywane w formacie hOCR; w związku z tym został opracowany również pomocniczy program hocr2djvused konwertujący hOCR na format akceptowany przez istniejące narzędzia DjVu (djvused). W ramach projektu zostały opublikowane 2 wydania wersji 0.2, 3 wydania wersji 0.3, 8 wydań wersji 0.4, 2 wydania wersji 0.5, 2 wydania wersji 0.6 i 10 wydań wersji 0.7. Poszczególne wydania miały na celu uwzględnienia uwag i postulatów użytkowników programu, w szczególności w wersji 0.3 został wprowadzony pomocniczy program djvu2hocr, w wersji 0.4 została dodana obsługa Cuneiform, w wersji 0.5 obsługa Ocrad i Gocr, w wersji 0.6 obsługa programu Tesseract, która w wersji 0.7 została uaktualniona do obsługi istotnie odmiennej wersji 3 tego programu. Ostatnia wersja przygotowana w ramach projektu to 0.7.10, ale pomimo jego

zakończenia autor w miarę możliwości czasowych dalej rozwija program — najnowsza wersja to 0.7.11 z 28.05.2012.

> If you are creating e-books in DjVu format, there is a useful helper called *ocrodjvu* by Jakub Wilk [14] (Debian Squeeze and Ubuntu Lucid). Ocrodjvu takes over control of OCRopus and automates the whole process of text extraction and insertion into an existing DjVu-based e-book; as an alternative to Tesseract/OCRopus, you can also use Cuneiform (see the "Open Source OCR" box).

Rysunek 5. Daniel Stender, *Linux Magazine*, August 2010, Issue 117, p. 83 (czasopismo ma również inne wersje językowe, w tym polską)

Dodatkowym uzupełnieniem ocrodjvu jest pomocniczy skrypt scans2djvu+hocr dystrybuowany razem z tym programem. Automatyzuje on proces przetwarzania skanów umieszczonych we wskazanych katalogach, dokonując ich rozwarstwienia i konwersji za pomocą didjvu, a następnie stosując program ocrodjvu do automatycznego rozpoznawania znaków, przy czym wyniki mogą być zapisane w pliku hOCR. Na uwagę zasługuje fakt, że szczegóły przetwarzania są dokumentowane w plikach XMP.

Pełna informacja o programie znajduje się na stronie `http://jwilk.net/software/ocrodjvu`. W formie pakietu program jest dostępny w dystrybucjach systemu GNU/Linux: Debian, Ubuntu i openSUSE.

Do sprawozdania załączone są strony podręcznikowe ocrodjvu, hocr2djvused i djvu2hocr oraz informacja o popularności programu w systemie Debian (zał. nr 9 i 10).

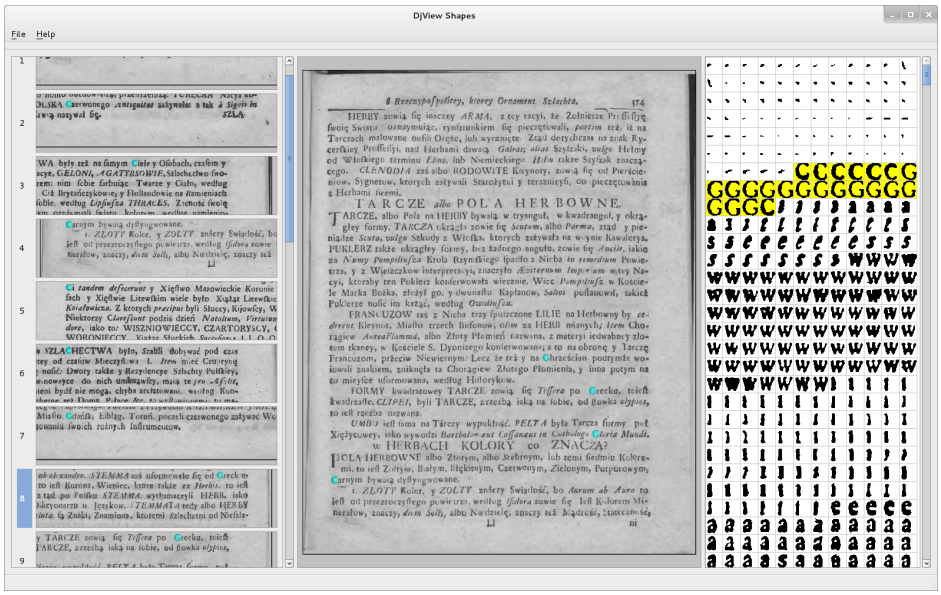
7.2. Ułatwienia trenowania programów do OCR

