UNIVERSITÄT REGENSBURG

Annotationswerkzeuge für diachrone Korpora

Klassifikation und Evaluation von Annotationswerkzeugen

Magisterarbeit im Fach Informationswissenschaft
Institut für Medien-, Informations- und Kulturwissenschaft

von:	Manuel Burghardt
Adresse:	Straubinger Str. 10 94360 Mitterfels
Matrikelnummer:	116 303 0
Erstgutachter:	Prof. Dr. Christian Wolff
Zweitgutachter:	Prof. Dr. Rainer Hammwöhner
Laufendes Semester:	Sommersemester 2008
Abgabedatum:	01. Juli 2008

Zusammenfassung

Diese Arbeit untersucht verschiedene computergestützte Werkzeuge zur Annotation diachroner Korpora und bewertet deren Eignung für das Projekt *Diachrone Syntax Deutsch* (DiSynDe). Anfangs wird überblicksartig das DiSynDe Annotationsszenario samt der besonderen Merkmale historisch-diachroner Korpora dargestellt, um grundlegende Termini und Anforderungen vorweg zu nehmen. Es folgt die Analyse eines breiten Spektrums verfügbarer Annotationswerkzeuge sowie eine generelle Klassifikation der unterschiedlichen Werkzeuge nach ausgewählten Unterscheidungskriterien. Anhand dieser Klassifikation sollen dann aus mehr als 50 Programmen geeignete Werkzeuge zur Annotation diachroner Korpora ermittelt und anschließend evaluiert werden. Dabei werden in einem eigenen Kapitel gängige Standards und Methoden zur Evaluation von Software vorgestellt. Darauf aufbauend wird eine geeignete Evaluation für Annotationswerkzeuge entworfen und durchgeführt. Die Ergebnisse der Evaluation werden nach der Durchführung ausgewertet und interpretiert, um Aussagen über die Qualität der einzelnen Werkzeuge machen zu können. Der Schlussteil beinhaltet sowohl eine Diskussion über individuelle Stärken und Schwächen des am besten bewerteten Tools sowie einen Ausblick zu offenen Fragen und ungelösten Problemen im weiteren Projektumfeld von DiSynDe.

Abstract

This work analyzes a range of computer-aided tools for the annotation of diachronic corpora and evaluates their adequacy for the *Diachrone Syntax Deutsch* (DiSynDe) project. At the beginning a survey of the DiSynDe annotation scenario will be given, including the description of distinct features of historic-diachronic corpora, in order to introduce some basic terminology and requirements. Subsequently there will be an analysis of the wide range of available annotation tools, as well as a general classification of the various tools by selected discriminating criteria. On the basis of this classification more than 50 applications are tested for their adequacy and then evaluated in detail if found appropriate. Another chapter is about common standards and current methods for the evaluation of software in general, and about how an appropriate evaluation for annotation tools can be designed and implemented. The results of the evaluation are assessed and interpreted in order to derive conclusions about the quality of each tool that was tested. The final part includes a discussion about particularly strong and weak points of the most adequate tool, as well as an outlook on unanswered questions and unsolved problems in the wider context of DiSynDe.

Inhaltsverzeichnis

1 Einleitung

> Diachrone Arbeiten im Rahmen der seit den 60er Jahren in der Syntaxtheorie einflussreichen Generativen Grammatik [,] wie sie für die Geschichte des Englischen vorliegen, sind […] für das Deutsche kaum zu verzeichnen. In den letzten Jahren bezeugen jedoch mehrere Dissertationen und andere Studien auch hier ein verstärktes theorieorientiertes Interesse. (NETZWERK HISTORISCHE SYNTAX, 2008)

1.1 Themenstellung und Zielsetzung

Das interdisziplinäre Forschungsprojekt *Diachrone Syntax Deutsch* (DiSynDe) verfolgt das Ziel ein diachrones Korpus historischer Texte des Deutschen aufzubauen. Um die Wiederverwendbarkeit und Portierbarkeit der syntaktisch annotierten Daten zu gewährleisten, wird dabei so weit wie möglich auf bestehende Annotationsstandards und -werkzeuge zurückgegriffen. Vor diesem Hintergrund werden im Folgenden entsprechende Annotationswerkzeuge zur Arbeitserleichterung und Qualitätssicherung untersucht, während sich eine parallel entstehende Magisterarbeit unter dem Titel „Informationsstrukturierung für die syntaktische Annotation eines diachronen Korpus des Deutschen“ (vgl. HEILEMANN 2008) mit geeigneten Standards für Annotationsschemata befasst.

1.2 Aufbau der Arbeit

Diese Arbeit unterstützt das Projekt DiSynDe durch die Evaluation von geeigneten Annotationswerkzeugen. Ausgangspunkt ist dabei die genaue Analyse der Projektziele und des damit einhergehenden Annotationsszenarios, welches vor allem durch die historische Textbasis, das diachrone Erkenntnisinteresse und die Heterogenität der einzelnen Annotationen gekennzeichnet ist.

Im darauf folgenden Kapitel wird das breite Spektrum bestehender Annotationswerkzeuge skizziert. Dabei werden Annotationsprogramme anhand ausgewählter Merkmale klassifiziert und in einer entsprechenden Ergebnismatrix dargeboten. Im Abgleich mit grundlegenden Anforderungen an ein Werkzeug zur Annotation dia-

chroner Korpora werden schließlich potentiell geeignete Kandidaten für das DiSynDe Projekt ausgewählt.

Es folgt ein kurzes Kapitel zu Standardisierungsbestrebungen im Bereich der Software-Evaluation, insbesondere der Evaluation von Annotationswerkzeugen. Hier werden vor allem Vorschläge und Normen der ISO (*International Organization for Standardization*) und der EAGLES (*Expert Advisory Group for Language Engineering Standards*) Projektgruppe vorgestellt.

Darauf aufbauend erfolgt die Planung und Durchführung einer umfangreichen Evaluation zur qualitativen Bewertung von Annotationswerkzeugen. Hierzu werden anfangs allgemeine Anforderungen an ein Annotationswerkzeug für diachrone Korpora beschrieben, um im nächsten Schritt ein feingliedriges Qualitätsmodell zu erstellen, welches auf den ISO Qualitätskriterien Funktionalität und Benutzbarkeit basiert. Mit dem Festlegen passender Werteskalen und Metriken sowie der Formulierung von Bewertungs- und Gewichtungsformalismen wird ein standardisiertes Evaluationsinstrumentarium zur Bewertung von Annotationssoftware geschaffen. Die Durchführung der Evaluation ermöglicht sowohl detaillierte als auch generalisierende Aussagen zur Eignung verschiedener Annotationswerkzeuge für ein diachrones Syntaxannotationsprojekt.

Im Schlussteil werden schließlich die individuellen Stärken und Schwächen des am besten geeigneten Tools in Hinblick auf die tatsächlichen Auswirkungen für das DiSynDe Annotationsszenario diskutiert. Ein Ausblick zeigt offene Fragen und Problembereiche im weiteren Projektkontext von DiSynDe auf.

2 Projektumfeld einer diachronen Syntaxanalyse

Seit dem Mittelalter sind Texte in deutscher Sprache handschriftlich oder später auch in Drucken überliefert und mittlerweile größtenteils in Editionen als Replike verfügbar. Digitalisate historischer Texte des Deutschen sind ebenfalls in hohem Maße vorhanden, allerdings oft nur unzureichend oder uneinheitlich annotiert und technisch aufbereitet. In einer vergleichenden Analyse bestehender historischer und digitaler Korpora fassen KROYMANN et al. (2004, S. 41) zusammen, dass die historischen Sprachstufen des Deutschen unterschiedlich gut abgedeckt sind. Prinzipiell ist zwar viel Material digital vorhanden, die Unterschiede zwischen den verschiedenen Texten sind jedoch meist noch so groß, dass diachrone Studien kaum möglich sind. Existierende Standardisierungsbestrebungen wie etwa die *Text Encoding Initiative* (TEI) oder der *Corpus Encoding Standard* (CES) sowie verschiedene frei verfügbare Annotationswerkzeuge leisten einen wichtigen Beitrag auf dem Weg hin zu einem einheitlichen Annotationsformat. Die Projektgruppe DiSynDe hat es sich zur Aufgabe gemacht, aufbauend auf existierenden Standards und unter Verwendung etablierter Annotationswerkzeuge, ein diachrones Korpus aufzubauen, welches dazu geeignet ist bislang offene Fragen im Bereich der historischen Syntax zu untersuchen und zu beantworten.

2.1 Historische und diachrone Textkorpora

Wie hat sich die Verbstellung in Hauptsätzen, denen ein Konditionalsatz vorausgeht vom Althochdeutschen zum Mittelhochdeutschen entwickelt? Welche formalen Konditionalsatztypen werden in Gefügen mit mehr als einem Konditionalsatz in Rechtsprosa und Fachliteratur des 12. Jahrhunderts verwendet? Welche Konstituentenabfolgen in Nominalgruppen gelten innerhalb von verbalen Klammern in der Textsorte *Unterhaltungsprosa* im Zeitraum 1500 bis 1650?

An Fragen wie diesen wird deutlich wozu historisch-diachrone Korpora mit entsprechenden Abfragetools eigentlich benötigt werden.

2.1.1 Merkmale historisch-diachroner Korpora

Der Begriff historisch-diachroner Korpora wird im Laufe dieser Arbeit immer wieder Verwendung finden und soll deshalb kurz erläutert werden.

Historische Korpora bezeichnen typischerweise Zusammenstellungen von Texten älterer, also historischer Sprachstufen, wie etwa des Althochdeutschen oder des Mittelenglischen. Unter Diachronie ist die historische Betrachtung von Sprache zu verstehen, d. h. vor allem die Untersuchung von Wandelprozessen auf den verschiedenen sprachlichen Ebenen, wie etwa Lautwandel, morphologischer und syntaktischer Wandel, lexikalischer und semantischer Wandel etc. STRÖMSDÖRFER und VENNEMANN (1995, S. 1131) sprechen von „sprachlichen Strukturen [die] sich nicht abrupt, sondern graduell mit fließenden Übergängen verändern" und fordert für die diachrone Sprachbetrachtung deshalb „eine Änderung der Auffassung des Sprachwandels als eine Abfolge von Sprachstadien zugunsten einer Auffassung des Sprachwandels als Prozeß". Diachrone Textkorpora kennzeichnen sich deshalb durch ihre Zusammensetzung aus Dokumenten unterschiedlicher Zeit- bzw. Sprachstufen. Besteht ein Korpus aus historischen Originaltexten, deren linguistische Besonderheiten diachron, also kontinuierlich entlang einer Zeitachse, mit allen eventuellen Sprachwandelprozessen annotiert wurden, so wird in Hinsicht auf die Kontinuität späterer Abfragen zwangsläufig ein gewisser Widerspruch deutlich.

Um diachrone Untersuchungen an einem Textkorpus durchführen zu können, sollten sich die Korpustexte im günstigsten Fall nur im Parameter Zeit unterscheiden, da wie der Terminus *Diachronie* wörtlich besagt, bei diesem methodischen Ansatz ein bestimmter Aspekt in unterschiedlichen Ausprägungen *durch die Zeit*[1] hinweg untersucht und verglichen wird. Dabei sind vor allem Annotationen zu älteren Sprachstufen durch ein hohes Maß an Diskontinuität gekennzeichnet. So kann beispielsweise der Aspekt *Textsorte* nicht immer kontinuierlich über mehrere Sprachstufen hinweg verglichen werden, da einige Textsorten, welche es im Frühneuhochdeutschen bereits gibt, im Althochdeutschen schlicht noch nicht existent waren. KROYMANN et al. sprechen deshalb auch von „Kontinuität in Teilbereichen" (2004, S. 4).

[1] Diachronie, von griechisch *dia* (durch) und *chronos* (Zeit) (vgl. KORTMANN 1999, S. 9)

Eine weitere Besonderheit historisch-diachroner Korpora zeigt sich in der Zusammensetzung der Textsammlungen und ist der historischen Datengrundlage geschuldet. Da die Verfügbarkeit historischer Texte in den meisten Fällen überschaubar und die sprachliche Entwicklung in diesen Bereichen in der Regel längst abgeschlossen ist, sind historische Korpora praktisch immer als Referenzkorpora realisiert. Dieser Korpustyp ist im Gegensatz zu so genannten Monitorkorpora durch eine feste Größe und Zusammensetzung gekennzeichnet, wohingegen Monitorkorpora nach einem bestimmten vordefinierten Schema ständig weiter wachsen. Werden gemeinhin Textkorpora, Sprachkorpora und multimodale Korpora unterschieden, so ist für Texte historischer Sprachstufen nur der Terminus Textkorpus relevant, da in diesem Bereich keinerlei Originaldaten zu gesprochener Sprache oder Ähnlichem vorhanden sind, sondern ausschließlich auf digitalisierte Textdokumente zurückgegriffen werden kann. Die Begriffe Korpus und Textkorpus werden deshalb im Zusammenhang mit historischen Dokumenten fortan synonym verwendet.

2.1.2 Erstellung historisch-diachroner Korpora

Die Erstellung eines diachronen Korpus historischer deutscher Texte erfordert im Wesentlichen dieselben Arbeitsschritte wie die allgemeine Korpusproduktion. Jedoch müssen stets formale und orthographische Besonderheiten historischer Texte berücksichtigt werden, welche bei fast jedem Fertigungsschritt auf dem Weg zum diachronen Korpus zu Tage treten.

Schon bei der Digitalisierung der Texte ist die Verwendung handschriftlicher Originaltexte oder gedruckter Editionen als Datengrundlage zu klären. Dabei ist sowohl der Grad der Diplomatizität, also der Urkundentreue als auch die Berücksichtigung etwaiger paläographischer Phänomene, also die schriftgeschichtliche Entwicklung der Texte, von Bedeutung. Hinzu kommt, dass viele historische Texte häufig spezifische Sonderzeichen enthalten, welche es in ein standardisiertes Zeichenkodierungsformat, wie etwa Unicode[2], zu überführen gilt. Besteht die Möglichkeit auf digitalisierte Texte aus anderen Projekten zurückgreifen zu können, tritt häufig das Problem uneinheitlicher Formatierungen auf. Initiativen wie TEI und

[2] Unicode ist ein internationaler Standard, in dem langfristig für jedes sinntragende Zeichen bzw. Textelement aller bekannten Schriftkulturen und Zeichensysteme ein digitaler Code festgelegt wird.

CES leisten auf diesem Gebiet wichtige Standardisierungsarbeit und sollten bei der Formatierung von Primärtexten zumindest als Ausgangspunkt herangezogen werden, um die Wiederverwendbarkeit der Digitalisate zu gewährleisten.

Bevor mit der Annotation der Texte begonnen werden kann, müssen diese der Prozedur der Tokenisierung unterzogen werden. Hierbei wird der Text unter Zuhilfenahme eines Algorithmus, welcher im einfachsten Fall anhand von Leer- und Satzzeichen die Grenzen zwischen den einzelnen Textwörtern erkennt, in so genannte Token[3] zerteilt.

An dieser Stelle kann nun der Annotator nahtlos anknüpfen und mit der Auszeichnung (engl.: *markup*) der sprachlichen Daten beginnen. Unter Annotation ist dabei im weitesten Sinne die Beifügung von Metadaten zu einer definierten Annotationsbasis (vgl. FOGLI et al. 2004, S. 98), also in diesem Fall der digitalisierten Texte, zu verstehen. LÜDELING et al. (2004, S. 7) wie auch KROYMANN et al. (2004, S. 4) unterscheiden zudem Header-Annotation, positionelle Annotation und strukturelle Annotation bei der Auszeichnung von Texten.

Header-Annotationen enthalten grundlegende bibliographische Informationen zum gesamten Text, wie etwa Autor, Textsorte, technische Vorverarbeitung und Ähnliches. Unter struktureller Annotation wird die Auszeichnung von physischer und logischer Textstruktur zusammengefasst, wie etwa die Annotation von Zeilennummern und Seitenzahlen, oder die Markierung von Kapiteln und Absätzen. Positionelle Annotation beschreibt hingegen die inhaltliche Auszeichnung der einzelnen Token anhand eines vorher definierten Annotationsschemas.

Wurden die Auszeichnungen in der Vergangenheit als so genannte Inline[4] Annotationen meist direkt in den Originaltext geschrieben, so scheinen sich Stand-off Modelle (vgl. THOMPSON & MCKELVIE 1997; RODRÍGUEZ et al. 2007) als Defacto-Standard immer mehr durchzusetzen. Bei der Stand-off Annotation wird eine strikte logische Trennung von Primärdaten und eigentlicher Annotation gefordert. Mithilfe der XML Technologien XLink und XPointer (vgl. DIPPER 2005, S. 41) ist es möglich Referenzen vom Originaltext auf separate Dateien zu setzen. Diese Da-

[3] Das Gegenstück zum Token ist der Type. Im Satz „Der Mensch ist dem Menschen ein Wolf" unterscheidet man beispielsweise für das Wort „Mensch" zwei Token aber nur einen Type, also zwei Vorkommen (Token) eines bestimmten Worttyps (Type).

[4] Bei der Inline Annotation werden Ausgangstext und Annotation miteinander verknüpft und in ein und derselben Datei gespeichert.

teien enthalten dann die eigentliche Annotation und sind mit der Annotationsbasis über einen virtuellen Zeiger verbunden. Auf diese Weise bleiben zum einen die Primärdaten im Originalzustand erhalten, zum anderen können durch die Trennung von Textbasis und Annotationsdatei beliebig viele Annotationsebenen hinzugefügt, und sogar überlappende Hierarchien modelliert werden. Die Idee für diese Art der Annotation enstammt ursprünglich aus dem Bereich multimodaler Korpora: „Ein Sprachsignal mit seiner Transkription und eventuell noch weiteren Informationen aus anderen Modi wie z. B. Mimik, Gestik und Prosodie […] bildet den Zeitstrahl (engl.: *timeline*), auf den sich alle weiteren Ebenen beziehen" (LÜDELING et al. 2004, S. 124). Die Timeline-Metapher wird in reinen Textkorpora häufig durch fortlaufende ID-Tags der einzelnen Annotationseinheiten realisiert, welche dann durch Referenz auf die jeweilige ID auf beliebig vielen Ebenen mit linguistischen Informationen annotiert werden können. Dabei entspricht bei rigoroser Umsetzung der Stand-off Idee jede Annotationsebene einer eigenen Datei, welche wiederum beliebig miteinander verknüpft werden können. Von Stand-off Annotation ist aber auch häufig dann die Rede, wenn Originaltext und Markup in derselben Datei gespeichert sind, aber dennoch durch ein Referenzierungssystem logisch voneinander getrennt sind. [5]

Die eigentliche Annotation kann entweder automatisch nach einem regelbasierten oder probabilistischen Modell, halbautomatisch oder manuell erfolgen. Da regelbasierte Algorithmen auf einem soliden linguistischen Modell, und probabilistische Ansätze auf einem möglichst großen und adäquaten Trainigskorpus aufsetzen, scheint für die Annotation historischer Texte eine semiautomatische Herangehensweise am praktikabelsten, da gänzlich manuelle Annotation zum einen äußerst zeitaufwendig ist, und zum anderen bei mehreren Annotatoren leicht qualitative Inkonsistenzen entstehen können. Brauchbare linguistische Modelle für eine teilweise Automatisierung existieren zumindest für Teilbereiche historischer Sprachstufen. Die intelligente Nachkorrektur, bei der insbesondere der Kontext mit berücksichtigt werden muss, wird durch die automatisierte Vorarbeit deutlich erleichtert, entfallen

[5] Dies ist beispielsweise bei GATE der Fall. Die Frage inwiefern bei GATE das Stand-off Konzept berücksichtigt ist beantwortet die Entwicklerin Diana Maynard in der offiziellen Mailinglist mit folgender Erläuterung: „We call it standoff markup as the original text is not modified as such, e. g. you can save the annotated text (xml file) separately from the original text. The annotations are indeed stored separately from the text, they're just not stored in a different file."

doch z. B. immer wiederkehrende Routinearbeiten, weil sie bereits vorher von einem regelbasierten Algorithmus abgearbeitet wurden. Mit steigender Zahl vorannotierter Texte wird auch die Verwendung des wahrscheinlichkeitsbasierten Modells immer interessanter. Viele computergestützte Annotationswerkzeuge implementieren solche Algorithmen zur semi-automatischen Auszeichnung von Texten und unterstützen den Annotator zusätzlich beim manuellen Textmarkup.

2.2 DiSynDe

Die Zahl bereits bestehender Annotationswerkzeuge ist immens. Dabei sind diese meist entweder auf bestimmte linguistische Fragestellungen zugeschnitten[6], oder im Sinne eines Frameworks oder Ähnlichem betont offen und allgemein gehalten. Deshalb soll an dieser Stelle kurz das Annotationsszenario für das diachrone Syntaxprojekt DiSynDe skizziert werden, um im Anschluss Anforderungskriterien an ein Annotationswerkzeug zu formulieren, welches sich für die Projektziele und die speziellen Bedürfnissen der Anwender besonders gut eignet.

2.2.1 Projektziele und Organisation

DiSynDe will zunächst einmal Formalismen und Techniken zur kontinuierlichen Annotation historischer Texte des Deutschen entlang einer Zeitachse erarbeiten, um die so erschlossenen Texte dann später diachronen Fragestellungen zu unterziehen. Auf dieser Grundlage soll nach und nach eine moderne deutsche Syntax abgeleitet werden. Die Motivation und Notwendigkeit eines solchen Unterfangens verdeutlicht SCHMID[7]:

> Die letzte große Gesamtdarstellung der historischen Syntax des Deutschen, nämlich die von Otto Behaghel, ist in vier Bänden zwischen 1923 und 1932 erschienen. […] Das will nicht besagen, dass zwischendurch auf dem Gebiete der historischen Syntax nichts geschehen wäre. Genannt seien nur die „Kurze deutsche Syntax auf historischer Grundlage“ von Ingerid Dal oder die Darstellungen von Robert Peter Ebert […], die sich entweder auf die Diachronie bestimmter Einzelaspekte konzentrieren […] oder auf syntaktische Gegebenheiten

[6] So gibt es einige Werkzeuge, die bestimmte Ebenen der Annotation fest vorgeben, und den Einsatz des Tools damit auf einen bestimmten Themenbereich einschränken.
[7] Prof. Hans Ulrich Schmid ist einer der Initiatoren des Projekts Diachrone Syntax Deutsch

> einzelner Sprachstufen wie die Syntaxteile in den Niemeyer-Grammatiken. […] Es wird also kaum zu leugnen sein, dass eine umfassende Darstellung der historischen Syntax des Deutschen, so etwas wie ein „neuer Behaghel" ein Forschungsdesiderat ist. (2007, S. 51)

Zum momentanen Zeitpunkt befindet sich DiSynDe noch in einer inoffiziellen Pilotphase, deren Ziele vor allem in der Erstellung grundlegender Annotationsschemata und der Untersuchung geeigneter Annotationswerkzeuge bestehen. Dabei ist das interdisziplinäre Unterfangen aus Philologen, Korpus- und Computerlinguisten vorerst in insgesamt fünf Arbeitsgruppen organisiert, von denen sich vier Gruppen mit unterschiedlichen Analyseebenen historischer Texte beschäftigen und ein fünftes Team mit der technischen Umsetzung des Annotationsvorhabens betraut ist. Die beiden Magisterarbeiten über diachrone Annotationstools und entsprechend geeignete Annotationsschemata fallen in den Zuständigkeitsbereich der Gruppe *Annotation und Technik*.

Arbeitsgruppe	Aufgabenbereich
1. Gruppe	Textebene/Wortarten
2. Gruppe	Ebene komplexer Satz
3. Gruppe	Ebene einfacher Satz
4. Gruppe	Ebene der Nominalgruppe
5. Gruppe	Annotation und Technik

Tabelle 1: Arbeitsaufteilung nach Gruppen im Projekt DiSynDe

Die oberste Analyseebene bei den Annotationsgruppen stellt die *Textebene*, als größte syntaktisch relevante und strukturbedingte Entität, dar. Es folgen die Analyseebenen *komplexer Satz*, *einfacher Satz* und *Nominalgruppe*. Die Ebene der *Wortarten* soll im Rahmen des Pilotantrags zunächst provisorisch von der Gruppe *Textebene* mit übernommen werden, da später möglicherweise auf entsprechend vorannotierte Texte aus dem Projekt Diachrone Syntax Deutsch (DDD) zurückgegriffen werden kann.

Obwohl das Annotationsvorhaben als hierarchisches Abarbeitungsschema entweder von der kleinsten zur größten Einheit oder umgekehrt gesehen werden kann, wurde zumindest für die Dauer der Pilotphase eine dynamische Analyse, ohne allzu starre Abgrenzung der einzelnen Bereiche, vereinbart. Vielmehr sollen gruppenübergreifende Untersuchungen und zirkulierende Annotationen zugunsten eines hierar-

chischen Ansatzes wertvolle Erkenntnisse über eventuelle Problembereiche der syntaktischen Analyse bringen.

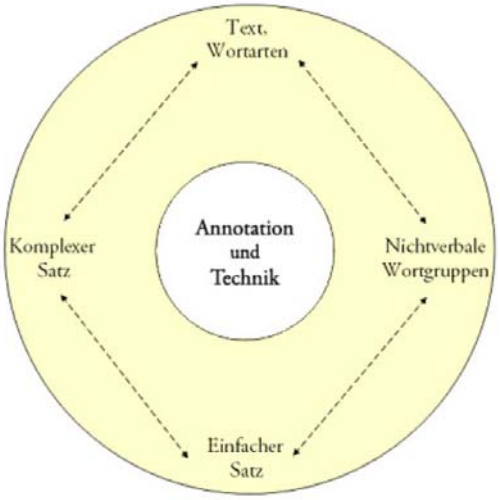

Abbildung 1: Der Annotationsprozess als zirkulierender Arbeitsfluss

Annotationsebene Text Beim Bestreben eine möglichst umfassende historische Syntax des Deutschen zu erstellen, bildet die Stufung *Text* die größte syntaktische Einheit der insgesamt vier Annotationsebenen. Dabei soll die Annotation auf dieser Stufung einerseits Informationen zum Text als Ganzes, andererseits satzinterne sowie satzübergreifende, textgrammatische Merkmale umfassen. Zusätzlich sollen während der Pilotphase in diesem Annotationsmodul Wortarten im Sinne eines *Part of Speech Taggings* (POS-Tagging) erfasst werden und als Grundlage für die Ebene der nicht-verbalen Wortgruppen dienen. Leistet die Erfassung von Textsorte und -funktion wichtige Vorarbeit für alle anderen Annotationsebenen, so kann es im Bereich der textgrammatischen Detailanalyse leicht zu Überschneidungen mit den anderen Ebenen kommen. Ein Ziel der Pilotphase von DiSynDe ist es solche Überschneidungen am Beispiel laufender Annotationen zu dokumentieren, um später Zuständigkeitsbereiche eindeutig abstecken zu können oder falls nötig bestimmte Phänomene an neuralgischen Stellen bewusst doppelt zu annotieren. Mögliche Beschreibungsziele auf dieser Auszeichnungsebene könnten beispielsweise eine Darstellung der Textgrammatik, der Interpunktion und der Textgliederung sowie eine Charakterisierung der Begleittexte (Paratexte), eine Bestimmung der Textfunktion

oder die Beschreibung der Textsorte umfassen. Die nicht immer klar definierbaren Grenzen zu anderen Analyse-Ebenen, insbesondere der Ebene *komplexer Satz*, sowie Schwierigkeiten bei der eindeutigen Bestimmung von Satzgrenzen in Texten älterer Sprachepochen des Deutschen, bezeichnen grundlegende Probleme dieser Arbeitsgruppe.

Annotationsebene komplexer Satz Die Ebene *komplexer Satz* fungiert als Bindeglied zwischen den Einheiten *Text* und *einfacher Satz*. Komplexe Sätze umfassen syntaktische Einheiten aus Haupt- und Nebensatz, Infinitiv- und Partizipialkonstruktionen. Die Annotation soll dabei weitestgehend theorieneutral erfolgen, um später vielfältige Untersuchungen, ausgehend von unterschiedlichsten theoretischen Ansätzen, zuzulassen. Mögliche Erkenntnisinteressen dieser Annotationsebene könnten etwa in der Erstellung einer Nebensatztypologie, der Beschreibung des Modus im abhängigen Satz, der Erfassung von Einleitewörtern, Korrelaten und Verbstellung sowie die Positionierung der unterschiedlichen Nebensatz-, Infinitiv- und Partizipialkonstruktionen im Gesamtgefüge liegen. Neben den bereits zuvor benannten Problemen der Überschneidung zwischen den einzelnen Ebenen, treten hier zusätzlich einige spezielle Problemstellungen auf. So erschwert die partielle Nichtunterscheidbarkeit von bestimmten Modi bei älteren Sprachstufen des Deutschen sowie die variable Stellung des finiten Verbs in abhängigen Sätzen eine eindeutige Annotation, und macht die Kooperation mit den benachbarten Ebenen vorerst unumgänglich.

Annotationsebene einfacher Satz Die Beschreibung der Elementarsätze beschränkt sich ausschließlich auf den Verbalsatz, d. h. verblose Setzungen werden bei der Annotation nicht weiter berücksichtigt. Methodisch wird bei der Beschreibung einfacher Sätze auf einen Valenzansatz zurückgegriffen, bei dem das gesamte Prädikat als Valenzträger gilt. Valenz beschreibt dabei „die Fähigkeit insbesondere von Verben, um sich herum Leerstellen zu eröffnen, die obligatorisch oder optional zu besetzen sind“ (KORTMANN 1999, S. 103). Auf dieser Ebene soll unter anderem die Ermittlung und Kategorisierung von Satzgliedern, die Erfassung unterschiedlicher Satzarten sowie eine Topologie des Verbalkomplexes und der Satzebene erfolgen. Außerdem umfassen die weiteren Beschreibungsziele eine Untersuchung der Valenz und auf dieser Grundlage eine Satzmusterbestimmung. Auch auf dieser Annotati-

onsebene kommt es zu Überschneidungen mit der übergeordneten Stufung *komplexer Satz*, werden doch auch hier Nebensätze, Infinitiv- und Partizipialkonstruktionen untersucht. Weitere Probleme sind bei der Kategorisierung der Satzglieder, der eindeutigen Abgrenzung von Verbalkomplexen sowie der quantitativen Untersuchung der Valenz zu erwarten.

Annotationsebene Nominalgruppe Als Datengrundlage für die vorerst kleinste Annotationseinheit *nominaler Wortgruppen* dienen Texte, die bereits nach Wortarten annotiert sind. In der Pilotphase wird das POS-Tagging vorerst von der hierarchisch gesehen höchsten Annotationsinstanz, der Arbeitsgruppe *Text*, vorgenommen. Im weiteren Projektverlauf kann hier möglicherweise auf wortartannotierte Texte des DDD zurückgegriffen werden. Vorgesehen ist hier ein deskriptiver Ansatz, welcher bei der Flexionsbestimmung einzelner Wörter einsetzt und stufenweise darauf aufbauend komplexere Einheiten beschreibt. Den Untersuchungsgegenstand dieses Projektteils bilden einfache und komplexe Nominalphrasen sowie komplexe Adverbialphrasen, wobei eine detaillierte Beschreibung des inneren Aufbaus der Satzglieder Hauptziel der Analyse ist. Dabei soll im Bereich der Nominalphrasen zum einen eine Typisierung der nicht-verbalen Wortgruppen, zum anderen eine Untersuchung der Flexion innerhalb der Nominalphrasen geleistet werden. Bei allen nicht-verbalen Wortgruppen, also auch den komplexen Adverbialphrasen, soll zudem eine Komplexitätsuntersuchung in Hinblick auf Umfang und Bestandteile sowie eine Topologie innerhalb der Wortgruppen erfolgen.

Arbeitsgruppe Annotation und Technik Das gesamte Annotationsszenario bei DiSynDe ist durch räumlich und zeitlich verteilt arbeitende Gruppen gekennzeichnet. Zusätzlich sind die Annotationsebenen in der Pilotphase nicht klar voneinander zu trennen, sondern überschneiden sich an vielen Stellen, was in der Praxis die Koexistenz mehrerer Annotationsvorschläge zu ein und demselben syntaktischen Phänomen zur Folge haben kann. Aufgabe der Gruppe *Annotation und Technik* ist es deshalb den verteilten Annotationsprozess bestmöglich durch standardisierte Annotationsrichtlinien und computergestützte Werkzeuge zu koordinieren und zu unterstützen. Ein einheitliches Annotationsschema für alle Ebenen der Auszeichnung legt den Grundstein für eine konsistente Annotation der historischen Dokumente, ein entsprechendes Annotationstool setzt diese Richtlinien graphisch um und unter-

stützt den Annotator durch funktionelles und benutzerfreundliches Design bei seiner Arbeit. Ein Annotationswerkzeug soll außerdem in der Lage sein die unterschiedlichen Ebenen parallel, also durch Filterungsmechanismen graphisch darzustellen und Änderungen auf allen Ebenen ermöglichen.

2.2.2 Pilotkorpus

Für die Dauer der Pilotphase sollen digitalisierte Texte aus dem Zeitraum vom 11. bis zum 17. Jahrhundert verwendet werden, wobei das Korpus aus knapp 40 Microsoft (MS) Word Dokumenten unterschiedlicher Länge besteht. Bei der Zusammenstellung dieses provisorischen Korpus wurde versucht, einen möglichst repräsentativen Ausschnitt einer umfassenden Textsammlung althochdeutscher, mittelhochdeutscher und frühneuhochdeutscher Schriften zu simulieren. Die Dokumente aus unterschiedlichen Epochen repräsentieren zudem unterschiedliche Texttypen, wie etwa Fachliteratur, geistliche Prosa, Rechtsprosa, Übersetzungsliteratur, Chronistik, Privatschriften und Unterhaltungsprosa. Hinsichtlich der Kodierung von Sonderzeichen und der Angabe bibliographischer Daten bestehen zwischen den einzelnen Textdokumenten des provisorischen Korpus teilweise starke Inkonsistenzen, welche es im Verlauf der Pilotphase zu beseitigen gilt.

3 Systematisierung von Annotationswerkzeugen

Annotation wurde bereits lange vor Anbruch des Computerzeitalters betrieben und ist seit jeher ein wichtiges Instrument um Wissen zu akkumulieren, es zu verwalten und anderen besser zugänglich zu machen. Doch auch wenn sich die damaligen Vorgehensweisen und Techniken teilweise erheblich von der heutigen Annotationspraxis unterschieden, so ist das Ziel der Auszeichnung doch das gleiche geblieben, nämlich eine vorher definierte Annotationsbasis mit Metadaten, Zusatzinformationen und Kommentaren anzureichern (vgl. FOGLI et al. 2004, S. 98). Längst muss es sich bei der Annotationsbasis nicht mehr nur um geschriebene Texte handeln. Vielmehr wird mit multimodaler Annotation das Bestreben bezeichnet, neben Text auch statische und bewegte Bilder, Sprache, Prosodie, Mimik, Gestik und vieles mehr zu annotieren. Spätestens hier ist die Unterstützung des Annotators durch die Maschine sinnvoll, werden doch häufig verschiedene Ebenen in unterschiedlichsten Modalitäten parallel annotiert.

Die Zahl der computerisierten Annotationswerkzeuge ist dabei genauso groß wie die Zahl der denkbaren Annotationsszenarien. Insgesamt zeichnet sich die Landschaft an bestehenden Annotationstools durch ein außerordentlich hohes Maß an Heterogenität aus, welches zum einen von den unterschiedlichen Annotationsbedürfnissen, zum anderen vom Nichtvorhandensein oder Nichteinhalten geeigneter Standards herrührt. Dieser Teil der Arbeit soll deshalb das weite Feld der existierenden Werkzeuge skizzieren und aufzeigen welche grundlegenden Anforderungen von welchen Werkzeugen am besten erfüllt werden. Ziel ist es, eine Klassifikation von Annotationswerkzeugen zu erstellen, welche die Auswahl geeigneter Kandidaten für das im vorhergehenden Kapitel dargelegte diachrone Annotationsszenario des DiSynDe Projekts erleichtert.

3.1 Annotationsmodalität und Softwaretyp

Die nachfolgende Einteilung, nach Annotationsmodalitäten und Typ der Annotationssoftware, verfolgt einen doppelten Zweck: Erstens soll anhand dieser beiden

Merkmale das breite Angebot an Annotationswerkzeugen systematisiert und überschaubar gemacht werden, zweitens sind beide Merkmale gleichzeitig Kriterien zur Vorauswahl geeigneter Tools für das diachrone Syntaxprojekt DiSynDe. Somit stellt die Systematisierung nach Modalität und Typ die erste Stufe eines zweistufigen Filterungsprozesses dar:

Recherche	Breite Recherche zu computergestützten Annotationswerkzeugen
1. Stufe	Systematisierung der recherchierten Werkzeuge nach den Kriterien *Annotationsmodalität* und *Softwaretyp* (*Tabelle 3*). Sofort verwendbare Werkzeuge[8], welche sich für die Annotation der Modalität Text eignen, werden im nächsten Schritt auf obligatorische Ausschlusskriterien geprüft.
2. Stufe	Filterung der verbleibenden Textannotationswerkzeuge nach obligatorischen Auswahlkriterien (*Tabelle 4*).

Tabelle 2: Zweistufiger Filterungsprozesses zur Auswahl von geeigneten Annotationswerkzeugen für diachrone Korpora

3.1.1 Spektrum der Annotationsmodalitäten

Wenn im Zusammenhang mit Sprachdaten die Rede von Modalitäten ist, sind immer die vielfältigen Möglichkeiten gemeint, die der Mensch zur Kommunikation und Interaktion entweder mit einem anderen Menschen, oder einer Maschine zur Verfügung hat. Technisch gesehen wird jede Modalität als Signalstrom eines bestimmten Typs behandelt, wobei mögliche Signaltypen beispielsweise Wörter, Gesten, Blicke oder Ähnliches sein können (vgl. MÜLLER & STRUBE 2001, S. 45).

