Originalveröffentlichung in: Zeller, Jan-Patrick/Menzel, Thomas/Bartels, Hauke (Hrsg.): Einheit(en) in der Vielfalt von Slavistik und Osteuropakunde.
– Berlin: Peter Lang, 2023. S. 309-327.

Peter Meyer (meyer@ids-mannheim.de)

Leibniz-Institut für Deutsche Sprache, Mannheim

Einsatz von EDV und Mikrocomputer in Lehrveranstaltungen zur digitalen Lexikografie

Schlüsselwörter: Computerlexikografie, Internetlexikografie, XML, Lehrprogramm, Datenbanken

1. Einleitung: Ein wenig Geschichte

Gerd Hentschel gehört zu den Pionieren der heutigen Computerlexikografie und der IT-gestützten Korpuserschließung. Eine seiner ersten Zeitschriftenpublikationen, mit dem Titel *Einsatz von EDV und Mikrocomputer in einem lexikographischen Forschungsprojekt zum deutschen Lehnwort im Polnischen* (Hentschel 1983), befasst sich mit der Frage, wie – unter den damaligen technischen Vorzeichen – Forschungs- und Dokumentationsarbeiten zu polnischen Germanismen sinnvoll durch die Verwendung von Computern unterstützt werden können. Die besagten Arbeiten mündeten später in die Online-Publikation des *Wörterbuchs der deutschen Lehnwörter in der polnischen Schrift- und Standardsprache* (WDLP).

Es ist aus heutiger Sicht bemerkenswert, mit welchen Beschränkungen die Arbeit mit dem Computer noch vor 40 Jahren zu kämpfen hatte. Aus gegebenem Anlass sei es gestattet, diesen Punkt etwas ausführlicher zu illustrieren. Bis zum Beginn der 80er-Jahre des 20. Jahrhunderts war der zentrale Großrechner (Mainframe) z. B. eines universitären Rechenzentrums das Standardwerkzeug, mit dem man über – ggf. vom Arbeitsplatz aus per Modem angeschlossene – zeichenbasierte Terminals arbeiten konnte, jedoch nur im Rahmen der jeweils zugeteilten Rechenzeit. Der erwähnte Aufsatz von 1983 steht schon ganz im Zeichen der sich gerade erst auf dem Massenmarkt etablierenden "Kleinrechner" oder "Mikrocomputer", die auch für Privatpersonen oder Universitätsinstitute erschwinglich wurden und die alltägliche Computerarbeit erheblich erleichterten. Auf dem Rechner, der dem in Hentschel (1983) beschriebenen Projekt zur Verfügung stand, werkelte ein Prozessor der – übrigens bis

heute lieferbaren und verwendeten – Z80-Familie, der maximal 64 **Kilobyte**[1] Arbeitsspeicher ansprechen konnte; schon ein einzelnes Foto auf einem heutigen Mobiltelefon benötigt ungefähr das 50fache davon. In diesen Speicher mussten das Betriebssystem, das im Projekt genutzte Datenbankmanagementsystem (dBase II) und natürlich auch die je aktuell bearbeiteten Daten passen.[2]

Die Beschränkungen der damaligen Hardware machten, wie in HENTSCHEL (1983) ausführlich beschrieben wird, einen komplexen arbeitsteiligen Ablauf erforderlich, in dem die Anbindung an einen Großrechner (z. B. als Schnittstelle zum Magnetbandarchiv für die Projektdaten) nach wie vor eine Rolle spielte, sei es über eine Telefonleitung oder durch das Transportieren von Daten auf Disketten, also letztlich *per pedes apostolorum*.

Nicht weniger gravierend dürfte jedoch das Fehlen einer sprach- und schriftsystemübergreifenden Zeichenkodierung gewesen sein, wie sie mit dem heutigen Unicode-Standard selbstverständlich geworden ist. Eher am Rande werden in den frühen Veröffentlichungen von Gerd Hentschel immer wieder die letztlich sehr aufwändigen Maßnahmen erwähnt und in Beispielen gezeigt, die nötig waren, um "Sonderzeichen" mit einem äußerst beschränkten Zeichensatz auszudrücken.

Die rasante Weiterentwicklung der Technologie in den 80er-Jahren spiegelt sich unmittelbar in den weiteren computerlexikografischen Veröffentlichungen von Gerd Hentschel wider. 1983 war der Computer noch lediglich ein unterstützendes Werkzeug im lexikografischen Prozess; er wurde zum einen zur

---

1 Im Aufsatz wird von "Z80A Zentraleinheit und 64 MByte Hauptspeicher" (HENTSCHEL 1983: 15) gesprochen; die angegebene Speichergröße ist jedoch klar ein Versehen. Der Z80 kann mit seiner 16-bit-Adressbreite nur 2^{16} Byte, also 64 KB, Arbeitsspeicher ansprechen; ein größerer Arbeitsspeicher kann nur mit Tricks wie dem sog. Bank Switching genutzt werden. Mehr als 128 KB RAM hatten Z80-basierte Rechner in der Praxis vermutlich selten; 64 MB wären überdies kaum bezahlbar gewesen.

2 Hier ist der Vergleich mit heutigen Verhältnissen instruktiv. Der Verfasser dieser Zeilen sammelte 1983 seine ersten Programmiererfahrungen mit dem in der Bundesrepublik sehr erfolgreichen Heimcomputer Sinclair ZX Spectrum, der ebenfalls mit einem Z80-Prozessor sowie mit maximal 48 KB Arbeitsspeicher bestückt war und mit dem heimischen Röhrenfernseher als Bildschirm und einem seinerzeit handelsüblichen Kassettenrekorder als Datenspeicher betrieben wurde. Den vorliegenden Aufsatz schrieb der Verfasser auf einem Notebook mit über 500.000fach größerem Arbeitsspeicher und einem Mikroprozessor, der das Zweimillionenfache an Transistoren zählt. Zu einer Steigerung der Produktivität in derselben Größenordnung hat dies allerdings nicht geführt.

strukturierten Datenaufnahme genutzt, insbesondere um Exzerptdaten aus den verschiedenen Quellwörterbüchern des WDLP sauber nach Informationstypen getrennt in separaten Dateien ablegen zu können. Zum anderen kam ein Datenbanksystem, seinerzeit dBase II, zur automatisierten Auswertung dieser Daten zum Einsatz. Mithilfe von kleinen Programmen, die man in der dBase-eigenen Programmiersprache formulieren musste, konnten zusammengehörige Informationen aus vielen verschiedenen Exzerptdateien zu einer einzelnen Datei zusammengefügt werden, etwa als Indizes oder für eine Grunddatei, die alle Daten für die Erstellung eines Eintrags enthielt.

