Jan Christoph Meister

JENSEITS VON LAGADO
LITERATURWISSENSCHAFTLICHES PROGRAMMIEREN
AM BEISPIEL DES KODIERUNGSPROGRAMMS Move Parser 3.1

LAGADONISCHE VORBEMERKUNG

Jeder Mythos hat seinen Ursprungsort: Dem PC, virtuelles Medium par excellence und derzeit technologischer Dreh- und Angelpunkt des Informationszeitalters, weist die Legende eine Garage im kalifornischen Silicon Valley zu – eine Geburtsstätte, die Mobilität und subversive bricolage zugleich symbolisiert. Tellerwäscher und ingenieuse Garagentüftler, so wissen wir, kurbeln in der Neuen Welt den Motor des Fortschritts an. Geistes- und zumal Literaturwissenschaftler, so will es manchem scheinen, vegetieren derweil in der alten.

Das Bild von der Philologie als rührselig-traditionsbewußter, (selbst)-vergessener Nachhut der Moderne bedarf einer Revision. Sicher, lange genug waren die litera- den numerophilen Wissenschaften beim Buhlen um CPU-Time auf Großrechnern unterlegen – mit dem PC aber bricht endlich auch für Philologen[1] die Ära einer digitalen, quantitativ vermessenen Wirklichkeit ihrer Gegenstände an. Anlaß genug, erhobenen Hauptes daran zu erinnern, daß das Projekt eines »Humanities Computing«, wie es zur Zeit hauptsächlich in der anglo-amerikanischen Welt diskutiert wird, in Europa schon über 250 Jahre lang vorgedacht war. Literarisch, versteht sich:

> The first professor I saw was in a very large room, with forty pupils about him. After salutation, observing me to look earnestly upon a frame which took up the greatest part of both the length and breadth of the room, he said perhaps I might wonder to see him employed in a project for improving speculative knowledge by practical mechanical operations. [...] Every one knew how laborious the usual method is of attaining to arts and sciences; whereas, by his contrivance, the most ignorant person, at a reasonable charge, and with a little bodily labour, might write books in philosophy, poetry, politics, law, mathematics, and theology, without the least assistance from genius or study. He then led me to the frame, about the sides whereof all his pupils stood in ranks. It was twenty feet square [...]. The superficies was composed of several bits of wood, about the bigness of a die [...] They were all linked together by slender wires. These bits of wood were covered, on every square, with paper pasted on them; and on these papers were written all the words of their language, in their several moods, tenses, and declensions, but without any order. The professor then desired me to observe; for he was going to set his engine at work. The pupils, at his command, took each of them hold of an iron handle, whereof there were forty fixed round the edges of the frame, and giving them a sudden turn, the whole disposition of words was entirely changed. He then commanded six-and-thirty of the lads to read the several lines softly, as they appeared on the frame; and where they found three or four words together that might make part of a sentence, they dictated to the four remaining boys [...]
> Six hours a day the young students were employed in this labour; and the professor shewed me several volumes in large folio, already collected, of broken sentences, which he intended to piece together, and, out of those rich materials, to give the world a complete body of all arts and sciences; which, however, might be still improved, and much expedited, if the public would raise a fund for making and employing five hundred such frames in Lagado [...][2]

Schon das 18. Jahrhundert also kennt die Vision einer Textwissenschaft, die Sinn produziert unter der Ägide von Großforschern, deren Programm sich mechanisch-kombinatorisch abarbeiten läßt. Computerphilologie, das wäre also nun endlich die Ausbeutung dieses antragswürdigen Forschungsprojekts, wenn, ja wenn nur »the public would raise a fund for making and employing five hundred such frames«, wie Jonathan Swift berichtet.