Dostosowywanie programów OCR do historycznych tekstów polskich jest w tej chwili jednym z zadań projektu SYNAT (`http://www.synat.pl/`), którego realizacja rozpoczęła się w r. 2010, a zakończenie planowane jest na r. 2013. Pewne informacje na ten temat można znaleźć w referacie A. Dudczaka, M. Kmieciaka i M. Werli *Country-scale infrastructure for creation of full text versions of historical documents from Polish Digital Libraries* (`http://lib.psnc.pl/publication/426`). Przedstawione niżej narzędzia nie dublują oprogramowania przygotowywanego w ramach projektu SYNAT, ale są w stosunku do nich komplementarne, wykorzystują bowiem specyficzne cechy formatu DjVu. Z tego samego powodu nie dublują one oprogramowania stworzonego w ramach wspomnianego wcześniej — właśnie zakończonego — europejskiego projektu IMPACT (*IMProving ACcess to Text*), choć cele projektów były podobne.

Jeśli w trakcie tworzenie dokumentu DjVu zostały zbudowane wspólne słowniki kształtów — co zależy od użytego oprogramowania, a także od samego dokumentu (czy kontrast między zadrukiem a tłem jest wystarczająco duży) — to słowniki te zawierają repertuar kształtów odpowiedniego fragmentu dokumentu, co stanowi z reguły dość dobre przybliżenie repertuaru znaków. Jest to tylko przybliżenie, ponieważ kształty w słownikach są spójne, co powoduje rozdzielenie dwuczęściowych liter typu *c* z kreską, *z* z kropką, a także litery *i*. Wydaje się jednak, że z praktycznego punktu widzenia nawet takie przybliżenie jest bardzo pożyteczne.

7.2.1. Ustalenie repertuaru znaków

Zawarte w dokumentach DjVu wspólne dla wielu stron słowniki powtarzających się kształtów pozwalają w szczególności na wybranie z tekstu reprezentatywnych fragmentów i wykorzystanie ich następnie do bezpośredniego interakcyjnego trenowania programu typu FineReader lub do tworzenia danych treningowych za pomocą odpowiednich interakcyjnych narzędzi, np. opracowywanych w projekcie SYNAT. Służy do tego zmodyfikowana przeglądarka dokumentów DjVu djview-shapes, oparta na przeglądarce djview4 i wcześniejszym prototypie.



Rysunek 6. djview-shapes: wykaz wystąpień kształtów podobnych do litery *C*

Program ten jest dostępny w repozytorium `https://bitbucket.org/mrudolf/djview-shapes`.

Program będzie z pewnością wykorzystywany w pracach dygitalizacyjnych Katedry Lingwistyki Formalnej, planowane jest także wykorzystywanie go na zajęciach dydaktycznych poświęconych paleografii polskiej. Sądzimy, że znajdzie on również inne zastosowania. W razie potrzeby będzie on dalej rozwijany.