In einem computerlexikografischen Lehrbuchkapitel wenige Jahre später (Hentschel/Gregor 1986) ist bereits selbstverständlich vom PC die Rede. Es wird nun ausdrücklich die Möglichkeit genannt, die Artikel eines Wörterbuchs per Programm vollautomatisch aus der Datenbank zu generieren, jedenfalls wenn diese einer einfachen "Slot-and-Filler"-Struktur gehorchen (Hentschel/Gregor 1986: 218). Auch die Idee eines digitalen lexikografischen Endprodukts, das Nutzern neue Wege des Datenzugriffs ermöglicht, ist schon da,[3] auch wenn die immer noch erheblichen Hardware-Beschränkungen weiterhin ein Problem bleiben.[4]

Ein Konferenzbericht aus dem Jahre 1987 erwähnt die "Modellierung von sprachlichem und außersprachlichem Wissen in Lexika der Künstlichen-Intelligenz-Systeme sowie den Wortschatzkomponenten in der maschinellen Übersetzung" (Hentschel 1987: 46) als relevantes zukünftiges Themenfeld.

Mit einem Aufsatz zur Nutzung relationaler Datenbanken für die Wörterbucherstellung ist schließlich bereits in Hentschel (1989) in wesentlichen Punkten die heutige Praxis digitaler Lexikografie erkennbar. Im Artikel wird

---

3 "Die wirklichen Vorteile der Elektronik werden sich erst dann zeigen, wenn die Lexika als elektronische Datenbanken veröffentlicht werden, bei denen der Benutzer dann alle vom Computer gebotenen Möglichkeiten der Informationserschließung zur Verfügung hat. Wegbereitend wird hier die Neubearbeitung des 'Oxford English Dictionary' sein, die unter Nutzung der Laser-Speichertechnologie den Weg in die Lexikographie des 21. Jahrhunderts weist [...]" (Hentschel/Gregor 1986: 219 f.).

4 Dies zeigt sich schön in folgendem Ratschlag an angehende Computerlexikografen: "Glauben Sie nicht zu sehr den Experten – welcher Couleur auch immer –, die grundsätzlich abraten, weil PCs für sinnvolle Arbeiten zu wenig leistungsfähig sind. Wenn Sie alle Möglichkeiten ausnutzen, die heute zur Effizienzsteigerung von PCs zur Verfügung stehen (Arbeitsspeicher-Ausbau, Festplatten, RAM-Disks), können Sie Ihr Lexikon am eigenen Schreibtisch schreiben – sofern es sich nicht gerade um die 'Encyclopaedia Britannica' handelt" (Hentschel/Gregor 1986: 221).

die relationale Datenmodellierung des WDLP anhand von Entity-Relationship-Diagrammen im Detail beschrieben. Um aus einer Vielzahl von Datenbanktabellen eine digitale Druckvorlage zu generieren, wurde per Programm aus der Datenbank eine Quelldatei für das Textsatzsystem TeX[5] erstellt, ein Prozess, der heute in wenigen Minuten erledigt wäre, damals aber für die mehr als 2000 Artikel des WDLP auf einem handelsüblichen PC drei Tage in Anspruch nahm (Hentschel 1989: 65). Auch die Möglichkeit, Nutzern das Wörterbuch als durchsuchbare Datenbank auf CD-ROM zur Verfügung zu stellen, wird erwogen; die Realisierung eines vollständig digitalen lexikografischen Prozesses wird ausdrücklich als anzustrebendes Ideal benannt.[6] Abschließend findet sich sogar schon die Vision, das WDLP an eine umfassendere lexikografische Datenbank zu europäischen Sprachkontakten anzuschließen; damit nimmt Gerd Hentschel in gewisser Weise die weitere Geschichte des WDLP vorweg, das zwar tatsächlich zunächst als TeX-generierte PDF-Ansicht online veröffentlicht wurde, wenig später dann aber mit umfangreichen Recherchemöglichkeiten in das *Lehnwortportal Deutsch* (LWP), ein am Leibniz-Institut für Deutsche Sprache (IDS) betriebenes Online-Informationssystem zu deutschen lexikalischen Entlehnungen in andere Sprachen, integriert und mit anderen Lehnwortressourcen vernetzt wurde.

Damit sind wir in der jüngeren und jüngsten Geschichte der von Gerd Hentschel mit Leidenschaft betriebenen digitalen Lehnwortlexikografie angelangt: Um das WDLP mit den Datenstrukturen des Lehnwortportals kompatibel zu machen, wurden die relationalen Daten in das sowohl menschen- als auch computerlesbare textbasierte Repräsentationsformat XML konvertiert, um das es im Weiteren gehen soll und das in weiten Bereichen der Computerlexikografie derzeit, als internes Datenformat ebenso wie als externes Austauschformat, eine wichtige Rolle spielt. Von Anfang an XML-basiert sind zwei neuere, von der Deutschen Forschungsgemeinschaft geförderte Projekte, die

---

5 Aus dieser Quelldatei erzeugt das TeX-System eine Druckpräsentation, heutzutage typischerweise eine PDF-Datei.

6 "Pod pojęciem łańcucha leksykograficznego rozumiemy różnorodne posunięcia w pracy leksykograficznej – od zbierania materiału przez przedstawienie rezultatów po proces używania słownika. Dwa lata temu Zampolli i Calzolarii (1985: 71 i nast.) opisali w pełni skomputeryzowaną pracę stacji leksykograficznej jako wizję i zadanie na przyszłość. Rozpoczynając pracę inicjatorzy tego projektu mieli na uwadze przedsięwzięcie takie, o jakim piszą wspomniani autorzy. Jednak ograniczenia finansowe i warunki lokalne sprawiają, że dalecy jesteśmy od pierwotnej wizji" (Hentschel 1989: 64).

Gerd Hentschel in Kooperation u. a. mit dem IDS durchführt und die letztlich in zwei weitere größere Lehnwörterbücher münden werden, zum einen ein Wörterbuch der parallelen Lehnwörter aus dem Deutschen im Polnischen und in den ostslavischen Sprachen (vgl. Meyer 2015) und zum anderen ein Wörterbuch zu deutschen Lehnwörtern in polnischen Dialekten (vgl. Meyer/Hentschel 2021).