Bei multimodalen Korpora können die meisten auftretenden Modalitäten in die beiden Gruppen Sprache und Körperbewegungen eingeteilt werden. Zur Kategorie Sprache zählen sowohl Texte als auch gesprochene Sprache, welche vor der eigentlichen Annotation meist erst transkribiert werden muss. Durch das Setzen von Marken entlang eines Zeitstrahls ist eine zeitliche Zuordnung von Transkription und akustischem oder visuellem Signal möglich (vgl. ENGLERT 1999, S. 88). Dabei ist die Transkription als Vorstufe der Annotation zu sehen, da die Verschriftlichung

[8] „Sofort verwendbar" bedeutet, dass es sich um die konkrete Implementierung eines Annotationswerkzeugs handelt, und nicht etwa ein formales Framework vorliegt, welches nicht direkt anwendbar ist.

eines beliebigen Signals unabdingbar ist, um überhaupt erst metasprachliche Auszeichnungen, wie etwa grammatische Glossen und Ähnliches, in textueller Form beifügen zu können. Im Kontext des DiSynDe Projekts kann die Digitalisierung von Handschriften in gewisser Weise ebenfalls als Transkription gesehen werden, welche aber nicht vom Annotationswerkzeug geleistet werden muss. Tools, die lediglich den Arbeitsschritt einer orthographischen oder phonetischen Umschrift unterstützen, werden dadurch nicht automatisch zur Kategorie der Textannotationswerkzeuge gezählt, da Transkription lediglich eine Vorverarbeitung der Daten für den eigentlichen Annotationsprozess darstellt. Die Kategorie der Körperbewegungen bzw. Körpersprache umfasst unter anderem Gesichtsausdruck, Kopfbewegungen, Augenbewegungen, Blicke, Lippenbewegungen, Handbewegungen, Haltung, räumliche Ausrichtung etc. (vgl. RYDEMAN 2003, S. 6). Von multimodalen Annotationen wird dann gesprochen, wenn mindestens zwei Kommunikationsmodi mit Metainformation angereichert werden, also beispielsweise Sprache und Gestik. Die Arbeitsgruppe *Natural Interactivity and Multimodality* (NIMM) befasst sich im Rahmen des Projekts *International Standards for Language Engineering* (ISLE) seit dem Jahr 2000 mit Problemen multimodaler Annotation und Interaktion, und leistet wertvolle Basisarbeit auf diesem Gebiet. Das DiSynDe Szenario hingegen sieht bedingt durch die historische Textbasis nur eine unimodale Annotation vor, nämlich die Auszeichnung von digitalisierten Originaltexten. Präferiert werden demnach Werkzeuge die speziell auf die Annotation von Text ausgelegt sind, und nicht etwa Text nur in einem multimodalen Kontext neben Sprache und Gestik berücksichtigen. *Tabelle 3* zeigt einen Großteil der derzeit existierenden Annotationswerkzeuge samt der Modalitäten, für deren Auszeichnung sie geeignet sind.

3.1.2 Typen von Annotationssoftware

Neben der Unterscheidung nach Annotationsmodalitäten lassen sich Werkzeuge auch in bestimmte Softwaretypen unterteilen. Das breite Spektrum an Annotationssoftware gliedert sich dabei grob in zwei Kategorien: Zum einen finden sich zahlreiche fertig implementierte Werkzeuge, welche häufig durch Schlichtheit sowie gute Visualisierung auffallen und ohne hohen Einrichtungsaufwand sofort eingesetzt werden können. DIPPER et al. sprechen in diesem Zusammenhang vom Charakte-

ristikum „ready and easy to use“, (2004, S. 56), was nichts anderes bedeutet, als dass die Werkzeuge ohne großen Installations- und Konfigurationsaufwand sofort einsatzbereit zur Verfügung stehen und auch die Bedienung intuitiv und ohne unnötig hohe Einarbeitungszeiten von statten gehen sollte. Diese Art von Werkzeugen eignet sich vor allem für Annotatoren, die keine weiterreichenden Kenntnisse im Bereich der Programmierung besitzen, sondern lediglich als Endnutzer eines bereits fertig gestellten und funktionierenden Produkts auftreten.

Zum anderen gibt es komplexe Softwarelösungen, welche in erster Linie für die Riege der Systementwickler interessant sind. Sie umfassen mächtige Entwicklungsumgebungen, formale Datenmodelle und abstrakte Klassenbibliotheken zur Erstellung und Modifizierung individueller Annotationstools. Da diese Programme Detailwissen im Bereich Programmierung sowie ein hohes Maß an Abstraktionsvermögen erfordern, können sie nicht ohne eine vorhergehende Anpassung durch Systementwickler von den Endnutzern verwendet werden. Anwendungen, welche mit dem Vermerk Framework, API oder Toolkit gekennzeichnet sind, eignen sich somit nicht sofort zur Annotation einer bestimmten Modalität, sondern liefern vielmehr das Handwerkszeug um Annotationsschemata oder eigene Tools für beliebige Modalitäten zu generieren.

Ein *Framework* zeichnet sich vor allem durch sein auf Wiederverwendbarkeit ausgelegtes, komponentenbasiertes Design aus. PREE spricht von einer „Sammlung verschiedener, individueller Komponenten mit definiertem Kooperationsverhalten“ (1997, S. 7) zur Erfüllung einer bestimmten Aufgabe, wobei ein solch definiertes Kooperationsverhalten häufig auch als *Architektur* (vgl. CUNNINGHAM 2000, S. 39ff.) bezeichnet wird. Nach PREE ist eine Architektur als Untermenge eines Frameworks zu sehen, welche das Framework von „gewöhnlichen Klassenbibliotheken“ (1997, S. 19) unterscheidet. Die Komponenten eines Frameworks sind häufig in Form von wiederverwendbaren Klassenbibliotheken und *Application Programming Interfaces* (APIs) realisiert, über welche die Bibliotheken angesteuert werden können. Besitzt ein Werkzeug ein API, so kann dieses im Prinzip als Programmierschnittstelle verstanden werden, welche die Anbindung anderer Programme an das System ermöglicht (vgl. SCHNEIDER & WERNER 2001, S. 237). Alle im Rahmen dieser Untersuchung als Framework klassifizierten Werkzeuge beinhalten demnach eine Sammlung von Komponenten in Form von Klassen oder Softwaremodulen

und eine zugrunde liegende Architektur, welche dass Zusammenspiel zwischen den einzelnen Komponenten und die Eingliederung neuer Module über ein API regelt.

Der Vermerk *Annotationswerkzeug* deutet auf ein fertig implementiertes Tool hin, welches zur Annotation der angegebenen Modalitäten verwendet werden kann. Wenn neben der Hauptaufgabe der Annotation auch noch andere Dienste, wie etwa statistische Analysetools, Lexikoncompiler, XML-Browser, TTS-Synthese[9], Sprachtrainer, Konvertierungstools oder Korpusverwaltungstools zur Verfügung stehen, so werden diese bei der Typisierung der Software zwar ebenfalls angegeben, sind aber für die eigentlichen Projektanforderungen vorerst unerheblich.

In die nähere Auswahl der Werkzeuge für diachrone Syntaxannotation gelangen zunächst all jene Tools, die für die Annotation der Modalität *Text* geeignet sind. Da es im Rahmen dieser Magisterarbeit aus Zeit- und Kostengründen (vgl. DIPPER et al. 2004, S. 54) nicht möglich sein wird ein eigenes Werkzeug von Grund auf neu zu implementieren, werden neben sofort verwendbaren und betriebsbereiten Tools nur solche Frameworks in Betracht gezogen, die schon ein zumindest grundlegendes Textannotationswerkzeug bereitstellen.

Der nachfolgende Abschnitt dokumentiert welche Werkzeuge bei einer Filterung nach Annotationsmodalität und Softwaretypus als ungeeignet eingestuft werden müssen.

3.1.3 Filterung nach Annotationsmodalität und Softwaretyp

Die Übersicht am Ende dieses Abschnitts zeigt insgesamt 51 Annotationswerkzeuge in alphabetischer Reihenfolge, systematisiert nach Annotationsmodalität und Typ der Software. Als Grundlage für die Recherche und Kategorisierung dienten in erster Linie einschlägige Webportale zum Thema Annotation, wie etwa die Übersichtsseite des *Linguistic Data Consortium* (LDC) oder das *Linguistic Annotation Wiki* (LAW) sowie zahlreiche Berichte (vgl. CAPPELLI et al. 1998; DYBKJÆR et al. 2001a) und Fachaufsätze[10] aus dem Bereich der computergestützten Annotation. Darüber

[9] *Text to speech* - Dabei wird geschriebener Text automatisch in vom Computer gesprochene Sprache umgewandelt.

[10] vgl. BIGBEE et al. 2001; DIPPER et al. 2004; GARG et al. 2004; IDE & BREW 2000; MITKOV et al. 2000; MÜLLER & STRUBE 2001; RYDEMAN 2003; SIM et al. 2005 und TEICH et al. 2001. Die vollständigen bibliographischen Angaben finden sich im Literaturverzeichnis wieder.

hinaus wurden weitere Werkzeuge durch Schlagwortsuche selbständig im Web recherchiert. Eine vollständige Übersicht zu allen Annotationswerkzeugen mit weiterführenden Links ist auf der Projekthomepage www.disynde.de zu finden.

Hellrot hinterlegte Tabellenfelder markieren ungeeignete Werkzeuge sowie deren Ausscheidungsgrund. Alle Tools die ausschließlich zur Annotation von gesprochener Sprache und Geräuschen (DAT, EXMARALDA, Praat, Snack, Transcriber, WaveSurfer) oder Sprache und Prosodie (Multext Tools) gedacht sind, eignen sich für die Annotation eines diachronen Textkorpus nicht. Ebenso ungeeignet sind die multimodalen Tools aus dem Bereich des Videomarkup (Anvil, CAVA, CBAS, ELAN, EUDICO, Interact, MediaStreams, MultiTool, SyncWriter, TASX, Transformer, vPrism), welche die Modalitäten Sprache und Gestik annotierbar machen. Werkzeuge, welche spezielle Tagsets für Gesichtsausdrücke und Ähnliches bereitstellen, und somit neben der Auszeichnung von Körperbewegungen und Sprache eine Mimikannotation der Sprecher (CSLU Toolkit, Observer, SignStream, SmartKom) ermöglichen, sind für das Projektszenario von DiSynDe ohne weiteren Nutzen.

Unter den multimodalen Tools stechen einige Programme, wie etwa CLAN und NITE hervor, welche neben der Annotation von Videomaterial zusätzlich die Annotation von geschriebenem Text ermöglichen. Dabei kann CLAN nicht beliebige Texte annotieren, sondern nur solche editieren und modifizieren, die im CHILDES (*Child Language Data Exchange System*) Format, einem textuellen Auszeichnungsformat für gesprochene Kindersprache, vorliegen. Das NITE Projekt ermöglicht theoretisch die Annotation jeder beliebigen Modalität, somit auch die Auszeichnung von Text, allerdings gibt es für das abstrakte Framework, welches als Nachfolger des MATE Projekts gilt, noch kein implementiertes Textannotationstool. Die beiden Programme DitAT und MATE versprechen sowohl eine Annotation von gesprochener als auch geschriebener Sprache, und sollen deshalb genauso wie die reinen Textannotationswerkzeuge zunächst näher untersucht werden. Von den insgesamt 28 Textauszeichnungstools sind vier Werkzeuge (AGTK, ATLAS, LT XML, NITE) für die diachrone Syntaxannotation durch Benutzer ohne Programmierkenntnisse ungeeignet, da sie als formale Frameworks vorliegen und nicht sofort als Annotationshilfe eingesetzt werden können. Immerhin können die Frameworks AGTK und ATLAS indirekt durch ihre konkreten Implementierungen ACE Annotation Toolkit und Callisto weiteren Praxistests unterzogen werden.

Toolname	Modalitäten	Softwaretyp
ACE Annotation Toolkit	Text	Annotationswerkzeug (basiert auf dem AGTK)
ACT	Text	Annotationswerkzeug
AGTK	Text	Framework
Alembic Workbench	Text	Framework und Annotationswerkzeug
Annotate	Text	Annotationswerkzeug
Anvil	Sprache, Gestik	Annotationswerkzeug
Arboreal	Text	Annotationswerkzeug, XML-Browser
ATLAS	Text	Framework
CAVA	Sprache, Gestik	Annotationswerkzeuge
Callisto	Text	Annotationswerkzeug (basiert auf jATLAS, Nachfolger der Alemic Workbench)
CBAS	Sprache, Gestik	Annotationswerkzeug
CLAN	Text, Sprache, Gestik	Annotationswerkzeug für Texte eines bestimmten Formats (CHILDES), Analysetool
CLaRK	Text	Annotationswerkzeug, Lexikonerstellung
CSLU Toolkit	Sprache, Mimik	Framework, Annotationswerkzeug, Analysetool, TTS, Sprachtrainer
DAT	Sprache (Dialoge)	Annotationswerkzeug (benutzt das DAMSL[11] Schema)
Dexter	Text	Annotationswerkzeug
DitAT	Text, Sprache	Annotationswerkzeug
ELAN	Sprache, Gestik	Annotationswerkzeug
EUDICO	Sprache, Gestik	Framework, Workbench (Integration in GATE geplant)
EXMARaLDA	Sprache	Annotationswerkzeug, Korpusmanager, Analysetool
FLEX	Text	Annotationswerkzeug für Feldforschung, Lexikonerstellung
GATE	Text	Framework und Annotationswerkzeug
Interact	Sprache, Gestik	Annotationswerkzeug
ITE	Text	Annotationswerkzeug
LT XML	Text	Framework, Greptool
MATE	Text, Sprache	Framework
MediaStreams	Sprache, Gestik	Ikonisches Annotationswerkzeug
MMAX	Text	Annotationswerkzeug
MMAX 2	Text	Annotationswerkzeug (Nachfolger von MMAX)
Multext Tools	Sprache, Prosodie	Annotationswerkzeug
MultiTool	Sprache, Gestik	Annotationswerkzeug, Analysetool

[11] *Dialog Act Markup in Several Layers* – DAMSL ist ein spezielles Schema zur Annotation von Dialogen

NITE (NXT)	Text, Sprache, Gestik, Mimik	Framework (Nachfolger von MATE)
Observer	Mimik, Gestik	Annotationswerkzeug
oXygen	Text	XML Annotationswerkzeug
Palinka	Text (Anaphern)	Annotationswerkzeug (Nachfolger von Clinka)
Praat	Sprache	Annotationswerkzeug, Analysetool, TTS
RST Tool	Text (rhetorische Strukturen)	Annotationswerkzeug
SignStream	Sprache, Mimik, Gestik	Annotationswerkzeug, Analysetool
SmartKom	Sprache, Mimik, Gestik	Framework (benutzen Anvil zur Annotation)
Snack	Sprache	Framework
SyncWriter	Sprache, Gestik	Annotationswerkzeug
Synpathy	Text (Syntax)	Annotationswerkzeug
Systemic Coder	Text	Annotationswerkzeug, Analysetool
TASX	Sprache, Gestik	Framework und Annotationswerkzeug
Toolbox	Text	Annotationswerkzeug für Feldforschung, Lexikonerstellung (Nachfolger von Shoebox)
Transcriber	Sprache	Annotationswerkzeug
Transformer	Sprache, Gestik	Annotationswerkzeug
vPrism	Sprache, Gestik	Annotationswerkzeug
WaveSurfer	Sprache	Annotationswerkzeug
UAM CorpusTool	Text	Annotationswerkzeug (Nachfolger von Systemic Coder), Analysetool
Wordfreak	Text	Annotationswerkzeug

Tabelle 3: Filterung nach Annotationsmodalität und Softwaretypus

3.2 Grundlegende Auswahlkriterien

Da nach dem relativ groben Filterungsprozess anhand der Kriterien *Annotationsmodalität* und *Softwaretyp* von den 51 vorgestellten Werkzeugen immer noch 23 Textannotationswerkzeuge verbleiben, gilt es nun einige grundlegende Auswahlkriterien für ein diachrones Annotationsszenario an die verbleibenden Kandidaten anzulegen.

3.2.1 Verfügbarkeit und Aktualität der Applikation

Ein solches Kriterium stellt die technische Verfügbarkeit eines Werkzeugs dar. So finden sich einige Tools, welche zwar in der einschlägigen Forschungsliteratur immer wieder erwähnt werden, aber im Netz nicht mehr verfügbar sind. Dabei bedeu-

tet Verfügbarkeit[12] für die verbleibenden Textannotationswerkzeuge in allen Fällen das Vorhandensein eines funktionierenden Downloadlinks.

Gründe für einen ungültigen oder nicht vorhandenen Downloadlink sind in den meisten Fällen entweder das hohe Alter eines Werkzeugs, oder der hohe Neuheitswert einer Anwendung. So liegt das viel versprechende Annotationstool Dexter zum Zeitpunkt der Untersuchung noch in einer unausgereiften Betaversion vor. DitAT ist zwar bereits verfügbar (Version 0.8), aber noch nicht vollständig mit allen geplanten Features realisiert, und wird deshalb aktuell noch weiterentwickelt. Somit ist die Aktualität eines Werkzeugs eng verknüpft mit seiner technischen Verfügbarkeit. Die Aktualität bezeichnet den Entwicklungsstatus eines Werkzeugs oder des dazugehörigen Projekts und zeigt anhand der Versionsgeschichte sowie der letzten Updates an, ob ein Tool schon seit längerem nicht mehr gewartet wird, ob eine Software sich noch mitten in der Entwicklung befindet oder ob ein Projekt sogar ganz eingestellt wurde. Das tschechische Annotationsprogramm ACT, zur Verarbeitung von altkirchenslavischen Manuskripten, konnte Anfang 2007 noch problemlos heruntergeladen werden. Zum momentanen Zeitpunkt steht das gesamte Projekt offensichtlich still und ist nicht mehr im Netz aufrufbar. Die MATE Homepage findet sich zwar noch im Internet und liefert grundlegende Informationen über den Umfang der Applikation, eine herunterladbare Version ist aber nicht mehr verfügbar. Der Grund hierfür ist wahrscheinlich die Einstellung des Projekts aufgrund von Stabilitätsproblemen der Software. Auf den Seiten des NITE Projekts wird der NITE XML Toolkit als offizieller MATE-Nachfolger vorgestellt. MMAX findet sich ebenfalls noch häufig in der Literatur und auf der offiziellen Homepage wieder, wird aber zugunsten des Nachfolgers MMAX2 seit 2003 nicht mehr aktiv weiterentwickelt. Der Nachfolger liegt zum Zeitpunkt der Untersuchung in einer stabilen Betaversion vor, die bereits wichtige Funktionen, wie etwa Mehrebenenannotation im Stand-off XML Format, implementiert. Das auf Annotationsgraphen basierende Annotate wird zwar seit 1998 nicht mehr offiziell weiterentwickelt, ist aber immerhin noch verfügbar wenn eine ausgefüllte Lizenzvereinbarung an die Universität des Saarlands geschickt wird.

[12] Eine Ausnahme bildet das kommerzielle Videoannotationstool Interact der Firma Mangold, welches nach erfolgter Onlinebestellung über eine zugesandte CD-ROM verfügbar ist.

3.2.2 Flexibilität der Annotationsschemata

Ein Annotationswerkzeug für diachrone Korpora muss nicht nur die Anpassung und Erstellung von Annotationsschemata unterstützen, sondern darüber hinaus die Verknüpfung mehrerer Schemata mit unterschiedlichen Annotationsebenen ermöglichen. Ein Annotationsschema stellt eine Art abstrakte Vorschrift dar, welche vorgibt mit welchen Auszeichnungselementen ein Text überhaupt annotiert werden darf. Werden die Daten, wie in den meisten Fällen, im XML-Format annotiert und gespeichert, so ist das zugrunde liegende Schema meist als *Document Type Definition* (DTD) realisiert. Eine DTD gibt vor, welche Elemente mit welchem Inhalt und in welcher Reihenfolge in einem Dokument zur Auszeichnung vorkommen dürfen. Im annotierten Text werden solche Elemente durch Textmarken, so genannten *Tags*, realisiert. Ein Annotationsschema ist dementsprechend ein bestimmter Satz an Tags, ein so genanntes *Tagset*. Da sich die Projektziele nach der Pilotphase nochmals erheblich ändern können, ist es unumgänglich, dass das Tool die Modifikation bestehender Schemata und das Hinzufügen neuer Vorschriften erlaubt und unterstützt. Optimalerweise bietet das Werkzeug die Möglichkeit Tagsets über eine graphische Oberfläche (UAM CorpusTool, Systemic Coder) anzupassen, komplett neu zu importieren oder gegebenenfalls für andere Projekte zu exportieren. Wird die Anpassung bestehender Tagsets oder die Einbindung eigener Annotationsschemata von fast allen Werkzeugen unterstützt, so erlauben doch nur wenige auch die parallele Definition eigener Annotationsebenen.

Eine Annotationsebene ist mit einem bestimmten Annotationsschema verknüpft und kann sowohl bei der Annotation als auch bei späteren Korpusabfragen nach Belieben ein- und ausgeblendet werden. DiSynDe sieht zum momentanen Zeitpunkt die Annotation historischer Texte auf fünf unterschiedlichen linguistischen Ebenen vor, um diese dann später diachronen Fragestellungen unterziehen zu können. Deshalb reicht es nicht aus Annotationsschemata für eine statische Ebene der Annotation modifizieren zu können, vielmehr ist es erforderlich beliebige, eigene Annotationsebenen frei definieren zu können und diese dann mit geeigneten Sche-

mata zu verknüpfen, mit dem Ziel der Erstellung einer *multi-level*[13] Annotation (vgl. MÜLLER & STRUBE 2003, S. 198).

Viele Tools scheitern an diesem Kriterium, da sie häufig für die Annotation einer oder mehrerer vordefinierter Annotationsebenen konzipiert wurden. Arboreal, CLaRK und oXygen erlauben die vollständige Anpassung eines Tagsets über die Einbindung einer eigenen DTD, jedoch leider nur für eine Ebene. Die Feldforschungswerkzeuge Toolbox und FLEX sind in erster Linie zur Erstellung von Lexika für eine bestimmte Sprache gedacht. Die Annotationsebenen beschränken sich deshalb ausschließlich auf die morphologische Schicht. Auch RST Tool, Palinka, Systemic Coder und Wordfreak erlauben nur die Anpassung einer einzelnen, vordefinierten Annotationsebene. Das RST Tool wurde beispielsweise speziell für die Annotation von rhetorischen Strukturen auf Textebene entwickelt. ACE, ACT, Alembic Workbench und ITE unterstützen zwar eine Annotation auf mehreren Ebenen, erlauben es aber nicht eigene Ebenen hinzuzufügen, sondern stellen einen Satz vordefinierter Ebenen zur Verfügung. Synpathy ist ein reiner Syntaxviewer zur Darstellung und Manipulation von Tiger-XML annotierten Dateien, erlaubt aber nicht das Hinzufügen neuer Ebenen. Dexter und DitAT planen die parallele Annotation auf mehreren Ebenen, haben diese Funktion aber noch nicht vollständig implementiert. Die beiden eingestellten Projekte MATE und MMAX unterstützen die Erstellung beliebiger Annotationsebenen ebenso wie der Alembic Workbench Nachfolger Callisto, MMAX2, die Systemic Coder Weiterentwicklung UAM CorpusTool sowie das GATE Projekt.

3.2.3 Wiederverwendbarkeit des Annotationsformats

Da die Erstellung von Korpora fast immer mit hohen Kosten und großem Zeitaufwand verbunden ist, gilt es schon bei der Auswahl der Annotationstools die Wiederverwendbarkeit der Annotationen für andere Projekte zu berücksichtigen. Auch die parallele Annotation auf mehreren Ebenen stellt zusätzliche Anforderungen an ein standardisiertes und wiederverwendbares Format. Mit XML als Quasistandard (vgl. MÜLLER & STRUBE 2003, S. 198) für das Format der Annotationen ist dies in ho-

[13] Multi-level: This term implies the ability to annotate, link between, and analyze different linguistic levels. Levels of analysis may include orthography, morphology, syntax, dialogue acts, co-reference, intonation, gestures, and so forth. (BIGBEE et al. 2001, S. 1)

hem Maße gewährleistet, verwenden doch auch zahlreiche Standardisierungsprojekte wie CES oder TEI die *Extensible Markup Language* (XML) als Grundlage. Ein obligatorisches Kriterium für die Auswahl eines Tools ist deshalb seine Fähigkeit annotierte Texte als XML-Datei zu speichern, oder zumindest über Konvertierungsmechanismen eine proprietäre Annotation als wohlgeformte XML-Datei exportieren zu können. Da eine weitere zentrale Forderung an Werkzeuge für diachrone Korpora die Unterstützung beliebig definierbarer Annotationsebenen ist, muss ein Tool welches die Annotationen im hierarchischen XML-Format, also nach dem Schema gerichteter, azyklischer Graphen repräsentiert, auf die Technik der Stand-off Annotation zurückgreifen (vgl. TEICH et al. 2003, S. 231ff.).

Das Prinzip der Stand-off Annotation besagt eine logische Trennung von Annotationsbasis und Annotation, da dies zusätzlich die Wiederverwendbarkeit der Daten durch unkompliziertes Hinzufügen oder Weglassen von beliebigen Annotationsebenen gewährleistet. Bis auf die Ausnahmen Alembic Workbench, Toolbox, FLEX, ACE und Annotate, erfüllen praktisch alle Tools die Forderung nach Annotationen im XML-Format. So verwendet die Alembic Workbench die *Standard Generalized Markup Language* (SGML) als Speicherformat, eine Metasprache mit der beliebige Auszeichnungssprachen definiert werden können. Toolbox und FLEX speichern die annotierten Daten lediglich als Klartext, ACE und Annotate greifen auf das Datenmodell der Annotationsgraphen zurück (vgl. BIRD & LIBERMAN 2000). Werkzeuge, die sowohl das Kriterium der Mehrebenenannotation als auch die Forderung nach XML als Annotationsformat erfüllen, bedienen sich in jedem Fall einer Stand-off Technik. Callisto, GATE, MMAX2 und das UAM CorpusTool haben die Stand-off Annotation äußerst effektiv, wenn auch teilweise auf unterschiedliche Weise, implementiert.

3.2.4 Filterung nach Auswahlkriterien

Die erste Filterung aller untersuchten Werkzeuge nach Annotationmodalität und Softwaretyp schränkt die Zahl verfügbarer Annotationstools bereits stark ein. Das Resultat sind 23 vollständig implementierte Anwendungen zur Annotation von Textdateien. Die zweite Stufe der Filterung reduziert die Anzahl potentieller Kandidaten nochmals beträchtlich. Von den verbliebenen Tools erfüllen nur vier Werk-

zeuge die obligatorischen Auswahlkriterien *Aktualität und Verfügbarkeit*, *Flexibilität der Annotationsschemata* und *Wiederverwendbarkeit des Annotationsformats* in befriedigendem Maße. Schon bei dieser Voruntersuchung wird allerdings deutlich, dass jedes der Tools ganz eigene, individuelle Stärken und Schwächen aufweist, welche es im Evaluationsteil genauer zu erfassen gilt.

Toolname	Verfügbarkeit und Aktualität	Format der Annotation	Flexibilität der Annotationsschemata
ACE Annotation Toolkit	Downloadlink für Version 1.0, letzte Aktualisierung 2005	Annotationsgraphen (AGTK Implementierung), kein Stand-off	Keine Anpassung oder Erweiterung der bestehenden vier Annotationsebenen möglich, das Hinzufügen eigener Annotationsebenen wird nicht unterstützt
ACT	Downloadlink momentan nicht verfügbar, letzte Aktualisierung 2003	XML als Speicherformat, Stand-off	Vorgegebene Ebenen und die Möglichkeit Elemente zu sog. *Complex Groups* zusammenzufassen, das Hinzufügen eigener Annotationsebenen wird allerdings nicht unterstützt
Alembic Workbench	Downloadlink für Version 4.61, Zeitpunkt der letzten Aktualisierung nicht bekannt	SGML/XML als Speicherformat, kein Stand-off	Tagsets können für drei bestehende Annotationsebenen angepasst werden, das Hinzufügen eigener Annotationsebenen wird nicht unterstützt
Annotate	Downloadlink für Version 2.0.4 nach Lizenvereinbarung, letzte Aktualisierung 1998	Annotationsgraphen auf Grundlage von AGTK, kein Stand-off	Mehrere Ebenen zur syntaktischen Annotation als Baum, aber keine Möglichkeit eigene Ebenen hinzuzufügen
Arboreal	Downloadlink für Version 5.14, letzte Aktualisierung 2006	XML als Speicherformat, kein Stand-off	Anpassung der Tagsets über DTD oder XSD möglich, das Hinzufügen eigener Annotationsebenen wird allerdings nicht unterstützt
Callisto	Downloadlink für Version 1.5.0, letzte Aktualisierung 2007	ATLAS Interchange Format und XML als Speicherformat, Stand-off	Über eine DTD und den ATLAS Standard für Metadaten (MAIA) können verschiedene Schemata definiert werden

ClaRK	Downloadlink für Version 1.0, letzte Aktualisierung 2005	XML als Speicherformat, kein Stand-off	Modifizierung von Tagsets über Anpassung bestehender DTDs möglich, das Hinzufügen eigener Annotationsebenen wird allerdings nicht unterstützt
Dexter	Downloadlink für Betaversion, letzte Aktualisierung 2007	XML als Speicherformat, Stand-off	Es können eigene Tagsets für beliebige Ebenen hinzugefügt werden
DitAT	Downloadlink für Version 0.8, zum Zeitpunkt der Untersuchung noch nicht vollständig implementiert, letzte Aktualisierung 2007	XML als Speicherformat, kein Stand-off (in Planung)	Die Annotation von Sprache über mehrere Ebenen ist geplant, aber zum Zeitpunkt der Untersuchung nicht implementiert
FLEX	Downloadlink für die Standardversion 4.0, letzte Aktualisierung 2007	Klartext als Speicherformat (aber XML Exportfunktion), kein Stand-off	Da FLEX vor allem für die Feldforschung entwickelt wurde, ist eine Annotation nur auf Morphemebene möglich, das Hinzufügen eigener Annotationsebenen wird nicht unterstützt
GATE	Downloadlink für Version 4.0, letzte Aktualisierung 2007	XML als Speicherformat, Stand-off	Es können eigene Tagsets für beliebige Ebenen hinzugefügt werden
ITE	Downloadlink für Endversion, letzte Aktualisierung 2003	XML als Speicherformat, Stand-off	Modifizierung der bestehenden vier Annotationsebenen durch Anpassung der Tagsets möglich, es können aber keine beliebigen Ebenen hinzugefügt werden
MATE	Downloadlink nicht mehr verfügbar, das Projekt wurde zugunsten des NITE XML-Toolkits eingestellt	XML als Speicherformat, Stand-off	Es können eigene Tagsets für beliebige Ebenen hinzugefügt werden
MMAX	Downloadlink für Version von 2003, das Projekt wurde zugunsten von MMAX2 eingestellt	XML als Speicherformat, Stand-off	Es kann nur auf einer Ebene annotiert werden

MMAX 2	Downloadlink für stabile Betaversion 1.12, wird noch weiterwickelt, letzte Aktualisierung 2006	XML als Speicherformat, Stand-off	Es können eigene Tagsets für beliebige Ebenen hinzugefügt werden
oXygen	Downloadlink für Endversion, letzte Aktualisierung 2007	XML als Speicherformat, kein Stand-off	Anpassung der Tagsets über DTD oder XSD, es können beliebige Ebenen hinzugefügt werden
Palinka	Downloadlink für Endversion, letzte Aktualisierung 2005	XML als Speicherformat, kein Stand-off	Das Hinzufügen eigener Annotationsebenen wird nicht unterstützt
RST Tool	Downloadlink für Version 3.4.1, letzte Aktualisierung 2003	XML als Speicherformat, kein Stand-off	Das Tool ist lediglich zur Strukturierung von Text und Annotation auf Ebene von rhetorischen Strukturen gedacht, das Hinzufügen eigener Annotationsebenen wird nicht unterstützt
Synpathy	Downloadlink für Version 1.0, letzte Aktualisierung 2007	Tiger-XML als Speicherformat, kein Stand-off	Synpathy eignet sich nur zur Darstellung und Nachbearbeitung von Dateien im Tiger-XML Format, das Hinzufügen eigener Annotationsebenen wird nicht unterstützt
Systemic Coder	Downloadlink für Version 4.6.8, letzte Aktualisierung 2005	XML als Speicherformat, kein Stand-off	Das Hinzufügen eigener Annotationsebenen wird nicht unterstützt
Toolbox	Downloadlink für Version 1.5.1, letzte Aktualisierung 2007	Klartext als Speicherformat, kein Stand-off	Da Toolbox vor allem für die Feldforschung entwickelt wurde ist eine Annotation nur auf Morphemebene möglich, das Hinzufügen eigener Annotationsebenen wird nicht unterstützt
UAM CorpusTool	Downloadlink für Version 1.2, letzte Aktualisierung 2007	XML als Speicherformat, Stand-off	Es können eigene Tagsets für beliebige Ebenen hinzugefügt werden
Wordfreak	Downloadlink für Version 2.2, letzte Aktualisierung 2004	XML als Speicherformat, kein Stand-off	Das Hinzufügen eigener Annotationsebenen wird nicht unterstützt

Tabelle 4: Filterung nach Auswahlkriterien

3.3 Annotationswerkzeuge für diachrone Korpora

Bevor die potentiell geeigneten Annotationsprogramme Callisto, GATE, MMAX2 und das UAM CorpusTool in einer Evaluation genauer untersucht werden, soll an dieser Stelle eine kurze Beschreibung der Werkzeuge gegeben werden.

3.3.1 Callisto

Mit Callisto liegt ein Werkzeug vor, welches auf der API des ATLAS[14] Frameworks basiert. In einer praktischen Einführung zur Verwendung von ATLAS beschreiben LAPRUN et al. den allgemeinen Aufbau des Frameworks:

> ATLAS provides an architecture targeted at facilitating the development of linguistic annotation applications and is comprised of four main components: a data model, an Application Programming Interface (API), the ATLAS Interchange Format (AIF, 1999) and the Meta-Annotation Infrastructure (MAIA, 2002) for ATLAS. The data model at its core provides the abstractions on which the rest of the framwork is built. These abstractions can be implemented using any full-featured programming languages. (2000, S. 1928)

Die Java Implementierung jATLAS umfasst grundlegende Funktionen zum Erstellen, Editieren und Löschen von Annotationen. Neben der Metadateninfrastruktur MAIA (*Meta-Annotation Infrastructure for ATLAS*) findet sich auch das ATLAS *Interchange Format* (AIF), ein Stand-off XML basiertes Austauschformat für Annotationsgraphen, in Callisto wieder. Im Annotationsformat wird der Originaltext als Signalstrom in Form einer Base64[15] Verschlüsselung mitgespeichert, die eigentliche Annotation referenziert bestimmte Textregionen in diesem Signal und basiert auf dem formalen Modell von BIRD & LIBERMANS Annotationsgraphen (2000, S. 27). Alle Annotationsebenen werden mitsamt dem Originaltext in einer einzigen Datei gespeichert, sind aber gemäß den Forderungen des Stand-off Konzepts logisch voneinander getrennt. Um ein eigenes Annotationsschema in Callisto verwenden zu können, muss dieses in der Syntax einer DTD formuliert und im Callisto DTD Compiler in ein *Task* umgewandelt werden. Das Ergebnis dieser Kompilierung ist eine Java Archive Datei (JAR), welche das Schema im DTD-Format enthält und in das Annotationswerkzeug eingebunden werden kann. Callisto ist vollständig in Java

[14] *Architecture and Tools for Linguistic Analysis Systems*
[15] Base64 beschreibt ein Verfahren bei dem Daten als ASCII-Zeichenstrom kodiert werden.

implementiert und kann aufgrund seines modularen Designs gut angepasst und erweitert werden.