Mit dem MoveParser wird im Folgenden ein Programm vorgestellt, das weder solcher Hoffnung auf enzyklopädische Sinnproduktion noch der auf eine Automatisierung des Wissenserwerbs Vorschub leisten will. Vielmehr stellt der MoveParser den Versuch dar, die Balance zu halten zwischen der Notwendigkeit zur rigiden Formalisierung von Beschreibungsdaten, welche im Rahmen einer computergestützten empirischen Text- und Rezeptionsanalyse ausgewertet werden sollen, und der notwendigen Einbeziehung interpretativer Akte, ohne die die Gewinnung solcher Beschreibungsdaten im Umgang mit Texten undenkbar wäre. Der Leser ist zur praktischen Erprobung und Kritik dieses literaturwissenschaftlichen Werkzeugs einladen; die entsprechenden MoveParser-Dateien sind zu diesem Zwecke über die Website der Zeitschrift für Computerphilologie abrufbar.[3] Für Kommentare und Kritik, insbesondere aber auch Anregungen in Hinblick auf die mögliche Übertragung des MoveParser auf andere literaturwissenschaftliche Anwendungsbereiche wäre ich sehr dankbar![4]

1. DAS »EPITEST«-PROJEKT: ERZäHLTHEORETISCHER KONTEXT

MoveParser wurde im Rahmen meines aktuellen Forschungsprojektes »Epitest« als ein Hilfsprogramm zur Vorbereitung der computergestützten Analyse von Handlungsstrukturen entwickelt. Das erzähl- beziehungsweise handlungstheoretisch orientierte »Epitest«-Projekt selbst ist schon an anderer Stelle ausführlich dargestellt worden,[5] soll hier jedoch der Verständlichkeit halber kurz skizziert werden.

Ziel des »Epitest«(= »Episoden-Test«)-Projektes, das im Rahmen einer umfassenderen Studie zur Theorie der literarischen Handlung steht, ist der empirische Nachweis von Verknüpfungsvorschriften und -regeln, anhand derer natürliche Leser bei der Lektüre eines narrativen Textes die fiktionalen Geschehenselemente zu komplexen Handlungssequenzen integrieren. Dies soll im Zuge einer computergestützten Analyse von empirischen Rezeptionsdaten untersucht werden. Bei der Konzeption des MoveParser, der im Wesentlichen ein Hilfsprogramm zur Erstellung formaler Beschreibungsprotokolle fiktionaler Handlungen ist, standen zwei narratologische Modelle im Hintergrund, die im Kontext meiner Überlegungen zur Theorie literarischer Handlung zentrale Bedeutung haben. Zunächst handelt es sich dabei um das sogenannte »Move«-Modell fiktionaler Handlungen, das Thomas Pavel 1985 in der Studie Poetics of Plot vorgestellt hat.[6] Jeder einzelne Handlungsschritt in einem epischen oder dramatischen Text wird von Pavel als dreigliedriges Element aufgefaßt, das heißt als »Move«, der im Sinne von »Spielzug« einen vorgefundenen Zustand in einen zweiten überführt:

Move

/ \

Problem Solution

Die dreigliedrigen Elementar-Strukturen der Moves verbinden sich nun im Zuge der sequentiellen Lektüre zur komplexen Gesamtstruktur des »Spiels«, das heißt zu einer Handlung. Die Rezeption von komplexen Handlungsgefügen durch einen Leser ist damit an die Bedingung geknüpft, daß eine hinreichende Zahl von Zustands- und Ereignisbeschreibungen des Textes entweder als Anlaß (= Problem) oder als Konsequenz (= Solution) einer Zustandstransformation (= Move) interpretiert und daß außerdem diese Transformationen selbst wiederum kausal miteinander verknüpft werden können.