2. Computerlexikografie in der Lehre: Herausforderungen und Lösungen

Die prominente Rolle informatischer und IT-Konzepte in der heutigen Lexikografie und Linguistik stellt neue Anforderungen an die akademische Lehre. Entsprechend hat auch in der akademischen Lehre von Gerd Hentschel der Einsatz von IT-Technologien z. B. für linguistische und lexikografische Fragestellungen eine erhebliche Rolle gespielt.[7] In einem geisteswissenschaftlichen Umfeld bringt jedoch nicht jeder das Interesse an und die Neigung zu der erforderlichen formalen Denkweise mit; entsprechend klein fällt zwangsläufig die Zahl der Studierenden aus, die sich im Rahmen einer Einzelphilologie erfolgreich und längerfristig mit Themen auseinandersetzen, die mittlerweile als Digital Humanities institutionalisiert sind. Umso wichtiger ist es, auch angesichts der immer weiter steigenden Relevanz entsprechender Fertigkeiten, durch geeignete didaktische Konzepte und Werkzeuge einen niedrigschwelligen Zugang zu schaffen. Im Falle der Computerlexikografie kommt der oben erwähnte Trend zu XML entsprechenden Bemühungen entgegen. Im Vergleich zu XML ist die Arbeit mit relationalen Datenbanken nämlich eine verhältnismäßig hohe Hürde für Einsteiger: Die lexikografischen Daten zu einem Wörterbucheintrag sind in einer solchen Datenbank auf womöglich Dutzende von aufeinander bezogenen Tabellen verteilt. Damit man überhaupt auf konsistente Weise Daten in diese Tabellen eingeben und Informationen strukturiert ausgeben und weiterverarbeiten kann, sind ausreichende Kenntnisse in einer Allzweckprogrammiersprache erforderlich, wie sie sich kaum im Rahmen eines Seminars oder einer Übung auf befriedigendem Niveau vermitteln lassen.[8]

---

7 Hier wäre zuletzt eine zweisemestrige Übung zu "Einsatz und Programmierung mit Visual Fox in korpuslinguistischer Forschung" an der Universität Oldenburg im Sommersemester 2017 und Wintersemester 2017/18 zu nennen, die in das Arbeiten mit einem Datenbanksystem aus dem dBase-Umfeld einführte.

8 Üblicherweise wird heute die eigentliche Datenbankinteraktion mit einer deklarativen Abfragesprache wie SQL durchgeführt, Dateneingabe und -weiterverarbeitung

Programme wie Microsoft Access oder Libreoffice Base können zwar den Umgang mit relationalen Datenbanken erheblich vereinfachen und in einfachen Anwendungsfällen sogar die Programmierung entbehrlich machen, sind aber mit erheblichem Einarbeitungsaufwand verbunden und für computerlexikografische Anwendungen aus verschiedenen Gründen doch nur mit hohem Programmieraufwand nutzbar: Will man komplex formatierte Texte ausgeben oder eine für Lexikografen sinnvolle Dateneingabe schaffen, muss man die auf einen bestimmten Artikel bezogenen Daten aus sämtlichen Tabellen der Datenbank synthetisieren. Der typische Anwendungsfall der genannten Programme ist aber die gezielte Arbeit mit und Auswertung von jeweils nur wenigen miteinander verknüpften Tabellen.

Die erhebliche Kluft zwischen einem abstrakten Tabellensystem und den daraus abzuleitenden Texten von Wörterbuchartikeln ist sicher auch ein Grund für den Erfolg von XML in der Computerlexikografie. Ein als XML-Dokument codierter Wörterbuchartikel ist schlicht ein lesbarer "plain text"; für die Erzeugung, Verarbeitung und Prüfung von XML-Dokumenten gibt es eine breite Palette an Programmen und Programmierwerkzeugen, die offenen, hardwareunabhängigen Standards des World Wide Web Consortium genügen. Hinzu kommt, dass es speziell auch für die XML-Codierung von Wörterbuchdaten umfangreiche und erfolgreiche Standardisierungsbemühen gibt (TEI; TEI Lex-0).

Eine umfassende Einführung in XML-bezogene Technologien ist jedoch immer noch kein realistisches Ziel für eine lexikografische Lehrveranstaltung im Rahmen eines philologischen Studiums. Es ist aber sehr wohl möglich, am Beispiel von Internetwörterbüchern in einem einzelnen Kurs oder Modul einen soliden, mit praktischen Übungen unterfütterten Überblick über die Grundlagen der Arbeit mit XML – von der Datenmodellierung über die Eingabe bis zur Datenauswertung und Erzeugung von Artikelansichten im Browser – zu vermitteln. Ein Verständnis der dabei vermittelten grundsätzlichen theoretischen Konzepte ist auch nützlich, wenn später mit ganz anderen Technologien der Datenmodellierung und -präsentation gearbeitet werden soll.

Die Rahmenbedingungen für solche Lehrveranstaltungen stellen sich heute völlig anders dar als vor 40 Jahren; seinerzeit hätte man im Rahmen

---

aber mit einer prozeduralen Programmiersprache wie Java, Python o. ä. Die verschiedenen Varianten des dBase-Systems, das Gerd Hentschel über Jahrzehnte verwendet hat, verwenden jedoch eine für Datenbankaufgaben und -zugriffe optimierte eigene Sprache, die beide Aspekte vereint.

einer Lehrveranstaltung zu computerunterstützter Lexikografie eher nur eine grundsätzliche Programmier- und Datenbankkompetenz vermitteln können, so wenig waren die technischen und formalen Gegebenheiten konkreter Projekte übertragbar. Von den heute üblichen standardisierten Datenformaten und Zeichenkodierungen, die auf wenigen, weit verbreiteten und weitestgehend kompatiblen Architekturen bearbeitet und gespeichert werden und in Sekundenbruchteilen zwischen beliebigen Rechnern ausgetauscht werden können, konnte man nur träumen. Erst die oben in Ansätzen skizzierte Evolution hinsichtlich Rechen- und Speicherkapazitäten macht es möglich, die Vermittlung der digitalen Lexikografie ebenfalls mit digitalen Hilfsmitteln zu gestalten – womit wir beim eigentlichen Thema dieses Aufsatzes angelangt sind.