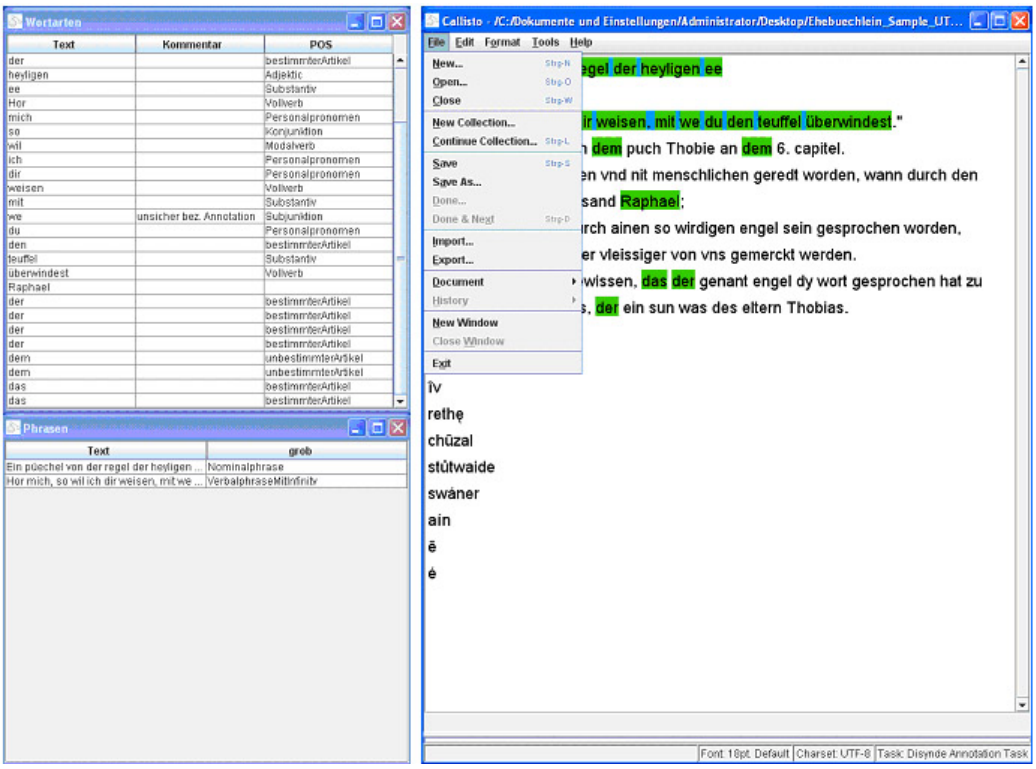

Abbildung 2: Screenshot des Callisto Annotationswerkzeugs

3.3.2 GATE

Von allen untersuchten Annotationswerkzeugen ist GATE (*General Architecture for Text Engineering*) zweifelsohne das Tool mit der eindrucksvollsten Entwicklungsgeschichte. So wurde bereits 1995 die Arbeit an einer wiederverwendbaren Architektur für *Text Engineering* (TE) Applikationen an der Sheffield University im Rahmen der Arbeitsgruppe *Infrastructure for Human Language Technology* begonnen, und ein Jahr später die erste offizielle Version veröffentlicht. Seitdem wurde GATE permanent weiterentwickelt, so dass es seit 2007 mittlerweile in Version 4.0 vorliegt und von 20 aktiven Entwicklern gewartet und weiter vorangetrieben wird. Laut Angaben von Herstellerseite soll die flexible TE Architektur allein im letzten Jahr über 20.000 Mal heruntergeladen worden sein (vgl. CUNNINGHAM 2007, S. 1). Neben

einem datenbankbasierten Speicheransatz bietet GATE auch die Möglichkeit Annotationen als Stand-off XML Dateien zu exportieren, wobei der Originaltext formal als azyklischer Graph repräsentiert wird, dessen Knoten verschiedene Annotationen zugeteilt werden können:

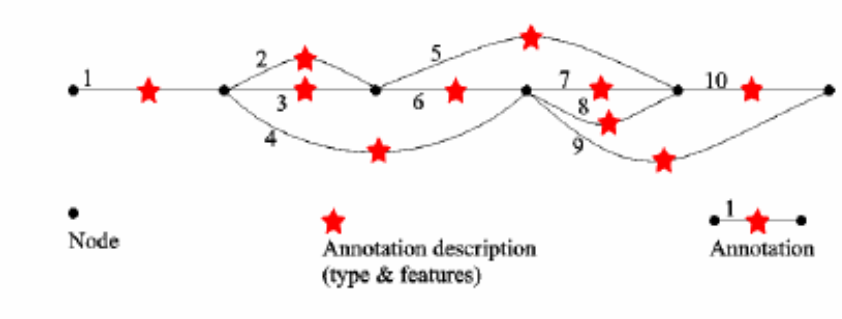

Abbildung 3: Modell eines Annotationsgraphen, nach TABLAN et al. (2004, S. 19)

GATE ist die einzige Applikation unter den verbleibenden Testkandidaten, welche neben einem Textannotationswerkzeug zusätzlich eine komplexe Infrastruktur für Programme aus dem TE Bereich bereitstellt. Dabei sieht sich GATE selbst als *Domain-Specific Software Architecture* (DSSA), welche zum einen als Framework eine komponentenbasierte[16] Architektur für Programmierer von NLP[17]-Applikationen zur Verfügung stellt, zum anderen als Entwicklungsumgebung für Korpora, mit fertig implementierten Werkzeugen zur Annotation und Abfrage Anwendern die Möglichkeit bietet eigene Texte zu annotieren (vgl. CUNNINGHAM 2000, S. 40).

Die komplexe Architektur von GATE lässt sich in drei Subsysteme untergliedern: Den *GATE Document Manager* (GDM), welcher als Repositorium für alle zu verarbeitenden Sprachressourcen dient, eine Sammlung wiederverwendbarer *Language Engineering* Objekte mit dem Namen CREOLE (*Collection of Reusable Objects for Language Engineering*) und das *GATE Graphical Interface* (GGI), welches zur Darstellung der GDM und CREOLE Ressourcen genutzt wird. GATE unterscheidet programmintern außerdem *Language Resources* (Sprachressourcen) und *Pro-*

[16] GATE ist zu 100% in Java implementiert, beim komponentenbasierten Design wurde auf die Java Beans Technologie zurückgegriffen (vgl. CUNNINGHAM et al. 2007, S. 75)

[17] *Natural Language Processing*, wird im Deutschen häufig mit *Verarbeitung natürlicher Sprache* (VNS) übersetzt

cessing Resources (Verarbeitungsressourcen), welche in so genannten Pipelines[18] fast beliebig kombiniert und verknüpft werden können. Dabei sind mit *Language Resources* reine Sprachdaten wie etwa Texte, Korpora, Lexika oder Ontologien gemeint. *Processing Resources* bezeichnen Programme und Algorithmen, wie beispielsweise Tokenizer, Tagger oder Parser, welche zur Verarbeitung von Sprachdaten genutzt werden können. Für GATE existieren vor allem im Bereich der *Processing Resources* unzählige externe Programme und Plugins, die in das Java-basierte Framework eingegliedert werden können. Somit bietet GATE ein Höchstmaß an Flexibilität und Anpassbarkeit weit über die Anforderungen des momentanen Annotationsszenarios hinaus. GATE ist vollständig in Java implementiert und lässt sich in der aktuellen Version auf jeder Plattform ausführen, welche eine *Java Virtual Machine* installiert hat.

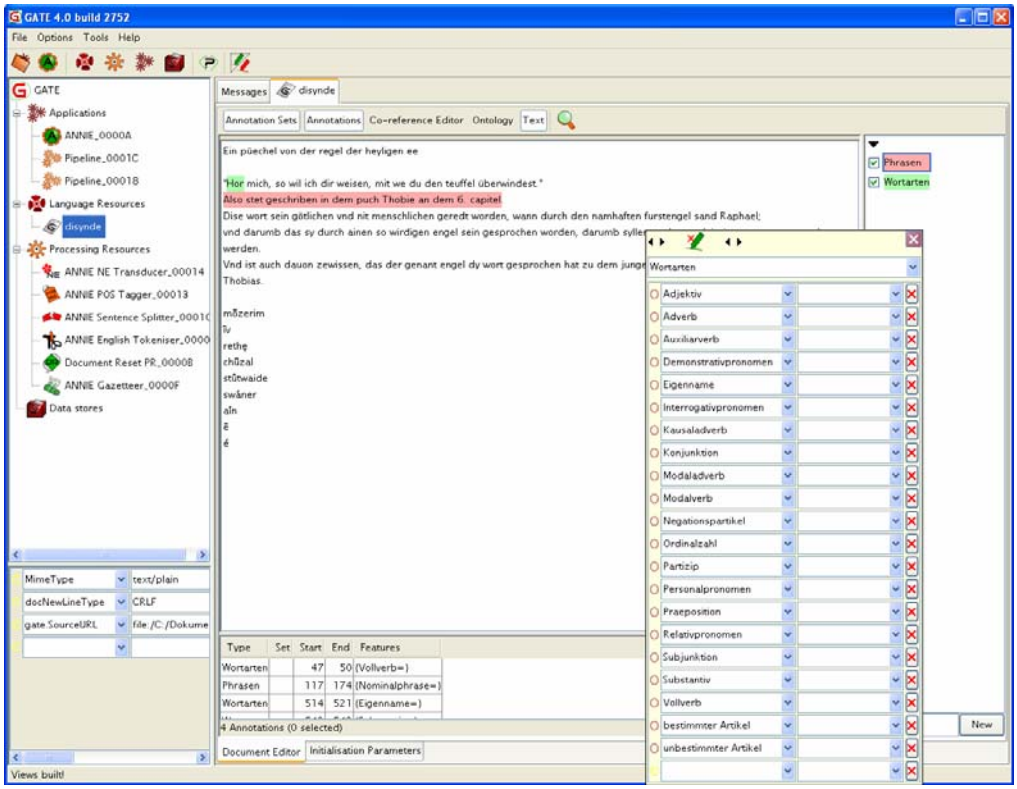

Abbildung 4: Screenshot des GATE Annotationswerkzeugs

[18] Die Metapher wurde aus der Unix Welt übernommen, wo über so genannte *pipes* der Output einer Applikation als Input für eine andere Applikation deklariert werden kann, und so die Möglichkeit besteht verschiedene Applikationen auf komplexe Weise miteinander zu verknüpfen.

3.3.3 MMAX2

MMAX2, eine Software für *Multi-Modal Annotation in XML* (MMAX), wurde im Rahmen des Forschungsprojekts EMBASSI[19] von der EML Research gGmbH ursprünglich zur Annotation von Koreferenz entwickelt. Wegen seines hohen Maßes an Anpassbarkeit und seiner Flexibilität in Hinblick auf Annotationsschemata, erfreute sich das Werkzeug in verschiedensten Annotationsprojekten großer Beliebtheit, wurde aber wegen einiger grundlegender Schwachstellen im April 2003 zum letzten Mal aktualisiert und auf dem Stand der Version 0.94 eingefroren. Mit MMAX2 wurde Anfang 2005 ein würdiger Nachfolger geschaffen, welcher fast alle Einschränkungen des Vorgängers aufhebt. So wurde bereits in der Betaversion, neben einer verbesserten Visualisierung und der Möglichkeit komplexe Relationen zwischen Annotationselementen zu erstellen, auch die Definition beliebig vieler Auszeichnungsebenen als Neuerung implementiert.

Dabei setzt MMAX2 auf eine rigorose Umsetzung des Stand-off Prinzips. Die Rohdaten, welche in diesem Kontext als *base data* bezeichnet werden, können entweder in Form einer XML Datei oder als Klartext importiert werden (vgl. MÜLLER & STRUBE 2006, S. 3f.). Ein Projektwizard unterstützt den Anwender beim Erstellen neuer Annotationsprojekte. Bei der Stand-off Annotation werden der Originaltext und sämtliche Annotationsebenen strikt voneinander getrennt, was bei der MMAX2 Implementierung bedeutet, dass die tokenisierte *base data* als indexierte XML-Datei abgelegt wird. Die einzelnen Wörter werden dabei mit fortlaufenden IDs versehen. Außerdem wird für jede Annotationsebene eine XML-Datei erstellt, welche dann die Wortliste des Originaltexts über die automatisch erstellten Indices referenziert. Die Dateien, welche die Annotationsebenen enthalten, sind ihrerseits mit vorher definierten Annotationsschemata im XML-Format verknüpft.

Doch MMAX2 setzt nicht nur rigoros die Trennung von Originaltext, Annotation und Annotationsschema um, sondern lagert auch die visuelle Anpassung der einzelnen Ebenen sowie die Speicherung von Metadaten zum gesamten Projekt in se-

[19] „Das Ziel des Leitprojektes ‚EMBASSI' ist es, ein ganzheitliches Assistenzkonzept zu entwickeln, das den Nutzer bei der Bedienung von Alltagstechnologie optimal unterstützt." [http://www.embassi.de] – Zugriff am 10.06.2008.

parate Dateien aus. Ein zentrales Stylesheet erlaubt die Modifikation der Gesamtdarstellung der Annotation über XSL[20] Transformation.

Verschiedene Browser zur Darstellung der Annotationen im KWIC (*Keywords in Context*) Format sowie eine komfortable Query Konsole mit einer mächtigen Abfragesprache namens *MMAX Query Language* runden das Bild eines in hohem Maße anpass- und individualisierbaren Annotationswerkzeugs ab.

Nachdem die Vorgängerversion von Anfang an kostenlos zur Verfügung stand, war für MMAX2 ursprünglich die Erhebung einer Lizenzgebühr geplant, um die Wartung und Weiterführung des Projekts durch die beiden Hauptentwickler Christoph Müller und Michael Strube zu ermöglichen (vgl. MÜLLER 2004, S. 2). Mittlerweile wird die Weiterentwicklung der Software von der Klaus Tschira Stiftung gefördert. So wurde das Projekt im Jahre 2007 der *open source* Gemeinde übergeben und ist seit Juni 2007 in der stabilen Version 1.12 herunterladbar. Da MMAX2 vollständig auf die Technologien Java und XML setzt, und mittlerweile sogar eine gut dokumentierte API zur Verfügung stellt, bietet es nicht nur eine offene Architektur mit vielen Freiräumen für Entwickler und Programmierter, sondern auch ein hohes Maß an Portierbarkeit auf nahezu jedes beliebige System mit einem *Java Runtime Environment*. Laut Entwicklerteam läuft die Applikation stabil auf Windows, Linux und MAC OS Systemen.

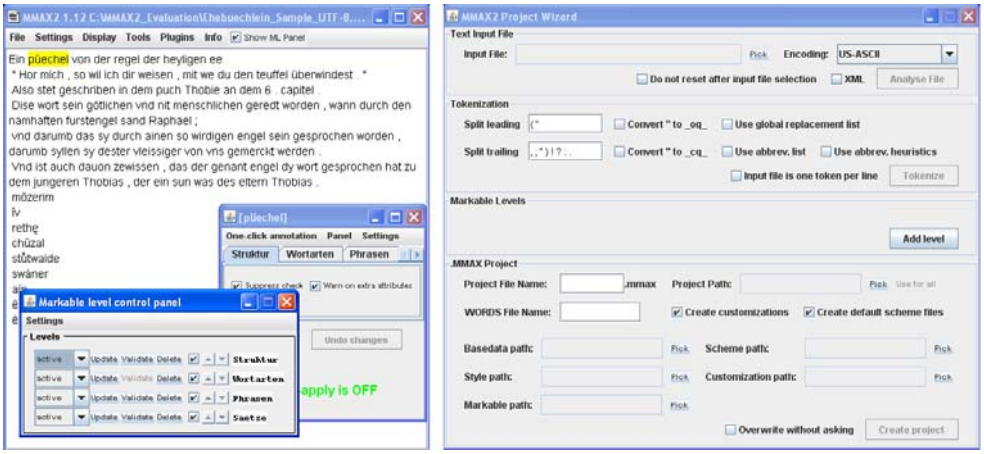

Abbildung 5: Screenshot des MMAX2 Annotationswerkzeugs

[20] Die *Extensible Stylesheet Language* ist eine Sprache zur Formatierung von XML-Dokumenten.

3.3.4 UAM CorpusTool

Mit dem UAM (*Universidad Autonoma de Madrid*) CorpusTool stellt der Entwickler des Systemic Coder ein modernes Annotationswerkzeug für Textkorpora zur Verfügung, welches volle Flexibilität der Annotationsschemata durch einen graphischen Schemaeditor, und Annotation auf beliebig vielen Ebenen durch Stand-off XML gewährleistet. Das Tool in Version 1.3.3 ist weitestgehend ausgreift, wird jedoch trotzdem aktuell noch weiterentwickelt. Von der Installation bis zum ersten Programmstart fällt die Software durch ihre einfache Bedienung auf. Wizards und Assistenten unterstützen den Anwender bei fast jedem Schritt auf dem Weg zur Erstellung eines neuen Projekts.

Dabei steht als erstes die Definition der Annotationsbasis in Form von beliebigen Textdateien an, welche keinerlei Vorverarbeitung durch Tokenizer oder ähnlichem bedürfen. Sobald die Textdateien mit grundlegenden Metainformation zu Sprache und Zeichensatz versehen sind, werden die Rohtexte in das aktuelle Annotationsprojekt aufgenommen. Über den *Add Files Assistant* können dem Projekt zu jedem Zeitpunkt Texte hinzugefügt, oder wieder entnommen werden. Der *Create Layer Assistant* unterstützt den Annotator bei der Neuerstellung von Annotationsebenen. UAM CorpusTool unterscheidet hierbei zwischen Annotationsebenen, welche sich auf das gesamte Dokument beziehen, und solchen, welche auf das Markup bestimmter Textsegmente abzielen. Jede Auszeichnungsebene muss mit einem entsprechenden Annotationsschema verknüpft werden, welches entweder von anderen Ebenen kopiert, oder über den graphischen Schemaeditor neu erstellt werden kann. Spätestens bei der Bedienung des Schemaeditors wird die Verwandtschaft des Werkzeugs zum Systemic Coder deutlich. Wie schon der Vorgänger stellt auch UAM CorpusTool Annotationsschemata als hierarchische Netzwerke (*system networks*) dar, wie sie aus der systemischen Linguistik bekannt sind. Der Editor erlaubt die Definition beliebiger Verzweigungen und ermöglicht außerdem die Glossierung der einzelnen Knoten, welche zusätzlich mit linguistischen Beispielen verknüpft werden können.

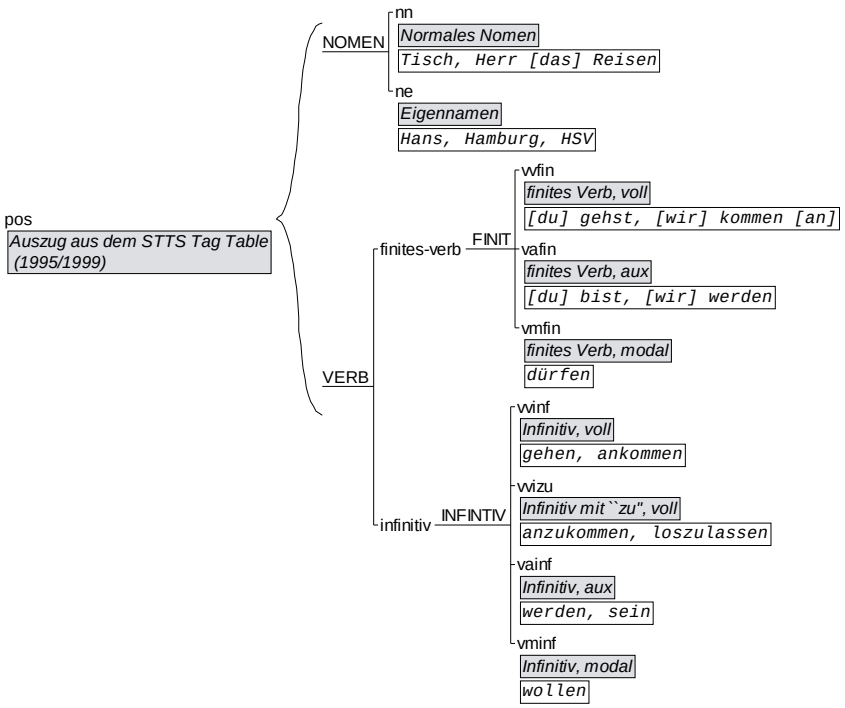

Abbildung 6: Screenshot des MMAX2 Schemaeditors

Die Schemata werden bei Beendigung des Toos als XML-Dateien gespeichert und können während der laufenden Annotation jederzeit im Editor geändert werden. Alle Änderungen im Schema, wie beispielsweise die Umbenennung eines Knotennamens, werden automatisch auf die bestehende Annotation übertragen. Annotierte Elemente werden im Gesamttext hervorgehoben und können jederzeit verändert oder gelöscht werden.

Nach Speicherung der Annotation kann diese auch als XML-Markup im Werkzeug angezeigt werden. Die Annotationen werden im Stand-off Format als separate XML-Dateien für jede Ebene, getrennt vom Originaltext, gespeichert. Die Referenzierung des Originaltextes durch die Annotationsebenen erfolgt stets durch Angabe einer Start- und einer End-ID, da CorpusTool den Originaltext beim anfänglichen Import in das Projekt zeichenweise zerlegt und dabei jedes Zeichen mit einer ID versieht. In einem Ordner namens *Corpus* werden alle Originaltexte abgespeichert, der Ordner *Analyses* enthält die annotierten Ebenen eines Textes sowie eine Datei mit Metainformationen zum Text. Die Annotationsschemata finden sich im Verzeichnis *Schemes*. CorpusTool bietet neben der Hauptfunktionalität Textdateien auf beliebigen Ebenen mit frei definierbaren Tagsets zu annotieren zudem die Möglich-

keit fertige Annotationen über eine Query Konsole abzufragen und über eine Sammlung statistischer Werkzeuge quantitativ auszuwerten. Das Konzept der Einfachheit wird konsequent in allen Bereichen des Tools durchgehalten, was bis auf die unorthodoxe Visualisierung durchweg positiv zu bewerten ist. Bei der Gestaltung der Oberfläche ist das Bestreben des Autors nach Übersichtlichkeit und Schlichtheit durchaus erkennbar, lediglich die Umsetzung bricht an vielen Stellen mit etablierten Konventionen des Interaktionsdesigns und der Softwareergonomie (vgl. TIDWELL 2005; HERCZEG 2006). So kann Typographie, Größe, Form, Farbgebung und Anordnung von Interaktionselementen, Arbeitsoberflächen und Ausgabefenstern durchaus als unkonventionell bezeichnet werden. UAM CorpusTool wurde mit Hilfe der Skriptsprachen Python und Tcl/Tk implementiert und ist vorerst nicht beliebig portierbar, sondern nur für Windows und Macintosh Systeme verfügbar.

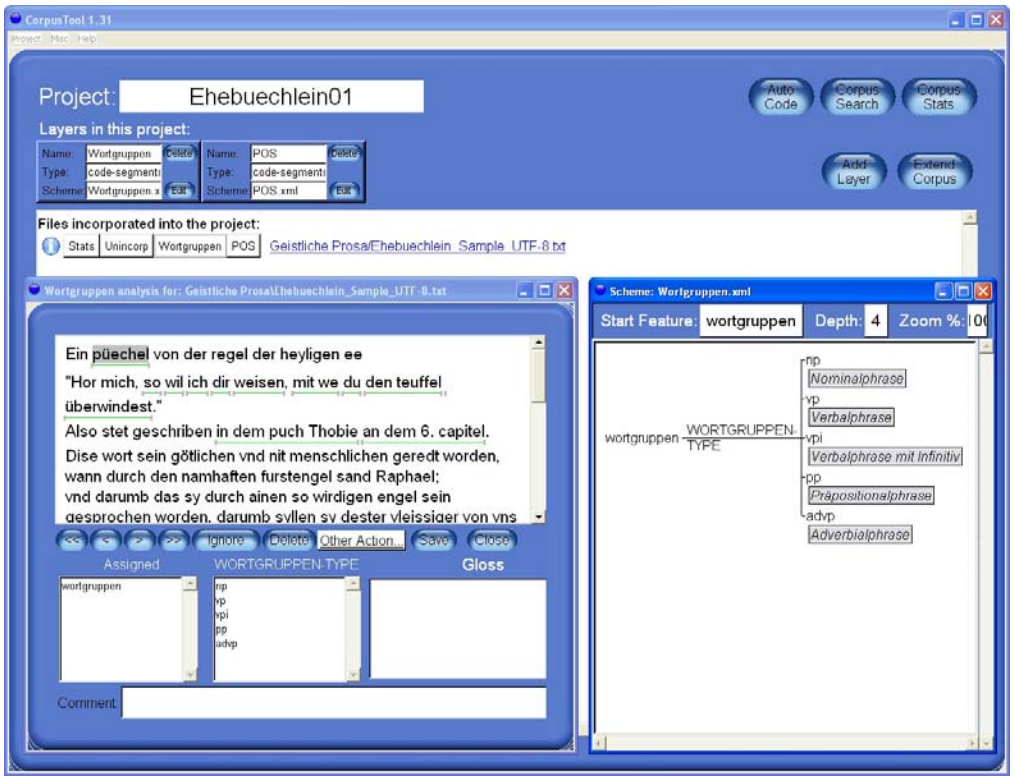

Abbildung 7: Screenshot des UAM CorpusTool

4 Standardisierungsbestrebungen in der Software- Evaluation

Wie geeignet ist ein Programm unter Berücksichtigung verschiedener Umgebungsvariablen zur Erfüllung einer bestimmten Aufgabe eigentlich? Durch die Operationalisierung des Qualitätsbegriffs unter Miteinbeziehung verschiedener kontextueller Parameter liefern Software-Evaluationen Antworten auf diese und ähnliche Fragen. HEINRICH formuliert den allgemeinen Zweck einer Evaluation so:

> Zweck von Evaluation ist es, in einer bestimmten Problemsituation auf der Grundlage sachlicher Merkmale im Hinblick auf die in der Praxis angestrebten Ziele […] die in Betracht kommenden Alternativen entsprechend zu ordnen und so eine Stellungnahme zu ermöglichen, die zu einer Entscheidung führt. Sie sind ein wesentlicher Bestandteil der Willensbildung. (ALBERT 1987, zitiert nach HEINRICH 2000, S. 9)

Die konkrete Problemsituation ist in diesem Fall ein diachrones Annotationsszenario für historische Texte des Deutschen, aus welchem sich bestimmte Anforderungen ergeben. Die in Betracht kommenden Alternativen zur Lösung der Problemsituation stellen im Vorfeld sorgfältig ausgewählte Annotationswerkzeuge dar. In diesem Teil der Arbeit soll untersucht werden, auf welche Standards aus dem Bereich der Software-Evaluation und der Qualitätssicherung bei der qualitativen Bewertung von Callisto, GATE, MMAX2 und UAM CorpusTool zurückgegriffen werden kann.

Dabei finden sich für die Verwendung standardisierter Ansätze gleich mehrere gute Gründe. So wird im EAGLES Bericht von einer Evaluationsprozedur geträumt, die eines Tages soweit standardisiert ist, dass beliebige Softwareprodukte vollautomatisch evaluiert werden können:

> […] the ultimate goal for evaluation for Language Engineering would be to have some automated procedure into which new products are fed which are then evaluated and compared with other products (EAGLES 1999a, S. 18)

Ob es solch ein parametrisierbares Evaluationsframework in absehbarer Zeit geben wird, sei dahingestellt[21], Tatsache ist jedoch, dass es auf dem Gebiet der Standardisierung von Software-Evaluation bereits in den 90er Jahren einige bedeutsame Entwicklungen gegeben hat. Standardisierte Vorgehensweisen versprechen nicht nur die künftige Automatisierung von Evaluationen, sondern bieten schon jetzt wiederverwendbare Formalismen und Strategien zur Bewertung von Programmen aus unterschiedlichsten Anwendungsgebieten.

Für die standardisierte Evaluation von Annotationswerkzeugen scheinen vor allem die Qualitätsnormen der ISO (*International Organization for Standardization*) und der IEC (*International Electrotechnical Commision*) sowie das Framework zur Evaluation von VNS-Software (Software zur Verarbeitung Natürlicher Sprache) der EAGLES Evaluationsarbeitsgruppen von Nutzen zu sein.

4.1 ISO/IEC Normen

Sowohl ISO als auch IEC entwickeln internationale Standardisierungsrichtlinien für nahezu alle industriellen Bereiche und sind als unabhängige Normierungsinstanz weltweit anerkannt:

> ISO (International Organization for Standardization) and IEC (International Electrotechnical Commission) together form the specialized system for worldwide standardization. International Standards are developed by technical committees, whose membership comes from national bodies who are members of ISO or from IEC participants. ISO and IEC committees collaborate in fields of mutual interest. (EAGLES 1999a, S. 5)

Dabei beschäftigt sich die ISO Serie 9000 in erster Linie mit Standardisierungsvorschlägen zur Qualitätssicherung, wobei für die Evaluation eines Annotationstools vor allem der ISO 9126 Standard zur Sicherung der Produktqualität von Software relevant ist. Die erste Auflage der Norm entstand bereits 1991 im Rahmen eines JTC (*Joint Technical Committee*) der beiden Organisationen ISO und IEC. Da vor allem in der deutschsprachigen Literatur gemeinsam erarbeitete Normen meist ohne explizite Nennung der IEC zitiert wird, soll hier analog verfahren werden. Wenn

[21] Es gibt bereits einige semi-automatische Evaluationstools wie z.B. *AutoEval* und *Missplel*, allerdings nicht für den speziellen Bereich der Annotationswerkzeuge (vgl. BIGERT et al. 2003).

also im Folgenden von ISO 9126, ISO 14598, oder ISO 25000 die Rede ist, impliziert dies immer auch das Mitwirken des IEC.

In der Urversion des ISO Standards wurden die Grundpfeiler des Qualitätsmodells in Form von sechs essentiellen Qualitätskriterien für Software gesetzt: Funktionalität, Zuverlässigkeit, Benutzbarkeit, Effizienz, Änderbarkeit und Übertragbarkeit. Für jedes dieser Qualitätskriterien existieren diverse Unterkriterien, welche je nach Evaluationsgegenstand zur weiteren Untersuchung herangezogen werden können. Durch dieses System aus Qualitätskriterien auf oberster Ebene (*top level characteristics*) und daraus abgeleiteten Unterkriterien (*sub-characteristics*) ergibt sich eine baumartige Evaluationshierarchie, an deren Ende Attribute stehen, die mit vorher festgelegten Metriken eindeutig gemessen und bewertet werden können:

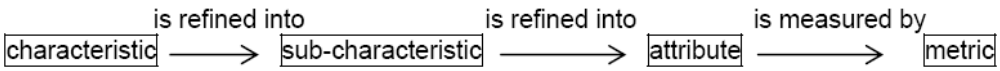

Abbildung 8: Beziehung zwischen den einzelnen Elementen des ISO 9126 Qualitätsmodells, nach LOSAVIO (2003, S. 136)

Im Zeitraum 2001-2004 veröffentlichte das ISO/IEC SC7 W6 (*sub-committee 7, working group 6*) schließlich eine erweiterte Version des Standards. In der neuen Auflage der Norm wird nun ein Qualitätsmodell vorgestellt, welches den Qualitätsbegriff aus drei unterschiedlichen Perspektiven angeht: „ISO/IEC 9126 improved the original model by defining subcharacteristics, separating internal and external characteristics and adding quality in use characteristics“ (AZUMA 2001, S. 340). Der Standard gliedert sich dementsprechend in die vier Teildokumente: ISO 9126-1: *Qualitätsmodell*, ISO TR 9126-2: *Externe Metriken*, ISO 9126-3: *Interne Metriken* und ISO 9126-4: *Metriken für Gebrauchsqualität*[22].

Das grundlegende Qualitätsmodell sieht nun also eine dreigeteilte Auffassung von Softwarequalität vor. Die interne Qualität einer Software beschreibt deren messbaren Produkteigenschaften und bezieht sich auf statische Attribute des Programms, wie etwa die Anzahl an Codezeilen oder Funktionen. Typischerweise sind Software-Entwickler daran interessiert die interne Qualität eines Programms zu er-

[22] Gebrauchsqualität ist in der deutschen Forschungsliteratur zu diesem Thema die übliche Übersetzung für den englischen Ausdruck *quality in use*.

fassen, um etwa Erkenntnisse darüber zu gewinnen ob, und in welchem Rahmen, die Software veränderbar ist, oder ob die Implementierung bestimmter Standards erfüllt ist. Externe Qualität bezieht sich hingegen auf den Zweck, den die Software erfüllen soll, also auf das Verhalten der Software in einer bestimmten Systemumgebung oder einem spezifizierten Aufgabenbereich. Charakteristisch für diese Qualitätsperspektive wäre z. B. die Frage, wie viel Zeit eine Applikation benötigt, um eine bestimmte Aufgabe zu erfüllen. Sowohl interne als auch externe Qualität sind durch die sechs Qualitätskriterien, aus denen sich entsprechende Metriken ableiten lassen, operationalisierbar. Beide Qualitätsbegriffe werden durch eine objektive, methodische Herangehensweise zur Ermittlung „harter Daten" (HEGNER 2003, S. 16) beschrieben. Der Begriff der Gebrauchsqualität wird hingegen durch die subjektive Messung so genannter „weicher Daten" (ebd.) definiert und bezeichnet den Effekt, den die Software auf den Benutzer hat. Hierzu wurden neue Qualitätskriterien, wie etwa Effektivität, Effizienz, Sicherheit und Zufriedenheit eingeführt. Da Gebrauchsqualität die Interaktion zwischen Benutzer und Software bewertet, kann sie in gewisser Weise als kombinierter Effekt interner und externer Qualitätskriterien bezeichnet werden.

Inhaltlich eng verknüpft mit dem ISO 9126 Standard ist die ISO 14598 Norm für die Evaluation von Softwareprodukten, spezifiziert sie doch ein generisches Modell zur schrittweisen Durchführung einer Evaluation. Dabei werden im ersten Schritt grundlegende Anforderungen, wie z. B. Zweck der Evaluation, Produkttyp und Qualitätsmodell, festgelegt. Im zweiten Schritt werden dann entsprechende Metriken, Einstufungsniveaus und Bewertungskriterien definiert. Die eigentliche Durchführung stellt den letzten Schritt der Evaluation dar. Hier werden schließlich Messungen vorgenommen, Vergleiche angestellt und Bewertungen abgegeben.

Um den rasanten Entwicklungen im Bereich der Informationstechnologie Rechnung zu tragen, arbeiten ISO und IEC seit 2005 an der nächsten Generation von Softwarequalitätsstandards mit dem Arbeitstitel SquaRE (*Software Product Quality Requirements and Evaluation*). Die neue ISO 25000 Serie soll nach und nach den ISO 9126 Standard ersetzen und strebt dabei vor allem eine terminologische Angleichung der thematisch verwandten Standards ISO 9126 und ISO 14598 an (vgl. ABRAN et al. 2005, S. 37ff.). Ziel der neuen Serie ist es, eine einheitliche Sammlung von Standards zur Verfügung zu stellen, welche im Rahmen einer homogenen Ter-

minologie die drei Bereiche Anforderungsspezifikation, Messung und Durchführung der Evaluation gleichermaßen abdecken. Insgesamt soll die ISO 25000 Serie 14 detaillierte Dokumente aus den fünf thematischen Hauptbereichen *Qualitätsmanagement*, *Qualitätsmodelle*, *Qualitätsmaße*, *Qualitätsvoraussetzungen* und *Evaluation von Qualität* enthalten (SURYN & ABRAN 2003, S. 7ff.).

Einen ähnlichen Ansatz verfolgt die EAG-EWG (*EAGLES Evaluation Working Group*), allerdings in spezialisierter Form.

4.2 EAGLES Evaluation Working Group

Die europäische Forschungsgruppe EAGLES begann Ihre Arbeit 1993 mit dem Ziel Standards für den Bereich des *Language Engineering* (LE) zu entwickeln. Dabei oblag es der EAG-EWG standardisierte und wiederverwendbare Evaluationsstrategien für Software aus dem Sektor der natürlichen Sprachverarbeitung zu entwickeln. Da das breite Spektrum an VNS-Software von Rechtschreibkorrekturprogrammen bishin zu maschinellen Übersetzungssystemen reicht, wurde schnell klar, dass es nicht möglich sein würde eine universale Evaluationsvorschrift zu erstellen, welche für alle denkbaren Ausprägungen des VNS-Software Bereichs gleichermaßen gut geeignet ist.

Erklärtes Ziel war deshalb die Entwicklung eines standardisierten und doch flexiblen Frameworks, welches durch Erweiterung und Modifikation die Konstruktion beliebiger spezifischer VNS-Software-Evaluationen ermöglichen sollte. Da der ISO 9126 Standard bereits ein generelles Framework zum Design von Software-Evaluationen unter Verwendung eines operationalisierten Qualitätsbegriffs vorgibt, wurde er als Ausgangspunkt für die Erstellung eines spezifischeren, direkt benutzbaren Frameworks herangezogen. So wie der Standard ISO 9126 mit seinem Softwarequalitätsmodell die Spezialisierung eines allgemein formulierten Qualitätsbegriffs darstellt, so kann auch das EAGLES Framework als weitere Spezialisierung der Softwarequalitätsnorm gesehen werden, hält es doch Evaluationsrichtlinien und Qualitätsmodelle für den spezifischen Bereich von VNS-Software bereit. Durch die konkrete Umsetzung der allgemein gehaltenen ISO Standards hat EAGLES ein Framework für die Evaluation von VNS-Software geschaffen, indem es das Qualitätsmodell aus ISO 9126 mit der Strukturierung des Evaluationsprozesses aus ISO

14598 kombiniert und so direkt benutzbar macht. Die vielfach praktisch angewandten EAGLES Guidelines helfen den ISO-Autoren wiederum ihre Standards weiter zu verbessern und zu modifizieren:

> The current round of EAGLES work has brought the EvaluationWorking Group into close contact with ISO activities. […] The experience gained in carrying out the specialisation becomes feedback for ISO on the theory of software quality, and may, over time contribute to the creation of ISO standards for specialised softwares. (EAGLES 1999a, S. 17)

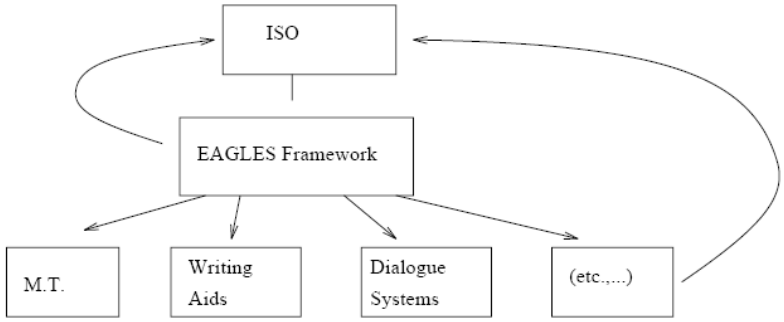

Abbildung 9: Zusammenhang zwischen ISO und EAGLES, nach EAGLES (1999, S. 17)

So entstand in der Anfangsphase des Projekts von 1993-1995 die erste Definition eines ISO-basierten Qualitätsmodells für VNS-Software, in Form eines hierarchisch strukturierten Merkmalskatalogs, auf dessen unterster Ebene messbare Attribute mit einer Verknüpfung zu entsprechenden Metriken standen (vgl. HOVY et al. 2003, S. 46f.). Beschränkten sich die Möglichkeiten des Frameworks anfangs noch auf die Evaluation von Schreib- und Übersetzungshilfen, so gab es in der zweiten Projektphase von 1995-1996 nochmals beträchtliche Erweiterungen und Verallgemeinerungen, um die EAGLES Guidelines auch zur Evaluation anderer VNS-Applikationen verwenden zu können.