Mit dem Move-Modell ist die Oberflächendimension des Handlungsgeschehens hinreichend beschreibbar. Unabhängig von dem an der fiktionalen Oberfläche erzählerisch angedeuteten oder gar explizierten kausalen und psychologischen Wirkungszusammenhang hat jede Handlung jedoch auch so etwas wie ein semantisches Unterfutter: In der Abfolge der Zustandstransformationen kann man eine systematische Gegenüberstellung und ›Abarbeitung‹ allgemeiner semantischer Terme erkennen, unter denen Erzähleinheiten (in der Regel von der Satzebene an aufwärts) subsumiert werden können. Diese sogenannte »Tiefenstruktur« von Handlungen abzubilden, ist eines der Hauptanliegen von A. J. Greimas' semiotisch orientierter Erzählgrammatik, in der die einzelnen Handlungsereignisse der Oberflächenstruktur (hier also: der Moves) auf zugrundeliegende Konstellationen semantischer Terme zurückgeführt werden.[7] Welche Terme allerdings jeweils konkret einer gegebenen Einheit zugeordnet werden, läßt sich nie mit Gewißheit voraussagen. Die theoretisch unendlich große Menge abstrakter Begriffspaare wie »Liebe/Haß«, »groß/klein«, »gut/böse« et cetera, als deren Abfolge das Transformationsgeschehen der einzelnen Moves sich fassen läßt, ist zwar durch gattungs-, themen- oder textsortenspezifische Bedingungen eingegrenzt; dennoch bleiben in der Regel noch genügend Benennungsmöglichkeiten übrig, um eine breite Streuung von idiosynkratischen Varianten zu erwarten. Bittet man indes verschiedene Leser, die mit der Move-Darstellungskonvention vertraut sind, die von ihnen auf der Grundlage einer identischen Textbasis[8] rezipierten Handlungen als Verkettung solcher semantischen Terme darzustellen, so treten insbesondere in Hinblick auf die jeweils realisierten Verknüpfungsmuster streckenweise frappierende Übereinstimmungen hervor. Während die vorgenommenen Abgrenzungen der Erzähleinheiten in einer relativ homogenen Lesergruppe dabei meist weitgehend identisch sind, unterscheiden sich umgekehrt die intuitiv zugeordneten Beschreibungsbegriffe von Leser zu Leser. Unterschiedlich bewertete Elemente fügen sich also zu teilweise identischen Strukturen. Das legt die Vermutung nahe, daß es durchaus so etwas wie ›Universalien‹ der erzählten Handlung gibt, und zwar im Sinne von rein formalen, inhaltsunabhängigen Regeln, wohingegen der semantische Gehalt eines Handlungsmusters darüber entscheidet, ob diese ›Handlung‹ als Interpretationstatsache den je spezifischen Produktions- beziehungsweise Rezeptionskonventionen entspricht, die als Wertsetzungen außerhalb des Textes verankert sind.

2. MOVEPARSER 3.1

In einem im Wintersemester 1996/97 an der Universität Hamburg veranstalteten Seminar habe ich nun erstmals versucht, mit einer Gruppe von Studentinnen und Studenten eine größere Menge empirischer Rezeptionsdaten zu generieren, wobei zugleich verschiedene Versionen des MoveParser-Programms entwickelt und getestet wurden. Unsere gemeinsame Textbasis waren dabei Auszüge aus Goethes Unterhaltungen deutscher Ausgewanderten, die sich u.a. aufgrund der Gliederung in Novellen und eine stark handlungszentrierte Erzählweise für Handlungsanalysen anbieten.

Der MoveParser unterstützt als Software den Leser bei der Identifizierung und semantischen Kodifizierung der Einzelereignisse, aus denen sich sukzessive die komplexe Gesamthandlung eines fiktionalen epischen Textes aufbaut. Im Zuge einer strikt sequentiellen Lektüre kann man so Zug um Zug beziehungsweise ›Move um Move‹ Textpartien als »Problem« oder »Solution« markieren und diesen dann jeweils beschreibende Begriffe zuordnen, die als semantische Terme im Sinne der zugrundeliegenden Erzählgrammatik gelten. Das »Parsing« findet also bislang nicht etwa automatisch – unter Zugrundlegung bekannter Regeln – statt, sondern im Zuge einer von den Lesern geleisteten Interpretation. Mit der vom Programm gespeicherten sogenannte »Move Base«-Datei steht schließlich eine Datenbank zur Verfügung, in der die interpretativ gewonnenen empirischen Beschreibungsdaten zu einer vorgegeben Textgrundlage (= »Text Base«-Datei) gesammelt vorliegen. Dieser Datenbestand kann dann auf rekurrente Muster und Terme analysiert werden.