Ein grundsätzliches praktisches Problem in einem ersten Kurs zu *XML für Lexikografen* ist die Auswahl der Software, mit der im Unterricht gearbeitet werden soll. In akademischen Kontexten wünscht man sich kostenfreie Lösungen. Angesichts eines typischerweise breiten Spektrums an IT-Ausstattung und -vorkenntnissen bei den Teilnehmern werden gewöhnlich plattformübergreifend einsetzbare Werkzeuge benötigt. Professionelle XML-Editoren, die für mehrere Betriebssysteme verfügbar sind,[9] sind jedoch nicht umsonst zu haben, erfüllen nicht alle Wünsche für die computerlexikografische Lehre – dazu später mehr – und sind mit ihrer großen Palette an hochspezialisierten Funktionen und Eingabehilfen auch nur bedingt für den Anfängerunterricht geeignet. Eine Alternative kann die Verwendung eines generischen, mit "Add-Ons" oder "Plugins" um diverse XML-spezifische Fähigkeiten erweiterten Code-Editors oder einer Programmierumgebung sein.[10] Der Aufwand, der für die Bereitstellung und vor allem für den Support einer solchen manuell konfigurierten Lösung betrieben muss, ist nach den Erfahrungen des Verfassers jedoch schlicht zu hoch: Die Beschäftigung mit den Idiosynkrasien und Tücken der Bedienung des Werkzeugkastens tritt schnell gegenüber den eigentlichen Inhalten in den Vordergrund. Schließlich kann noch die Verwendung eines dedizierten Programms zum Bearbeiten von Wörterbüchern (dictionary writing system) in Erwägung gezogen werden. Hier bietet sich z. B. die

9 Hier ist derzeit vor allem der im akademischen Bereich sehr beliebte Oxygen XML Editor zu nennen.

10 Beispiele sind der nicht mehr weiterentwickelte Editor Atom (atom.io) oder auch Visual Studio Code (code.visualstudio.com); beides sind Open-Source-Projekte. Die Eclipse IDE (www.eclipse.org/ide/) ist ein Beispiel für eine ebenfalls freie, modulare, konfigurierbare Entwicklungsumgebung, die für solche Zwecke genutzt werden kann.

frei verfügbare Online-Plattform Lexonomy (vgl. Měchura 2017) an, auf der Artikel als XML-Dokumente eingegeben werden, während Datenmodellierung und äußere Artikelgestalt auf intuitive Weise über eine grafische Benutzeroberfläche, ohne Eingabe von "Code", spezifiziert werden. Die Gefahr ist aber auch hier, dass statt verallgemeinerbarer Konzepte und Kompetenzen in erster Linie der Umgang mit einem konkreten Programm vermittelt wird.

Gegenstand des vorliegenden Beitrags ist das Programm **x4ml** ("XML/HTML workbench *for m*asterful *l*exicographers"), das vor dem Hintergrund der geschilderten Probleme als Unterrichtssoftware zur Einführung in den Einsatz von XML und HTML in der Internetlexikografie entwickelt wurde – nicht zuletzt auch aufgrund von Erfahrungen, die in der bislang letzten computerlexikografischen Veranstaltung von Gerd Hentschel gewonnen wurden, einer gemeinsam mit dem Verfasser im Frühjahr 2021 abgehaltenen Übung zum Thema *XQuery in lexikographischen Datenbanken zu slavischen Sprachen*.

3. Zurück zu den Anfänge(r)n: Ein Rundgang durch x4ml

Im Folgenden werden der Umgang mit dem Programm x4ml und seine wichtigsten Funktionalitäten in groben Zügen umrissen. Eine detaillierte Anleitung ist Bestandteil des Open-Source-Pakets; hier soll der Schwerpunkt auf dem didaktischen Konzept liegen, das der Entwicklung zugrunde liegt. Ganz nebenbei ergibt sich dabei ein Streifzug durch die mit x4ml abgedeckten Technologien, die hier natürlich nicht im Einzelnen erläutert, aber doch immerhin an einem elementaren Beispiel vorgeführt werden können.[11] Zielgruppe sind Studierende ohne spezielle IT-Vorkenntnisse; elementarste allgemeine Kompetenzen im Umgang mit dem PC werden jedoch vorausgesetzt.

11 Ein knapper Überblick zu den relevanten Themen findet sich in den Lehrbuchkapiteln Meyer/Herold/Lemnitzer (2016) und Herold/Meyer/Müller-Spitzer (2016).

3.1 Grundsätzliches

x4ml wird als eine einzige Java-Binärdatei (JAR-Datei) ausgeliefert und kennt zwei verschiedene Betriebsweisen. Im **Desktopmodus** kann das Programm auf dem eigenen Rechner im Normalfall einfach durch Doppelklick auf die Binärdatei gestartet werden. Einzige Voraussetzung ist eine vorher zu installierende, frei verfügbare Java-Laufzeitumgebung. Im **Servermodus** wird x4ml als Webapplikation mit einigen Kommandozeilenparametern auf einem Server gestartet; Benutzer können sich dann über das Internet anmelden und müssen keine Software auf ihrem eigenen Rechner installieren. In beiden Fällen findet die Nutzerinteraktion in einem Browserfenster statt. Vor der ersten Nutzung muss man ein Projektverzeichnis auswählen, in dem alle zu erstellenden bzw. zu bearbeitenden Dateien liegen (je nach Modus auf dem eigenen Rechner oder auf dem Server). Während der eigentlichen Arbeit mit x4ml entfällt für die Nutzer jegliches Nachdenken über das Speichern oder auch nur Wiederfinden von Daten – alles findet im Projektverzeichnis statt, Änderungen an Daten werden automatisch gespeichert. Schon diese simple Designentscheidung reduziert den im Unterricht anfallenden Betreuungsbedarf erheblich.

Das im nachstehenden Screenshot (Abb. 1) gezeigte x4ml-Browserfenster ist in zwei Hälften geteilt: Links lässt sich jeweils ein XML-Dokument bearbeiten, das im typischen Anwendungsfall die lexikografischen Daten eines Wörterbucheintrags enthält; im rechten Bereich hingegen kann man jeweils eine Datei öffnen und bearbeiten, die auf den Inhalten der aktuell links geöffneten XML-Datei operiert, nämlich entweder ihre Struktur überprüft (DTD, RelaxNG) oder bestimmte Informationen aus ihr extrahiert (XPath, XQuery) oder aber sie in eine Artikelansicht für ein hypothetisches Online-Wörterbuch transformiert (XSLT oder spezielle x4ml-HTML-Templates – s. u.). Beide genannten Bereiche sind wiederum vertikal in einen Editor (oben) und ein Ausgabefeld (unten) gegliedert. Die Ausgabe für den XML-Editor links informiert darüber, ob der eingegebene XML-Code syntaktisch wohlgeformt ist, und falls nicht, wo der Fehler liegt. Im rechten unteren Ausgabebereich werden die Resultate, also der "Output" der Strukturprüfung, Datenextraktion bzw. HTML-Seitengenerierung angezeigt, je nach aktuell rechts geöffneter Datei.