Das Partner Programm ISLE, welches ursprünglich mit der EAG-EWG zusammen an einem allgemeinen VNS-Evaluationsframework arbeitete, begann 1999 die Arbeit an einem speziell für den Bereich der maschinellen Übersetzung zugeschnittenen Evaluationsframework. Das Ergebnis war FEMTI (*Framework for Machine Translation Evaluation*), ein Framework zur Evaluation von Übersetzungssystemen, welches deutliche Ähnlichkeiten zu ISO 9126 und EAGLES aufweist.

Nachdem in diesem Abschnitt die gängigen Konzepte und Projekte im Bereich der standardisierten Software-Evaluation aufgezeigt wurden, soll im nachfolgenden Abschnitt ausgehend von den ISO 9126 und EAGLES Guidelines ein Evaluationsplan für die qualitative Bewertung von Annotationswerkzeugen für diachrone Korpora erstellt werden.

4.3 Standardisierung des Evaluationsprozesses

Wie bei jeder Evaluation ist es auch bei der Bewertung von Softwareprodukten sinnvoll, sich an einer vorher spezifizierten Ablauflogik zu orientieren, welche den Evaluationsprozess modelliert und dadurch vorgibt. In der Literatur gibt es durchaus unterschiedliche Vorschläge und Richtlinien für den Ablauf und die Gestaltung von Software-Evaluationen, letzten Endes haben jedoch alle einige grundlegenden Prozessstadien gemeinsam.

Am Anfang steht immer die Formulierung von Werte- oder Anforderungskriterien, gefolgt von der Erstellung geeigneter Metriken und Einstufungsniveaus. Unter Metriken werden dabei einerseits Messwerte für die verschiedenen Systemattribute verstanden, die auf einer definierten Messskala abgelesen werden können, und andererseits Messmethoden, um diese Werte zu ermitteln (vgl. UNDERWOOD & LISOWSKA 2006, S. 2481). Einstufungsniveaus geben den Grad der Erfüllung eines bestimmten Attributs wieder und dienen als Grundlage für eine abschließende Bewertung. Nach diesen beiden vorbereitenden Schritten kann schließlich der analytische Teil der Evaluation beginnen. Es erfolgen Messungen anhand der vorher aufgestellten Metriken und ein Vergleich der Ergebnisse unter Zuhilfenahme einer Bewertungsskala. Die Gewichtung und Relationierung der einzelnen Messungen zu einem aussagekräftigen Gesamturteil wird häufig auch als Synthese bezeichnet (vgl. HEGNER 2003, S. 9). ISO 14598 schlägt hierzu ein grundlegendes Prozessmodell für Software-Evaluationen vor, welches von ISO 9126, EAGLES und FEMTI (vgl. HOVY 2002, S. 45) gleichermaßen aufgegriffen wird. Da die ISO 9126 Vorschläge jedoch zu allgemein gehalten sind, und FEMTI die Evaluation von Systemen zur maschinellen Übersetzung im Speziellen beinhaltet, soll nachfolgend für die Evaluation von Annotationswerkzeugen die Evaluationslogik des EAGLES Frameworks verwendet werden.

Dabei hat EAGLES auf Grundlage des ISO 9126 Standards einen 7-stufigen Leitfaden[23] zur Durchführung von VNS-Software-Evaluationen erstellt, welches 1999 auf der EELS Konferenz in Hoevelaken erstmals präsentiert wurde. Er stellt die Quintessenz des knapp 300 Seiten umfassenden Berichts zum Thema „Evaluation of Natural Language Processing Systems" (vgl. EAGLES 1999a) dar und soll als allgemeine Richtlinie für die Evaluation von VNS-Software dienen. Die sieben Stufen des EAGLES Leitfadens beinhalten neben den bereits genannten grundlegenden Prozessstadien einer Evaluation außerdem noch die Modellierung virtueller Nutzer und deren Arbeitskontext (vgl. SPARCK-JONES & GALLIERS 1992, S. 11) sowie die Spezifizierung des Testmaterials. In der nachfolgenden Auflistung sind überblicksartig die sieben Stufen einer Software-Evaluation nach dem EAGLES Framework zu sehen:

The EAGLES 7-step recipe (vgl. EAGLES 1999b):

Step 1: Why is the evaluation being done?
Step 2: Elaborate task model.
Step 3: Define top level quality characteristics.
Step 4: Produce detailed requirements for the system under evaluation, on the basis of step 2 and 3.
Step 5: Devise the metrics to be applied to the system for the requirements produced under step 4.
Step 6: Design the execution of the evaluation.
Step 7: Execute the evaluation.

In den nächsten Abschnitten werden diese Einzelschritte konkret auf die Evaluation von Annotationswerkzeugen für das historische Syntaxprojekt DiSynDe angewandt. Dabei wird im ersten Schritt kurz die eigentliche Motivation hinter der Evaluation aufgezeigt. Im Anschluss soll der Gebrauchskontext durch Modellierung der späteren Aufgabe und einer Charakterisierung der beteiligten Nutzer skizziert werden, um darauf aufbauend vorerst noch allgemein gehaltene Anforderungen an ein Annotationswerkzeug formulieren zu können. Auf Basis dieser Anforderungen wird

[23] Nur online verfügbar, unter [http://www.issco.unige.ch/projects/eagles/ewg99/7steps.html] – Zugriff am 10.06.2008.

im vierten Schritt ein mehrstufiges Qualitätsmodell erstellt werden, auf dessen oberster Ebene sich ausgewählte Qualitätskriterien des ISO 9126 Standards befinden. Die unterste Ebene stellen messbare Systemattribute dar, während im fünften Schritt die Erstellung von Metriken zur Messung, Gewichtung, Einstufung und Bewertung der einzelnen Attribute vorgesehen ist. Da insgesamt 30 Systemattribute untersucht werden sollen, gestaltet sich dieser Schritt entsprechend umfangreich. Bei der Planung zur Durchführung der Evaluation werden die Testmaterialien und das Testszenario spezifiziert. Der letzte Schritt, die tatsächliche Durchführung der Evaluation, beinhaltet schließlich die Messung der verschiedenen Systemattribute, welche in komprimierter tabellarischer Form dargeboten werden, sowie die Auswertung und Diskussion der Evaluationsergebnisse.

5 Evaluation von Annotationswerkzeugen

Die nachfolgenden Unterkapitel 5.1–5.7 orientieren sich inhaltlich im Wesentlichen am 7-Punkte Plan des EAGLES Frameworks zur Konzipierung und Durchführung einer Evaluation von VNS-Software.

5.1 Motivation der Evaluation

Den ersten Schritt auf dem Weg hin zu einer umfassenden Software-Evaluation stellt die Formulierung des Evaluationszwecks dar. Dazu ist es unerlässlich das Evaluationsobjekt und dessen Anwendungskontext genau zu spezifizieren. Wie im zweiten Kapitel bereits ausführlich dargelegt wurde, steht das diachrone Syntaxprojekt DiSynDe vor der anspruchsvollen Aufgabe, historische Texte des Deutschen auf unterschiedlichsten linguistischen Ebenen entlang einer Zeitachse zu annotieren. Das letztendliche Ziel des Projekts ist die Erstellung eines syntaktisch annotierten, diachronen Korpus des Deutschen. Die Motivation dieser Evaluation besteht deshalb in erster Linie darin, zu untersuchen, wie gut und auf welche Weise bestehende Werkzeuge das Projekt bei der Annotationsarbeit unterstützen können.

Dabei wurden im vorhergehenden Kapitel über 50 Annotationswerkzeuge auf grundlegende Basisfunktionalitäten und -kriterien hin untersucht, um nach einem zweistufigen Filterungsprozess potentielle Kandidaten für das DiSynDe Szenario zu ermitteln. Das Ergebnis dieser Vorauswahl sind vier Annotationswerkzeuge, welche gleichzeitig die Evaluationsobjekte der angestrebten vergleichenden Evaluation darstellen. Die Werkzeuge sollen hinsichtlich der beiden Qualitätskriterien Funktionalität und Benutzbarkeit evaluiert werden, welche es in den nachfolgenden Schritten sukzessive in kleinere, messbare Systemattribute zu zerlegen gilt. Die Einzelmessungen der Systemattribute zeigen einerseits individuelle Stärken und Schwächen eines Werkzeugs auf, und lassen sich andererseits gewichten und zusammenfassen, um Gesamtaussagen über Qualität und Eignung der Software machen zu können.

5.2 Modellierung des Gebrauchskontexts

Nachdem festgelegt wurde, was genau untersucht werden soll und welche Motivation sich hinter der Evaluation verbirgt, gilt es ein adäquates Aufgabenmodell mit allen zu erwartenden Endnutzern zu erstellen.

5.2.1 Aufgabe und Arbeitsumfeld

Die Aufgabe, zu deren Erfüllung die Annotationswerkzeuge genutzt werden sollen, besteht grob gesehen darin, ein Korpus aus historischen Originaltexten unter Einsatz computergestützter Annotationswerkzeuge in ein diachrones Korpus aus syntaktisch annotierten Texten zu verwandeln. Digitalisiert, formal normalisiert und mit einem einheitlichen Dateiheader versehen, werden die Originaltexte vom Annotator mit Hilfe des Annotationswerkzeugs geöffnet. Das OCR[24]-Scannen der Texte, deren formale Aufbereitung unter besonderer Berücksichtigung der korrekten Darstellung von Sonderzeichen erfolgt, und die Erstellung eines Dateiheaders sind zwar nötige Vorbereitungsschritte auf dem Weg hin zur syntaktischen Annotation, gehören aber nicht zum eigentlichen Annotationsprozess.

Der Annotator reichert dem Zufolge lediglich den Originaltext auf der ihm zugewiesenen Annotationsebene mit syntaktischen Informationen aus einem entsprechenden Annotationsschema an, und wird dabei vom Werkzeug durch graphische Aufbereitung der Daten und qualitätssichernde Maßnahmen unterstützt. Der auf einer bestimmten sprachlichen Ebene annotierte Text wird als Annotationsdatei aus dem Tool exportiert und in einem Korpus teilannotierter Texte zwischengespeichert. Ein Text bleibt solange in diesem *Zwischenkorpus*, bis er von allen beteiligten Annotatoren auf allen angedachten Ebenen unter Zuhilfenahme des Annotationswerkzeugs ausgezeichnet ist. Erst dann wird das Dokument Teil des diachronen Endkorpus, welches nur vollständig annotierte Texte enthält.

[24] *Optical Character Recognition* bezeichnet die optische Schrift- und Zeichenerkennung per Scanner.

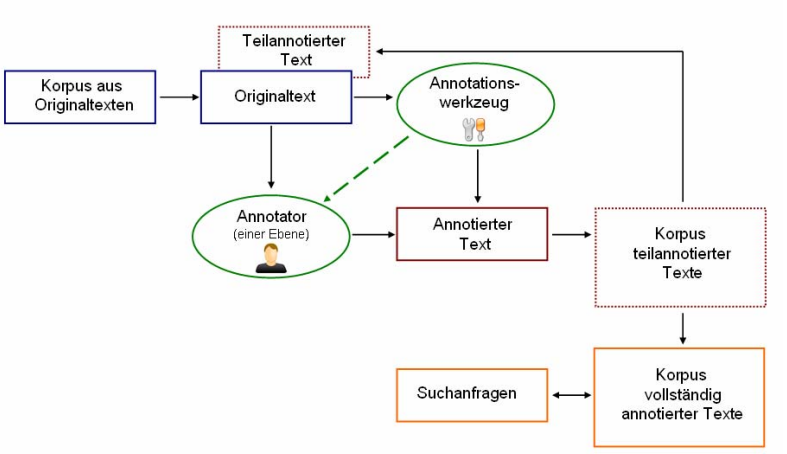

Abbildung 10: Arbeitsablauf und Informationsfluss während des Annotationsprozesses - die Schritte *OCR-Scannen*, *Normalisierung* und *Erstellung eines Dateiheaders* werden vorausgesetzt.

5.2.2 Nutzergruppen

Bei der Formulierung von Anforderungen an ein Annotationswerkzeug sind nicht nur die Funktionalitäten in Hinblick auf die Projektziele, sondern vor allem auch die besonderen Bedürfnisse der Anwenderschaft zu berücksichtigen. REIDSMA et al. (2004, S. 3) geben eine grundlegende Klassifikation für Benutzer von Annotationswerkzeugen und unterscheiden dabei Annotatoren, Annotationskonsumenten und Entwickler, wobei sich die letzte Gruppe wiederum in Korpus- und Systementwickler unterteilen lässt. Auch DYBKJÆR et al. (2001b, S. 3) schlagen in ihrem Abschlussbericht über Anforderungskriterien für multimodale Annotationswerkzeuge eine ähnliche Einteilung vor, sparen jedoch die Klasse der Annotationskonsumenten, also der Benutzer die das annotierte Korpus später benutzen und abfragen wollen, aus.

Die DiSynDe Gruppe, welche für die technische Umsetzung des Projekts durch die Bereitstellung von Annotationswerkzeugen und -schemata verantwortlich ist, fällt eindeutig in die Kategorie der Entwickler. Zum einen kommt dieser Gruppe die Aufgabe der Korpusentwicklung durch die Spezifikation und Anpassung eines geeigneten Annotationsschemas, zum anderen die der Systemweiterentwicklung durch Modifizierung und Anpassung eines geeigneten Annotationswerkzeugs, zu. Die besonderen Bedürfnisse dieser Anwendergruppe unterscheiden sich grundle-

gend von denen der Annotatorengruppen, und sollen deshalb nicht Gegenstand der angedachten Evaluation sein.

Die Gruppe der Annotatoren zeichnet sich hingegen durch ihr geringes Interesse an technischen Details, wie etwa der internen Datenrepräsentation, der Implementierungssprache, oder der Systemarchitektur aus. Vielmehr fordern diese Nutzer ein computergestütztes Werkzeug, welches sie auf intuitive und leicht verständliche Art und Weise bei ihrer Annotationsarbeit unterstützt. Typischerweise sind diese Anwender Experten auf ihrem Fachgebiet und sehen ein Annotationswerkzeug lediglich als Instrument, um ihre Arbeit noch effizienter zu gestalten (vgl. DYBKJÆR et al. 2001b, S. 3). Die Mitglieder des DiSynDe Projekts können als typische Annotatoren mit Expertenwissen im Bereich der historischen Syntax des Deutschen charakterisiert werden. Da die Annotationen während der gesamten Pilotphase jedoch ständig auf Qualität und Konsistenz überprüft werden müssen, sind die Annotatoren auch in gewisser Weise als Annotationskonsumenten zu verstehen, da sie ständig den letztendlichen Zweck der Auszeichnung, nämlich die Erstellung einer historischen Syntax durch diachrone Korpusabfragen, im Auge behalten müssen. Somit gilt es einen virtuellen Nutzer zu modellieren, der in erster Linie möglichst effizient seine Annotationsaufgabe verrichten möchte und dabei doch stets den letztendlichen Zweck der Annotation im Hinterkopf behält. Die besonderen Bedürfnisse und Anforderungen dieser Nutzergruppe werden im nächsten Schritt modelliert.

5.3 Anforderungsanalyse

Ausgehend von den eingangs beschriebenen Projektzielen und der Charakterisierung der Anwenderschaft, lassen sich nun allgemeine Anforderungen an ein Annotationswerkzeug für diachrone Korpora erstellen. REIDSMA et al. verdeutlichen den Zusammenhang zwischen Projektkontext und Anforderungsanalyse: „First, based on the analysis of the project needs, a list of requirements for annotation tools is defined […]. Next, the ‚evaluation criteria' are derived" (2004, S. 1).

Als Ausgangspunkt für die Definition projektspezifischer Anforderungen dient der Aufsatz von DIPPER et al. (2004, S. 55f.) zur Evaluation von einfachen Annotationswerkzeugen. Die hier definierten Anforderungen sollen im nächsten Evaluati-

onsschritt ausgehend von den ISO 9126 Qualitätskriterien und deren Unterkriterien in einen detaillierten Katalog messbarer Attribute übertragen werden.

Zeichensatz Die zu verarbeitenden Rohdaten liegen vor dem Annotationsprozess als uneinheitliche Digitalisate historischer Texte vor. Uneinheitlich deshalb, weil zumindest das vorläufige Pilotkorpus ausschließlich Texte enthält, welche in Bezug auf Form und Dateiheader keinem gemeinsamen Standard entsprechen. Dieser Missstand kann unter Zuhilfenahme verschiedener Standardisierungsinitiativen, wie etwa TEI oder CES relativ gut beseitigt werden. Auf Ebene der Zeichensätze gilt es vor allem orthographische Probleme zu beseitigen, welche meist von den exotischen Sonderzeichen der historischen Texte herrühren. Bei der genauen Analyse des Pilotkorpus wird deutlich, dass diese Zeichen häufig bereits beim Vorgang des OCR-Scannens nicht richtig erkannt werden, und deshalb einer manuellen, Unicodekonformen, Nachkorrektur bedürfen. Aus diesem Grund muss das spätere Annotationswerkzeug in jedem Fall Unicode unterstützen, um nachträglich orthographische Änderungen im Primärtext vornehmen zu können, aber auch um bereits richtig eingescannte Texte im Annotationsprogramm korrekt darstellen zu können. Auch nach Überführung der Rohdaten in das entsprechende Annotationsformat müssen alle Sonderzeichen erhalten bleiben.

Mehrebenenannotation Da der Primärtext auf unterschiedlichen sprachlichen Ebenen mit Information angereichert werden soll, ist es wichtig, dass das Tool für jede einzelne Ebene Annotationsschemata unterstützt, aber auch die Möglichkeit bietet verschiedene Ebenen miteinander zu vernetzen, bzw. Relationen und Zeiger zu setzen. Wenn es zu Überschneidungen und Verschachtelungen der einzelnen Ebenen kommt, ist eine angemessene Visualisierung unumgänglich. Bei ebenenübergreifenden Phänomenen sollte außerdem ein Mechanismus vorhanden sein, mit dem es möglich ist, ein und demselben Element mehrere parallele Annotationen zukommen zu lassen, da sich die Zuständigkeitsbereiche der einzelnen Ebenen nicht immer klar abgrenzen lassen und neuralgische Phänomene deshalb häufig doppelt annotiert werden. Die Umsetzung einer Mehrebenenannotation und die Forderung nach einem XML-konformen Annotationsformat implizieren die Verwendung eines logisch strukturierten Stand-off XML Formats.

Flexibilität der Annotationsschemata Aus dem Vorhaben die Texte auf mehreren sprachlichen Ebenen zu annotieren, ergibt sich automatisch eine entsprechende Mannigfaltigkeit der Annotationsschemata[25]. Zwar untersucht HEILEMANN (2008) in seiner Arbeit über „Informationsstrukturierung für die syntaktische Annotation eines diachronen Korpus des Deutschen" bereits standardisierte Schemata, um die Einheitlichkeit der Annotationen auf allen Ebenen zu gewährleisten, dennoch sollte auch ein entsprechendes Annotationswerkzeug zusätzliche Konsistenz- und Vollständigkeitsprüfungen der Annotationen ermöglichen. Dabei sind Konsistenz und Vollständigkeit wichtige Kriterien für die Qualitätssicherung der Annotationen (DIPPER et al. 2004, S. 55), vor allem in Hinblick auf eine spätere Kooperation mit anderen diachronen Annotationsprojekten wie etwa dem DDD. An eine Zusammenführung zweier unterschiedlicher Projekte ist nicht zu denken, wenn die Textkorpora in sich unstimmig, weil uneinheitlich und unvollständig annotiert, sind.

Eine weitere wichtige Forderung in diesem Bereich ist die Definition und Einbindung eigener Schemata für jede Ebene der Annotation. Eine Anpassung und Änderung der verwendeten Annotationsschemata, möglichst über eine graphische Oberfläche und während des laufenden Annotationsprozesses, ist deshalb notwendig, weil die Schemata in der Pilotphase noch nicht vollständig ausgereift sind. Vielmehr geht es in dieser Phase des Projekts gerade darum verschiedene Schemata auszuprobieren und bei auftretenden Problemen zu modifizieren. Falls im Laufe des Annotationsprozesses also klar wird, dass das verwendete Tagset ein bestimmtes Phänomen nicht zufriedenstellend abdeckt, muss es möglich sein das Schema abzuändern oder entsprechend zu erweitern.

Flexibilität der Ein- und Ausgabe Durch die Heterogenität der Originaltexte, die unterschiedlichen Annotationspraktiken der einzelnen Gruppen und nicht zuletzt die Forderung nach Wiederverwendbarkeit der Daten, ergibt sich als weitere Anforderung an die Software ein hohes Maß an Flexibilität bezüglich der Ein- und Ausgabe. Flexibilität bei der Dateneingabe bedeutet die Möglichkeit Dokumente

[25] Dies wurde bereits beim DiSynDe Workshop zur Koordination des Pilotprojekts deutlich (Oktober 2007), denn bei den exemplarischen Analysen der einzelnen Annotationsgruppen wichen die Darstellungsmodi zum Teil erheblich voneinander ab.

verschiedenster Formate zu importieren oder falls nötig in ein passendes Dateiformat zu konvertieren. Die Werkzeuge müssen außerdem in der Lage sein, die annotierten Texte in einem standardisierten Format, also XML basiert, abzuspeichern. Falls aus Gründen der Systemarchitektur die Annotationen in einem proprietären Format gespeichert werden, sollte zumindest eine Exportfunktion für ein standardisiertes XML Format vorhanden sein.

Schlichtheit der Anwendung Mit der zunächst allgemein gehaltenen Forderung nach Schlichtheit der Anwendung soll den Bedürfnissen der bereits im Vorfeld charakterisierten Benutzer Rechnung getragen werden. Da das technische Verständnis der meisten DiSynDe Annotatoren relativ gering ist, sollte die Anwendung sowohl in ihrer Funktionsweise als auch in der Visualisierung einfach gehalten sein und technische Aspekte, wie etwa Markupsprachen- und Implementierungsdetails, weitestgehend verbergen. Dabei gilt auf funktionaler Ebene, dass die Software all das leisten sollte, was für eine diachrone Annotation erforderlich ist, aber nach Möglichkeit auch nicht mehr. Darüber hinausgehende Leistungsmerkmale sollten zumindest ausgeblendet werden können. Die Visualisierung eines angemessenen Tools muss ein möglichst intuitives und effizientes Arbeiten, ohne großen Lernaufwand, unterstützen. Der Gebrauch von Metaphern aus anderen, bereits bekannten Softwareprodukten steigert die Akzeptanz des Annotationswerkzeugs für unerfahrene Benutzer. Durch einfaches und ergonomisches Design soll zudem die Attraktivität des Tools und die Erlernbarkeit für technisch unversierte Benutzer erhöht werden. Mit *Schlichtheit der Anwendung* ist darüber hinaus aber auch die unkomplizierte Beschaffung, Installation und Konfiguration der Software gemeint.

Adaptierbarkeit der Software Neben der grundlegenden Forderung, die Applikation so schlicht wie möglich zu halten, sollte dennoch auch die Möglichkeit bestehen die Software an Änderungen und Entwicklungen des Annotationsszenarios anzupassen. Mit dem Begriff *Adaptierbarkeit* ist die Anpassung des Tools an die Bedürfnisse der Annotatoren durch diese selbst gemeint. So kann es sinnvoll sein anfangs bestimmte Funktionen auszublenden, um die Benutzer nicht zu überfordern und ihnen eine kognitive Überlastung zu ersparen. Später ist oft eine Anpassung an die gesteigerten Systemkenntnisse des Benutzers nötig. Um auch fortgeschrittenen

Annotatoren ein möglichst effizientes Arbeiten zu ermöglichen, soll das System deshalb z. B. durch selbst definierbare Shortcuts, anpassbare Menüleisten, oder die Möglichkeit immer wiederkehrende Arbeitsabläufe zu automatisieren dem Anwender entgegenkommen. Es kann nicht davon ausgegangen werden, dass alle Mitglieder der Annotatorengruppen von Anfang denselben Wissenstand besitzen, bzw. gleich schnell lernen mit dem System umzugehen. Die Möglichkeit einer individuellen Anpassung an den jeweiligen Anwender ist also eine wichtige Forderung.

Automatisierbarkeit der Software Um den Annotationsprozess auf lange Sicht effektiv unterstützen zu können, ist es erforderlich, dass die Software aus bereits gespeicherten Annotationen lernt und entsprechende Alternativen für selektierte Annotationseinheiten unterbreitet. Optimalerweise ist es sogar möglich, die Anwendung zu trainieren oder zu programmieren, um eine spätere Automatisierung der Annotation, zumindest in Teilbereichen, wie etwa dem POS-Tagging, zu ermöglichen. Auch die Verwendung eines Tokenizers, zur Erstellung grundlegender Annotationseinheiten, zählt zu diesem Anforderungsbereich.

Verteilung der Arbeitsabläufe Die Arbeitsabläufe im DiSynDe Projekt können in dreierlei Hinsicht als *verteilt* bezeichnet werden. Erstens erfolgt eine gruppenweise Arbeitsteilung und damit Verteilung je nach Expertise der einzelnen Mitglieder. Zweitens sind die verschiedenen Gruppen räumlich voneinander getrennt, wobei nicht nur die unterschiedlichen Gruppen, sondern in den meisten Fällen auch die Mitglieder ein und derselben Gruppe große räumliche Entfernungen trennen. Die Kommunikation zwischen den Gruppen und somit die Synchronisierung der Arbeitsabläufe ist also erschwert. Ein ideales Annotationstool bietet deshalb Möglichkeiten zur asynchronen Kommunikation, beispielsweise durch Kommentare oder ein integriertes Diskussionsforum. Durch die räumliche und zeitliche Verteilung, ist eine Synchronisierung und Koordination der einzelnen Arbeitsschritte durch die Annotationssoftware ebenfalls sinnvoll. Falls diese Anforderung nicht oder nur unzureichend von den Evaluationskandidaten erfüllt wird, sollte untersucht werden, welche anderen Programme zur asynchronen Kommunikation und zum Datenmanagement mit dem Annotationswerkzeug kombiniert werden können.

5.4 Qualitätsmodell

Bevor in diesem Abschnitt ein Modell zur Operationalisierung des Qualitätsbegriffs von Annotationswerkzeugen erstellt wird, soll kurz auf die unterschiedlichen Typen von Qualität und deren Umsetzung in dieser Untersuchung eingegangen werden.

5.4.1 Aufbau des Qualitätsmodells

Wie im vorhergehenden Kapitel bereits beschrieben, liegt dem ISO 9126 Ansatz in seiner aktuellen Auflage eine Dreiteilung des Qualitätsbegriffes zugrunde, der vom EAGLES Framework gleichermaßen verwendet wird.

Bei der Evaluation von Annotationswerkzeugen für diachrone Korpora steht vor allem die interne und die externe Qualität der Tools im Vordergrund stehen. Da die gesamte Evaluation unter besonderer Berücksichtigung der Endnutzer erfolgt, mag es im ersten Moment widersprüchlich erscheinen, den Fokus nicht auf gebrauchsqualitative Merkmale zu legen, da diese Sichtweise von Qualität *per definitionem*[26] die Fähigkeit eines Softwareprodukts angibt, einen Anwender bei der Erfüllung einer bestimmten Aufgabe in einem spezifizierten Gebrauchskontext zu unterstützen.

Eine direkte Evaluation der Tools in Hinblick auf deren Gebrauchsqualität durch Fragebögen oder Usabilitytests ist im Rahmen dieser Arbeit jedoch nicht ohne Weiteres möglich, da im derzeitigen Projektstadium weder verwendbare Annotationsschemata, noch klare Grenzen zwischen den einzelnen Annotationsebenen spezifiziert sind. In Anbetracht der Tatsache, dass die späteren Nutzer in den meisten Fällen noch keinerlei Erfahrung mit computergestützten Annotationshilfen haben, ist eine Usabilitytestreihe, bei der alle vier Tools unter gleich bleibenden Bedingungen getestet werden, im Rahmen dieser Arbeit zeitlich nicht durchführbar. Die extreme räumliche Verteilung der Benutzer auf unterschiedliche Standorte in Deutschland und Österreich sowie die Tatsache, dass für vier unterschiedliche Annotationsgruppen auch jeweils individuelle Testszenarien erdacht werden müssen, erschwert die Durchführung zusätzlich. Somit kann die Interaktion zwischen Anno-

[26] Quality in use: The capability of the software product to enable specified users to achieve specified goals with effectiveness, productivity, safety and satisfaction in specified contexts of use. (EAGLES 1999, S. 39)

tatoren und Annotationssoftware vorerst nur durch die Modellierung virtueller Nutzerklassen analysiert werden.

Auf der Basis von internen und externen Qualitätskriterien soll deshalb die Gebrauchsqualität von Annotationswerkzeugen für diachrone Korpora zumindest prognostiziert werden. Dieses Vorgehen ist im Bereich der Software-Evaluation weit verbreitet. So verwendet selbst die ISO 9126 Norm den Begriff einer „estimated or predicted quality in use" (vgl. UNDERWOOD & LISKOWSKA 2006, S. 2). Um Annahmen zur Gebrauchsqualität von Annotationstools über interne und externe Qualitätskriterien machen zu können, ist es wichtig die Attribute und Metriken so zu wählen, dass später auch wirklich Aussagen über die Fähigkeit der Software einen virtuellen Nutzer bei einer bestimmten Aufgabe zu unterstützen hergeleitet werden können. Da die Gebrauchsqualität nicht direkt messbar ist, der Benutzer jedoch nichts desto trotz im Mittelpunkt der Analyse stehen soll, kann von einer „benutzerzentrierten und aufgabenbasierten Evaluation" (vgl. UNDERWOOD & LISOWSKA 2006, S. 1) gesprochen werden.

Es gilt nun also die im vorhergehenden Schritt allgemein formulierten Anforderungen an ein Annotationswerkzeug für diachrone Korpora in korrelierende interne und externe Qualitätskriterien zu übertragen. Laut DIN 66272[27] (1994, S. 5) zur Bewertung von Softwareprodukten, welche identisch mit der ISO 9126 Norm ist, drücken qualitative Anforderungen die Ansprüche der Umgebung an das betrachtete Softwareprodukt aus und werden durch geeignete ISO 9126 Qualitätskriterien und deren Subkriterien operationalisiert.

[27] Das Deutsche Institut für Normierung vertritt die deutschen Interessen in internationalen Normengremien wie ISO oder IEC. Oftmals werden ISO Normen ins Deutsche übersetzt und dann als DIN Normen bezeichnet.

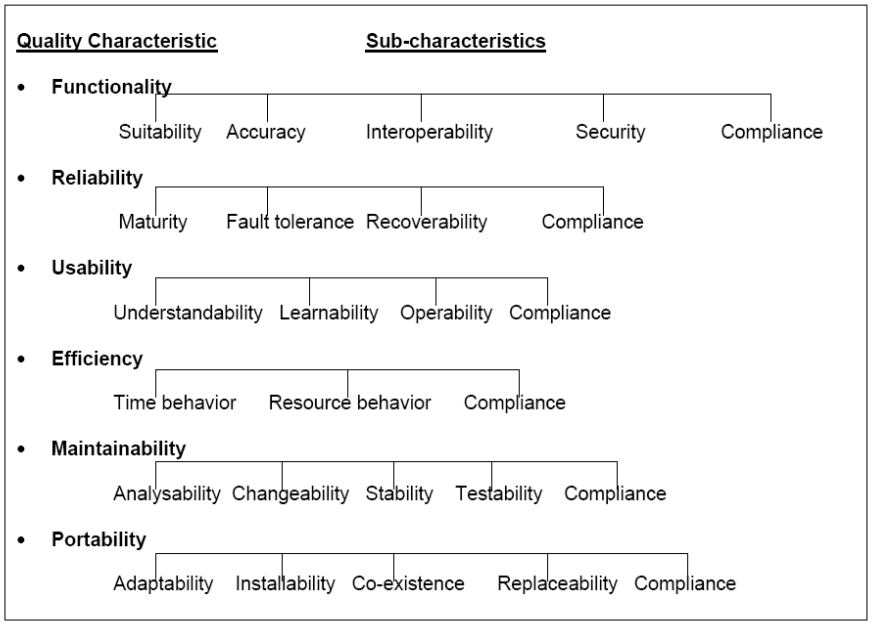

Abbildung 11: ISO 9126 Qualitätskriterien und deren Unterkriterien, nach LOSAVIO et al. (2003, S. 138)

Laut ISO 9126 können von den insgesamt sechs Qualitätskriterien je nach Szenario diejenigen Kriterien verwendet werden, welche das Evaluationsobjekt am besten beschreiben. Für die Evaluation von Annotationswerkzeugen scheinen unter Berücksichtigung des vorher erstellten Nutzermodells die Kriterien Funktionalität und Benutzbarkeit am besten geeignet um Aussagen über die qualitative Eignung der Software machen zu können (vgl. DIPPER et al. 2004, S. 56). Ausgehend von diesen Qualitätskriterien und ausgewählten Unterkriterien wird im Folgenden ein detaillierter Katalog aus Systemattributen und korrelierenden Metriken erstellt, welche es erlauben jedem einzelnen Attribut für jedes Werkzeug einen bestimmten Messwert zuzuordnen.

Da ein Softwareprodukt immer ein komplexes Gebilde mit vielen unterschiedlichen Merkmalen ist, müssen diese auf ein ganzheitliches Qualitätsmodell abgebildet werden, um Gesamtaussagen über die Qualität der Software machen zu können: „[...] quality characteristics are broken down into subcharacteristics, which in turn are broken down into measurable attributes“ (EAGLES 1999a, S. 46). In diesem hierarchischen Modell befinden sich auf der untersten Ebene messbare Attribute, die es erlauben Rückschlüsse über Qualitätskriterien auf übergeordneten Ebenen zu

ziehen, wobei auf der höchsten Ebene das Softwareprodukt als Ganzes steht. Ausgangspunkt für das stufenweise Vorgehen sind die sechs Qualitätskriterien der ISO 9126 Norm und deren Unterkriterien, wobei jedes Kriterium bei der Durchführung der Evaluation auf valide und zuverlässige Weise messbar sein muss. Deshalb werden die ausgewählten Qualitätskriterien Funktionalität und Benutzbarkeit so lange zergliedert, bis messbare Attribute übrig bleiben. Im EAGLES Framework werden grundlegende Anforderungen an solche Attribute definiert:

> „The central aim of the preparation stage of the evaluation is arriving at a set of attributes which satisfy the following conditions.
> - Their values should be obtainable by observation, by direct measurement or by deriving them from the values of other attributes.
> - The attributes should be adequate for expressing all explicit or implicit user requirements.
> - The attributes should be general enough to be applied to sets of systems with similar tasks, and to different classes of users or usages of those systems." (EAGLES 1995, S. 20-21)

Unter Berücksichtigung dieser Anforderungen ist das Ergebnis ein Katalog mit insgesamt 30 Systemattributen, die nach Qualitätskriterien und deren Unterkriterien klassifiziert werden können.

Softwareprodukt (operationalisiert durch zwei Qualitätskriterien mit insgesamt fünf Unterkriterien und 30 Attributen)				
Funktionalität (operationalisiert durch zwei Unterkriterien mit insgesamt 13 Attributen)		**Benutzbarkeit** (operationalisiert durch drei Unterkriterien mit insgesamt 17 Attributen)		
Angemessenheit	Interoperabilität	Erlernbarkeit	Bedienbarkeit	Konformität
9 Attribute	4 Attribute	5 Attribute	8 Attribute	4 Attribute

Tabelle 5: Hierarchisches Qualitätsmodell nach ISO 9126

Nachfolgend soll kurz erläutert werden, was sich hinter den ISO Kriterien im Detail verbirgt, und wie die Anforderungen an ein Annotationswerkzeug konkret auf das Qualitätsmodell abgebildet werden können.

5.4.2 Qualität durch Funktionalität

Die Funktionalität von Annotationswerkzeugen beschreibt in welchem Maße Funktionen zur Erfüllung einer bestimmten Aufgabe durch einen Benutzer vorhanden sind. In einer Anmerkung der ISO 9126 Norm wird die Funktionalität eines Softwareprodukts, in Abgrenzung zu den restlichen Qualitätskriterien[28], folgendermaßen definiert: „[Eine] Menge von Merkmalen [die] charakterisiert, was die Software zur Erfüllung von Erfordernissen tut, während die anderen Mengen hauptsächlich charakterisieren wann und wie sie das tut“ (DIN 66272 1994, S. 4). Funktionalität beschreibt also grob das Verhältnis von Werkzeug und Aufgabe (vgl. DIPPER et al. 2004: 3). Für die Annotatoren des DiSynDe Projekts bedeutet Funktionalität die grundlegende Fähigkeit des Werkzeugs den Annotationsprozess zu erleichtern, zu überwachen und zu beschleunigen.

Funktionalität (I): Angemessenheit Die Angemessenheit von Annotationswerkzeugen beschreibt deren funktionelle Eignung hinsichtlich einer bestimmten Aufgabe. Dies umfasst unter anderem die korrekte Darstellung und Verarbeitung der Primärdaten, also der Rohtexte, und der Sekundärdaten, also der Annotationen. In Hinblick auf die Verarbeitung der Primärdaten ist vor allem wichtig, dass Texte im Unicodeformat[29] und anderen Zeichenkodierungen[30] korrekt dargestellt werden können und die Werkzeuge in der Lage sind verschiedene Textdateiformate zu öffnen. Auch bei den Sekundärdaten muss sicher gestellt sein, dass der Unicodekonforme Originaltext vollständig erhalten bleibt und korrekt im Annotationsformat gespeichert wird. Außerdem sollte ein angemessenes Tool Änderungen im Ori-

[28] Qualitätskriterien werden im nachfolgenden Zitat nach der ISO-Terminologie als *Mengen* bezeichnet.

[29] In der Praxis ist die meist genutzte Anwendung zur Speicherung und Darstellung von Unicodezeichen das 8-bit Unicode Transformation Format (UTF-8). Dabei wird jedes Zeichen durch eine individuelle Bytekette kodiert.