In der vorliegenden Version 3.1 ist der MoveParser ein in Microsoft Visual Basic geschriebenes Programm, das unter Windows 3.1 beziehungsweise Windows 95 läuft. Sowohl die im Programm bearbeitete elektronische Textgrundlage, also die Text Base-Datei, als auch die vom Leser sukzessive erstellte Move Base-Datei sind reine DOS-Texte, können also auch mit anderen gängigen Anwendungen (wie zum Beispiel Textverarbeitungsprogrammen) manipuliert werden. Das gleiche gilt für eine dritte Datei, die ebenfalls bei Verwendung des Programms dynamisch anwächst, die sogenannte »Labels«-Datei. In dieser werden die nach acht festgelegten Kategorien geordneten semantischen Terme festgehalten, die die Leser im Zuge der Beschreibung von Moves verwenden. In jeder dieser Kategorien steht den Benutzern des Programms bereits eine Vorauswahl von Beschreibungsbegriffen zur Verfügung; diese Auswahl kann jedoch jederzeit nach Bedarf ergänzt werden. In Hinblick auf die Beschreibungsbegriffe ist nur darauf zu achten, daß jeder Begriff irgendwo in dem Gesamtinventar der Terme einen Gegenbegriff haben muß; dies ist ein Erfordernis der zugrundeliegenden semantischen Theorie. Bei Einfügung eines neuen Begriffs fordert das Programm den Benutzer deshalb automatisch auf, einen entsprechenden Gegenbegriff zu definieren und speichert diesen sodann zusammen mit dem ersten in seiner Labels-Datei.

Typischerweise absolviert der Benutzer bei der Festlegung und Kodifizierung eines einzelnen Moves vier Schritte:

1. In dem im Textfenster aufgerufenen Text wird ein Abschnitt markiert, der als »Problem« im Sinne des Move-Modells angesehen wird, das heißt als Eröffnung einer Möglichkeit zur Zustandsveränderung in der fiktionalen Welt. Die nach gängiger Windows-Konvention per Maus vorgenommene Markierung wird dabei automatisch als Seiten/Zeilen-Index erfaßt.

2. Ist der erste Schritt vollzogen, wird automatisch ein sogenanntes »Pull Down-Menü« aktiviert. Unter der Menüüberschrift »MoveParser« wird in der Rubrik »Focus« eine Erzähl- oder Figurenperspektive ausgewählt, unter der der markierte Textausschnitt als »problematisch« (im Sinne von: eine Handlungsmöglichkeit oder -notwendigkeit eröffnend) erscheint.

3. Nach der Zuordnung des »Focus« wird vom Leser im selben Menü festgelegt, unter welcher der vorgegebenen acht Kategorien der zuzuordnende beschreibende Term steht und welcher Term dann innerhalb der gewählten Kategorie das »Problem« (im Sinne einer Situation, die eine Handlungsmöglichkeit beziehungsweise -notwendigkeit impliziert) des markierten Textabschnitts abstrakt

beschreibt. Findet sich in der aktivierten Vorauswahl kein passender Begriff, so kann ein neuer eingegeben werden; in diesem Fall fordert das Programm dann auch zur Definition eines Gegenbegriffes auf.

4. Schritte 1 und 3 werden jetzt zur Festlegung beziehungsweise Beschreibung der dem »Problem« zugeordneten »Solution« wiederholt. Unter »Solution« wird die faktisch im Handlungsgeschehen vollzogene Zustandsveränderung verstanden, die auf die Handlungsmöglichkeit des »Problem« antwortet.

Sobald die vier Bearbeitungsschritte durchlaufen worden sind, zeigt das Programm in einem zweiten Fenster den kodifizierten Move als Listeneintrag an. Jeder Listeneintrag kann nachträglich aus der Liste aufgerufen werden; außerdem können die den »Problem«- beziehungsweise »Solution«-Instanzen zugeordneten Textabschnitte eingesehen werden.