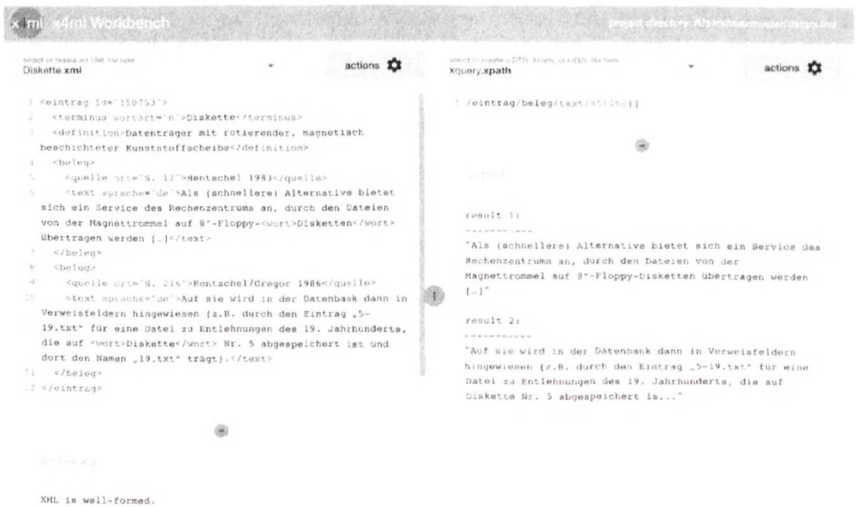

Abb. 1: x4ml-Browserfenster mit einem XML-Dokument auf der linken und einer XPath-Auswertung auf der rechten Seite.

Die beiden Ausgabebereiche werden bei jeder Änderung an den Daten oder an den ausgewählten Dateien instantan aktualisiert; es entfällt die z. B. in einem XML-Editor erforderliche manuelle Konfiguration ("wende Strukturschema X auf XML-Dokument Y an") und das manuelle Anstoßen von Operationen. Nutzer können sich so besser auf Inhalte konzentrieren, nach kurzer Gewöhnung wird das Werkzeug x4ml im Idealfall gewissermaßen unsichtbar.

In den nachfolgenden Abschnitten wird in aller Kürze die Verwendung von x4ml mit den verschiedenen unterstützten Dateiformaten beschrieben, so dass auch technisch nicht vorgebildete Interessierte einen Eindruck von der Arbeit mit dem Programm gewinnen können.

3.2 Kodierung der lexikografischen Daten mit XML

Im einfachsten, an gewöhnliche Printlexikografie angelehnten Fall ist der Wörterbuchartikel die grundlegende Informationseinheit des Wörterbuchs. Mit XML werden die Daten eines Eintrags durch Auszeichnungen in Winkelklammern, den Tags, in hierarchisch strukturierter Weise ausgezeichnet, wie hier

am Beispiel eines hypothetischen Artikels in einem *Wörterbuch zur Terminologie der frühen Computerlexikografie* vorgeführt werden soll:[12]

```
<eintrag id="150753">
  <terminus wortart="n">Diskette</terminus>
  <definition>Datenträger mit rotierender, magnetisch beschichteter Kunststoffscheibe</definition>
  <beleg>
    <quelle ort="S. 12">Hentschel 1983</quelle>
    <text sprache="de">Als (schnellere) Alternative bietet sich ein Service des Rechenzentrums an, durch den Dateien von der Magnettrommel auf 8"-Floppy-<wort>Disketten</wort> übertragen werden […]</text>
  </beleg>
  <beleg>
    <quelle ort="S. 216">Hentschel/Gregor 1986</quelle>
    <text sprache="de">Auf sie wird in der Datenbank dann in Verweisfeldern hingewiesen (z.B. durch den Eintrag „5-19.txt" für eine Datei zu Entlehnungen des 19. Jahrhunderts, die auf <wort>Diskette</wort> Nr. 5 abgespeichert ist und dort den Namen „19.txt" trägt).</text>
  </beleg>
</eintrag>
```

Das XML-Dokument besteht aus Informationsblöcken, den XML-Elementen, die mit einem Starttag wie <definition> eingeleitet und einem namensgleichen Endtag (</definition> mit einem Schrägstrich nach der öffnenden Winkelklammer) abgeschlossen werden. Der frei wählbare Elementname zwischen den Winkelklammern klassifiziert den Elementinhalt. Jedes Element enthält Text und/oder weitere XML-Elemente. In den Starttags können sogenannte Attribute auftauchen, die typischerweise Metainformationen zum Elementinhalt bereitstellen und das syntaktische Format attributname="attributwert" haben.

Der XML-Editor im linken Arbeitsbereich von x4ml verfügt mit guten Gründen nur über einfachste Eingabehilfen, wie automatische Einrückung und Hinzufügung eines passenden Endtags bei Eingabe eines Starttags. Durch Einfärbung wird zudem die syntaktische Struktur des Dokuments deutlich

12 Um die Exposition übersichtlich und verständlich zu halten, wird auf eine maximal granulare Strukturierung – etwa bei den Literaturverweisen – sowie auf Nutzung von Standards bewusst verzichtet. Auch der sog. XML-Prolog wird hier nicht angegeben. Die typografischen Auszeichnungen dienen der besseren Lesbarkeit und sind nicht Bestandteil des XML-Dokuments, das, technisch gesehen, einfach eine Abfolge von Zeichen ist.

gemacht ("syntax highlighting"). Für Anfänger ist es sinnvoll, XML vollständig manuell einzugeben und jeweils unmittelbare Rückmeldung zu erhalten, ob der aktuelle Stand syntaktisch korrekt ist. Erst wenn die Systematik verstanden ist, kann ein professioneller Editor von lästigen Dingen wie der manuellen Eingabe von Tags abstrahieren.

Die beiden Editoren von x4ml verfügen über einen "Aktionen"-Button, mit dem Sonderfunktionen von x4ml ausgeführt werden können. Im Fall von XML ist hier nur eine automatische Einrückung ("pretty printing") zu nennen, die, wie im obigen Beispiel, für mehr Übersicht über die Baumstruktur des Dokuments sorgt.

Wir wenden uns nun den XML-verarbeitenden Datentypen zu, die x4ml im rechten Arbeitsbereich anzeigt.