[30] Weitere gängige Zeichenkodierungen stellen beispielsweise das ASCII-Format oder die ISO 8859 Kodierung dar

ginaltext während des gesamten Annotationsprozesses erlauben, und die Möglichkeit bieten über einer Tokenizer grundlegende Annotationseinheiten wie z. B. Wörter automatisch zu definieren. Um das Kriterium der funktionalen Angemessenheit zur vollen Zufriedenheit der DiSynDe Annotatoren zu erfüllen, muss ein Annotationswerkzeug darüber hinaus die teilweise oder vollständige Automatisierung des Annotationsprozesses durch die Verwendung von Lexika oder regulären Ausdrücken unterstützen. Des Weiteren muss die Möglichkeit bestehen, Relationen zwischen Annotationseinheiten zu definieren, um so etwa ebenenübergreifende Phänomene oder Koreferenzen zwischen Einheiten innerhalb einer Ebene annotieren zu können. Ein angemessenes Werkzeug erlaubt die nachträgliche Modifikation von Annotationsschemata während des gesamten Annotationsprozesses sowie eine Konsistenzprüfung durch Validierung der Annotation gegen das zugrundegelegte Schema.

Das Unterkriterium Angemessenheit umfasst insgesamt neun messbare Attribute, welche weitestgehend die Anforderungen an *Zeichensatz*, *Flexibilität der Annotationsschemata*, *Flexibilität der Ein- und Ausgabe* und die *Automatisierbarkeit der Software* wiederspiegeln.

(01) Unicode Unterstützung
(02) Alternative Zeichenkodierung
(03) Unterstützte Dateiformate
(04) Modifizierbarkeit der Rohtexte
(05) Tokenisierung der Rohtexte
(06) Automatisierbarkeit der Annotation
(07) Skopus der Annotationseinheiten
(08) Änderung der Schemata
(09) Validierung gegen Schema

Tabelle 6: Messbare Systemattribute für die Angemessenheit von Annotationswerkzeugen

Funktionalität (II): Interoperabilität Mit der Interoperabilität von Annotationswerkzeugen soll deren Kompatibilität zu anderen bestehenden Annotations- und Korpusprogrammen beschrieben werden. Umfangreiche Import- und Exportfunktionen befähigen ein Werkzeug auch Annotationen aus anderen Projekten verarbeiten zu können. Die Interoperabilität eines Werkzeugs erhöht sich maßgeblich, wenn neben dem systemeigenen Stand-off Format zusätzlich Annotationen anderer Stand-

off und Inline Formate verarbeitet werden können, sei es über eine definierte Import-/Exportschnittstelle, oder über externe Konvertierungsprogramme. Gleichzeitig garantiert ein wohlgeformtes und standardisiertes XML-Format der eigenen Annotationen ein hohes Maß an Wiederverwendbarkeit und die reibungslose Eingliederung anderer Projekte, wie etwa dem DDD. Ein standardisiertes Format greift im günstigsten Fall auf Stand-off Konzepte zurück, die bereits von anderen Unternehmungen erfolgreich implementiert wurden. Wohlgeformtheit bezieht sich auf die rigorose Einhaltung aller vom W3C aufgestellten XML-Regularien, wie z. B. die Forderung nach genau einem Wurzelelement pro Dokument oder die paarige Verschachtelung von Start- und Endtags bei inhaltstragenden Elementen.

Diese Unterkategorie der Funktionalität wird durch vier Attribute beschrieben und trägt den Anforderungen *Mehrebenenannotation* und *Flexibilität der Ein- und Ausgabe* Rechnung.

(10) Im-/Export von Stand-off Formaten
(11) Im-/Export Inline-Formaten
(12) Wohlgeformtheit der Annotation
(13) Kompatibilität des Annotationsformats

Tabelle 7: Messbare Systemattribute für die Interoperabilität von Annotationswerkzeugen

5.4.3 Qualität durch Benutzbarkeit

Anders als beim Kriterium der Funktionalität, steht bei der Benutzbarkeit der Anwender des Annotationswerkzeugs mit seinen individuellen Bedürfnissen an Interaktionsverhalten und Visualisierung der Software im Vordergrund. Ein hoher Grad an Benutzbarkeit impliziert immer auch einen möglichst geringen Aufwand zum Erlernen und zur Bedienung der Software. Beschreibt das Kriterium Funktionalität *wie* sich ein Werkzeug bei der Bewältigung einer bestimmten Aufgabenstellung verhält, so bildet das Qualitätskriterium Benutzbarkeit das Verhältnis zwischen Werkzeug und Benutzer ab (vgl. DIPPER et al. 2004, S. 3).

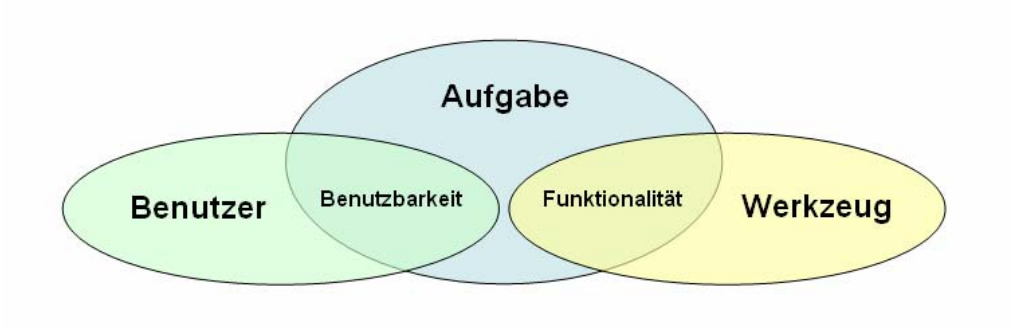

Abbildung 12: Benutzbarkeit und Funktionalität im Verhältnis zur Aufgabe

Da der Benutzer im Mittelpunkt der gesamten Software-Evaluation stehen soll, wird diesem Kriterium besondere Aufmerksamkeit geschenkt, beinhaltet es doch laut ISO-Norm „die individuelle Bewertung einer […] Benutzung durch eine festgesetzte oder vorausgesetzte Gruppe von Benutzern" (DIN 66272 1994: 4). So beschreiben insgesamt 17 Attribute die Benutzbarkeit der Annotationswerkzeuge, und 13 Attribute deren Funktionalität.

Benutzbarkeit (I): Erlernbarkeit Die Erlernbarkeit beschreibt den Aufwand des Benutzers, sich Kompetenzen über die Funktionsweise und Handhabung einer Software zu verschaffen. Typischerweise umfasst dies Kenntnisse um Ablaufsteuerung, Eingabe, Ausgabe etc. (vgl. DIN 66272 1994, S. 8). Eine gute Programmdokumentation sowie zusätzliche Lernhilfen in Form von Tutorials oder Übungsbeispielen machen es dem Benutzer leichter sich in ein Programm einzuarbeiten und erhöhen die Lernförderlichkeit des Werkzeugs. Integrierte Hilfesysteme und *Tooltips*[31] steigern die Selbstbeschreibungsfähigkeit[32] einer Anwendung und beseitigen Unklarheiten, die während der Benutzung des Programms auftreten können, schnell und effektiv. Wenn zur Lösung von weiterreichenden Problemen zusätzliche Strategien, wie etwa eine Mailinglist oder FAQs[33] angeboten werden, erhöht dies die Erlernbarkeit eines Annotationswerkzeugs zusätzlich.

[31] Tooltips liefern Kurzinformationen zu unbekannten Funktionen oder Schaltflächen sobald man den Mauszeiger über das entsprechende Element bewegt.
[32] Die Konzepte Lernförderlichkeit und Selbstbeschreibungsfähigkeit entstammen der ISO Norm 9241-110 über „Grundsätze der Dialoggestaltung".
[33] FAQs beinhalten häufig gestellte Fragen (*Frequently Asked Questions*) und standardisierte Antworten.

Die Forderung nach *Schlichtheit der Anwendung* wird durch dieses Unterkriterium der Benutzbarkeit, welches durch fünf Attribute repräsentiert wird, maßgeblich modelliert.

(14) Dokumentation
(15) Hilfesystem
(16) Kurzinfo
(17) Praktische Lernhilfen
(18) Problemlösungsstrategien

Tabelle 8: Messbare Systemattribute für die Erlernbarkeit von Annotationswerkzeugen

Benutzbarkeit (II): Bedienbarkeit Mit dem Unterkriterium Bedienbarkeit wird beschrieben wie aufwendig die tatsächliche Handhabung des Programms für den Benutzer ist. Gute Bedienbarkeit beginnt schon bei der Installation und Konfiguration des Annotationswerkzeugs, welche möglichst ohne Komplikationen und komplexe technische Kenntnisse erfolgen sollte. Besonderes Augenmerk wird auf die Bedienbarkeit in Hinblick auf die Modifizierbarkeit der Annotationsschemata gelegt. Ein graphischer Schemaeditor, der eine Änderung des Tagsets während des laufenden Annotationsprozesses erlaubt, unterstützt die Anwender in idealer Weise. Wenn das Schema zusätzliche Mechanismen enthält, um auf unvorhergesehene Phänomene flexibel reagieren zu können, ist dieser Bereich der Benutzbarkeit optimal abgedeckt. Darüber hinaus verbessern ebenenweite Aktionen, wie etwa das Ein- oder Ausblenden bestimmter Annotationsebenen und Suchfunktionen für Primär- und Sekundärdaten, die Bedienung der Software während des Annotationsprozesses. Die individuelle Anpassung von Visualisierung und Funktionalität auf Benutzerebene, also ohne direkt den Quellcode des Programms modifizieren zu müssen, tragen zur Adaptierbarkeit und damit zu einer erhöhten Bedienbarkeit des Werkzeugs bei. Die Organisation des Annotationsprojekts DiSynDe in mehrere räumlich und zeitlich verteilte Arbeitsgruppen stellt eine besondere Anforderung an die Bedienbarkeit eines Annotationswerkzeugs, hinsichtlich individueller Zugriffsrechte für einzelne Ebenen, dar.

In diesem komplexen Untersuchungsbereich decken insgesamt acht Attribute die Themen *Mehrebenenannotation*, *Flexibilität der Annotationsschemata*, *Schlichtheit der Anwendung* und *Verteilung der Arbeitsabläufe* ab.

(19)	Installationsaufwand
(20)	Ebenenbezogene Aktionen
(21)	Editierung der Schemata
(22)	Flexibilität der Schemata
(23)	Zugriffsrechte für Annotationsebenen
(24)	Suchfunktion für Textdaten
(25)	Anpassung der Darstellung
(26)	Anpassung der Funktionalität

Tabelle 9: Messbare Systemattribute für die Bedienbarkeit von Annotationswerkzeugen

Benutzbarkeit (III): Konformität Das Unterkriterium Konformität existiert für fünf von sechs ISO Qualitätskriterien. Im Zusammenhang mit der Benutzbarkeit von Software beschreibt es, inwieweit das Werkzeug Normen oder Vereinbarungen in diesem Feld umsetzt. Dazu zählen unter anderem standardisierte Visualisierungs- und Interaktionskonzepte aus dem Bereich der Softwareergonomie. Vor allem die Aufteilung und Gestaltung der Arbeitsfläche sowie konventionalisierte Metaphern und Funktionen zur Bearbeitung von Textdaten tragen zur Konformität bei.

Die Konformität der Benutzbarkeit beschreibt durch vier messbare Attribute vor allem die Anforderungen *Schlichtheit der Anwendung* sowie teilweise die Forderung nach einer *Mehrebenenannotation*.

(27)	Aufteilung der Arbeitsfläche
(28)	Gestaltung der Arbeitsfläche
(29)	Standardisierte Aktionen
(30)	Selektion von Annotationseinheiten

Tabelle 10: Messbare Systemattribute für die Konformität von Annotationswerkzeugen

Detaillierte Metriken zur Messung der insgesamt 30 Systemattribute werden im nächsten Abschnitt vorgestellt.

5.5 Metriken

Aufbauend auf einem Qualitätsmodell mit messbaren Attributen auf unterster Ebene sollen nun entsprechende Metriken für diese erstellt werden:

> Metrics are an essential concomitant of the quality model lying at the base of any evaluation. In ISO terms quality characteristics are broken down into subcharacteristics, which in turn are broken down into measurable attributes. Looked at from the bottom up, starting from the metrics and leading upwards towards the top level quality characteristics, it is possible to think of the product as being specified by the metrics that will be applied to it. (EAGLES 1999a, S. 46)

5.5.1 Aufbau der Metriken

Eine Metrik besteht aus einem Maß und einer Messmethode (vgl. UNDERWOOD & LISOWSKA 2006, S. 2481). Als Maß soll im Folgenden ein Attribut, mitsamt den möglichen Werten, die es annehmen kann bezeichnet werden (vgl. EAGLES 1995, S. 29). Die zu erwartenden Werte bilden eine nominale Skala, wobei das Messen auf nominalem Niveau die einfachste Form der Datenerhebung darstellt, können doch lediglich Aussagen über Gleichheit oder Ungleichheit der Werte getroffen werden. Eine spätere Übertragung der nominalen Werte auf eine Verhältnisskala ist deshalb unerlässlich. Die Messmethoden dienen dazu, konkrete Werte auf einer definierten Skala für das Attribut ermitteln zu können. Dabei gibt es für jedes der insgesamt 30 Attribute eine eigene Metrik, mit individuellen Werteskalen und Messmethoden.

Werteskalen Während ISO 9126 vorsieht, dass Metriken idealerweise zu quantifizierbaren Werten führen, zeigt die Realität, dass dies nicht immer möglich ist. Das EAGLES Framework ist bei diesem Thema weitaus flexibler und vertritt folgenden Grundsatz:

> Attributes are typed according to their possible values. The range of values (scale) can for example be boolean (yes/no), nominal or classificatory (a set of unordered values), comparative (a set of ordered values), ordinal (a range of values whose differences can be compared), or metric (real valued with fixed origin and unit). Some attributes can have other feature structures as values. (EAGLES 1999a, S. 19)

Attribute müssen nach dieser Auffassung keinesfalls immer nur zählbare, numerische Werte annehmen, sondern können genauso gut qualitativ auf einer Nominalskala beschrieben werden. Die Evaluation von Funktionalität und Benutzbarkeit kann nur durch die subjektive Beurteilung des Durchführers der Evaluation bewerkstelligt werden, da keines der untergeordneten Attribute direkt quantifizierbar ist. Da die Beurteilung nichtsdestotrotz innerhalb definierter Nominalskalen erfolgt,

und nur eine bestimmte Menge von Werten zulässig ist, nimmt die Messung der Attribute eine „Zwischenstellung zwischen einem ‚subjektiven' und einem ‚objektiven' Erhebungsverfahren" (MUMMENDEY & GRAU 2008, S. 16) ein, wie sie auch bei der Fragebogenmethodik beobachtet werden kann.

Der EAGLES Terminologie entsprechend sind die zu erwartenden Werte für diese Evaluation entweder binärer oder klassifikatorischer Natur (vgl. EAGLES 1995, S. 31). Das Systemattribut *Wohlgeformtheit der Annotation* kann als Beispiel für die Klasse der Binärwerte herangezogen werden, denn das Annotationsformat welches ein Werkzeug liefert, ist nach Definition des W3C[34] entweder wohlgeformt, oder nicht. Soll hingegen ein Wert für den *Umfang der Dokumentation* eines Werkzeugs gemessen werden, so stehen verschiedene klassifikatorische Werten auf einer Nominalskala zur Auswahl. Eine Sonderform klassifikatorischer Werte, welche in dieser Evaluation gehäuft auftritt, stellen kombinierte klassifikatorische Werte dar, bei denen für ein und dasselbe Attribut mehrere einander ergänzende Werte kombiniert ausgewählt werden können. Dabei kann jedoch nicht jeder Wert beliebig mit einem anderen kombiniert werden, vielmehr schließen viele Werte einander aus.

Auswahlregeln Um die Kombination von solchen Teilwerten zu koordinieren, gibt es für jedes Attribut eine logische Auswahlregel. Werte die durch ein nichtausschließendes logisches Oder (OR) verknüpft sind, können entweder einzeln oder auch kombiniert ausgewählt werden. Sprechweise: *a* oder *b* (oder beide).

a OR b

Werte, die durch ein ausschließendes logisches Oder (XOR) verknüpft sind, exkludieren einander, d. h. es ist genau einer der beiden Werte möglich. Die Messung des Wertes *a* schließt den Wert *b* aus, und umgekehrt. Sprechweise: entweder *a* oder *b*.

a XOR b

[34] W3C Recommendation für XML [http://www.w3.org/TR/2006/REC-xml-20060816] – Zugriff am 10.06.2008.

Darüber hinaus können Klammerungen ganze Wertgruppen über Adjunktion (OR) oder Disjunktion[35] (XOR) mit anderen Werten verknüpfen. Im nachfolgenden Beispiel dürfen etwa die Werte *a*, *b* und *c* beliebig miteinander kombiniert werden, aber bereits die Auswahl eines einzigen der drei Werte schließt den Wert *d* aus. Ebenso schließt die Auswahl von *d* automatisch die Werte *a*, *b* und *c* aus.

(a OR b OR c) XOR d

Messmethoden Um konkrete Werte für ein Attribut ermitteln zu können, werden Messmethoden benötigt. Da die im Qualitätsmodell definierten Attribute aber nicht durch das Messen einer benötigten Zeitspanne, oder das Zählen von bestimmten Ereignissen direkt quantifizierbar sind, umfasst die nominale Werteskala faktische Antworten auf eine eindeutige Frage zu werkzeugspezifischen Ausprägung eines Attributs. Die Messmethoden der im Folgenden definierten Metriken bestehen also aus eindeutig beantwortbaren Fragen zu Umfang und Umsetzung der Implementierung einzelner Systemattribute, die letztendlichen Messwerte sind mögliche Antworten auf diese Fragen. Der Durchführer der Evaluation bewertet die vordefinierten Antworten auf die Fragen binär als *zutreffend* (+) oder *nicht zutreffend* (-). Für den Fall, dass es mehrere zutreffende Antworten auf eine Messfrage gibt, zeigen die oben definierten logischen Auswahlregeln gültige Kombinationsmöglichkeiten an.

Ein kurzes Anwendungsbeispiel soll das Zusammenspiel des Qualitätsmodells und der zugrundegelegten Metriken veranschaulichen. Um etwa Aussagen über die Funktionalität eines Annotationswerkzeugs machen zu können, wurde das Kriterium im Qualitätsmodell unter anderem in das Unterkriterium Angemessenheit zergliedert, welches wiederum in mehrere messbare Attribute aufgeteilt wurde. Ein Attribut zur Evaluation der Angemessenheit eines Werkzeugs stellt z. B. die *Unicode Unterstützung* des Programms dar. Die Metrik für dieses Attribut besteht aus einer Messmethode in Form einer konkreten Frage zur Darstellung und Speicherung von Rohtextdaten im Annotationswerkzeug. Die möglichen Antworten stellen eine nominale Werteskala dar und geben Auskunft darüber ob, die UTF-8-kodierten Pri-

[35] Einige Autoren verwenden den Begriff *Disjunktion* auch für das nicht ausschließende Oder. Hier soll allerdings das nicht-ausschließende Oder als *Adjunktion*, und das ausschließende Oder als *Disjunktion* bezeichnet werden. Im Zusammenhang mit XOR ist darüber hinaus häufig auch von *Kontra-* oder *Antivalenz* die Rede. (vgl. KELLY 2003, S. 23; Wikipedia:Aussagenlogik, 2008)

märdaten korrekt dargestellt und gespeichert werden können. Soweit die grundlegende Metrik mit Werteskala und Messmethode für ein exemplarisches Attribut. Da der eigentliche Zweck der messbaren Einzelattribute die finale Aufrechnung mit anderen Attributen zu einem aussagekräftigen Gesamturteil ist, bedarf es eines Formalismus um die unterschiedlichen Nominalwerte zusammenfassen zu können.

Skalentransformation Hierzu sollen die Werte der nominalen Messskalen in prozentuale Verhältnisskalen übertragen werden. Im Gegensatz zu Nominalskalen stellen quantifizierbare Verhältnisskalen das höchstmögliche Messniveau dar (vgl. SCHARF & SCHUBERT 2001, S. 366). So ist es möglich vielfältige Rechenoperationen, wie etwa die Addition auf die Messwerte anzuwenden sowie Aussagen über deren Verhältnismäßigkeit zu treffen. Eine beispielhafte Verhältnisskala mit den Werten A=15, B=30 und C=50 erlaubt etwa die Rückschlüsse, dass B doppelt so groß ist wie A und beide Werte zusammen kleiner sind als C. Auf einer Nominalskala kann nur ausgesagt werden, dass sich alle drei Werte voneinander unterscheiden. Die Frage, ob ein Wert das Attribut besser erfüllt als ein anderer, kann nicht auf quantitativem Wege beantwortet werden, sondern erfordert die subjektive Interpretation des Nominalwertes.

Zum besseren Verständnis für die Verwendung einer prozentualen Verhältnisskala soll an dieser Stelle kurz das eigentliche Erkenntnisinteresse der Evaluation verdeutlicht werden: Über die einzelnen Attribute eines Annotationswerkzeugs sollen Aussagen über individuelle Stärken und Schwächen sowie Gesamtaussagen zur Qualität des Tools gemacht werden. Die bloße Feststellung, dass die verschiedenen Werkzeuge ein Attribut auf gleiche oder unterschiedliche Weise erfüllen, liefert jedoch die gewünschten Erkenntnisse noch nicht. Deshalb muss vor der Durchführung der Evaluation festgelegt werden, welcher Messwert das Attribut zu welchem Grad erfüllt. Ein Attribut kann dabei entweder gar nicht oder maximal gut erfüllt sein. Um einen solchen Erfüllungsgrad intuitiv auszudrücken, ist eine Prozentskala besonders gut geeignet.

Nach der Übertragung der Werte auf die Verhältnisskala kann für jedes Attribut ein prozentualer Gesamtwert ermittelt werden, der angibt zu welchem Grad das Attribut erfüllt wurde: „For the final product there are quality goal values that must be reached or surpassed for each attribute. When the values are reached or surpassed

then the architecture is said to satisfy the quality characteristics required“ (LOSAVIO et al. 2003, S. 138). Ein solcher Gesamtwert ist vor allem für Attribute mit einer kombiniert klassifikatorischen Werteskala unerlässlich, werden doch mehrere Teilwerte zu einem Gesamtwert zwischen 0 und 100% aufaddiert, der später leicht interpretiert und weiterverarbeitet werden kann. Binäre Werte zeigen immer entweder eine vollständige Erfüllung eines Attributs, oder eine gar nicht vorhandene Erfüllung an, und sind deshalb immer entweder genau 100% oder genau 0%. Bei klassifikatorischen Werteskalen können verschiedene Teilwerte mit unterschiedlichen prozentualen Gewichtungen belegt werden. So trägt etwa das Vorhandensein eines Tutorials (40%) mehr zur Erfüllung des Attributs *Praktische Lernhilfen* bei, als eventuell verfügbare Übungsdateien (20%), Lehrvideos (20%) oder Forschungsliteratur (20%). Eine hundertprozentige Erfüllung ist dann gegeben, wenn alle vier Lernhilfen angeboten werden. Wenn bei einem Werkzeug keiner der vorgegebenen Nominalwerte gemessen werden kann, so wird es mit 0% auf der Verhältnisskala bewertet.

Einstufungsniveaus Der Prozentwert, welcher für ein bestimmtes Attribut ermittelt wird, gibt bereits Aufschluss darüber, in welchem Maße es als erfüllt angesehen werden kann, jedoch nicht darüber, welcher prozentuale Grad der Erfüllung für die späteren Nutzer der Annotationswerkzeuge als akzeptabel bzw. als nicht-akzeptabel gilt. Die Werte müssen deshalb erst noch interpretiert werden, um Aussagen über die Eignung des Systems machen zu können. Ein Schwellwert gibt für jedes Attribut an, welche Prozentzahl akzeptabel und welche nicht-akzeptabel ist. In der ISO 9126 Terminologie wird an dieser Stelle der Begriff der *Einstufungsniveaus* eingeführt (vgl. DIN 66272 1994, S. 3). Unterhalb einer bestimmten Prozentgrenze ist ein Messwert nicht mehr akzeptabel. Jeder Wert, der größer oder gleich diesem Grenzwert ist, ist im Rahmen des DiSynDe Szenarios akzeptabel. Durch Einstufungsniveaus werden die Anforderungen der eingangs modellierten Nutzergruppe des Werkzeugs berücksichtigt und operationalisiert. Warum manche Werte mehr zur Akzeptabilität des Werkzeugs durch die späteren Nutzer beitragen als andere, wird im Anschluss für jedes Attribut spezifisch erläutert. Einstufungsniveaus variieren je nach den Bedürfnissen der verschiedenen Anwender und werden darüber hinaus für jedes Attribut individuell erstellt, da bei manchen Attributen bereits ein niedrigerer Wert als ak-

zeptabel angesehen werden kann als bei anderen. So wird einerseits deutlich, welche Attribute von welchen Werkzeugen zufriedenstellend erfüllt werden, bzw. wo individuelle Schwachstellen bei den Annotationstools liegen können, andererseits können so genannte Leistungsniveaus[36] berechnet werden (vgl. DIN 66272 1994, S. 3). Leistungsniveaus fassen die Einstufungsniveaus mehrerer Attribute zu gewichteten Durchschnittswerten zusammen und ermöglichen so qualitative Aussagen zu den fünf Unterkriterien Angemessenheit, Interoperabilität, Erlernbarkeit, Bedienbarkeit und Konformität, den übergeordneten Kriterien Funktionalität und Benutzbarkeit sowie dem Annotationswerkzeug als Ganzes.

Gewichtung Dabei wird eine Gewichtung von Muss- und Soll-Attributen mit einberechnet. Ein Muss-Attribut verkörpert eine obligatorische Anforderung und wird deshalb gegenüber einem Soll-Attribut, welches lediglich optionale Anforderungen wiedergibt, doppelt gewichtet. Im nachfolgenden Abschnitt sollen detaillierte Metriken für jedes einzelne Systemattribut erstellt werden. Die jeweiligen Auswahlregeln, Gewichtungen und Einstufungsniveaus werden durch die Bedürfnisse eines virtuellen DiSynDe Anwenders legitimiert.

5.5.2 Attributkatalog und Werteskalen

Die folgenden Metriken sind als Katalog von insgesamt 30 messbaren Attributen realisiert, welche aus den ISO Qualitätskriterien und deren Unterkriterien abgeleitet wurden. Die Messmethode ist jeweils durch eine konkrete Fragestellung definiert, das Maß ergibt sich aus einer nominalen Werteskala mit potentiellen Antworten auf die Messfrage. Die Antworten werden auf eine prozentuale Verhältnisskala übertragen, um Aussagen über den Grad der Erfüllung des Attributs machen zu können und um die Festlegung konkreter Einstufungsniveaus zu ermöglichen. Die Zuweisung und Höhe der Prozentwerte wird für jedes Attribut individuell anhand der Benutzeranforderungen für das DiSynDe Projekt erläutert. Grundsätzlich gilt, dass ein Einzelwert mit einem höheren Prozentwert korreliert, wenn er gegenüber anderen Werten auf der Nominalskala mehr zur Erfüllung des Attributs beiträgt. Von

[36] Der Grad, zu dem die Erfordernisse erfüllt werden, dargestellt durch eine Menge von Werten für die Qualitätsmerkmale

den prozentualen Korrelaten hängen auch die Einstufungsniveaus für akzeptable und nicht akzeptable Werte ab. Da sich ein Maß teilweise aus mehreren kombinierten Messwerten der Skala zusammensetzt, wird zusätzlich zu jedem Attribut eine logische Auswahlregel für die möglichen Kombinationen von Einzelwerten mit angegeben, welche nie den Gesamtwert von 100% überschreiten dürfen.

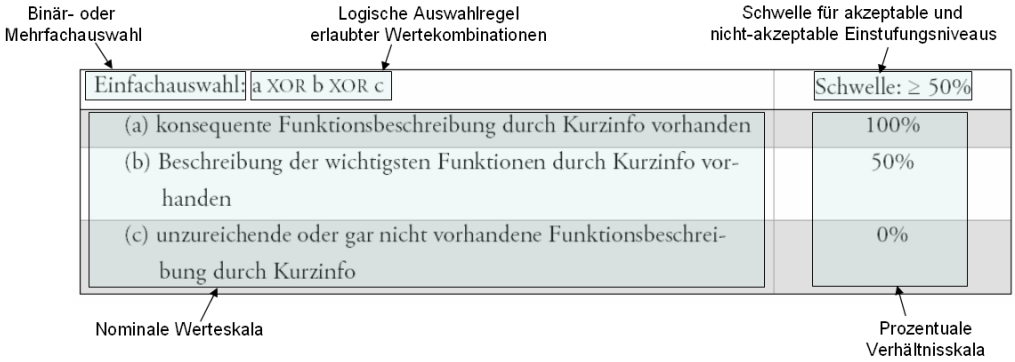

Abbildung 13: Erläuterungen zur tabellarischen Darstellung der Metriken

(01) Unicode Unterstützung: Werden Unicode-Texte im Annotationswerkzeug korrekt angezeigt und auch im Annotationsformat vollständig gespeichert?

Hier wird überprüft, ob das Tool in der Lage ist Unicode-kodierte Rohtexte im UTF-8 Format korrekt darzustellen und diese auch in der Annotationsdatei mit allen vorhandenen Sonderzeichen zu speichern. Das Attribut besitzt eine kombiniert klassifikatorische Werteskala, wobei sowohl die Anzeige als auch die Speicherung der Primärdaten je 50% zum maximalen Erfüllungsgrad des Attributs beitragen, da beide Kriterien für das DiSynDe Szenario gleich bedeutend sind. Wird ein Text nicht vollständig angezeigt oder korrekt gespeichert, so wird dies mit 0% bewertet, da eine teilweise Darstellung der historischen Sonderzeichen den Projektanforderungen nicht gerecht wird. Das Attribut gilt nur dann als akzeptabel, wenn sowohl bei der Darstellung als auch im Annotationsformat alle Sonderzeichen korrekt interpretiert werden, weshalb die Schwelle für akzeptable Werte bei 100% liegt. Da die Unterstützung von Unicode durch das UTF-8 Format immanent wichtig für die Annotation historischer Texte ist, wird dieses Attribut als obligatorisch eingestuft und bei der Gesamtbeurteilung gegenüber optionalen Attributen doppelt gewichtet.

Mehrfachauswahl: a OR b	Schwelle: ≥ 100%
(a) Anzeige der Rohtexte als UTF-8 möglich	50%
(b) Speicherung der Annotation als UTF-8 möglich	50%

Tabelle 11: Unicode Unterstützung - Metrik

(02) Alternative Zeichenkodierung: Werden Texte in alternativen Zeichenkodierungen vom Annotationswerkzeug korrekt angezeigt?

Dieses Attribut kann als Ergänzung zur Forderung nach Unicode Unterstützung gesehen werden, beschreibt es doch die Fähigkeit eines Tools Rohtexte neben dem UTF-8 Format auch in anderen populären Zeichenkodierungen, wie etwa ASCII oder ISO 8859, darzustellen. Prozentuale Gewichtung und Auswahl der einzelnen Werte sind analog zum vorhergehenden Kriterium zu verstehen, da es aber nicht zwingend erforderlich ist neben Unicode auch noch andere Zeichenkodierungen zu unterstützen, wird dieses Attribut in der Gesamtwertung nur als optionales Kriterium gewertet.

Mehrfachauswahl: a OR b	Schwelle: ≥ 100%
(a) Anzeige der Rohtexte als ASCII, ISO-8859 etc. möglich	50%
(b) Speicherung der Annotation als ASCII, ISO-8859 etc. möglich	50%

Tabelle 12: Alternative Zeichenkodierung - Metrik

(03) Unterstützte Dateiformate: Welche Dateiformate kann das Tool öffnen und gegebenenfalls richtig interpretieren?

An dieser Stelle wird bewertet, welche unterschiedlichen Dateiformate das Tool öffnen und im Falle von Markupformaten, wie etwa XML, interpretieren kann. Da das DiSynDe Pilotkorpus gänzlich aus MS-Word Dokumenten besteht, wird die Unterstützung von Markupformaten nur mit 20% bewertet, wobei hier zusätzlich unterschieden wird, ob das Markup interpretiert werden kann[37]. Die Pilottexte können mit geringem Aufwand vom MS-Word Format in *.TXT Dateien umgewandelt werden, weshalb beide Formate mit 40% auf der Verhältnisskala bewertet

[37] Man spricht in der englischsprachigen Literatur an dieser Stelle häufig von so genannter *markup awareness* (vgl. CUNNINGHAM et al. 2007, S. 46)

werden. Das obligatorische Attribut *Unterstützte Dateiformate* repräsentiert eine Muss-Anforderung, gilt jedoch bereits als akzeptabel wenn entweder MS-Word Dokumente oder einfache Textdateien verarbeitet werden können.

Mehrfachauswahl: a OR b OR (c XOR d)	Schwelle: ≥ 40%
(a) Textdateien: *.TXT	40%
(b) MS-Word Dokumente: *.DOC	40%
(c) Markupsprachen: *.XML , *.HTML, *.SGML (ohne Interpretation der Tags)	20%
(d) interpretierte Markupsprachen: *.XML , *.HTML, *.SGML (mit Interpretation der Tags)	20%

Tabelle 13: Unterstützte Dateiformate - Metrik

(04) Modifizierbarkeit der Rohtexte: Ist eine Modifizierung der Rohtexte innerhalb des Annotationswerkzeugs möglich, und wenn ja, zu welchem Zeitpunkt des Annotationsprozesses?

Mit Hilfe dieser Fragestellungen kann untersucht werden, ob es möglich ist den Rohtext zu modifizieren nachdem er in das Annotationswerkzeug geladen wurde. Dies ist nötig falls etwa orthographische Fehler in den Primärdaten erst während des Annotationsprozesses entdeckt werden und nachträglicher Verbesserung bedürfen. Die Nominalskala erlaubt auch in diesem Fall eine Mehrfachauswahl von Werten. Das Attribut gilt zu 50% als erfüllt, wenn eine Modifizierung der Daten vor Beginn der Annotation im Werkzeug möglich ist. Wenn die Texte darüber hinaus auch noch während des laufenden Annotationsvorgangs geändert werden können, ist das Attribut zu 100% erfüllt. Die Modifizierbarkeit der Texte ist optional, da das eingangs modellierte Annotationsszenario bereits von formal und orthographisch korrekten Texten ausgeht. Außerdem können Texte notfalls auch außerhalb des Werkzeugs geändert, und danach neu in die Programmoberfläche geladen werden.

Mehrfachauswahl: a OR b	Schwelle: ≥ 50%
(a) Modifizierung während der Annotation möglich	50%
(b) Modifizierung vor der Annotation möglich	50%

Tabelle 14: Modifizierbarkeit der Rohtexte - Metrik

(05) Tokenisierung der Rohtexte: Unterstützt das Annotationswerkzeug die interne oder externe Tokenisierung der Rohtexte?

Die Tokenisierung wird als Muss-Attribut gewertet, stellt sie doch einen wichtigen Vorverarbeitungsschritt bei jeder Annotation dar, bei dem die kleinsten anzunehmenden Annotationseinheiten als Token erfasst werden. Ein Token ist dabei „üblicherweise definiert als eine von Leerzeichen oder Interpunktionen begrenzte Folge von Buchstaben oder Ziffern", wobei es häufig auch wünschenswert ist „eine feinere Unterteilung zu schaffen oder umgekehrt größere Leerzeichen übergreifende Einheiten als ein Token aufzufassen" (EVERT & FITSCHEN 2001, S. 371). Die Skala kombiniert Werte, welche die Möglichkeiten im Bereich interner und externer Tokenisierung erfassen. Die Tokenisierung wird als *intern* bezeichnet, wenn sie in die Werkzeugoberfläche integriert ist. Wenn zudem die Abgrenzungen eines einzelnen Token individuell definiert werden kann, wird zusätzlich die Forderung nach Parametrisierung der Tokenisierung erfüllt. Je nachdem ob der interne Tokenizer konfiguriert werden kann oder nicht, wird dieses Attribut mit einem 60- bzw. 40- prozentigen Erfüllungsgrad bewertet. Die Einbindung externer Tokenizer wird zusätzlich mit 40% bemessen, so dass im Idealfall eine parametrisierbare Tokenisierung innerhalb des Tools sowie eine optionale Einbindung externer Ressourcen möglich sind. Das Kriterium gilt als erfüllt, wenn mindestens eine beliebige Möglichkeit der Tokenisierung besteht.

Mehrfachauswahl: (a XOR b) OR c	Schwelle: ≥ 40%
(a) interne Tokenisierung mit Möglichkeit der Parametrisierung	60%
(b) interne Tokenisierung ohne Möglichkeit der Parametrisierung	40%
(c) externe Tokenisierung möglich	40%

Tabelle 15: Tokenisierung der Rohtexte - Metrik

(06) Automatisierbarkeit der Annotation: Unterstützt das Annotationswerkzeug die teilweise oder vollständige Automatisierbarkeit des Annotationsprozesses?

Durch regelbasierte oder probabilistische Ansätze kann für gegenwartssprachliche Texte häufig eine voll- oder zumindest semi-automatische Annotation erreicht werden. Für Korpora historischer Texte hält SYRING (2004, S. 317) jedoch fest:

> Bei Korpora die z. B. Texte des 18. Jahrhunderts oder mittelalterliche Quellen enthalten, versagt die automatische Annotation zumindest dann, wenn eine urkundengetreue Wiedergabe zugrundegelegt […] wird. In der Regel wird man daher bei der Annotation älterer Texte auf manuelle oder halbautomatische Verfahren zurückgreifen.