Selbstverständlich ließe sich all dies auch ohne Computerprogramm bewerkstelligen; tatsächlich ging der Entwicklung des MoveParser ein entsprechender Versuch in einem Oberseminar voraus, bei dem die beteiligten Studenten und Studentinnen Moves und Move-Strukturen manuell kodifizierten. Worin also liegt der Vorteil bei der Verwendung eines derartigen Programms? Ich möchte hier drei Aspekte anführen, die mir im Rahmen einer empirisch orientierten erzähltheoretischen beziehungsweise handlungslogischen Analyse relevant erscheinen.

1. Formale Konsistenz der Beschreibungssprache: Neben der Arbeitsersparnis und Anschaulichkeit, die mit dem Computerprogramm für den Leser gegenüber rein manuellen Kodifizierungen erreicht wird, zeichnet sich das Ergebnis des hier erstellten Move-Inventars zu einem Text zunächst dadurch aus, daß es ein standardisiertes Format und damit die Berücksichtigung der spezifischen syntaktischen Konventionen garantiert, die für die spätere computergestützte Auswertung eines einzelnen Protokolls wesentlich sind. Diese Auswertung in Hinblick auf

a) Kombinationsmöglichkeiten von einzelnen Moves zu Move-Ketten;

b) den Vergleich dieser automatisch generierten Strukturen mit der faktisch vom Leser realisierten komplexen Move-Struktur

wird von einem zweiten Programm übernommen, das aufgrund der spezifischen Anforderungen einer solchen komplexen Analyse in einer anderen Programmiersprache (Prolog) geschrieben worden ist. Auf dieses – programmiertechnisch wesentlich raffiniertere, aber bislang nur als Pilotversion erstellte – Programm »Epitest« kann hier nicht näher eingegangen werden.[9]

2. Induktives Beschreibungsverfahren: Das Computerprogramm erzwingt weitgehend eine geordnete Abfolge der Bearbeitungsschritte und eine sequentielle, induktiv organisierte Erhebung der Beschreibungsdaten. Zwar kann der Benutzer des MoveParser wie bei der natürlichen Textlektüre jederzeit seine Hypothesen über das Handlungskonstrukt der Gesamthandlung revidieren, indem bereits erfaßte Moves neu definiert und so mit der auf der Grundlage einer mittlerweile komplexeren Datenbasis generierten Hypothese harmonisiert werden. In aller Regel führt dies jedoch nicht zur Löschung des ersten Eintrags aus der Datenbasis, sondern zur Eingabe einer neuen, zusätzlichen Beschreibungsvariante – die alte ›Bedeutung‹ geht also nicht verloren, sondern bleibt erhalten und steht so weiterhin für die anschließenden Analysen zur Verfügung. Das Programm ermutigt damit zur Erfassung von interpretativ gewonnenen hypothetischen Bedeutungsvarianten, wie sie bei einer strikt sequentiellen Lektüre generiert werden. Nicht die ›fertige‹, nach der

Gesamtlektüre entworfene Hypothese über Ordnung und Bedeutung der Handlung als solche, sondern die noch für verschiedene Entwicklungsmöglichkeiten ›offene‹ Perspektive auf einzelne Moves als individuelle Handlungsschritte steht im Vordergrund.

3. Vergleichbarkeit der individuellen Move-Protokolle: Die von den einzelnen Lesern mit dem MoveParser erarbeiten Move-Protokolle und -Listen, aber auch die Listen der semantischen Beschreibungsterme können schließlich, da methodisch und formal einheitlich konstituiert, auch in ihrer Gesamtheit rechnergestützt analysiert werden. Damit eröffnet sich erstens die Möglichkeit einer statistischen Auswertung der Protokolle insgesamt, etwa in Hinblick auf rekurrente Move-Muster, oder der Analyse der Label-Datei mit ihren Termlisten in Hinblick auf das gehäufte Vorkommen eines Terms innerhalb spezifischer binärer Begriffskonstellationen. Zweitens garantiert das Verfahren die Vergleichbarkeit mit den in einer Kontrollgruppe erstellten Move-Protokollen, wobei ein solcher Vergleich wiederum computergestützt erfolgen kann.