3.3 Datenmodellierung mit DTD und RelaxNG

Dreh- und Angelpunkt jedes digitalen lexikografischen Projektes ist das Schema, also die formale Beschreibung der Struktur der Daten. x4ml bietet neben den älteren, trotz ihrer Beschränkungen immer noch recht beliebten Dokumenttypdefinitionen (DTD) auch das deutlich flexiblere RelaxNG an, das in der sogenannten kompakten Notation auch viel lesbarer ist. Für unseren Beispielartikel sieht ein RelaxNG-Schema wie folgt aus:

```
element eintrag {
    attribute id {text},
    element terminus {
      attribute wortart {text},
      text
    },
    element definition {text},
    element beleg {
        element quelle {
            attribute ort {text},
            text
        },
        element text {
            attribute sprache {'de' | 'pl' | 'en'},
            (text | element wort {text})+
        }
    }+
}
```

Man erkennt leicht die grundsätzliche Idee: Ein eintrag-Element verfügt über ein textuelles Attribut id und enthält (gewissermaßen als Lemma)

ein terminus-Element, ein definition-Element sowie ein oder mehrere beleg-Elemente. Die Kardinalität "ein oder mehrere" wird durch ein Pluszeichen ausgedrückt.

Ist in x4ml rechts ein Schema-Dokument wie das oben gezeigte geöffnet, so wird automatisch geprüft, ob das links geöffnete XML-Dokument valide ist, also dem Schema gehorcht. Lernende haben es nunmehr mit zwei verschiedenen Typen von Fehlermeldungen zu tun, denn neben fehlgeschlagener Validierung kommen auch syntaktische Fehler im Schema selber in Frage.

Hat man mehrere Einträge – XML-Dokumente – für ein Übungswörterbuch angelegt, kann man sich per Mausklick im Menü des "Aktionen"-Buttons auch alle diese Dokumente zugleich im Ausgabebereich rechts unten validieren lassen. Diese Funktion steht analog auch für alle nachfolgend besprochenen Dokumenttypen zur Verfügung.

3.4 Informationsextraktion mit XPath und XQuery

Mit der Abfragesprache XPath kann man in x4ml leicht die Grundlagen für Funktionalitäten wie Generierung von Lemmalisten und Implementierung von Such- und Filtermöglichkeiten kennenlernen. So liefert der simple Pfadausdruck /eintrag/terminus/text() den Textinhalt im terminus-Element im eintrag-Element – also den idealen Kandidaten für eine Lemmazeichengestaltangabe –, standardmäßig jedoch nur für das links aktuell angezeigte XML-Dokument. Wendet man den Ausdruck per "Aktionen"-Menü separat auf alle XML-Dokumente des eigenen Projektverzeichnisses an, erhält man eine Lemmaliste des Wörterbuchs. x4ml bietet aber zusätzlich einen mächtigen "Datenbankmodus" an, der, vereinfacht gesagt, sämtliche XML-Dokumente im Projektverzeichnis gemeinsam abfragt. Der Ausdruck count(/eintrag/beleg) liefert etwa, angewendet auf die einzelnen XML-Dokumente, für jedes Dokument die Zahl der in ihm enthaltenen beleg-Elemente; im Datenbankmodus wird hingegen die Gesamtzahl der Belege aus allen Dokumenten ermittelt.

Neben XPath unterstützt x4ml auch die viel mächtigere Abfragesprache XQuery in der aktuellen Version 3.1. XQuery enthält XPath als Untermenge.

3.5 Präsentation mit HTML

Mit der Textauszeichnungssprache HTML lassen sich Dokumente beschreiben, die in einem Internetbrowser dargestellt (gerendert) werden können. Die Grundlagen von HTML sind leicht zu erlernen, und es hilft Anfängern, dass die Syntax im Wesentlichen die von XML ist. Im rechten Bereich von x4ml lassen sich HTML-Dokumente erstellen, hinsichtlich der Einrückung formatieren,

nach der aktuellen HTML5-Spezifikation validieren und – nicht zuletzt – auch in einem separaten Browserfenster anzeigen. Studierende können so in einem ersten Schritt einfach eine konkrete Bildschirmansicht eines (fiktiven) Wörterbuchartikels erstellen und dabei natürlich auch Darstellungsvorgaben zu Farben, Typografie usw. mit der "Stylesheet-Sprache" CSS formulieren, Multimediaelemente einbinden usw.

Interessant wird es aber erst, wenn gefragt wird, wie man aus den XML-Artikeldaten automatisch eine HTML-Darstellung ableiten kann. x4ml unterstützt die Standardtechnologie XSLT, um XML in HTML zu transformieren. Der Einstieg in XSLT, das letztlich eine durchaus komplexe funktionale Programmiersprache ist, ist jedoch im Rahmen eines ersten Kurses zur digitalen Lexikografie kaum zu leisten, gerade dann, wenn das Ausgabeformat HTML selber auch erst einmal erlernt werden möchte. Darum bietet x4ml eine niedrigschwellige Alternativlösung an, die die in der Webentwicklung verbreiteten HTML-Template-Sprachen als Vorbild nimmt. Die Idee ist hier, dass an beliebiger Stelle im HTML-Quelltext statt eines statischen Textes ein XPath/XQuery-Ausdruck in {{geschweiften Doppelklammern}} stehen darf. Dieser Ausdruck wird dann auf dem XML-Dokument im linken Arbeitsbereich ausgewertet und das Resultat anstelle des geklammerten XPath-Ausdrucks in den HTML-Code eingefügt. Im rechten unteren Ausgabebereich kann man jederzeit kontrollieren, ob diese Ersetzung das gewünschte Ergebnis geliefert hat.

Zur Verarbeitung von mehrfach vorkommenden Elementen, wie den Belegen in unserem Beispiel, können spezielle vordefinierte HTML-Kommentare verwendet werden. Der HTML-Code zwischen einem Start-Kommentar und einem End-Kommentar wird dann für alle per XPath ausgewählten XML-Elemente nacheinander wiederholt.