Doch nicht nur die urkundengetreue Wiedergabe historischer Texte, sondern auch die enorme Vielfalt an parallel existierenden Formen und Sonderfällen sowie die im Vergleich zu modernen Korpora relativ geringen Textmengen, erschweren vollautomatische Annotationsansätze in diesem Bereich. Deshalb soll realistischerweise auch nur untersucht werden, inwieweit die Werkzeuge zumindest eine teilweise automatisierbare Annotation unterstützen. Implementiert ein Werkzeug beispielsweise reguläre Ausdrücke[38] zur regelbasierten, semi-automatischen Annotation bestimmter Annotationseinheiten, so wird der Grad der Automatisierbarkeit für das DiSynDe Szenario mit 60% beschrieben. Ein Lexikon-basierter Ansatz, der automatisch Vorschläge für ähnliche Annotationseinheiten unterbreitet, steuert zusätzlich 40% zur maximalen Erfüllung des Kriteriums bei und markiert gleichzeitig die Mindestgrenze der Akzeptabilität. Auch wenn die Automatisierbarkeit während der Pilotphase nur eine untergeordnete Rolle spielt, so wird sie für ein effizientes Arbeiten in einem späteren Stadium des Projekts doch unabdingbar sein. Aus diesem Grund wird dieses Attribut als obligatorisch eingestuft und später doppelt gewichtet.

Mehrfachauswahl: a OR b	Schwelle: ≥ 40%
(a) Automatisierbarkeit durch Verwendung von regulären Ausdrücken möglich	60%
(b) Automatisierbarkeit durch Verwendung eines Lexikons möglich	40%

Tabelle 16: Automatisierbarkeit der Annotation - Metrik

(07) Skopus der Annotationseinheiten: Welchen Skopus können die Annotationseinheiten einnehmen, und welche Möglichkeiten der Vernetzung von Annotationseinheiten unterstützt das Annotationswerkzeug?

Dieses Attribut ist in gewisser Hinsicht als Ergänzung der Tokenisierung zu verstehen. Hier soll erfasst werden, ob Token wirklich beliebig definiert werden kön-

[38] Reguläre Ausdrücke stellen eine generalisierte Sprache zur Verarbeitung von Textdaten dar. Dabei werden sie „von vielen Programmen dazu genutzt, eine Menge von Strings zu beschreiben, ohne jeden einzelnen String dieser Menge aufführen zu müssen“ (KLIER 2002, S. 193).

nen, und ob darüber hinaus die Möglichkeit besteht sie durch gerichtete Relationen zu verknüpfen. Die Definition beliebiger Annotationseinheiten, mit einem Start- und einem Endpunkt, deckt bereits 40% des maximalen Erfüllungsgrades dieses Attributs ab. Ebenso viele Punkte werden für die Unterstützung von Koreferenzen innerhalb einer Ebene vergeben. Um das Attribut zu 100% zu erfüllen, muss das Annotationswerkzeug zusätzlich die Vernetzung verschiedener Token zwischen unterschiedlichen Ebenen ermöglichen. Die Schwelle zur Akzeptanz ergibt sich aber bereits aus den beiden erstgenannten Werten. Da die Definition beliebiger Annotationseinheiten und deren Verknüpfung eine grundlegende Anforderung an Werkzeuge für diachrone Korpora stellt, wird ein angemessener *Skopus der Annotationseinheiten* als Muss-Attribut eingestuft.

Mehrfachauswahl: a OR b OR c	Schwelle: ≥ 80%
(a) Beliebige Annotationseinheiten mit einem Start- und einem Endpunkt	40%
(b) gerichtete Relationen zwischen Annotationseinheiten auf einer Ebene möglich	40%
(c) gerichtete Relationen zwischen Annotationseinheiten und Vernetzung über verschiedener Ebenen hinweg möglich	20%

Tabelle 17: Skopus der Annotationseinheiten - Metrik

(08) Änderung der Schemata: Inwieweit können Annotationsschemata während des laufenden Annotationsprozesses verändert werden?

Vor allem in der Pilotphase von DiSynDe sind die Annotationsschemata noch nicht vollständig ausgereift, sondern werden ständig an die Annotationspraxis angepasst. Deshalb wird bei diesem ebenfalls obligatorischen Attribut unterschieden, welche Änderungen im Tagset des Schemas erlaubt sind. Am wahrscheinlichsten ist der Fall einer nachträglichen Einfügung unvorhergesehener Tags zum bereits bestehenden Tagset. Unterstützt ein Annotationswerkzeug eine solche Änderung, so wird dies mit 40% auf der Verhältnisskala vermerkt. Können zusätzlich bestehende Tags abgeändert oder trotz bereits getätigter Annotationen komplett gelöscht werden, werden diese Faktoren ebenfalls mit je 30% bewertet.

Mehrfachauswahl: a OR b OR c	Schwelle: ≥ 100%
(a) nachträgliche Einfügung von Tags möglich	40%
(b) nachträgliche Änderung von Tags möglich	30%
(c) nachträgliche Löschung von Tags möglich	30%

Tabelle 18: Änderung der Schemata - Metrik

(09) Validierung gegen Schema: Kann die Annotation nach einer Änderung des Schemas gegen das zugrundegelegte Annotationsschema validiert werden?

Thematisch angelehnt an die Änderung von Schemata ist die Validierung von Annotationen gegen das zugrundegelegte Schema. Änderungen im Schema dürfen keine Inkonsistenzen zu bereits getätigten Annotationen verursachen. Deshalb müssen Annotationen innerhalb des Werkzeugs bezüglich des Schemas validiert werden können. Dieses Muss-Attribut besitzt nur eine binäre Werteskala, da eine teilweise Validierung den Ansprüchen an die Sicherung der Konsistenz der Annotationen nicht gerecht werden würde. Dementsprechend liegt ein akzeptables Einstufungsniveau nur bei der Implementierung einer internen Validierungsfunktion vor.

Einfachauswahl: a	Schwelle: ≥ 100%
(a) interne Validierung möglich	100%

Tabelle 19: Validierung gegen Schema - Metrik

(10) Im-/Export von Stand-off Formaten: Ist es möglich Stand-off Annotationsformate anderer Werkzeuge zu importieren oder zu exportieren?

Alle getesteten Werkzeuge liefern eigene Stand-off Modelle zur Speicherung der Annotationen. Das diesem Attribut übergeordnete Kriterium Interoperabilität beschreibt jedoch die Eignung eines Werkzeugs mit anderen Systemen zusammenzuwirken, wobei das Annotationsformat die Schnittstelle zur Kombinierbarkeit mit anderen Tools darstellt. Ist ein Annotationswerkzeug in der Lage mindestens ein Stand-off Format eines anderen Programms zu importieren und korrekt darzustellen, gilt dieses Attribut als zur Hälfte erfüllt. Wenn neben der Speicherung im werkzeugspezifischen Format zusätzlich die Möglichkeit besteht, die Annotationen in ein beliebiges anderes Stand-off Format zu exportieren, gilt das Attribut als vollständig erfüllt. Ein hohes Maß an Interoperabilität ist zwar vor allem für die unmittelbare

Zukunft von DiSynDe in Hinblick auf die Zusammenarbeit mit anderen Projekten wünschenswert, aber im Vergleich zu anderen Anforderungen nur als Soll-Attribut einzustufen. Akzeptanz hinsichtlich dieses Attributs besteht bereits, wenn entweder eine Import- oder eine Exportfunktion vorhanden ist.

Mehrfachauswahl: a OR b	Schwelle: ≥ 50%
(a) Import anderer Stand-off Annotationsformate möglich	50%
(b) Export anderer Stand-off Annotationsformate möglich	50%

Tabelle 20: Im-/Export von Stand-off Formaten - Metriken

(11) Im-/Export von Inline Formaten: Ist es möglich Inline Annotationsformate anderer Werkzeuge zu importieren oder zu exportieren?

Dieses Attribut schließt sich thematisch direkt an das Vorhergehende an, mit dem, Unterschied dass hier die Möglichkeiten zum Import und Export von Inline-Formaten untersucht werden. Für die Auswahlregel, Gewichtung und Bestimmung der Einstufungsniveaus gelten dieselben Angaben, wie für den *Im-/Export von Stand-off Formaten*. Um den scheinbaren Widerspruch zwischen dem Export von unflexiblen Inline Formaten und der Anforderung an ein standardisiertes Annotationsformat aufzulösen, sei an dieser Stelle nochmals betont, dass der Export eines Inline Formats lediglich eine optionale Ergänzung zum für alle Werkzeuge obligatorischen Stand-off Annotationsformat darstellt.

Mehrfachauswahl: a OR b	Schwelle: ≥ 50%
(a) Import von Inline-Annotationen möglich	50%
(b) Export als Inline-Annotation möglich	50%

Tabelle 21: Im-/Export von Inline Formaten - Metrik

(12) Wohlgeformtheit der Annotation: Ist der XML-Quelltext des Annotationsformats nach Definition des W3C wohlgeformt?

Mit dem einzigen Muss-Attribut in diesem Untersuchungsbereich soll die Wohlgeformtheit der XML-Annotation nach Definition des W3C untersucht werden. Die möglichen Messwerte sind binär, denn ein Dokument kann nicht nur teilweise wohlgeformt sein, sondern muss *alle* Anforderungen der Wohlgeformtheit erfüllen:

i. Ein XML-Dokument besteht aus dem Prolog und mindestens einem Element.
ii. Es gibt nur ein Hauptelement (Wurzelelement oder auch root element).
iii. Alle Elemente sind richtig verschachtelt. Sie dürfen sich nicht überlappen.
iv. Alle Attributwerte stehen in einfachen oder doppelten Anführungszeichen.
v. Ein Element darf nicht zwei Attribute mit demselben Namen besitzen.
vi. Kommentare dürfen nicht direkt in den Elementen eingefügt werden.
vii. Reservierte Zeichen wie „<“ oder „>“ sind in der speziellen Form < bzw. > anzugeben. (NOACK 1998, S. 37)

Zur Überprüfung dieser Anforderungen wurde der online verfügbare XML-Validator von www.validome.org benutzt:

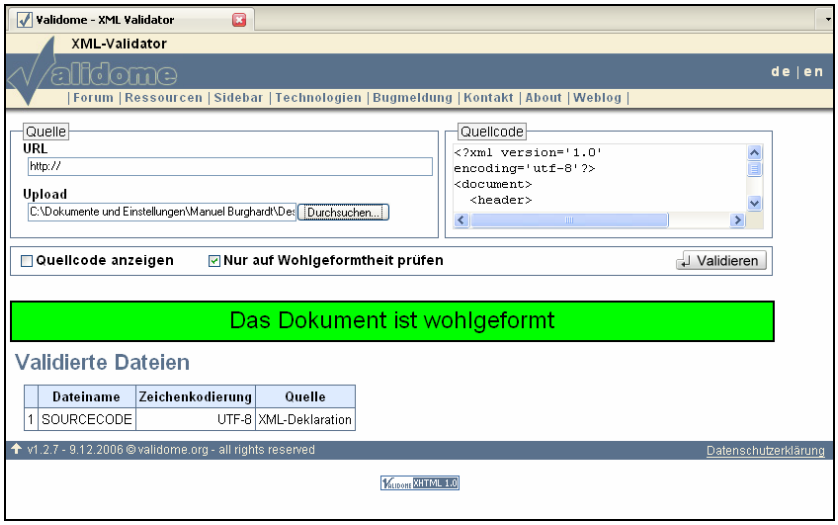

Abbildung 14: Wohlgeformtheitscheck für XML-Dateien, nach Validome (2008)

Einfachauswahl: a	Schwelle: ≥ 100%
(a) Wohlgeformtheit des Annotationsformats nach Definition des W3C	100%

Tabelle 22: Wohlgeformtheit der Annotation - Metrik

(13) Kompatibilität des Annotationsformats: Kann das Stand-off Format, welches das Annotationswerkzeug standardmäßig produziert, auch von anderen Systemen gelesen werden?

Nicht nur die Kompatibilität zu anderen Werkzeugen durch Im- und Exportfunktionen, sondern auch die Kompatibilität über das eigene Annotationsformat tragen zur Interoperabilität eines Tools bei. Wenn das standardmäßige Stand-off XML Format von mindestens einem anderen Werkzeug verarbeitet werden kann, gilt dieses binäre Attribut als erfüllt. Die Kompatibilität des Annotationsformats eines Tools zu anderen Systemen ist zwar wünschenswert, aber nicht zwingend notwendig um die DiSynDe Anforderungen zu erfüllen. Das Attribut wird deshalb als optional eingestuft.

Einfachauswahl: a	Schwelle: ≥ 100%
(a) Kompatibilität des Annotationsformats mit mindestens einem anderen Format	100%

Tabelle 23: Kompatibilität des Annotationsformats - Metrik

(14) Handbuch: Wie umfangreich ist das Handbuch für das Annotationswerkzeug?

Wenn Fragen beim Erlernen einer Software auftreten ist das Handbuch die erste Wahl um entsprechende Antworten zu finden. Deshalb wird das entsprechende Attribut als obligatorisch gewertet. Je nach Funktionsumfang des Annotationswerkzeugs muss das Handbuch angemessen lang sein. Eine pauschale Seitenzahl zum angemessenen Umfang eines Handbuchs kann an dieser Stelle jedoch nicht definiert werden, da eine möglichst hohe Seitenzahl nicht gleichzeitig die inhaltliche Adäquatheit der Anleitung belegt, sondern individuell davon abhängt, wie erklärungsbedürftig Funktionsumfang, Steuerbarkeit und Visualisierung eines Tools sind. Dabei bedeutet eine überdurchschnittlich ausführliche Softwareanleitung die detaillierte und verständliche Erklärung aller Funktionsweisen und Mechanismen des Werkzeugs, was einer vollen Erfüllung des Attributs gleichkommt. Eine ausreichende Dokumentation beschreibt zumindest die Kernfunktionen und -konzepte des Werkzeugs, wird aber nur mit 50% bewertet. Ein Handbuch, das nur einige Funktionen in unangemessener, nicht nachvollziehbarer Weise beschreibt, wird mir 0%

auf der Verhältnisskala vermerkt, ebenso wie ein gar nicht vorhandener Leitfaden zur Anwendung des Programms.

Einfachauswahl: a XOR b XOR c	Schwelle: ≥ 50%
(a) sehr ausführliche Dokumentation vorhanden	100%
(b) ausreichende Dokumentation vorhanden	50%
(c) unzureichende oder gar nicht vorhandene Dokumentation	0%

Tabelle 24: Handbuch – Metrik

(15) Hilfesystem: Wie umfangreich ist das eingebaute Hilfesystem für das Annotationswerkzeug?

Im Unterschied zum externen Handbuch muss das eingebaute Hilfesystem aus dem Werkzeug heraus verfügbar sein, und darüber hinaus eine angemessene Hilfestellung zu relevanten Problembereichen zur Verfügung stellen, um unklare Mechanismen schnell und verständlich zu erklären. In Hinblick auf die Werteskala und die entsprechende prozentuale Verteilung kann getrost auf die vorhergehende Argumentation des *Handbuch* Attributs verwiesen werden. Im Unterschied zu diesem wird das Vorhandensein eines eingebauten Hilfesystems jedoch lediglich als optionales Attribut eingestuft, da eine gute Dokumentation einen ähnlichen Zweck wie ein internes Hilfesystem erfüllen kann.

Einfachauswahl: a XOR b XOR c	Schwelle: ≥ 50%
(a) sehr ausführlichs Hilfesystem vorhanden	100%
(b) ausreichendes Hilfesystem vorhanden	50%
(c) unzureichendes oder gar nicht vorhandenes Hilfesystem	0%

Tabelle 25: Hilfesystem – Metrik

(16) Kurzinfo: Werden die Funktionen des Annotationswerkzeugs bei einem Mouseover[39] auf ein beliebiges Element konsequent durch eine aussagekräftige Kurzinfo beschrieben?

[39] Ein so genanntes Mouseover bezeichnet den Effekt der hervorgerufen, wird, wenn man den Mauszeiger in einem Programm über ein bestimmtes Element bewegt (ohne eine Maustaste zu drücken) und damit eine bestimmte Aktion, wie z.B. das Erscheinen eines Kontextmenüs, herbeiführt.

Mit diesem Attribut soll das Vorhandensein von Kurzinfos zu beliebigen Steuerelementen analysiert werden, welche durch Bewegen des Mauszeigers über das entsprechende Element erscheinen.

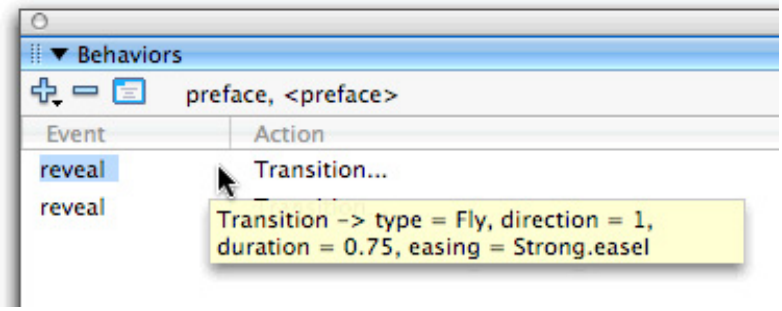

Abbildung 15: Beispiel für Tooltips bei Mouseover

Wie schon das Hilfesystem, ist auch dieses Attribut nicht obligatorisch, trägt aber dennoch entscheidend zum Erlernungsprozess einer Software bei. Es hält darüber hinaus die Frustrationsgrenze des ungeübten Nutzers niedrig, weil dieser an jeder Stelle Kurzinformationen zur Funktionalität unbekannter Steuerelemente erhalten kann. Die volle Erfüllung des Attributs bedingt eine konsequente Beschreibung aller vorhandenen Funktionen. Die Beschreibung der wichtigsten Funktionen führt zu einem 50-prozentigen Erfüllungsgrad, sporadische oder gar nicht vorhandene Kurzinfos werden mit einem Wert von 0% bemessen.

Einfachauswahl: a XOR b XOR c	Schwelle: ≥ 50%
(a) konsequente Funktionsbeschreibung durch Kurzinfo vorhanden	100%
(b) Beschreibung der wichtigsten Funktionen durch Kurzinfo vorhanden	50%
(c) unzureichende oder gar nicht vorhandene Funktionsbeschreibung durch Kurzinfo	0%

Tabelle 26: Kurzinfo – Metrik

(17) Praktische Lernhilfen: Gibt es zusätzlich zum Handbuch praktische Lernhilfen, welche die Erlernbarkeit des Annotationswerkzeugs steigern?

Neben dem Handbuch stellen praktische Lernhilfen ein weiteres wichtiges Instrument zum Erlernen einer Software dar. Der Effekt, den zusätzliche Lernhilfen

auf die Erlernbarkeit haben können, ist so positiv zu bewerten, dass dieses Attribut als obligatorisch gewichtet wird. Dabei können Tutorials und Leitfäden zur konkreten Anwendung als besonders effektiv eingestuft werden (40%), wird hier doch Schritt für Schritt erklärt wie bestimmte Praxisziele am besten zu erreichen sind. Videos verfolgen eine ähnliche Funktion, werden allerdings geringer als Tutorials bewertet (20%), weil das im Video gesehene meist nicht sofort im Annotationswerkzeug umgesetzt werden kann. Falls das Lehrvideo parallel zum Annotationswerkzeug geöffnet wird, ist es wahrscheinlich, dass das Video schneller abspielt als der Lernende die Information am praktischen Beispiel nachvollziehen kann. Vorlagen oder Übungsdateien steigern die Erlernbarkeit durch anwendbare Beispiele und ermöglichen so einen schnellen Einstieg in die Programmpraxis. Verglichen mit einem guten Tutorial tragen sie aber nur halb so viel zur Erfüllung des Attributs bei (20%), weil sie meist schon eine gewisse Grundkenntnis des Systems voraussetzen, um nachvollziehbar zu sein. Ähnlich verhält es sich mit Sekundärliteratur zur Software (20%), welche häufig auf Grundlagen des Handbuchs oder des Tutorials aufbaut und vornehmlich weiterführende Aspekte behandelt. Diese Art der Lernhilfe ist aber vor allem im fortgeschrittenen Lernprozess äußerst nützlich. Im Idealfall können bei einem Werkzeug alle vier Ausprägungen gemessen werden, was zu einem Gesamtwert von 100% führt. Das Einstufungsniveau für akzeptable Werte beginnt allerdings schon bei 40%, so dass das Vorhandensein eines Tutorials oder zweier beliebiger anderer Lernhilfen bereits zur zufriedenstellenden Erfüllung des Attributs ausreicht.

Mehrfachauswahl: a OR b OR c OR d	Schwelle: ≥ 40%
(a) Tutorial oder How-To vorhanden	40%
(b) Vorlagen oder Übungsdateien vorhanden	20%
(c) Lernvideo vorhanden	20%
(d) Sekundär-/Forschungsliteratur vorhanden	20%

Tabelle 27: Praktische Lernhilfen – Metrik

(18) Problemlösungsstrategien: Welche Lösungsstrategien stehen bei Problemen mit dem Annotationswerkzeug zur Verfügung?

Im fortgeschrittenen Lernstadium kann es häufig vorkommen, dass Probleme mit dem Annotationswerkzeug auftreten, welche auch durch vorhandene Lernhilfen

und ausführliche Handbücher, Hilfesysteme und Kurzinfos nicht gelöst werden können. Deshalb ist es wichtig, dass zusätzlich Problemlösungsstrategien vorhanden sind. Mailinglists stellen eine häufig verbreitete Methode zur Lösung von Problemen dar. Hier können konkrete Fragen per E-Mail an andere Benutzer der Software gesendet werden, welche in vielen Fällen praktische Hinweise aus eigener Erfahrung mit dem Werkzeug zur Lösung eines Problems beisteuern können. Eine Archivfunktion, welche die Fragen und eventuelle Antworten aller Benutzer dokumentiert und zugänglich macht, stellt einen idealen Ausgangspunkt dar, um Antworten auf Fragen zur Software zu recherchieren. Eine Mailinglist mit Archiv soll deshalb bereits 50% zur maximalen Attributerfüllung beitragen, falls kein Archiv vorhanden ist, werden nur 30% vergeben. Genauso hoch wird auch der direkte E-Mail Support von Herstellerseite gewichtet, wo Probleme und Fragen zur Software direkt an die Entwickler des Werkzeugs geschickt werden können. Der Nachteil hierbei ist, dass je nach Beschäftigung der Systementwickler eine Antwort unter Umständen sehr lange auf sich warten lässt. Viele Hersteller bieten auf ihrer Homepage oder in der Dokumentation eine Liste häufig gestellter Fragen, mit standardisierten Antworten, an. Diese FAQs behandeln aber meist nur sehr grundlegende Fragen und können deshalb bei speziellen Problemstellungen in der Regel nicht weiterhelfen. Sie tragen mit 20% von allen möglichen Problemlösungsstrategien am wenigsten zur Erfüllung dieses optionalen Attributs bei.

Mehrfachauswahl: (a XOR b) OR c OR d	Schwelle: ≥ 50%
(a) Mailinglist Archiv mit einsehbarem Archiv vorhanden	50%
(b) Mailinglist ohne einsehbares Archiv vorhanden	30%
(c) direkter E-Mail Support von Herstellerseite vorhanden	30%
(d) FAQs von Herstellerseite vorhanden	20%

Tabelle 28: Problemlösungsstrategien - Metrik

(19) Installationsaufwand: Welcher zeitliche und technische Aufwand ist nötig, um das Annotationswerkzeug erstmalig zu installieren?

Während des Projekts wird von einer Vielzahl von Annotatoren verteilt an unterschiedlichen Orten mit einem Annotationswerkzeug gearbeitet, d. h. es muss jedem DiSynDe Annotator an jedem Ort möglich sein, die Software selbst mit minimalen technischen Kenntnissen zu installieren. Der geringste Aufwand besteht, wenn die

Installation durch bloßes Klicken auf eine ausführbare Datei vollzogen werden kann, was einer 100-prozentige Erfüllung hinsichtlich eines möglichst geringen Installationsaufwands entspricht. Liegt die Installationsdatei als JAR-Datei vor, so sinkt der Wert auf der Verhältnisskala des obligatorischen Systemattributs auf 60%, da dies zumindest die Installation einer Java Laufzeitumgebung, und deshalb grundlegende technische Kenntnisse voraussetzt. Wenn zum Betreiben des Tools eine komplexe Konfiguration nötig ist und während der Installation beispielsweise ein Server und eine Datenbank eingerichtet werden müssen, so ist dieser Aufwand dem durchschnittlichen DiSynDe Mitglied nicht zuzumuten, und trägt deshalb nur mit 20% zur Erfüllung des Attributs bei. Der Aufwand wird als unangemessen hoch bewertet wenn Schwierigkeiten bei der Installation auftreten, welche nicht ohne fundiertes technisches Wissen beseitigt werden können. Die Akzeptanzschwelle liegt bei mindestens 60%, wenn also die Installation nicht durch einfaches Ausführen einer EXE- oder einer JAR-Datei vorgenommen werden kann, gilt der Aufwand als unangemessen hoch.

Einfachauswahl: a XOR b XOR c XOR d	Schwelle: ≥ 60%
(a) geringer Aufwand, nur Klick auf ausführbare Datei	100%
(b) angemessener Aufwand, Entpacken einer JAR-Datei	60%
(c) hoher Aufwand, Einrichten eines Servers oder einer Datenbank	20%
(d) unangemessener Aufwand, Schwierigkeiten bei der Installation	0%

Tabelle 29: Installationsaufwand – Metrik

(20) Ebenenbezogene Aktionen: Inwiefern werden Aktionen unterstützt, welche sich auf eine komplette Ebene auswirken?

Bei der Bedienbarkeit von Annotationswerkzeugen ist es wichtig, dass nicht nur einzelne Annotationseinheiten bearbeitbar sind, sondern gegebenenfalls auch komplexe Aktionen ausgeführt werden können, welche sich auf eine komplette Ebene beziehen. Dabei ist die wichtigste Aktion das Ein- und Ausblenden von Annotationsebenen, denn im Normalfall sollen nicht immer alle Ebenen gleichzeitig, sondern nur die momentane Bearbeitungsebene sichtbar sein. Das Einblenden anderer Ebenen ist dann relevant, wenn die Annotationen benachbarter Ebenen zur Orientierung herangezogen werden sollen. Auch das Löschen einer gesamten Ebene trägt insgesamt zur Bedienbarkeit bei, da so unter Umständen mehrere fehlerhafte Anno-

tationen auf einmal entfernt werden können oder eine Ebene, die sich im Laufe des Projekts als ungeeignet entpuppt, unkompliziert und vollständig aus der Annotation entfernt werden kann. Sowohl das Ein- und Ausblenden einzelner Ebenen als auch die Löschfunktion tragen je 50% zur Erfüllung dieses Muss-Attributs bei, akzeptable Werte werden bereits erreicht wenn eines der beiden Merkmale vorhanden ist.

Mehrfachauswahl: a OR b	Schwelle: ≥ 50%
(a) Einblenden/Ausblenden der einzelnen Ebenen möglich	50%
(b) Löschen der gesamten Ebene möglich	50%

Tabelle 30: Ebenenbezogene Aktionen – Metrik

(21) Editierung der Schemata: Können die Annotationsschemata komfortabel über das Annotationswerkzeug editiert werden?

Da vor allem in der Pilotphase die Annotationsschemata laufend angepasst und optimiert werden müssen, ist es wichtig, dass dies möglichst komfortabel geschehen kann. Dabei wird zum einen untersucht, ob diese Änderungen über einen graphischen Schemaeditor vorgenommen werden können, und zum anderen ob Änderungen im Schema einen Neustart der Applikation nötig machen. Obwohl ein graphischer Schemaeditor sehr wichtig für technisch unbedarfte Annotatoren ist, haben doch nur wenige Tools eine solche Funktion implementiert. Wenn Schemata geändert werden sollen, muss dies in den meisten Fällen über einen externen XML-Editor geschehen, da alle Annotationsschemata entweder als DTD oder XML Schema Datei realisiert sind. Die Änderung über einen externen Editor wird nur mit 20% bewertet, wohingegen ein graphischer Schemaeditor innerhalb des Tools 60% zur Attributerfüllung beiträgt. Im Idealfall unterstützt ein Werkzeug beide Möglichkeiten der Schemamodifikation, was einem Gesamtwert von 80% entspricht. Um die 100% Marke zu erreichen, muss das Tool zusätzlich in der Lage sein, Änderungen im Schema während des laufenden Annotationsprozesses zu verarbeiten, da ein Neustart nach jeder Schemaänderung den Arbeitsfluss enorm bremst. Dieses Muss-Attribut gilt als akzeptabel erfüllt, wenn mindestens ein graphischer Schemaeditor vorhanden ist.

Mehrfachauswahl: a OR b OR c	Schwelle: ≥ 60%
(a) Editierung der Annotationsschemata über graphischen Schemaeditor innerhalb des Werkzeugs möglich	60%
(b) Editierung der Annotationsschemata mit einem beliebigen XML-Editor außerhalb des Werkzeugs möglich	20%
(c) Editierung des Schemas während des laufenden Annotationsprozesses möglich (Update-Funktion)	20%

Tabelle 31: Editierung der Schemata – Metrik

(22) Flexibilität der Schemata: Inwiefern kann der Annotator innerhalb des Schemas flexibel auf unvorhergesehene Annotationsphänomene reagieren?

Mit diesem Attribut soll erfasst werden, wie starr bzw. flexibel ein verwendetes Schema und dessen Syntax ist, um unvorhergesehene, also im Schema nicht explizit erwähnte, Annotationsphänomene trotzdem angemessen annotieren zu können. Der Grad der maximalen Erfüllung setzt sich hier aus drei möglichen Teilwerten zusammen, wobei die Akzeptanzschwelle bei 50% liegt. Bietet ein Schema die Möglichkeit Elemente zu definieren, deren Wert als beliebiger String auch während des laufenden Annotationsprozesses angegeben werden kann, liegt ein effizienter Mechanismus vor um Phänomene zu annotieren, welche im Tagset noch nicht bedacht wurden. Falls bestimmte Phänomene immer wieder auftauchen, sollten diese fest in das Schema integriert werden, um eine nachhaltige Konsistenz der Annotationen zu gewährleisten. Eine ähnliche Funktion erfüllen Kommentar-Elemente innerhalb eines Schemas. Auch sie können beliebige Strings als Wert annehmen, werden in der Annotationsdatei aber immer als <comment> dargestellt. Werden also mehrere unterschiedliche Texteinheiten annotiert, so können diese in der späteren Annotation kaum voneinander unterschieden werden, da alle als *Kommentar* eingestuft werden. Da hier die Kommentarfunktion in gewisser Hinsicht missbraucht wird, um auf im Schema nicht bedachte Annotationsphänomene zu reagieren, trägt diese Ausprägung nur 30% zur maximalen Erfüllung des Attributs bei. In einigen Ausnahmefällen ist es möglich, während des Annotationsprozesses beliebige neue Tags zu generieren, welche dann auch in der Annotationsdatei gespeichert werden. Dieser Mechanismus wird allerdings nur mit 20% bewertet, da die neu generierten Tags nicht automatisch in das Annotationsschema übertragen werden. Inkonsistenzen

zwischen einzelnen Annotationen, die neben einem gemeinsamen Schema zusätzlich beliebige Tags enthalten können, sind somit vorprogrammiert.

Mehrfachauswahl: a OR b OR c	Schwelle: ≥ 50%
(a) Flexibilität durch im Schema frei definierbare Elemente, die als Wert einen beliebigen String annehmen können	50%
(b) Flexibilität durch im Schema vorgegebene Kommentar-Elemente, die als Wert einen beliebigen String annehmen können	30%
(c) Flexibilität, durch die Generierung beliebiger neuer Tags während des gesamten Annotationsprozesses, welche aber nicht in das verwendete Annotationsschema nachgetragen werden	20%

Tabelle 32: Flexibilität der Schemata – Metrik

(23) Zugriffsrechte: Können Zugriffsrechte für einzelne Annotationsdateien und deren Ebenen vergeben werden?

Die Vergabe von Zugriffsrechten stellt eine grundlegende Möglichkeit zur Steuerung verteilter Arbeitsabläufe dar und soll deshalb als obligatorisch gelten. Dabei wird einerseits unterschieden, ob Zugriffsrechte nur für verschiedene Annotationsdateien oder auch für einzelne Ebenen innerhalb der Annotationen vergeben werden können. Der kontrollierte Zugriff auf Dateiebene ist vor allem dazu geeignet, fertige Annotationen nur noch einem bestimmten Nutzerkreis zugänglich zu machen, um so beispielsweise sicherzustellen, dass eigentlich abgeschlossene Annotationen nicht versehentlich noch nachträglich modifiziert werden. Die Möglichkeit Zugriffsrechte für einzelne Ebenen einer Annotationsdatei zu vergeben, ermöglicht darüber hinaus die Koordination und Kontrolle der verteilt arbeitenden Annotationsgruppen. So kann gewährleistet werden, dass nicht die letzte Arbeitsgruppe Änderungen auf der untersten Annotationsebene vornimmt, ohne mit der verantwortlichen Arbeitsgruppe Rücksprache zu halten. Beide Möglichkeiten der Zugriffssteuerung werden mit je 50% bemessen. Zur Erfüllung des Attributs reicht das Vorhandensein eines der beiden vorgestellten Mechanismen.

Mehrfachauswahl: a OR b	Schwelle: ≥ 50%
(a) Vergabe von Zugriffsrechten für verschiedene Ebenen einer Annotationsdatei möglich	50%
(b) Vergabe von Zugriffsrechten für verschiedene Annotationsdateien möglich	50%

Tabelle 33: Zugriffsrechte - Metrik

(24) Suchfunktion für Textdaten: Können Rohtexte und Annotationen über eine Suchfunktion erschlossen werden?

Obwohl dieses Attribut verglichen mit anderen Systemattributen nur als optional eingestuft wird, ist vor allem bei längeren Dokumenten eine Funktion zum gezielten Durchsuchen des Textes von hohem praktischem Nutzen. Sowohl die Suche im Rohtext als auch in der fertigen Annotation soll mit je 50% beurteilt werden. Eine Suchfunktion für den Rohtext kann während der laufenden Annotation die Arbeit erleichtern, da beispielsweise gezielt orthographisch ähnliche Phänomene ausfindig gemacht werden können, und so die Annotationsentscheidung im Abgleich mit anderen relevanten Vorkommen im Text gefällt werden kann. Zum Durchsuchen annotierter Daten sind so genannten Query- oder Abfragefunktionen nötig, welche spätere Korpusabfragen simulieren, um Aufschluss über Nutzen und Relevanz bestimmter Annotationen geben. Das Attribut gilt als akzeptabel erfüllt, wenn eine der beiden Suchfunktionen implementiert ist.

Mehrfachauswahl: a OR b	Schwelle: ≥ 50%
(a) Wortsuchfunktion für Rohtexte vorhanden	50%
(b) Query-Tool zum Durchsuchen der Annotationen vorhanden	50%

Tabelle 34: Suchfunktion für Textdaten – Metrik

(25) Anpassung der Darstellung: Inwiefern kann die Darstellung des Annotationswerkzeugs angepasst werden?

Gute Bedienbarkeit zeichnet sich nicht zuletzt durch ein gewisses Maß an Anpassbarkeit aus, so dass die Bedienung individuell auf verschiedene Benutzervorlieben abgestimmt werden kann. Die *Anpassung der Darstellung* eines Annotationswerkzeugs ist deshalb ein obligatorisches Attribut, innerhalb des Unterkriteriums Bedienbarkeit. Hierbei werden fünf grundlegende Möglichkeiten der Anpassung

unterschieden, von denen jede mit je 20% bewertet wird. Das Werkzeug ist hinsichtlich dieses Attributs akzeptabel, wenn es mindestens zwei der Anpassungsmöglichkeiten unterstützt. Dazu zählen unter anderem die Anpassung der Gesamtdarstellung durch die Verwendung unterschiedlicher Skins, welche sich meist auf das jeweilige *Look and Feel* der unterschiedlichen Betriebssysteme beschränken sowie die beliebige Anordnung aller Steuerelemente und Fenster an einem unsichtbaren Raster, um so die Aufteilung der Arbeitsfläche beliebig gestalten zu können. Stellen die Modifizierbarkeit der einzelnen Fenster sowie die individuelle Darstellung von Text eher allgemeine Anpassungsmöglichkeiten von Software dar, so ist die Visualisierung der Annotationseinheiten auf den speziellen Bereich der Annotationswerkzeuge beschränkt. Annotationseinheiten können je nach persönlichen Vorlieben des Annotators durch verschiedene Farben, Schriftarten, Schriftschnitte und Klammerungen vom restlichen Text abgehoben werden.

Mehrfachauswahl: a OR b OR c OR d OR e	Schwelle: ≥ 40%
(a) Anpassung durch unterschiedliche Skins	20%
(b) optionale Anordnung aller Elemente und Arbeitsflächen an einem Raster	20%
(c) Anpassung der Fenster	20%
(d) Anpassung der Schrift	20%
(e) Anpassung der Visualisierung von Annotationseinheiten	20%

Tabelle 35: Anpassung der Darstellung – Metrik

(26) Anpassung der Steuerung: Inwiefern kann die Steuerung des Annotationswerkzeugs angepasst werden?

Mindestens genauso wichtig wie das vorhergehende Attribut ist die Anpassung der Steuerung. Da jeder Benutzer andere Vorlieben und Kenntnisse bei der Interaktion mit Annotationswerkzeugen hat, bedeutet die individuelle Anpassung von Steuerungsmechanismen gleichzeitig eine effizientere Arbeitsweise. Die gängigste Form der Anpassung in diesem Bereich stellt wohl die Definition von beliebigen Verknüpfungen und Shortcuts für besonders häufig verwendete Funktionen dar. Für das Vorhandensein dieses Merkmals werden 40% auf der Werteskala vermerkt. Die Möglichkeit der individuellen Anordnung von Steuerelementen in Menüs sowie das vollständige Ausblenden oder Deaktivieren nur selten oder gar nicht ver-

wendeter Funktionen, trägt ebenfalls zu einer effizienteren Arbeitsweise bei und wird mit jeweils 30% bemessen. Obwohl hier ein Muss-Attribut vorliegt, reichen bereits 30% für eine akzeptable Erfüllung aus.