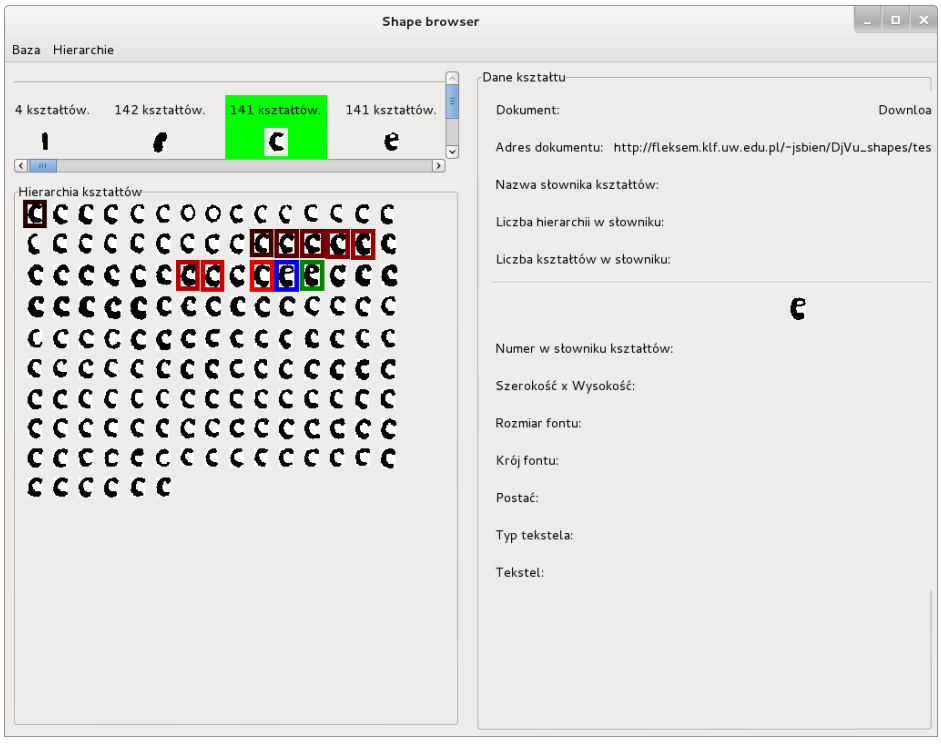
Warto dodać, że opracowanie narzędzia do tworzenia inwentarza znaków — działającego na zupełnie innych zasadach — było jednym z celów projektu IMPACT, ale prace te zostały przerwane bez podania przyczyn (być może koncepcja programu nie sprawdzała się w praktyce). Uzyskane dotąd wyniki zostały określone jako *experimental prototype* (por. `http://www.digitisation.eu/tools/experimental-prototypes/inventory-extraction/`) i udostępnione publicznie na swobodnej licencji Apache 2.0 (`https://github.com/impactcentre/inventory-extraction`). Niestety oprogramowanie to jako danych wejściowych wymaga wyników komercyjnych programów używanych w projekcie lecz niedostępnych dla zwykłego użytkownika (np. FineReader *System Development Toolkit*), w rezultacie przydatność tego programu — przynajmniej w obecnej postaci — jest znikoma i nie stanowi on alternatywy dla djview-shapes.

7.2.2. Przygotowywanie tekstów wzorcowych

Najbardziej uniwersalnym sposobem trenowania programów do OCR jest przygotowanie zestawu tekstów wzorcowych. W projekcie IMPACT procedura ta wyglądała w ten sposób, że wyniki programu FineReader w wersji SDK (*System Development Kit*) były ręcznie korygowane za pomocą specjalnego programu. Naszym zdaniem nie ma potrzeby stosowania odrębnych narzędzi do tworzenia tekstów wzorcowych i do zwykłej korekty, ponieważ jest to w istocie ten sam proces.