Die mit einem solchen Template definierte Artikelansicht kann man sich per "Aktionen"-Button schließlich für alle XML-Dokumente im Projektverzeichnis auch in Gestalt eines kleinen Online-Wörterbuchs anzeigen lassen: Man wählt links aus einer Lemmaliste einen Artikel aus, die template- oder XSLT-generierte Vorschau erscheint rechts. Welches XML-Element als Lemma herangezogen werden soll, kann per XPath-Ausdruck frei festgelegt werden. So resultiert aus nachstehendem HTML-Template die Wörterbuchansicht im darauffolgenden Screenshot (Abb. 2).

```
<!doctype html>
<html lang="de">
  <head>
    <meta charset="utf-8">
    <title>{{//terminus}} - Terminologie der frühen Computerle-
xikographie</title>
  </head>
  <body>
    <h3>{{//terminus}} <i>({{//terminus/@wortart}})</i></h3>
    '{{//definition}}'
    <!-- repeat for all $b in //beleg -->
      <p><b>{{$b/quelle}}, {{$b/quelle/@ort}}</b></p>
      <p style="margin-left:20px;">
          {{$b/text}}
      </p>
    <!-- end repeat for all $b -->
  </body>
</html>
```

Terminologie der frühen Computerlexikographie

Diskette

dBase

Diskette *(n)*

'Datenträger mit rotierender, magnetisch beschichteter Kunststoffscheibe'

Hentschel 1983, S. 12

Als (schnellere) Alternative bietet sich ein Service des Rechenzentrums an, durch den Dateien von der Magnettrommel auf 8"-Floppy-Disketten übertragen werden [...]

Hentschel/Gregor 1986, S. 216

Auf sie wird in der Datenbank dann in Verweisfeldern hingewiesen (z.B. durch den Eintrag „5–19.txt" für eine Datei zu Entlehnungen des 19. Jahrhunderts, die auf Diskette Nr. 5 abgespeichert ist und dort den Namen „19.txt" trägt).

Abb. 2: In x4ml erzeugte Wörterbuchansicht mit ausgewähltem Artikel *Diskette*.

3.6 Ein Lehrprogramm aus der Praxis, für die Praxis

Mit x4ml kann, wie gezeigt wurde, ein vollständiger "Technologiestack" für die Entwicklung eines Internetwörterbuchs von Lernenden auf der Ebene der zugrundeliegenden Computersprachen nachvollzogen werden, ohne dass Programmierkompetenz im engeren Sinne erforderlich ist. Eine vorläufige Fassung des Programms diente erfolgreich als technische Grundlage für das fünftägige

Aufbaumodul *Modeling and representing data in digital lexicography*, das im Rahmen des internationalen lexikografischen Studiengangs *Europäischer Master für Lexikographie* (EMLex; vgl. SCHIERHOLZ 2010) vom 28. März bis 1. April 2022 online an der Universidade do Minho, Portugal, durchgeführt wurde. Aufgrund der Fernunterrichtssituation war eine reibungslose Nutzbarkeit des Programms durch die fast 20 Teilnehmer von großer Wichtigkeit.

Das Programm x4ml wird am 15. Juli 2023 auf der Plattform GitHub als Open-Source-Software freigegeben.[13]

4. Ausblick

XML ist als Datenformat für mäßig komplexe lexikografische Projekte und als standardisiertes Austauschformat hervorragend geeignet und ermöglicht einen sanften Einstieg in die Welt der digitalen Lexikografie. Die streng hierarchische Struktur von XML-Dokumenten kann jedoch nur Baumstrukturen direkt abbilden. Sobald Querbezüge innerhalb oder gar zwischen XML-Dokumenten auftauchen, die diese Baumstruktur durchbrechen, muss man zu Hilfskonstruktionen greifen, indem man beispielsweise XML-Elemente mit IDs versieht, üblicherweise per Attribut. Auf diese IDs kann dann an anderer Stelle verwiesen werden. Ein elementares Beispiel dafür ist eine Wörterbuchartikel-Mikrostruktur, die separat Ausdrucksvarianten und Bedeutungen eines Lexems angibt und zusätzlich spezifizieren soll, welche Varianten in welchen Wortbedeutungen vorkommen. Solche Verweisstrukturen lassen sich schon nicht mehr gut händisch in einem XML-Editor pflegen, sondern verlangen, um Inkonsistenzen und Bearbeitungsfehler zu vermeiden, nach einer programmunterstützten Eingabe. Sehr schnell kommt man dann aber an einen Punkt, an dem man beginnt, die von relationalen Datenbanken gewohnte Funktionalität in XML-Datenbanken per Hand nachzuimplementieren: Was tun etwa, wenn Elemente mit bestimmten IDs vom Bearbeiter entfernt werden, aber Verweise auf die IDs noch vorhanden sind? Suchvorgänge in XML-Datenbanken, die solche ID-Verweise verfolgen, skalieren überdies erheblich schlechter (werden also im Allgemeinen bei größeren Datenmengen langsamer) als das Operieren mit Joins, also Tabellenverknüpfungen, in relationalen Datenbanken.[14] Die ältere, lang bewährte Technologie mit ihrem

13 Die URL des GitHub-Repository lautet: https://github.com/PeterMeyerIDS/x4ml.

14 Alternativ zu einer ID-basierten Lösung lassen sich Sprachelemente, die unter den Kürzeln XLink und XPointer firmieren, einsetzen. Hier treten jedoch vergleichbare Probleme auf.

generischen Datenmodell zeichnet sich insgesamt durch den größeren Anwendungsbereich und i. Allg. höhere Abfragegeschwindigkeit aus.

Die Zeit bleibt nicht stehen. Spätestens dann, wenn eine institutionenübergreifende Verfügbarkeit von miteinander vernetzten lexikografischen, allgemeiner: linguistischen Ressourcen angestrebt wird, kommen mittlerweile Technologien des Semantic Web zum Einsatz und gewinnen unter dem Stichwort Linguistic Linked Open Data (LLOD) auch in der Lexikografie an Bedeutung (vgl. hierzu bereits Chiarcos/McCrae/Cimiano/Fellbaum 2013). Statt Tabellen oder Bäumen sind hier Graphen, also Netzwerke von beliebig miteinander verknüpfbaren Elementen, als Datenstruktur gefragt.[15] Mit dem Einsatz von Graphen entstehen jedoch ganz neue Ebenen der Komplexität für die Bearbeitung und die Abfrage von Daten. Gleichzeitig verschieben sich aber die Grenzen des technisch Möglichen in ungeahntem Tempo.[16] Die weitere Entwicklung der Schnittstelle von Lexikografie und Informationstechnologie, die Gerd Hentschel seit vier Jahrzehnten mitgestaltet, dürfte also spannend bleiben.

Bibliografie

ChatGPT = ChatGPT system (OpenAI, 2021), Feb 9 Version, 2023. Internet: https://chat.openai.com/ [12.2.2023].