Mehrfachauswahl: a OR b OR c	Schwelle: ≥ 30%
(a) Verknüpfung häufig verwendeter Funktionen mit einem Shortcut	40%
(b) Anpassung der Menühierarchien zugunsten häufig verwendeter Funktionen	30%
(c) Ausblenden selten verwendeter Funktionen	30%

Tabelle 36: Anpassung der Steuerung - Metrik

(27) Aufteilung der Arbeitsoberfläche: Wie ist die Arbeitsfläche des Annotationswerkzeugs aufgeteilt?

Eine logisch aufgeteilte Arbeitsoberfläche kann entscheidend zur Konformität einer Software beitragen. Bei der Unterscheidung der grundlegenden Strukturierungsmöglichkeiten soll auf eine Klassifikation nach TIDWELL (vgl. 2006, S. 28ff.) zurückgegriffen werden:

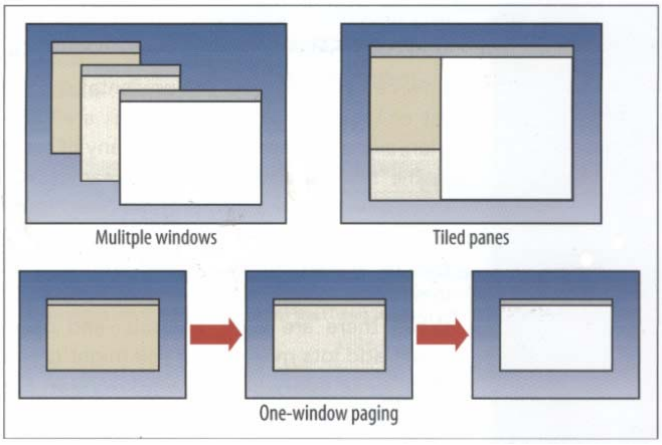

Abbildung 16: Möglichkeiten zur physischen Strukturierung der Arbeitsoberfläche, nach TIDWELL (2006, S. 28)

Hier werden zur grundlegenden physischen Strukturierung *multiple windows*, das *one-window paging* und so genannte *tiled panes* unterschieden. Dabei scheint der Ansatz mit mehreren gekachelten Bereichen innerhalb eines Fensters (*tiled panes*) für Annotationswerkzeuge die praktikabelste Form der Arbeitsoberflächenstrukturie-

rung. Die Darstellung eines Hauptarbeitsbereichs mit mehreren andockenden Werkzeugpaletten, wird bereits in einer Vielzahl von anderen Applikationen aus ähnlichen Anwendungsbereichen eingesetzt und sollte Benutzern von Annotationswerkzeugen deshalb schnell vertraut sein. Die Forderung nach Konformität der Arbeitsoberflächenaufteilung wird durch diesen Messwert zu 100% erfüllt. Die Darstellung im *one-window paging* Modus findet eigentlich typischerweise im Bereich der Webapplikationen Verwendung und scheint in Hinsicht auf Konformität der Visualisierung für Annotationswerkzeuge nur teilweise geeignet. Ist die Arbeitsfläche eines Tools dennoch in mehreren sequentiell angeordneten Fenstern organisiert, so soll das Attribut als zur Hälfte erfüllt gelten, was für ein akzeptables Leistungsniveau bereits ausreicht. Die Aufteilung in viele einzelne Fenster, welche willkürlich angeordnet werden können und einander überlappen, sind selten eine gute Wahl zur Strukturierung der Arbeitsoberfläche eines Programms (vgl. TIDWELL 2006, S. 28) und eignen sich auch für den Bereich der Annotationswerkzeuge nicht besonders gut. Dieser letzte Nominalwert entspricht 0% auf der Verhältnisskala, kann jedoch trotzdem in Kombination mit den Werten (a) oder (b) auftreten. Die Konformität der Arbeitsflächenaufteilung ist zwar wichtig, da aber die Gewöhnung an ungewöhnliche Darstellungsformen im besten Falle nur eine Frage der Zeit ist, liegt hier ein Soll-Attribut vor.

Mehrfachauswahl: (a XOR b) OR c	Schwelle: ≥ 50%
(a) mehrere gekachelte Bereiche in einem Fenster (*tiled panes*)	100%
(b) sequentielle Darstellung der einzelnen Bereiche in einem eigenen Fenster (*one-window paging*)	50%
(c) parallele Darstellung der einzelnen Bereiche in einem eigenen Fenster (*multiple windows*)	0%

Tabelle 37: Aufteilung der Arbeitsoberfläche – Metrik

(28) Gestaltung der Arbeitsoberfläche: Inwiefern sind Grundsätze der Gestaltung bei der Visualisierung der Arbeitsoberfläche von Annotationswerkzeugen berücksichtigt?

Dieses Attribut knüpft thematisch an die *Aufteilung der Arbeitsoberfläche* an und soll deshalb auch gleich gewichtet werden. Allerdings wird hier untersucht, inwiefern Grundsätze zur Gestaltung von Softwareoberflächen berücksichtigt sind (vgl.

HERCZEG 1994, S. 71-79). Am wichtigsten soll dabei eine stimmige Farbgebung mit möglichst wenigen unterschiedlichen Farbtönen (vgl. TIDWELL 2006, S. 294ff.) und ein harmonisches Größenverhältnis der einzelnen Elemente zueinander sein (je 30%). Ein unaufdringlicher Hintergrund (vgl. TIDWELL 2006, S. 291ff.) und aussagekräftige Piktogramme, welche die Funktion einer Schaltfläche angemessen visualisieren, tragen ebenfalls je 20% zur maximalen Erfüllung bei. Die Einstufungsniveaus für Akzeptabilität und Nicht-Akzeptabilität der Messwerte sind bei einer Schwelle von 60% definiert. Es gibt also vielfältige Kombinationsmöglichkeiten, um ein akzeptables Einstufungsniveau zu erreichen.

Mehrfachauswahl: a OR b OR c OR d	Schwelle: ≥ 60%
(a) Stimmige Farbpalette der einzelnen Elemente	30%
(b) Stimmiges Größenverhältnis der einzelnen Elemente zueinander	30%
(c) Schaltflächen werden als Piktogramme visualisiert	20%
(d) unaufdringlicher Hintergrund	20%

Tabelle 38: Gestaltung der Arbeitsoberfläche – Metrik

(29) Standardisierte Funktionen: Welche standardisierten Funktionen für den Steuerungsprozess einer Software sind in den Annotationswerkzeugen implementiert?

Nicht nur eine konventionalisierte und angemessene Visualisierung trägt zur Konformität einer Software bei, sondern auch standardisierte, systemübergreifende Funktionalitäten während des Steuerungsprozesses. Dabei sollen vor allem solche Funktionen untersucht werden, welche dem Benutzer ein gewisses Sicherheitsgefühl während der Anwendung einer Software vermitteln. Ein Grundbedürfnis von Anwendern bei der Benutzung beliebiger Software, stellt die möglichst umfassende Kontrolle über die Aktionshistorie dar, in welcher verschiedene Systemzustände und Benutzeraktionen in sequentieller Abfolge erfasst sind. Über eine *Undo*-Funktion können Aktionen der Historie rückgängig gemacht werden. Dieses Konzept ist in den meisten gängigen Anwendungen, in denen Text in irgendeiner Form editiert wird, implementiert. Eine vorhandene *Undo*-Funktion wird mit 50% bewertet, da der Benutzer das Konzept bereits aus anderen Anwendungen kennt und das beruhigende Gefühl vermittelt wird, dass eventuelle Fehler beim Umgang mit dem Werkzeug mit Hilfe dieser Funktion einfach rückgängig gemacht werden können. Die

gegenläufige Funktion zum *Undo* stellt das *Redo* dar, welches „zur Auswahl von wiederauszuführenden, rückgängig gemachten Aktionen“ (HERCZEG 2006, S. 169) in der Historie genutzt werden kann. Da das *Redo* nur sinnvoll verwendet werden kann, wenn auch ein *Undo* implementiert ist, und weil es weniger oft angewendet wird, soll es nur mit 30% bewertet werden. Ein weiteres standardisiertes Konzept, welches dem Anwender ein Gefühl von Sicherheit gibt, stellt die *Autosave*-Funktion dar. Hierbei wird der aktuelle Arbeitsstand in regelmäßigen, häufig selbst definierbaren Abständen, gesichert. Hinsichtlich der direkten Kontrolle der Aktionshistorie trägt diese Funktion jedoch nur mit 20% zur vollen Erfüllung dieses optionalen Attributs bei, welches bei einem Wert von größer oder gleich 50% als akzeptabel gilt und in jedem Fall das Vorhandensein eines *Undo* impliziert.

Mehrfachauswahl: a OR b OR c	Schwelle: ≥ 50%
(a) *Undo* möglich	50%
(b) *Redo* möglich	30%
(c) *Autosave* möglich	20%

Tabelle 39: Standardisierte Funktionen – Metrik

(30) Selektion von Annotationseinheiten: Werden bei der Selektion der Annotationseinheiten gängige Interaktionskonzepte zur Textauswahl berücksichtigt?

Wurden die bisherigen Attribute aus dem Bereich Konformität lediglich als optional eingestuft, so stellt die konventionalisierte *Selektion von Annotationseinheiten* ein absolutes Muss dar, hat doch jeder Benutzer des Annotationswerkzeugs zumindest grundlegende Kenntnisse im Umgang mit gängigen Texteditoren oder Textprogrammen. Deshalb soll an dieser Stelle untersucht werden inwiefern Konventionen aus dem Bereich der Textauswahl auch bei der Selektion von Annotationseinheiten im Rohtext eingehalten werden. Den bekanntesten Mechanismus zum Selektieren bestimmter Textbereiche stellt die „click, drag, release“ (TIDWELL 2006, S. 246) Methode dar, wo durch Drücken der Maustaste der Anfang des Selektionsbereichs, durch Bewegen des Mauszeigers der Umfang des Bereichs und durch Loslassen der gedrückten Taste das Ende des Textbereichs definiert wird. Da diese Methode am besten geeignet ist um beliebige Texteinheiten zu selektieren, erfüllt sie das Attribut zu 30%. Die zweite, jedoch nicht ganz so verbreitete, Methode zur Se-

lektion frei definierbarer Textbereiche, stellt die Kombination aus *Shift* + *Richtungstasten* dar. Dieser Mechanismus ist im Gegensatz zur *click, drag, release* Methode nicht automatisch jedem Anwender eines Textwerkzeugs bekannt, kann aber bei entsprechender Kenntnis durchaus effektiv eingesetzt werden und soll deshalb mit 20% zur maximalen Erfüllung beitragen. Konzepte zur Auswahl ganzer Wörter, wie etwa der Doppelklick auf ein Wort, oder die Kombination von *Shift* + *Strg* + *Richtungstasten* zur Wortselektion, werden mit jeweils 15% bewertet, da sie, verglichen mir den vorhergehenden Methoden, nur eingeschränkt zur Textselektion geeignet sind. Das Bewegen des Cursors durch den Text mit Hilfe der Richtungstasten trägt nur indirekt zur Auswahl von Annotationseinheiten bei und wird deshalb nur mit 10% auf der Verhältnisskala angegeben. Die Schwelle von 45% kann über vielfältige Kombinationsmöglichkeiten von Werten erreicht werden, setzt jedoch immer das Vorhandensein mehrerer grundlegender Mechanismen zur Textselektion voraus.

Mehrfachauswahl: a OR b OR c OR d OR e OR f	Schwelle: ≥ 45%
(a) Klicken, Ziehen, Loslassen: Selektion eines frei definierten Textbereichs	30%
(b) Shift + Richtungstasten: Selektion eines frei definierten Textbereichs	20%
(c) Doppelklick: Ganzes Wort auswählen	15%
(d) Shift + Strg + Richtungstasten: Ganzes Wort auswählen	15%
(e) Einfachklick: Textzeiger an diese Stelle bewegen	10%
(f) Richtungstasten: Textzeiger in Richtung der gedrückten Taste bewegen	10%

Tabelle 40: Selektion von Annotationseinheiten - Metrik

5.5.3 Bewertungsregeln

Der Bewertungsprozess verfolgt unterschiedliche Erkenntnisinteressen. Zum einen soll jedes einzelne Attribut auf sein erreichtes Einstufungsniveau hin überprüft werden, um spezifische Stärken und Schwächen für jedes Tool ermitteln zu können. Zum anderen sollen Gesamtaussagen zur Qualität der Annotationswerkzeuge gemacht werden. Die Bewertung der fünf Subkriterien Angemessenheit, Interoperabilität, Erlernbarkeit, Bedienbarkeit und Konformität erfolgt durch Aufrechnung der einzelnen Attributmessungen zu einem gewichteten Gesamtwert. Dabei werden für jedes Werkzeug alle optionalen Attribute mit akzeptablem

Einstufungsniveau addiert (A) und zur Summe aller akzeptablen obligatorischen Attribute (B) gezählt, wobei diese doppelt gewichtet werden. Dieser Gesamtwert dient schließlich als Teiler für die Gesamtsumme aller möglichen, optionalen (C) und obligatorischen Attribute (D) eines Subkriteriums, wobei obligatorische Attribute gegenüber optionalen Attributen wiederum doppelt gezählt werden. Das Ergebnis ist immer eine Dezimalzahl zwischen 0 und 1, welche dann in eine entsprechende Prozentzahl umgewandelt wird. So kann für jedes Werkzeug bestimmt werden, zu wie viel Prozent es akzeptable Werte für ein bestimmtes Subkriterium abliefert.

$$(C + 2*D) : (A + 2*B)$$

Die Bewertung der übergeordneten Qualitätskriterien Funktionalität und Benutzbarkeit erfolgt nach demselben Schema. Analog können auch alle ermittelten Einstufungsniveaus zusammen addiert werden, um so eine tendenzielle Gesamtaussage über die Qualität der vier Annotationswerkzeuge machen zu können (vgl. BÄCHLE 1996, S. 6).

5.6 Planung der Evaluationsdurchführung

Im letzten Schritt vor der tatsächlichen Durchführung der Evaluation, müssen schließlich geeignete Testmaterialien, unter Berücksichtigung des vorher definierten Anforderungskontextes, erstellt werden. Dies beinhaltet sowohl die Auswahl eines repräsentativen Textausschnitts als auch die Schaffung eines vorläufigen Annotationsschemas. Die Testmaterialien sollen den späteren Aufgaben- und Anwendungsbereich möglichst adäquat wiedergeben und eventuelle Problemsituationen kenntlich machen. Außerdem muss an dieser Stelle geklärt werden wer die Evaluation ausführt, und welchen Kenntnisstand der Software die Testperson hat, bzw. ob die Zuhilfenahme des Handbuchs oder anderer Hilfsmittel zur Erfüllung der Testaufgaben erlaubt sind.

5.6.1 Testmaterialien

Bei der Wahl eines geeigneten Textausschnitts kann auf grundlegende Vorarbeiten früherer DiSynDe Workshops zurückgegriffen werden. Hier wurde bereits ein vorläufiges Pilotkorpus unter Leitung von Prof. Hans-Ulrich Schmid erstellt, welches verschiedene deutsche Texte vom 11. bis zum 17. Jahrhundert enthält. Aus diesem provisorischen Korpus wurde wiederum ein exemplarischer Textausschnitt mit

möglichst vielen unterschiedlichen Annotationsphänomenen ausgewählt, und im Zuge einer Vorstudie in tabellarischer Form von allen Arbeitsgruppen annotiert. Dieser Textausschnitt eignet sich hervorragend als Testtext für die Evaluation von Annotationswerkzeugen, da er zum einen von den DiSynDe Annotatoren, also den späteren Nutzern der Werkzeuge, selbst ausgewählt wurde, und zum anderen bereits grundlegende Annotationen auf unterschiedlichen Ebenen enthält, welche es erlauben exemplarische Tagsets für bestimmte Ebenen abzuleiten, um so ein provisorisches Annotationsschema zu erstellen. Dabei scheinen vor allem die Annotationen für die beiden Ebenen *Wortarten* und *Wortgruppen* in sich stimmig und durchdacht, wohingegen die anderen Annotationsebenen im *Ehebüchlein* nicht immer nachvollziehbar, bzw. zum Teil unvollständig annotiert sind. Um die Annotationswerkzeuge hinsichtlich ihrer Eignung für Mehrebenenannotation zu testen, reichen jedoch bereits zwei exemplarische Ebenen als Grundlage für ein provisorisches Annotationsschema aus:

Ebene der Wortarten	Ebene der Wortgruppe
sb – Substantiv	np - Nominalphrase
sb.prop – Eigenname	vp - Verbalphrase
vb.voll – Vollverb	pp - Präpositionalphrase
vb.modal – Modalverb	adv.p – Adverbialphrase
…	…

Tabelle 41: Ausschnitt eines provisorischen Annotationsschemas

Bei dem ausgesuchten Textauschnitt handelt es sich um die ersten vier Sätze eines geistlichen Prosawerks, mit dem Originaltitel „Ein püechel von der regel der heyligen ee“, oder sinngemäß übersetzt: „Ein Ehebüchlein“. Der Originaltitel entstand vermutlich im 14. Jahrhundert, die hier verwendete Quelle entstammt Band 112 der *Zeitschrift für deutsches Altertum*, herausgegeben von Michael Dallapiazza im Jahre 1983:

> Ein püechel von der regel der heyligen ee
>
> „Hor mich, so wil ich dir weisen, mit we du den teuffel überwindest.“ Also stet geschriben in dem puch Thobie an dem 6. capitel. Dise wort sein götlichen vnd nit menschlichen geredt worden, wann durch den namhaften furstengel sand Raphael; vnd darumb das sy durch ainen so wirdigen engel sein gesprochen worden, darumb syllen sy dester vleissiger

> von vns gemerckt werden. Vnd ist auch dauon zewissen, das der genant engel dy wort gesprochen hat zu dem jungeren Thobias, der ein sun was des eltern Thobias. (vgl. DALLAPIAZZA 1983, S. 261f.)

Zusätzlich wurde der Textausschnitt um einige Einzelwörter aus anderen Texten des Pilotkorpus ergänzt, um die Verarbeitung von Sonderzeichen besser untersuchen zu können:

mǒzerim, îv, rethę, chûzal, stůtwaide, swáner, aín, ê, é

5.6.2 Testszenario

Die Spezifizierung des Testszenarios beinhaltet eine kurze Skizzierung der angedachten Annotationsaufgabe sowie eine Charakterisierung des Durchführers der Evaluation.

Die Annotationsaufgabe besteht darin, den ausgewählten Testtext auf zwei sprachlichen Ebenen zu annotieren. Dabei wird auf ein provisorisches Annotationsschema zurückgegriffen, welches alle zu erwartenden Annotationsphänomene, wohlgemerkt nur für den Testtext, enthält. Für die Durchführung gibt es keinen festgesetzten zeitlichen Rahmen, da nicht gemessen werden soll, wie schnell bestimmte Ziele erreicht werden, sondern auf welche Weise dies geschieht. Außerdem dürfen zu jedem Zeitpunkt das Handbuch oder andere Lernhilfen bzw. Problemlösungsstrategien zur Erfüllung der Annotationsaufgabe benutzt werden. Der evaluierende Annotator hat bereits Erfahrungen mit allgemeinen Funktionsweisen von Annotationswerkzeugen, was aber keinen Einfluss auf die Bewertung hat, sondern lediglich eine objektive Einschätzung der einzelnen Tools begünstigt. Die gesamte Evaluation wird unter besonderer Berücksichtigung der DiSynDe Anforderungen durchgeführt, d.h. bei der Evaluation wird die Nutzung der Werkzeuge soweit möglich aus Sicht eines virtuellen DiSynDe Annotators simuliert.

5.7 Durchführung der Evaluation

Nach diesen vorbereitenden Maßnahmen steht schließlich die Durchführung der Evaluation auf dem Plan. Die Durchführung beinhaltet konkrete Messungen unter Verwendung der Testmaterialien sowie eine Einstufung der Messwerte in Hinblick

auf die Erfüllung der anfangs definierten Anforderungen. Am Ende der Evaluation steht die Bewertung des evaluierten Programms durch Zusammenfassen der einzelnen Messungen.

5.7.1 Messergebnisse

Die nachfolgenden Tabellen enthalten in komprimierter Form die Messergebnisse der Evaluation sowie einige grundlegenden Informationen zu den einzelnen Metriken, wie z. B. die Gewichtung oder die logische Auswahlregel. Die Färbung der Prozentwerte gibt Auskunft darüber, ob ein Annotationswerkzeug einen akzeptablen Wert (blau) bei der Evaluation erreicht hat oder nicht (rot). Die vier Tools werden mit den jeweiligen Anfangsbuchstaben abgekürzt: C = Callisto, G = GATE, M = MMAX2, U = UAM CorpusTool.

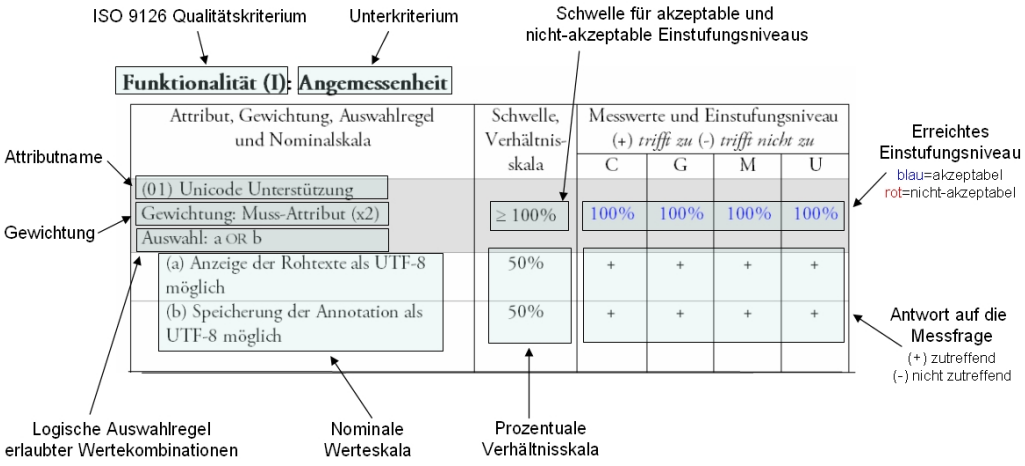

Abbildung 17: Legende zu den Ergebnisstabellen der Messungen

Funktionalität (I): Angemessenheit

Attribut, Gewichtung, Auswahlregel und Nominalskala	Schwelle, Verhältnis-skala	Messwerte und Einstufungsniveau (+) *trifft zu* (-) *trifft nicht zu*			
		C	G	M	U
(01) Unicode Unterstützung Gewichtung: Muss-Attribut (*2) Auswahl: a OR b	≥ 100%	100%	100%	100%	100%
(a) Anzeige der Rohtexte als UTF-8 möglich	50%	+	+	+	+
(b) Speicherung der Annotation als UTF-8 möglich	50%	+	+	+	+
(02) Alternative Zeichenkodierung Gewichtung: Soll-Attribut (*1) Auswahl: a OR b	≥ 100%	100%	100%	100%	100%
(a) Anzeige der Rohtexte als ASCII, ISO-8859 etc. möglich	50%	+	+	+	+
(b) Speicherung der Annotation als ASCII, ISO-8859 etc. möglich	50%	+	+	+	+
(03) Unterstützte Dateiformate Gewichtung: Muss-Attribut (*2) Auswahl: a OR b OR (c XOR d)	≥ 40%	60%	100%	40%	40%
(a) Textdateien: *.TXT	40%	+	+	+	+
(b) MS-Word Dokumente: *.DOC	40%		+		
(c) Markupsprachen: *.XML , *.HTML, *.SGML (ohne Interpretation der Tags)	20%	+	-	-	-
(d) interpretierte Markupsprachen: *.XML, *.HTML, *.SGML (mit Interpretation der Tags)	20%	-	+	-	
(04) Modifizierbarkeit der Rohtexte Gewichtung: Soll-Attribut (*1) Auswahl: a OR b	≥ 50%	0%	100%	100%	0%
(a) Modifizierung während der Annotation möglich	50%	-	+	+	-
(b) Modifizierung vor der Annotation möglich	50%	-	+	+	-
(05) Tokenisierung der Rohtexte Gewichtung: Muss-Attribut (*2) Auswahl: (a XOR b) OR c	≥ 40%	40%	100%	100%	0%
(a) interne Tokenisierung mit Möglichkeit der Parametrisierung	60%	-	+	+	-
(b) interne Tokenisierung ohne Möglichkeit der Parametrisierung	40%	+	-	-	-
(c) externe Tokenisierung möglich	40%	-	+	+	-

(06) Automatisierbarkeit der Annotation Gewichtung: Muss-Attribut (*2) Auswahl: a OR b	≥ 40%	0%	100%	0%	40%
(a) Automatisierbarkeit durch Verwendung von regulären Ausdrücken möglich	60%	-	+	-	-
(b) Automatisierbarkeit durch Verwendung eines Lexikons möglich	40%	-	+	-	+
(07) Skopus der Annotationseinheiten Gewichtung: Muss-Attribut (*2) Auswahl: a OR b OR c	≥ 80%	80%	80%	80%	40%
(a) Beliebige Annotationseinheiten mit einem Start- und einem Endpunkt	40%	+	+	+	+
(b) gerichtete Relationen zwischen Annotationseinheiten auf einer Ebene möglich	40%	+	+	+	-
(c) gerichtete Relationen zwischen Annotationseinheiten und Vernetzung über verschiedener Ebenen hinweg möglich	20%	-	-	-	-
(08) Änderung der Schemata Gewichtung: Muss-Attribut (*2) Auswahl: a OR b OR c	≥ 100%	40%	100%	100%	100%
(a) nachträgliche Einfügung von Tags möglich	40%	+	+	+	+
(b) nachträgliche Änderung von Tags möglich	30%	-	+	+	+
(c) nachträgliche Löschung von Tags möglich	30%	-	+	+	+
(09) Validierung gegen Schema Gewichtung: Muss-Attribut (*2) Auswahl: a	≥ 100%	0%	0%	100%	0%
(a) interne Validierung möglich	100%	-	-	+	-

Tabelle 42: Messwerte für den Bereich Funktionalität (I): Angemessenheit

Funktionalität (II): Interoperabilität

Attribut, Gewichtung, Auswahlregel und Nominalskala	Schwelle, Verhältnisskala	Messwerte und Einstufungsniveau (+) trifft zu (-) trifft nicht zu			
		C	G	M	U
(10) Im-/Export von Stand-off Formaten Gewichtung: Soll-Attribut (*1) Auswahl: a OR b	≥ 50%	0%	100%	0%	50%
(a) Import anderer Stand-off Annotationsformate möglich	50%	-	+	-	+
(b) Export anderer Stand-off Annotationsformate möglich	50%	-	+	-	-

(11) Im-/Export von Inline Formaten Gewichtung: Soll-Attribut (*1) Auswahl: a OR b	≥ 50%	0%	100%	0%	50%
(a) Import von Inline-Annotationen möglich	50%	-	+	-	-
(b) Export als Inline-Annotation möglich	50%	-	+	-	+
(12) Wohlgeformtheit der Annotation Gewichtung: Muss-Attribut (*2) Auswahl: a	≥ 100%	100%	100%	100%	100%
(a) Wohlgeformtheit des Annotationsformats nach Definition des W3C	100%	+	+	+	+
(13) Kompatibilität des Annotationsformats Gewichtung: Soll-Attribut (*1) Auswahl: a	≥ 100%	100%	100%	100%	0%
(a) Kompatibilität des Annotationsformats mit mindestens einem anderen Format	100%	+	+	+	-

Tabelle 43: Messwerte für den Bereich Funktionalität (II): Interoperabilität

Benutzbarkeit (I): Erlernbarkeit

Attribut, Gewichtung, Auswahlregel und Nominalskala	Schwelle, Verhältnisskala	Messwerte und Einstufungsniveau (+) trifft zu (-) trifft nicht zu			
		C	G	M	U
(14) Handbuch Gewichtung: Muss-Attribut (*2) Auswahl: a XOR b XOR c	≥ 50%	50%	100%	50%	50%
(a) sehr ausführliche Dokumentation vorhanden	100%	-	+	-	-
(b) ausreichende Dokumentation vorhanden	50%	+	-	+	+
(c) unzureichende oder gar nicht vorhandene Dokumentation	0%	-	-	-	-

(15) Hilfesystem Gewichtung: Soll-Attribut (*1) Auswahl: a XOR b XOR c	≥ 50%	50%	0%	0%	50%
(a) sehr ausführliches Hilfesystem vorhanden	100%	-	-	-	-
(b) ausreichendes Hilfesystem vorhanden	50%	+	-	-	+
(c) unzureichendes oder gar nicht vorhandenes Hilfesystem	0%	-	+	+	-
(16) Kurzinfo Gewichtung: Soll-Attribut (*1) Auswahl: a XOR b XOR c	≥ 50%	0%	100%	50%	50%
(a) konsequente Funktionsbeschreibung durch Kurzinfo vorhanden	100%	-	+	-	-
(b) Beschreibung der wichtigsten Funktionen durch Kurzinfo vorhanden	50%	-	-	+	+
(c) unzureichende oder gar nicht vorhandene Funktionsbeschreibung durch Kurzinfo	0%	+	-	-	-
(17) Praktische Lernhilfen Gewichtung: Muss-Attribut (*2) Auswahl: a OR b OR c OR d	≥ 40%	60%	100%	60%	60%
(a) Tutorial oder How-To vorhanden	40%	+	+	+	+
(b) Vorlagen oder Übungsdateien vorhanden	20%	+	+	-	+
(c) Lernvideo vorhanden	20%	-	+	-	-
(d) Sekundär-/Forschungsliteratur vorhanden	20%	-	+	+	-
(18) Problemlösungsstrategien Gewichtung: Soll-Attribut (*1) Auswahl: (a XOR b) OR c OR d	≥ 50%	80%	80%	30%	60%
(a) Mailinglist Archiv mit einsehbarem Archiv vorhanden	50%	-	+	-	-
(b) Mailinglist ohne einsehbares Archiv vorhanden	30%	+	-	-	+
(c) direkter E-Mail Support von Herstellerseite vorhanden	30%	+	+	+	+
(d) FAQs von Herstellerseite vorhanden	20%	+	-	-	-

Tabelle 44: Messwerte für den Bereich Benutzbarkeit (I): Erlernbarkeit

Benutzbarkeit (II): Bedienbarkeit

Attribut, Gewichtung, Auswahlregel und Nominalskala	Schwelle, Verhältnis-skala	Messwerte und Einstufungsniveau (+) trifft zu (-) trifft nicht zu			
		C	G	M	U
(19) Installationsaufwand Gewichtung: Muss-Attribut (*2) Auswahl: a XOR b XOR c XOR d	≥ 60%	60%	100%	60%	100%
(a) geringer Aufwand, nur Klick auf ausführbare Datei	100%	-	+	-	+
(b) angemessener Aufwand, Entpacken einer JAR-Datei	60%	+	-	+	-
(c) hoher Aufwand, Einrichten eines Servers oder einer Datenbank	20%	-	-	-	-
(d) unangemessener Aufwand, Schwierigkeiten bei der Installation	0%	-	-	-	-
(20) Ebenenbezogene Aktionen Gewichtung: Muss-Attribut (*2) Auswahl: a OR b	≥ 50%	50%	100%	100%	100%
(a) Einblenden/Ausblenden der einzelnen Ebenen möglich	50%	+	+	+	+
(b) Löschen der gesamten Ebene möglich	50%	-	+	+	+
(21) Editierung der Schemata Gewichtung: Muss-Attribut (*2) Auswahl: a OR b OR c	≥ 60%	20%	40%	40%	100%
(a) Editierung der Annotationsschemata über graphischen Schemaeditor innerhalb des Werkzeugs möglich	60%	-	-	-	+
(b) Editierung der Annotationsschemata mit einem beliebigen XML-Editor außerhalb des Werkzeugs möglich	20%	+	+	+	+
(c) Editierung des Schemas während des laufenden Annotationsprozesses möglich (Update-Funktion)	20%	-	+	+	+

(22) Flexibilität der Schemata Gewichtung: Muss-Attribut (*2) Auswahl: a OR b OR c	≥ 50%	50%	70%	50%	30%
(a) Flexibilität durch im Schema frei definierbare Elemente, die als Wert einen beliebigen String annehmen können	50%	+	+	+	-
(b) Flexibilität durch im Schema vorgegebene Kommentar-Elemente, die als Wert einen beliebigen String annehmen können	30%	-	-	-	+
(c) Flexibilität, durch die Generierung beliebiger neuer Tags während des gesamten Annotationsprozesses, welche aber nicht in das verwendete Annotationsschema nachgetragen werden	20%	-	+	-	-
(23) Zugriffsrechte Gewichtung: Muss-Attribut (*2) Auswahl: a OR b	≥ 50%	0%	50%	0%	0%
(a) Vergabe von Zugriffsrechten für verschiedene Ebenen einer Annotationsdatei möglich	50%	-	-	-	-
(b) Vergabe von Zugriffsrechten für verschiedene Annotationsdateien möglich	50%	-	+	-	-
(24) Suchfunktion für Textdaten Gewichtung: Soll-Attribut (*1) Auswahl: a OR b	≥ 50%	50%	50%	50%	50%
(a) Wortsuchfunktion für Rohtexte vorhanden	50%	+	+	-	-
(b) Query-Tool zum Durchsuchen der Annotationen vorhanden	50%	-	-	+	+

(25) Anpassung der Darstellung Gewichtung: Muss-Attribut (*2) Auswahl: a OR b OR c OR d OR e	≥ 40%	80%	60%	60%	40%
(a) Anpassung durch unterschiedliche Skins	20%	-	+	-	-
(b) optionale Anordnung aller Elemente und Arbeitsflächen an einem Raster	20%	+	-	-	-
(c) Anpassung der Fenster	20%	+	-	+	+
(d) Anpassung der Schrift	20%	+	+	+	+
(e) Anpassung der Visualisierung von Annotationseinheiten	20%	+	+	+	-
(26) Anpassung der Steuerung Gewichtung: Muss-Attribut (*2) Auswahl: a OR b OR c	≥ 30%	0%	0%	0%	0%
(a) Verknüpfung häufig verwendeter Funktionen mit einem Shortcut	40%	-	-	-	-
(b) Anpassung der Menühierarchien zugunsten häufig verwendeter Funktionen	30%	-	-	-	-
(c) Ausblenden selten verwendeter Funktionen	30%	-	-	-	-

Tabelle 45: Messwerte für den Bereich Benutzbarkeit (II): Bedienbarkeit

Benutzbarkeit (III): Konformität

Attribut, Gewichtung, Auswahlregel und Nominalskala	Schwelle, Verhältnisskala	Messwerte und Einstufungsniveau (+) trifft zu (-) trifft nicht zu			
		C	G	M	U
(27) Aufteilung der Arbeitsoberfläche Gewichtung: Soll-Attribut (*1) Auswahl: (a XOR b) OR c	≥ 50%	100%	100%	0%	50%
(a) mehrere gekachelte Bereiche in einem Fenster (*tiled panes*)	100%	+	+	-	-
(b) sequentielle Darstellung der einzelnen Bereiche in einem eigenen Fenster (*one-window paging*)	50%	-	-	-	+
(c) parallele Darstellung der einzelnen Bereiche in einem eigenen Fenster (*multiple windows*)	0%	+	-	+	-

(28) Gestaltung der Arbeitsoberfläche Gewichtung: Soll-Attribut (*1) Auswahl: a OR b OR c OR d	≥ 60%	80%	100%	80%	50%
(a) Stimmige Farbpalette der einzelnen Elemente	30%	+	+	+	+
(b) Stimmiges Größenverhältnis der einzelnen Elemente zueinander	30%	+	+	+	-
(c) Schaltflächen werden als Piktogramme visualisiert	20%	-	+	-	-
(d) unaufdringlicher Hintergrund	20%	+	+	+	+
(29) Standardisierte Funktionen Gewichtung: Soll-Attribut (*1) Auswahl: a OR b OR c	≥ 50%	20%	0%	70%	0%
(a) *Undo* möglich	50%	-	-	+	-
(b) *Redo* möglich	30%	-	-	-	-
(c) *Autosave* möglich	20%	+	-	+	-
(30) Selektion von Annotationseinheiten Gewichtung: Muss-Attribut (*2) Auswahl: a OR b OR c OR d OR e OR f	≥ 45%	75%	100%	45%	40%
(a) Klicken, Ziehen, Loslassen: Selektion eines frei definierten Textbereichs	30%	+	+	+	+
(b) Shift + Richtungstasten: Selektion eines frei definierten Textbereichs	20%	+	+	-	-
(c) Doppelklick: Ganzes Wort auswählen	15%	-	+	+	-
(d) Shift + Strg + Richtungstasten: Ganzes Wort auswählen	15%	+	+	-	-
(e) Einfachklick: Textzeiger an diese Stelle bewegen	10%	+	+	-	+
(f) Richtungstasten: Textzeiger in Richtung der gedrückten Taste bewegen	10%	-	+	-	-

Tabelle 46: Messwerte für den Bereich Benutzbarkeit (III): Konformität

5.7.2 Quantitative Auswertung

Anhand der ermittelten Messwerte können nun einerseits Stärken und Schwächen der Annotationswerkzeuge hinsichtlich einzelner Attribute ausgemacht werden. Andererseits ist die Aufrechnung beliebiger Attributgruppen zu gewichteten Durchschnittswerten und einer allgemeinen *Produktinformation* möglich:

> Comparative product tests can be described roughly as consisting of two successive phases: a testing phase and an information phase (Siderius, 1989). The second phase consists of interpreting the test results as product information […]" (EAGLES 1995, S. 46)

So zeigt sich beim Gesamtranking, dass 83,3% der Messwerte für das GATE Framework als akzeptabel eingestuft werden können, was einen beachtlichen Vorsprung vor MMAX2, mit insgesamt 72,9%, bedeutet. Der Abstand zwischen dem in der Gesamtwertung drittplatzierten Callisto mit 64,6% und dem viertplatzierten UAM CorpusTool mit 62,5%, beträgt hingegen nur wenige Prozentpunkte.