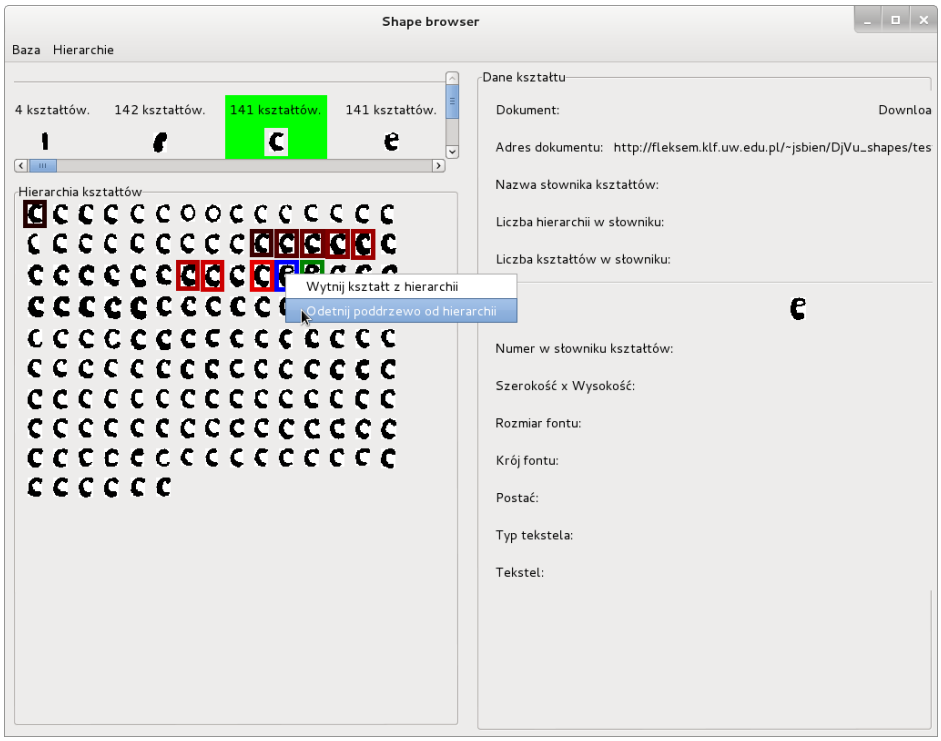
Wszystkie znane nam narzędzie tego typu wymagają od użytkownika sukcesywnej akceptacji kolejnych znaków tekstu, prezentowanych w kolejności ich występowania w tekście. Za wyjątek można uznać opracowany w ramach projektu IMPACT system CONCERT, który jednak jest przeznaczony do innych celów, por. `http://www.digitisation.eu/tools/ocr-post-correction-and-enrichment/collaborative-correction-platform/`; w dodatku narzędzie to nie tylko nie jest swobodnie dostępne, ale jest dostosowane wyłącznie do kosztownej wersji komercyjnego programu FineReader.

Wykorzystanie formatu DjVu — w którym wspólne słowniki kształtów są tworzone na potrzeby kompresji — pozwala prezentować użytkownikowi całe zestawy kształtów, które mogą być opisywane jednocześnie. Zestawy kształtów pobierane ze słownika mają strukturę drzewiastą — jeśli w drzewie znalazły się jakieś znaki niewłaściwe, wówczas użytkownik powinien odciąć odpowiednią gałąź drzewa usuwając je w ten sposób z zestawu.



Rysunek 7. Hierarchia kształtów litery *c* z „zabłąkaną” literą *e*

Uzyskany w ten sposób zbiór kształtów odpowiadający jednej literze może zostać „zaetykietowany” przez przypisanie mu współrzędnej kodowej zgodnej z powszechnie stosowanym standardem Unicode.



Rysunek 8. Eliminacja „zabłąkanej” litery *e*

Sposób zapisywania w dokumencie DjVu informacji o kształtach wspólnych jest podporządkowany jednemu celowi, mianowicie wizualizowaniu strony. W konsekwencji uzyskanie wykazu wystąpień kształtów na stronach wymaga specjalnego przetworzenia dokumentu w celu stworzenia odpowiedniego indeksu. Z drugiej strony zadania takie, jak przygotowywanie tekstów wzorcowych z reguły mają charakter zespołowy. W konsekwencji uznaliśmy za właściwe zastosowanie architektury klient-serwer i przechowywanie indeksu kształtów w bazie danych MySQL na serwerze.

Chociaż na krótką metę rozwiązanie takie może być trochę kłopotliwe, jest ono bardziej perspektywiczne. Z informatycznego punktu widzenia jest wygodne, że wstawianie kształtów do bazy danych jest wykonywane przez osobny program, co pozwala na eksperymentowanie z innymi algorytmami identyfikowania i porównywania kształtów niż stosowane w formacie DjVu. Z punktu widzenia użytkownika architektura klient-docelowo serwer pozwala względnie wygodnie wykorzystywać indeksy nie tylko dla wybranych publikacji lub ich fragmentów, ale także np. dla zbiorów publikacji pochodzących z tych samych drukarń. Daje to w szczególności możliwość kontynuowania w nowoczesny sposób przedsięwzięcia zainicjowanego w 1936 r. publikacją pierwszego tomu dzieła *Polonia typographica saeculi sedecimi: zbiór podobizn zasobu drukarskiego tłoczni polskich XVI stulecia* i przerwanego po publikacji tomu XII w r. 1981; przykładem podobnego projektu zagranicznego może być *Cassetin* (`http://www.tug.org/TUGboat/tb24-3/andre.pdf`).