Chiarcos, Christian; McCrae, John; Cimiano, Philipp; Fellbaum, Christiane 2013: Towards Open Data for Linguistics: Lexical Linked Data, in: Oltramari, Alessandro; Vossen, Piek; Qin, Lu; Hovy, Eduard (Hgg.), New Trends of Research in Ontologies and Lexical Resources. Heidelberg, S. 7–25.

15 Für Graphen können spezialisierte Graphendatenbank-Managementsysteme verwendet werden. Deren Datenmodell lässt sich aber durchaus auch in relationalen Datenbanksystemen abbilden, wie beispielsweise das Open-Source-Projekt Apache AGE (https://age.apache.org/) zeigt. Auch das eingangs erwähnte Lehnwortportal Deutsch (LWP) verwendet in seiner ursprünglichen Implementierung eine relationale Datenbank, um Graphensuchen in einem Netzwerk von Entlehnungs-, Variations- und Ableitungsrelationen zu ermöglichen; ein neu konzipiertes System mit nativer Graphendatenbank geht voraussichtlich Ende 2023 online (vgl. Meyer 2022). In jedem Fall ein weiteres Beispiel für die Leistungsfähigkeit der relationalen Technologie!

16 Dafür ein aktuelles Beispiel: Das Large Language Model (LLM) hinter dem System ChatGPT ist aus dem Stand in der Lage, die fiktiven Beispieleinträge des vorliegenden Beitrags zu analysieren, auf sinnvolle Weise in XML zu kodieren und dafür ein RelaxNG-Schema anzugeben.

Hentschel, Gerd 1983: Einsatz von EDV und Mikrocomputer in einem lexikographischen Forschungsprojekt zum deutschen Lehnwort im Polnischen, in: Sprache und Datenverarbeitung 1983/1–2, S. 11–15.

Hentschel, Gerd 1987: Computereinsatz in der praktischen Lexikographie: Nachbetrachtungen zum GLDV-Tutorial 'Maschinelle Lexikographie', in: LDV-Forum 5/1, S. 45–46.

Hentschel, Gerd 1989: Słownik zapożyczeń niemieckich w polszczyźnie jako relacyjna baza danych, in: Lubaś, Władysław (Hg.), Wokół słownika współczesnego języka polskiego II: materiały konferencji w Paszkówce, 26-28 XI 1986 r. Wrocław, S. 57–67.

Hentschel, Gerd; Gregor, Bernd 1986: Lexikographie, in: Gregor, Bernd; Krifka, Manfred (Hgg.), Computerfibel für die Geisteswissenschaften. München, S. 212–221.

Herold, Axel; Meyer, Peter; Müller-Spitzer, Carolin 2016: Datenmodellierung, in: Klosa, Annette; Müller-Spitzer, Carolin (Hgg.), Internetlexikografie. Ein Kompendium. Unter Mitarbeit von Martin Loder. Berlin–Boston, S. 111–152.

LWP = Lehnwortportal Deutsch. Leibniz-Institut für Deutsche Sprache, Mannheim. Internet: http://lwp.ids-mannheim.de [31.12.2022].

Měchura, Michal 2017: Introducing Lexonomy: An Open-Source Dictionary Writing and Publishing System, in: Kosem, Iztok; Tiberius, Carole; Jakubíček, Miloš; Kallas, Jelena; Krek, Simon; Baisa, Vít (Hgg.), Electronic Lexicography in the 21st Century: Proceedings of eLex 2017 Conference. Leiden, S. 662–679.

Meyer, Peter 2015: Aligning Word Senses and more: Tools for Creating Interlinked Resources in Historical Loanword Lexicography, in: Kallas, Jelena; Kosem, Iztok; Krek, Simon (Hgg.), Electronic lexicography in the 21st century: linking lexical data in the digital age. Proceedings of the eLex 2015 conference, 11–13 August 2015, Herstmonceux Castle, United Kingdom. Ljubljana–Brighton, S. 198–210.

Meyer, Peter 2022: Lehnwortportal Deutsch: A New Architecture for Resources on Lexical Borrowings, in: Klosa-Kückelhaus, Annette; Engelberg, Stefan; Möhrs, Christine; Storjohann, Petra (Hgg.): Dictionaries and Society. Proceedings of the XX EURALEX International Congress, 12–16 July 2022, Mannheim, Germany. Mannheim, S. 578–583.

Meyer, Peter; Hentschel, Gerd 2021: Charting a Landscape of Loans. An e-Lexicographical Project on German Lexical Borrowings in Polish Dialects, in: Gavriilidou, Zoe; Mitits, Lydia; Kiosses, Spyros (Hgg.), Proceedings of XIX EURALEX Congress: Lexicography for Inclusion, Vol. II. Komotini, S. 615–621.

Meyer, Peter; Herold, Axel; Lemnitzer, Lothar 2016: Technische Rahmenbedingungen der Internetlexikografie, in: Klosa, Annette; Müller-Spitzer, Carolin (Hgg.), Internetlexikografie. Ein Kompendium. Unter Mitarbeit von Martin Loder. Berlin–Boston, S. 1–30.

Schierholz, Stefan J. 2010: EMLex: Europäischer Master für Lexikographie – European Master in Lexicography, in: Lexicographica 26, S. 343–350.

TEI = TEI Consortium, eds.: TEI P5: Guidelines for Electronic Text Encoding and Interchange. [Version 4.5.0]. [Last updated on 25th October 2022, revision 3e98e619e]. Internet: https://tei-c.org/Vault/P5/4.5.0/doc/tei-p5-doc/en/html/ [31.12.2022].

TEI Lex-0 = Tasovac, Toma; Romary, Laurent; Bański, Piotr; Bowers, Jack; de Does, Jesse; Depuydt, Katrien; Erjavec, Tomaž; Geyken, Alexander; Herold, Axel; Hildenbrandt, Vera; Khemakhem, Mohamed; Lehečka, Boris; Petrović, Snežana; Salgado, Ana; Witt, Andreas 2018: TEI Lex-0: A Baseline Encoding for Lexicographic Data. Version 0.9.1. DARIAH Working Group on Lexical Resources. Internet: https://dariah-eric.github.io/lexicalresources/pages/TEILex0/TEILex0.html [31.12.2022].

WDLP = Vincenz, Andrzej de; Hentschel, Gerd 2010: Wörterbuch der deutschen Lehnwörter in der polnischen Schrift- und Standardsprache. Von den Anfängen des polnischen Schrifttums bis in die Mitte des 20. Jahrhunderts, Oldenburg. Internet: http://diglib.bis.uni-oldenburg.de/bis-verlag/wdlp [31.12.2022].