Kriterium	CALLISTO	GATE	MMAX2	UAM
Gesamtqualität der Software	64,6%	83,3%	72,9%	62,5%
Gesamtranking	3. Platz	1. Platz	2. Platz	4. Platz
erfüllte Muss-Attribute	12 von 18	15 von 18	14 von 18	11 von 18
erfüllte Soll-Attribute	7 von 12	10 von 12	8 von 12	8 von 12

Tabelle 47: Übersicht zur Gesamtqualität der Software

Wie problematisch dieses rechnerische Gesamturteil jedoch in der Realität ist, verdeutlicht die nachfolgende Tabelle, welche die erreichten Leistungsniveaus für die Qualitätskriterien Funktionalität und Benutzbarkeit darstellt:

Kriterium	CALLISTO	GATE	MMAX2	UAM
Qualität der Funktionalität	57,1%	90,5%	81%	61,9%
Gesamtranking	4. Platz	1. Platz	2. Platz	3. Platz
erfüllte Muss-Attribute	5 von 8	7 von 8	7 von 8	5 von 8
erfüllte Soll-Attribute	2 von 5	5 von 5	3 von 5	3 von 5
Qualität der Benutzbarkeit	70,4%	77,8%	66,7%	63%
Gesamtranking	2. Platz	1. Platz	3. Platz	4. Platz
erfüllte Muss-Attribute	7 von 10	8 von 10	7 von 10	6 von 10
erfüllte Soll-Attribute	5 von 7	5 von 7	5 von 7	5 von 7

Tabelle 48: Übersicht zur Qualität von Funktionalität und Benutzbarkeit

Callisto erreicht im Ranking für den Bereich Benutzbarkeit den zweiten Platz, bildet aber in Hinblick auf die funktionale Qualität das Schlusslicht der Bewertung. Ähnliche Diskrepanzen ergeben sich auch für MMAX2, welches eine Zweit- und eine Drittplatzierung verbuchen kann, sowie das UAM CorpusTool, welches einen dritten und einen vierten Platz belegt. Die prozentualen Leistungsniveaus für die einzelnen Unterkriterien belegen die Problematik eines Gesamtrankings. Offensichtlich werden mit zunehmender Abstraktion der Messwerte, diese im selben Maße unschärfer und weniger aussagekräftig, da sie individuelle Merkmale der Tools zugunsten eines gerundeten Durchschnittswerts unkenntlich machen. Nichtsdestotrotz erfüllt die Gesamtbewertung ihren Zweck. So sticht ein Werkzeug deutlich hervor, welches in allen wichtigen Anforderungsbereichen kontinuierlich Stärken aufweist und nur bei wenigen Einzelattributen Schwachstellen offenbart.

Kriterium	CALLISTO	GATE	MMAX2	UAM
Qualität der Angemessenheit	56,3%	87,5%	87,5%	56,3%
erfüllte Muss-Attribute	4 von 7	6 von 7	6 von 7	4 von 7
erfüllte Soll-Attribute	1 von 2	2 von 2	2 von 2	1 von 2
Qualität der Interoperabilität	60%	100%	60%	80%
erfüllte Muss-Attribute	1 von 1	1 von 1	1 von 1	1 von 1
erfüllte Soll-Attribute	1 von 3	3 von 3	1 von 3	2 von 3
Qualität der Erlernbarkeit	85,7%	85,7%	71,4%	100%
erfüllte Muss-Attribute	2 von 2	2 von 2	2 von 2	2 von 2
erfüllte Soll-Attribute	2 von3	2 von 3	2 von 3	3 von 3
Qualität der Bedienbarkeit	60%	73,3%	60%	60%
erfüllte Muss-Attribute	4 von 7	5 von 7	4 von 7	4 von 7
erfüllte Soll-Attribute	1 von 1	1 von 1	1 von 1	1 von 1
Qualität der Konformität	80%	80%	80%	20%
erfüllte Muss-Attribute	1 von 1	1 von 1	1 von 1	0 von 1
erfüllte Soll-Attribute	2 von 3	2 von 3	2 von 3	1 von 3

Tabelle 49: Übersicht zur Qualität von Angemessenheit, Interoperabilität, Erlernbarkeit, Bedienbarkeit und Konformität

GATE liefert konsequent die besten Durchschnittswerte für alle Untersuchungsbereiche, mit Ausnahme des Kriteriums Erlernbarkeit, wo es sich immerhin den zweiten Platz mit Callisto teilen kann, und ist deshalb auch im Gesamtranking verdient auf dem ersten Platz vertreten. Außer im Falle von GATE, welches durchweg sehr hohe Messwerte erzielt, scheint also eine zusammenfassende Rangordnung für alle

getesteten Werkzeuge wenig aussagekräftig, da die anderen Werkzeuge sehr unterschiedliche Stärken und Schwächen bezüglich der verschiedenen Unterkriterien aufweisen. So ist beispielsweise das in der Gesamtwertung letztplatzierte UAM CorpusTool beim Punkt Erlernbarkeit mit 100% eingestuft und somit besser als alle anderen Werkzeuge in diesem Teilbereich.

Die Zusammenfassung aller Attribute in einem einzigen Leistungsniveau ist jedoch insofern aussagekräftig, als dass sie die Verwendung von GATE, als das durchweg am besten geeignetste Werkzeug, nahe legt. Die restliche Rangfolge muss aber unter Berücksichtigung der individuellen Stärken und Schwächen der einzelnen Annotationswerkzeuge ebenfalls sinnvoll interpretiert werden. Tendenziell scheint auch MMAX2 ein interessantes Werkzeug zu sein, weist aber zumindest in einigen Teilbereichen wie etwa Interoperabilität (60% = 4. Platz) und Erlernbarkeit (71,4% = 4. Platz) erhebliche Schwachstellen im Vergleich zu den anderen Tools auf. Callisto ist im Gesamtranking nur um zwei Prozentpunkte besser eingestuft als das UAM CorpusTool, was einerseits daran liegt, dass beide Werkzeuge in den Bereichen Angemessenheit und Bedienbarkeit exakt dieselben Leistungsniveaus erreichen und dabei doch ganz individuelle Stärken und Schwächen bei der Erfüllung verschiedener Unterkriterien zeigen, die sich bei der Aufrechnung zu einem Gesamtdurchschnittswert gegenseitig kompensieren.

Da die Evaluation klar für die Verwendung von GATE spricht, das Werkzeug aber dennoch einige individuelle Schwachstellen hat, sollen in der Abschlussdiskussion Vorschläge zur Kompensation diverser Unzulänglichkeiten der Software gemacht, und Grenzen und Möglichkeiten der praktischen Anwendung des Werkzeugs aufgezeigt werden.

6 Diskussion und Ausblick

Auf Basis einer grundlegenden Klassifikation und Vorsortierung unterschiedlichster Annotationswerkzeuge sowie einer standardisierten Evaluation ausgewählter Tools, kann abschließend eine empirisch belegbare Empfehlung zur Verwendung eines konkreten Annotationswerkzeugs für das DiSynDe Vorhaben abgegeben werden.

6.1 Diskussion der Evaluationsergebnisse

Mit GATE steht ein komplexes Werkzeug zur Verfügung, welches qualitativ durch alle Untersuchungsbereiche hindurch überzeugen kann und ganz besonders Stärken im funktionalen Bereich zeigt. An dieser Stelle sollen aber auch die individuellen Schwachstellen des Tools kritisch diskutiert werden. Diese Diskussion beinhaltet vor allem die Analyse der praktischen Konsequenzen einzelner Schwachstellen für das DiSynDe Anwendungsszenario sowie Lösungsansätze zur Beseitigung und Kompensation bestehender Defizite.

6.1.1 Schwachstellen

Mit Ausnahme eines obligatorischen Attributs konnten für die insgesamt 13 Attribute des Qualitätskriteriums Funktionalität durchweg akzeptable Einstufungsniveaus gemessen werden. In Sachen Benutzbarkeit kann GATE ebenfalls souverän überzeugen. Wenn auch hier insgesamt vier Schwachstellen ermittelt wurden, so konnten doch immerhin 13 von 17 Benutzbarkeitsattributen in akzeptabler Weise erfüllt werden. Insgesamt offenbart GATE nach der Evaluation drei Schwachstellen im Bereich der obligatorischen Attribute und zwei Defizite bei den optionalen Messattributen.

So ist bei GATE das Thema *Hilfesysteme* in keinster Weise berücksichtigt, was aber hinsichtlich der restlichen Messwerte für den Bereich Erlernbarkeit nicht weiter ins Gewicht fällt. Die Nachrüstung eines Hilfesystems kann nur durch Änderungen im Programmcode und die genaue Kenntnis um relevante Hilfethemen und ent-

sprechende Antworten erreicht werden. Der nötige Aufwand überwiegt den tatsächlichen Nutzen dadurch bei weitem. Trotzdem sollte es in der Praxis möglich sein GATE effektiv und schnell zu erlernen, da alle anderen Attribute im Bereich Erlernbarkeit mehr als akzeptabel erfüllt werden. GATE ist das einzige Tool mit einem 380 Seiten mächtigen Benutzerhandbuch und einer ganzen Palette von praktischen Lehrvideos. Eine hochaktuelle Mailinglist mit einem mehrere hundert Einträge umfassenden Archiv und einer entsprechenden Suchfunktion fungiert in gewisser Weise als ausgelagertes Hilfesystem, welches darüber hinaus täglich größer wird und auch frei formulierbare Fragen zu lässt.

Der Missstand fehlender *standardisierter Funktionen* kann leider nicht durch andere Stärken des Annotationswerkzeugs kompensiert werden. GATE unterstützt weder das *Undo/Redo* Paradigma, noch die Möglichkeit den Arbeitsfortschritt automatisch zu speichern. Da es sich bei diesem nicht erfüllten Attribut aber ebenfalls um eine optionale Anforderung handelt, sollten fehlende Standardfunktionen eine effiziente Benutzung des Werkzeugs trotzdem möglich machen, wenn auch mit Einbußen hinsichtlich der Konformität zu anderen Textbearbeitungsprogrammen.

Auch die Validierung der Annotation gegen ein zugrundegelegtes Schema wird von GATE nicht direkt unterstützt. Eine denkbare Umgehung für diese Schwachstelle könnte durch die externe Validierung mit entsprechenden Zusatzprogrammen erfolgen. Die Annotationsschemata in GATE sind als wohlgeformte XML Schema[40] Dateien realisiert. So könnten die Annotationsdateien, zumindest stichprobenartig, über einen beliebigen XML-Validator, wie etwa den kostenlosen und online verfügbaren Service von Validome[41], gegen die Schemata validiert werden, um gegebenenfalls Inkonsistenzen aufzuzeigen und entsprechende Änderungen im Schema vorzunehmen. Da dieses Vorgehen aber grundlegende Kenntnisse der XML-Syntax voraussetzt, müssen die Annotatoren entweder geschult und mit entsprechenden Tutorials versorgt werden, oder der ganze Validierungsvorgang an eine zentrale Stelle mit entsprechenden technischen Kompetenzen ausgelagert werden.

Die *Editierung der Schemata* stellt ein weiteres Defizit dar, welches mit Ausnahme des UAM CorpusTool auch bei allen anderen Werkzeugen vorhanden ist. Die Edi-

[40] Ähnlich wie die DTD bietet auch das *XML Schema* die Möglichkeit die Struktur eines XML-Dokuments zu definieren.

[41] W3C-konformer XML-Validator, verfügbar unter [http://www.validome.org/] – Zugriff am 10.06.2008.

tierung der Schemata während des Annotationsvorgangs ist in GATE zwar möglich, jedoch nur unter Zuhilfenahme externer XML-Editoren. Schemaänderungen erfordern also ähnlich wie die Schemavalidierung, technische Kompetenzen, welche es auszulagern oder durch Schulung herbeizuführen gilt. Die Implementierung eines eigenen graphischen Schemaeditors scheint nicht rentabel, da die Schemata gegen Ende der Pilotphase ohnehin nur noch geringfügiger Änderungen bedürfen und der Aufwand zur Programmierung eines solchen Editors sehr hoch einzuschätzen ist.

Fehlende Mechanismen zur *Anpassung der Steuerung* bezeichnen die letzte markante Schwachstelle der GATE Annotationssoftware. Dabei fällt auf, dass dies das einzige Attribut ist, welches von keinem der Werkzeuge akzeptabel erfüllt wurde. Ein möglicher Erklärungsansatz hierfür könnte im betont funktionalen Design der meisten Annotationswerkzeuge liegen, welches häufig auf Kosten der Benutzbarkeit, insbesondere der Adaptierbarkeit und Bedienbarkeit, geht. Da sich eine unflexible Steuerung mit zunehmender Annotationspraxis unter Umständen sehr negativ auf Effizienz und Motivation auswirken kann, muss über mögliche Kompensationsstrategien schon frühzeitig nachgedacht werden. Die einzige Möglichkeit dieser Schwachstelle aktiv entgegenzuwirken besteht in der Modifizierung der Software auf Codeebene. GATE, welches als Open-Source Projekt offiziell Änderungen in den Kerndateien erlaubt, kommt dem gewillten Java-Programmierer mit einer hervorragenden Programmdokumentation sowie einer komplexen API entgegen. Außerdem können die enormen Stärken von GATE im Automatisierungsbereich die mangelnde Anpassbarkeit der Steuerung bei der manuellen Annotation entkräften, da mit Hilfe komplexer Regeln und regulärer Ausdrücke, welche es im Laufe der Pilotphase zu erstellen gilt, die manuelle Annotation und die damit verbundene Forderung nach einer effizienten Steuerung, immer mehr in den Hintergrund treten.

6.1.2 Stärken

Nach dieser kritischen Analyse sollen aber auch die besonderen Stärken von GATE in der Diskussion berücksichtigt werden. An dieser Stelle sei positiv angemerkt, dass das Werkzeug von den insgesamt 25 akzeptabel erfüllten Attributen, 19 Attribute zu 100% erfüllt. In einigen Bereichen hebt sich GATE darüber hinaus besonders positiv von den anderen Programmen ab.

Die besonderen Möglichkeiten im Bereich der Tokenisierung von Originaltexten und der Automatisierbarkeit des Annotationsprozesses wurden bereits kurz angerissen. Tatsächlich aber liegt hier die eigentliche Stärke von GATE. Ursprünglich konzipiert als komplexe Architektur für die Implementierung beliebiger Anwendungen aus dem Bereich des *Text Engineering*, ist die Verwendung als Annotationswerkzeug nur eines von vielen möglichen Einsatzgebieten der *General Architecture for Text Engineering*. GATE stellt für den Automatisierungsbereich ein elaboriertes Konzept auf Basis von endlichen Automaten und regulären Ausdrücken zur Verfügung, welches die regelbasierte Programmierung von Annotationsautomatismen erlaubt: „[…] JAPE – a Java Annotation Patterns Engine […] provides finite state transduction over annotations based on regular expressions“ (CUNNINGHAM et al. 2007, S. 107). Darüber hinaus wurde bereits eine ganze Reihe von Automatisierungsvorschriften implementiert, welche als Plugins optional genutzt werden können. Die bekannteste Sammlung von Plugins für die Annotation von Textdaten stellt ANNIE (*A Nearly-New Information Extraction System*) dar, welches parametrisierbare *Tokenizer*, *Sentence Splitter*, *POS-Tagger* und vieles mehr enthält:

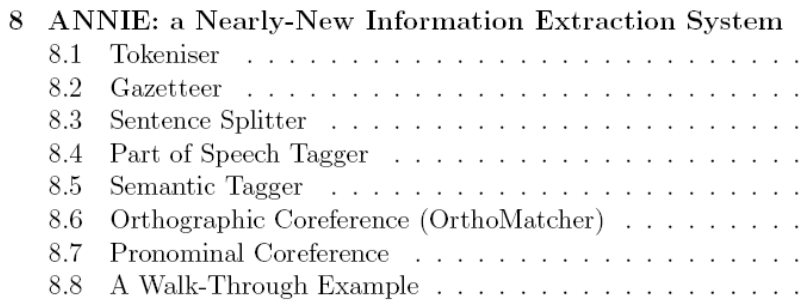

Abbildung 18: Ausschnitt aus dem Inhaltsverzeichnis des GATE Benutzerhandbuchs über den Funktionsumfang von ANNIE, nach CUNNINGHAM et al. (2007, S. iii)

Eine weitere Stärke von GATE besteht darin verschiedenste Dateiformate lesen zu können:

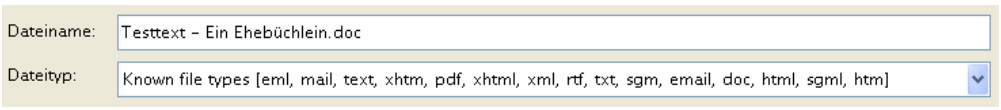

Abbildung 19: Bekannte Dateiformate in GATE

Da alle Texte des Pilotkorpus im MS-Word Format vorliegen, bliebe bei der Verwendung von GATE der Transformationsschritt hin zu einfachen Textdateien aus. Die optional zuschaltbare Funktion *markup-awareness* erlaubt es darüber hinaus Dokumente mit bereits bestehendem Markup hinsichtlich der verwendeten Tags und der Dokumentstruktur zu interpretieren. So können etwa teilannotierte Texte in einem beliebigen Markupformat importiert und mitsamt den Annotationen bearbeitet werden.

Ein Alleinstellungsmerkmal von GATE bildet schließlich der Mechanismus zur Vergabe von Zugriffsrechten, welcher bei keinem der anderen Tools auch nur ansatzweise vorhanden ist. Durch die im Handbuch vorgestellten Konzepte kann ein verteilter Arbeitsablauf grundlegend kontrolliert und dokumentiert werden: „The security model provides primitives such as users, groups, permissions and sessions similar to the ones provided by the operating systems […]" (CUNNINGHAM et al. 2007, S. 278). Der Nachteil besteht darin, dass die Zugriffsrechte momentan nur für das datenbankbasierte Annotationsformat von GATE implementiert sind und nicht auf das exportierbare Stand-off XML Format angewendet werden können. Außerdem ist es nicht möglich gezielte Zugriffe auf einzelne Ebenen zu definieren, sondern nur auf die Annotationsdatei als Ganze.

Die Abschlussdiskussion zu den Evaluationsergebnissen für GATE verdeutlicht, dass das Werkzeug in vielen Bereichen Stärken zeigt, die wenigen defizitären Bereiche hingegen sehr überschaubar sind. Individuelle Schwachstellen sind zwar vorhanden, können aber in allen Fällen entweder kompensiert oder vernachlässigt werden.

6.2 Ausblick

Im Bestreben das weite Feld bestehender Annotationswerkzeuge zu sondieren und ein geeignetes Tool für das DiSynDe Projekt zu ermitteln, konnten darüber hinaus wertvolle Erkenntnisse über offene Fragen und Problembereiche im weiteren Projektkontext gesammelt werden. Vor allem bei der Organisation verteilter Arbeitsabläufe sowie der Normalisierung der Rohtexte, gibt es einige grundlegende Probleme zu lösen.

Überschneidungen und doppelte Annotation dürfen, wenn überhaupt, nur in der Anfangsphase auftreten, und sollten an zentraler Stelle gesammelt und ausgewertet werden. Diese zentrale Stelle muss zum einen alle beteiligten Annotationsebenen überblicken, und zum anderen die späteren Korpusabfragen im Hinterkopf behalten. Die lockeren Grenzen zwischen den einzelnen Annotationsebenen sowie die Mehrfachannotation neuralgischer Phänomene müssen zu Gunsten eines klar strukturierten Arbeitsablaufs baldmöglichst beseitigt werden, da spätestens nach Beendigung der Pilotphase eine solch zentrale Kontroll- und Korrekturinstanz nicht mehr effizient realisierbar ist. Vielmehr müssen im Vorfeld eindeutige Grenzen zwischen den Aufgabengebieten der einzelnen Gruppen gezogen werden, um später deren Annotationen zu einer bestimmten Ebene, über ein durchdachtes Dokument- und Versionsmanagement zu einem einheitlichen Korpus zusammenfassen zu können. Da die verschiedenen Annotationsgruppen sowohl räumlich als auch zeitlich verteilt arbeiten und die Annotation einer Ebene im Idealfall als Ausgangspunkt für die benachbarte Ebene dient, muss darüber hinaus eine Client-Server basierte Infrastruktur zur Sammlung und Verwaltung der Einzelannotationen zur Verfügung gestellt werden.

Weitere offene Fragen ergeben sich hinsichtlich des zu annotierenden Textkorpus. So ist vor allem zu klären wie die Rohtexte in eine normalisierte, einheitliche Form gebracht werden können. Dies beinhaltet die systematische Angabe von bibliographischen Daten zu den einzelnen Texten sowie eine verbindliche Namenskonvention bezüglich der Dateinamen von Rohtexten und Annotationen. Die einheitliche Kodierung etwaiger Sonderzeichen im Unicodeformat sollte schon bei der Digitalisierung der Texte berücksichtigt werden, um spätere Inkonsistenzen bei der Darstellung innerhalb des Annotationswerkzeugs zu vermeiden. Zusätzlich stellt sich die Frage inwieweit graphische Phänomene im Text, also beispielsweise Informationen zu Zeilen, Seiten und Überschriften, bei der Annotation umgesetzt und berücksichtigt werden. Am nahe liegendsten ist die Einführung einer eigenen Annotationsebene für strukturelle, insbesondere graphische Aspekte. Es bleibt allerdings die Frage offen, ob eine solche graphische Annotation in den Zuständigkeitsbereich einer bereits bestehenden Annotationsgruppe fällt, oder ob sie die Neugründung einer zusätzlichen Arbeitsgruppe erforderlich macht.

Außerdem müssen so früh wie möglich Überlegungen zur allgemeinen Korpusarchitektur angestellt werden, die sowohl die technische Realisierung der Speicherung und Organisation der Annotationen als auch deren spätere Erschließung durch entsprechende Abfragemechanismen beinhaltet. Schnittstellen für die Kooperation mit anderen Syntaxprojekten müssen frühzeitig berücksichtigt werden, um eine spätere Zusammenarbeit zu gewährleisten.

Diese Arbeit zeigt, dass das diachrone Syntaxprojekt DiSynDe ein äußerst komplexes Unterfangen darstellt, welches jedoch an vielen Stellen durch geeignete Software unterstützt und vereinfacht werden kann. Ein Annotationswerkzeug für diachrone Korpora dient dabei nur der Arbeitserleichterung während des Annotationsprozesses. Darüber hinaus gilt es vor allem die Architektur, die Verwaltung und die Erschließung des Korpus frühzeitig zu planen und schließlich technisch umzusetzen.

Abbildungen

Tabellen

Literatur

Abran, Alain, Rafa et al. (2005). Harmonization Issues in the Updating of ISO Standards on Software Product Quality. In: *Metrics News. Journal of the Software Metrics Community*. Band 10 (2), S. 35–44.

Azuma, Motoei (2001). SQuaRE: The next Generation of ISO/IEC 9126 and 14598 International Standards Series on Software Product Quality. In: *Proceedings of the 12th European Software Control and Metrics Conference (ESCOM 2001)*. London, S. 337–346.

Bächle, Michael (1996). Anforderungen an das Qualitätsmanagement der Softwareentwicklung. Produkt- und Prozeßnormen. In: Jahnke, Bernd (Hrsg.). *Arbeitsberichte zur Wirtschaftsinformatik* (*Universität Tübingen)*. Band (14), S. 1–22.

Bigbee, Tony et al. (2001). Emerging Requirements for Multi-Modal Annotation and Analysis Tools. In: *Proceedings, Eurospeech 2001 Special Event: Existing and Future Corpora – Acoustic, Linguistic, and Multi-modal Requirements*. Aalborg, S. 1533–1536.

Bird, Steven & Mark Liberman (2000). A Formal Framework for Linguistic Annotation. In: *Speech Communication*, Band 33 (1), S. 23–60.

Bird, Steven & Mark Liberman (2008). *Linguistic Data Consortium. Overview: Linguistic Annotation*. [http:// www.ldc.upenn.edu/annotation/] – Zugriff am 10.06.2008.

Cappelli, Beppe et al. (1998). *MATE Deliverable D3.1: Specification of Coding Workbench*. [http://www. cogsci.ed.ac.uk/~amyi/mate/d3.1-s2.html] – Zugriff am 10.06.2008.

Cunningham, Hamish et al. (1999). *Experience with a Language Engineering Architecture: 3 years of GATE*. [http://www.dcs.shef.ac.uk/%7Ehamish/ GateAisb99.html] – Zugriff am 10.06.2008.

Cunningham, Hamish (2000). *Software Architecture for Language Engineering*. Ph.D. thesis. University of Sheffield.

Cunningham, Hamish et al. (2007). *Developing Language Processing Components with GATE Version 4 (a User Guide). For GATE version 4.0 (July 2007)*. [http://gate.ac.uk/sale/tao/ index.html] – Zugriff am 10.06.2008.

Cunningham, Hamish (2007). *GATE Flyer*. [http://gate.ac.uk/sale/gate-flyer/2007/gate-flyer-4-page.pdf] – Zugriff am 10.06.2008.

Dallapiazza, Michael (Hrsg.) (1983). Ein püchel von der regel der heyligen ee. In: *Zeitschrift für deutsches Altertum*, Band 112, S. 261–292.

DIN 66272 (1994). Bewerten von Softwareprodukten. Qualitätsmerkmale und Leitfaden zu ihrer Verwendung (Identisch mit ISO/IEC 9126: 1991). In: *DIN-Taschenbuch 354 – Software-Ergonomie*. Berlin: Beuth Verlag.

Dipper, Stefanie et al. (2004). Simple Annotation Tools for Complex Annotation Tasks: An Evaluation. In: *Proceedings of the LREC Workshop on XML-based Richly Annotated Corpora*. Lisbon, S. 54–62.

Dipper Stefanie (2005). XML-based Stand-off Representation and Exploitation of Multi-level Linguistic Annotation. In: *Proceedings of Berliner XML Tage 2005 (BXML 2005)*. Berlin, S. 39–50.

Dybkjær, Laila et al. (2001a). *Survey of Existing Tools, Standards and User Needs for Annotation of Natural Interaction and Multimodal Data. ISLE Natural Interactivity and Multimodality Working Group Deliverable D11.1*. [http://isle.nis.sdu.dk/reports/wp11/D11.1-14.2.2001. pdf] – Zugriff am 10.06.2008.

Dybkjaer, Laila et al. (2001b). *Requirements Specification for a Tool in Support of Annotation of Natural Interaction and Multimodal Data. ISLE Natural Interactivity and Multimodality Working Group Deliverable D11.2*. [http://isle.nis.sdu.dk/reports/wp11/D11.2-ISLE-29.7. 2001-F.pdf] – Zugriff am 10.06.2008.

EAGLES (1995). *EAGLES Evaluation Working Group. Evaluation of Natural Language Processing Systems. Final report. EAGLES DOCUMENT EAG-EWG-PR.2. September 1995*. [http:// www.issco.unige.ch/ ewg95/ewg95.html] – Zugriff am 10.06.2008.

EAGLES (1999a). *EAGLES Evaluation Working Group. Evaluation of Natural Language Processing Systems. Pre-final draft version of the Eagles Evaluation Working Group report. EAGLES DOCUMENT EAG-EWG-PR.1. March 1999*. [http://www.issco.unige.ch/projects/eagles/ index.html] – Zugriff am 10.06.2008.

EAGLES (1999b). *EAGLES Evaluation Working Group. The EAGLES 7-step recipe. April 1999.* [http://www. issco. unige.ch/projects/eagles/ewg99/7steps.html] – Zugriff am 10.06.2008.

Englert, Fred et al. (1999). *Phonetische Annotation spontansprachlichen Materials*. [http://titus.fkidg1.uni-frankfurt.de/curric/ gldv99/paper/englert/englertx.pdf] – Zugriff am 10.06.2008.

Evert, Stefan & Arne Fitschen (2001). Textkorpora. In: Carstensen, Kai-Uwe et al. (Hrsg.). *Computer-linguistik und Sprache. Eine Einführung*. Heidelberg/Berlin: Spektrum, S. 369–376.

EXMARaLDA (2008). *Linguistic Annotation Wiki*. [http://annotation.exmaralda.org/index.php/ Linguistic_ Annotation] – Zugriff am 10.06.2008.

Fogli, Daniela et al. (2004). On Electronic Annotation and Its Implementation. In: Costabile, M. F. (Hrsg.) (2004). *Proceedings of the Working Conference on Advanced Visual Interfaces (AVI)*. New York: ACM Press, S. 98–102.

Garg, Saurabh et al. (2004). Evaluation of Transcription and Annotation Tools for a Multi-modal, Multi-party Dialogue Corpus. In: *Proceedings of 4th International Conference on Language Resources and Evaluation (LREC)*. Lisbon, S. 2163–2166.

Hegner, Marcus (2003). *Methoden zur Evaluation von Software*. IZ-Arbeitsbericht Nr. 29. Bonn: Informationszentrum Sozialwissenschaften.

Heilemann, Michal (2008). *Informationsstrukturierung für die syntaktische Annotation eines diachronen Korpus des Deutschen*. Magisterarbeit, Universität Regensburg.

Heinrich, Lutz (2000). Bedeutung von Evaluation und Evaluationsforschung in der Wirtschaftsinformatik. In: Heinrich, Lutz & Irene Häntschel (Hrsg.). *Evaluation und Evaluationsforschung in der Wirtschaftsinformatik*. München/Wien: Oldenbourg, S. 7–22.

Herczeg, Michael (2006). *Interaktionsdesign. Gestaltung interaktiver und multimedialer Systeme*. München: Oldenbourg.

Herczeg, Michael (1994). *Software-Ergonomie: Grundlagen der Mensch-Computer-Kommunikation*. Bonn: Addison-Wesley.

Hovy, Eduard et al. (2003). Principles of context-based machine translation evaluation. In: *Machine Translation*. Band 17 (1), S. 43–75.

Ide, Nancy & Chris Brew (2000). Requirements, Tools, and Architectures for Annotated Corpora. In: *Proceedings of the EAGLES/ISLE Workshop on Data Architectures and Software Support for Large Corpora*. Paris: European Language Resources Association, S. 1–5.

Kelly, John (2003). *Logik im Klartext*. München: Prentice Hall.

Klier, Rainer (2002). *Perl*. München: Addison-Wesley.

Kortmann, Bernd (1999). *Linguistik: Essentials*. Berlin: Cornelsen.

Kroymann, Emil et al. (2004). Eine vergleichende Analyse von historischen und diachronen digitalen Korpora. Technischer Bericht 174 am Institut für Informatik. Berlin: Humboldt-Universität.

Laprun, Christophe et al. (2000). A Practical Introduction to ATLAS. In: *Proceedings of Third International Conference on Language Resources and Evaluation (LREC)*. Las Palmas de Gran Canaria, S. 1928– 1932.

Losavio, Francisca et al. (2003). Quality Characteristics for Software Architecture. In: *Journal of Object Technology*. Band 2 (2), 133–150.

Lüdeling, Anke et al. (2004). DeutschDiachronDigital – Ein diachrones Korpus des Deutschen. In: Braungart, Georg et al. (Hrsg.). *Jahrbuch für Computerphilologie*. Paderborn: Mentis Verlag, S. 119–136.

Mitkov, Ruslan et al. (2000). Coreference and anaphora: developing annotating tools, annotated resources and annotation strategies. In: *Proceedings of the Discourse Anaphora and Anaphora Resolution Colloquium (DAARC)*. Lancaster, S. 49–58.

Müller, Christoph (2004). *Why MMAX2 is not free*. [http://www.eml-research.de/downloads/nlp/whymmax2isnotfree.pdf] – Zugriff am 10.06.2008.

Müller, Christoph & Michael Strube (2001). MMAX: A Tool for the Annotation of Multi-modal Corpora. In: *Proceedings of 2nd IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems.* Seattle, S. 45–50.

Müller, Christoph & Michael Strube (2003). Multi-Level Annotation in MMAX. In: *Proceedings of the 4th SIGdial Workshop on Discourse and Dialogue*. Sapporo: Association for Computational Linguistics, S. 198–207.

Müller, Christoph & Michael Strube (2006). Multi-level annotation of linguistic data with MMAX2. In: Braun, Sabine et al. (Hrsg.). *Corpus Technology and Language Pedagogy: New Resources, New Tools, New Methods*. Frankfurt a. M.

Mummendey, Hans Dieter & Ina Grau (2008). *Die Fragebogen-Methode. 5., überarbeitete und erweiterte Auflage*. Göttingen: Hogrefe Verlag.

Netzwerk Historische Syntax (2008). *Internetpräsenz des Projekts Netzwerk Historische Syntax – Startseite*. [http://cgi.server.uni-frankfurt.de/fb10/ histsprawi/ netzwerk-hisyn/] – Zugriff am 10.06.2008.

Noack, Wilhelm (1998). *XML 1.1 – Grundlagen*. Bodenheim: Herdt-Verlag.

Pree, Wolfgang (1997). *Komponentenbasierte Softwareentwicklung mit Frameworks*. Heidelberg: dpunkt Verlag.

Reidsma, Dennis et al. (2004). *Designing Annotation Tools based on the Properties of Annotation Problems*. Report for the Centre for Telematics and Information Technology. University of Twente: Dept. of Computer Science, HMI Group.

Rodríguez, Kepa Joseba et al. (2007). Standoff Coordination for Multi-Tool Annotation in a Dialogue Corpus. In: *Proceedings of the Linguistic Annotation Workshop*. Prag: Association for Computational Linguistic, S. 148–155.

Rydeman, Bitte (2003). Using Multimodal Annotation Tools in the Study of Multimodal Communication Involving Non-speaking Persons. Course paper in Multimodal Communication, Department of Linguistics, Göteborg University.

Scharf, Andreas & Bend Schubert (2001): *Marketing. Einführung in Theorie und Praxis. 3. überarbeitete und erweiterte Auflage*. Stuttgart: Schäffer-Poeschel.

Schmid, Hans Ulrich (2007). Das Projekt einer Historischen Syntax des Deutschen. In: Bochmann, Klaus (Hrsg.). *Theorie(n) und Methoden der Sprachgeschichte. Materialien des Kolloquiums zu Ehren des 70. Geburtstages von Gotthard Lerchner*. Stuttgart/Leipzig: Hirzel, S. 51–57.

Schneider, Uwe & Dieter Werner (Hrsg.) (2001): *Taschenbuch der Informatik: 4., aktualisierte Auflage*. Leipzig: Carl Hanser Verlag.

Sim, Tze Jan et al. (2005). A Survey of Transcription, Annotation and Query Tools for Development of a Classroom Discourse Corpus. In: *Proceedings of the Conference of Redisigning Pedagogy: Research, Policy, Practice*. Singapore: Nanyang Technological University.

Sparck-Jones, Karen & Julia R. Galliers (1996). *Evaluating Natural Language Processing Systems: An Analysis and Review.* Berlin: Springer.

Strömsdörfer, Christian & Theo Vennemann (1995). Das Verhältnis des Syntaxwandels zur Theorie der Sprachzustände. In: Jacobs, Joachim et al. (Hrsg.). *Syntax. Ein internationales Handbuch zeitgenössischer Forschung*. Berlin/New York: deGruyter, S.1126–1135.

Suryn, Witold & Alain Abrain (2003). ISO/IEC SQuaRE. The second generation of standards for software product quality. In: *7th IASTED International Conference on Software Engineering and Applications*. California.

Syring, Wolf-Dieter (2004). Textannotation als wissenschaftliche Aufgabe. In: Poschenrieder, Thorwald (Hrsg.). *Die Indogermanistik und ihre Anrainer – Dritte Tagung der Vergleichenden Sprachwissenschaftler der Neuen Länder*. Innsbruck.

Tablan, Valentin (2004). *GATE – An Application Developer's Guide*. [http://gate.ac.uk/sale/pg/pg.pdf] – Zugriff am 10.06.2008.

Teich, Elke et al. (2003). Representing and querying multi-layer annotated corpora. In: Bird, Steven et al. (Hrsg.). *Proceedings of the IRCS Workshop on Linguistic Databases*. University of Pennsylvania, S. 228–237.

Thompson, Henry & David McKelvie (1997). *Hyperlink semantics for standoff markup of read-only documents*. [http://www.ltg.ed.ac.uk/~ht/sgmleu97.html] – Zugriff am 10.06.2008.

Tidwell, Jennifer (2006). *Designing Interfaces. Patterns for Effective Interaction Design*. Sebastopol: O'Reilly.

Underwood, Nancy & Agnes Lisowska (2006). The Evolution of an Evaluation Framework for a Text Mining System. In: P*roceedings of 5th International Conference on Language Resources and Evaluation*. Genoa, S. 2479–2484.

Validome:XML (2008). *Validome – Validator für XML-Dokumente*. [http://www.validome.org/xml/validate] – Zugriff am 10.06.2008.

Wikipedia:Aussagenlogik (2008). *Aussagenlogik – Wikipedia, Die freie Enzyklopädie*. [http://de.wikipedia. org/wiki/Aussagenlogik] – Zugriff am 10.06.2008.

Wittenburg, Peter et al. (2006). *Language Resource Archiving supporting Multimodality Research*. [http:// www.lat-mpi.eu/papers/papers-2006/MM-Resource-Archiving-v3.pdf] – Zugriff am 10.06.2008.

Eidesstattliche Erklärung

Ich versichere an Eides statt durch meine Unterschrift, dass ich die vorstehende Arbeit selbstständig und ohne fremde Hilfe angefertigt und alle Stellen, die ich wörtlich oder annähernd wörtlich aus Veröffentlichungen entnommen habe, als solche kenntlich gemacht habe, mich auch keiner anderen als der angegebenen Literatur oder sonstiger Hilfsmittel bedient habe. Die Arbeit hat in dieser oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen.

Regensburg, 01. Juli 2008

..……