Programy do obsługi wspólnych słowników kształtów na potrzeby przygotowywania tekstów wzorcowych są dostępne w repozytoriach `https://bitbucket.org/piotr_sikora/djvulibre-shape-tools` i `https://bitbucket.org/piotr_sikora/moredjvushapetools`. Progamy te będą z pewnością wykorzystywane w pracach dygitalizacyjnych Katedry Lingwistyki Formalnej, spodziewamy się również, że

będą one dalej rozwijane z uwzględnieniem doświadczeń własnych i innych użytkowników.

8. Uwagi końcowe

Część przygotowanych w ramach projektu narzędzi już znajduje się w eksploatacji w kraju i za granicą — uwagi do udostępnionych programów napływały m.in. z Czech, Francji, Indii, Hiszpanii, Izraela, Niemiec, Rosji, Ukrainy i Szwajcarii. Można mieć nadzieję, że użytkowników znajdą wkrótce również pozostałe programy.

9. Adresy internetowe wyników projektu

1. `https://bitbucket.org/jsbien/ndt` — główna witryna projektu.
2. `http://jwilk.net/software/python-djvulibre` — tzw. zestaw dowiązań (ang. *bindings*) wykorzystywany w innych programach.
3. `https://bitbucket.org/jwilk/marasca`, `https://bitbucket.org/jwilk/marasca-wbl` — różne składniki systemu Poliqarp for DjVu, w tym serwer i klient WWW.
4. `https://bitbucket.org/mrudolf/djview-poliqarp` — zdalny graficzny klient systemu Poliqarp for DjVu.
5. `https://bitbucket.org/tomek87/maleks` — przeglądarka materiałów leksykograficznych.
6. `http://jwilk.net/software/pdf2djvu` — rozbudowany konwerter dokumentów PDF na format DjVu.
7. `http://jwilk.net/software/scanhelper` — program ułatwiający skanowanie z wykorzystaniem automatycznego podajnika.
8. `http://jwilk.net/software/didjvu` — rozbudowany konwerter plików graficznych na format DjVu.
9. `http://jwilk.net/software/djvusmooth` — edytor dokumentów DjVu.
10. `http://jwilk.net/software/ocrodjvu` — program do automatycznego rozpoznawania znaków w dokumentach DjVu za pomocą swobodnych silników OCR.
11. `https://bitbucket.org/mrudolf/djview-shapes` — podstawowy program do analizy repertuaru znaków publikacji.
12. `https://bitbucket.org/piotr_sikora/djvulibre-shape-tools`, `https://bitbucket.org/piotr_sikora/moredjvushapetools`, — narzędzia ułatwiające trenowanie programów do OCR.
13. `http://fleksem.klf.uw.edu.pl/ndt` — demonstracyjne maszyny wirtualne w formacie OVA (*Open Virtual Appliance* — systemy GNU//Linux, dystrybucje Debian i Ubuntu))

10. Wykaz załączników

1. *Efficient search in hidden text of large DjVu documents*, artykuł opublikowany, `http://bc.klf.uw.edu.pl/177/`.

2. *Skanowane teksty jako korpusy*, artykuł w druku: *Prace Filologiczne* numer LX, http://bc.klf.uw.edu.pl/201/.
3. *The IMPACT project Polish Ground-Truth texts as a DjVu corpus*, artykuł złożony do druku w *Language Resources and Evaluation*, http://bc.klf.uw.edu.pl/296/.
4. *Dygitalizacja kartotek słownikowych*, artykuł złożony do druku w *Pracach Filologicznych*, http://bc.klf.uw.edu.pl/256/.
5. Podręcznik użytkownika przeglądarki materiałów leksykograficznych, http://bc.klf.uw.edu.pl/295/.
6. Strona podręcznikowa programu pdf2djvu.
7. Strona podręcznikowa programu didjvu.
8. Strona podręcznikowa programu djvusmooth.
9. Strony podręcznikowe programów ocrodjvu, hocr2djvused i djvu2hocr.
10. Informacje o popularności programów w dostępnych systemie Debian GNU/Linux:
    — http://qa.debian.org/popcon.php?package=pdf2djvu,
    — http://qa.debian.org/popcon.php?package=didjvu,
    — http://qa.debian.org/popcon.php?package=djvusmooth,
    — http://qa.debian.org/popcon.php?package=ocrodjvu.