3. KONTEXTBEDINGUNGEN PHILOLOGISCHEN PROGRAMMIERENS

Die Entwicklung eines auf eine spezielle Forschungsproblematik zugeschnittenen Computerprogramms ist nach meiner Erfahrung mit den Programmen MoveParser und »Epitest« für einen Literaturwissenschaftler ebenso frustrierend wie bereichernd. Zu den frustrierenden Erfahrungen zählt, daß auf fachliche Unterstützung durch kompetente Programmierer bei literaturtheoretisch orientierten Forschungsvorhaben kaum gerechnet werden darf. Informatiker, zumal wenn sie primär in Diplomstudiengängen lehren oder lernen, schätzen den Nutzen vergleichsweise esoterischer Projekte, wie sie die Modellierung literarischer Prozesse in der Regel darstellen, eher gering ein. Auch die linguistische Forschungspraxis, in der Computeranwendungen weitaus verbreiteter sind als in den Literaturwissenschaften, eröffnet nur bedingt Perspektiven auf die Gegenstände »Text«, »Literatur«, »Rezeption« et cetera, die für literaturwissenschaftliche Fragestellungen nutzbar zu machen sind. Das liegt nicht zuletzt daran, daß solche literaturwissenschaftliche Fragestellungen sich selten direkt auf ein gegebenes primäres Datum »Text« oder »Sprache« richten, sondern zumeist auf Interpretations- oder Beschreibungssachverhalte, die erst in Abhängigkeit von einem Primärtext erarbeitet werden müssen. Jede computergestützte literaturwissenschaftliche Analyse, die mehr als eine reine Auflistung und statistische Auswertung bloßer Zeichenfolgen ist, setzt daher ein Markup voraus – und sei es auch nur gemäß der rein deskriptiven SGML-Konvention. In die Markup-Konventionen fließen aber bereits in erheblichem Umfang theoretische Prämissen und methodologische Vorentscheidungen ein, die das Verfahren, die Zielrichtung und die Reichweite einer späteren Analyse der erhobenen Daten determinieren. Diesen Zusammenhang kann bei der Konzeption spezialisierter Software nur berücksichtigen, wer über entsprechende literaturwissenschaftliche Kenntnisse verfügt.

Bei der Entwicklung kommerzieller Software geht dem Programmentwurf voraus eine detaillierte Systemanalyse aller Verfahrens- und Entscheidungsprozesse, die den fraglichen Anwendungsbereich charakterisieren. Auch unter den gängigen textanalytischen Softwarepaketen finden sich entsprechende Beispiele einer solchen systematischen Kooperation von Informatikern und Philologen; so etwa bei TACT, dem an der Universität Toronto entwickelten Paket von »Textual Analysis Computing Tools«, das unter Leitung von Ian Lancashire von Mitgliedern des (im Rahmen eines mehrjährigen Kooperationsprojekts von IBM Canada finanziell

unterstützten) »Centre for Computing in the Humanities« erarbeitet wurde. Eine diesem Zentrum vergleichbare Einrichtung sucht man derzeit an den meisten deutschen Universitäten vergeblich; zu den rühmlichen Ausnahmen zählt die Eberhard-Karls-Universität Tübingen, an deren »Zentrum für Datenverarbeitung« Wilhelm Ott denn auch mit TUSTEP das hierzulande bekannteste wissenschaftliche Textverarbeitungsprogramm geschaffen hat.

Solange an der Mehrzahl unserer Universitäten institutionalisierte interdisziplinäre Forschungs- und Serviceeinrichtungen fehlen, die sich der Entwicklung und Betreuung geisteswissenschaftlicher Computeranwendungen annehmen, solange werden die meisten von uns, was die Praxis der Programmierung von philologischer Software angeht, weiterhin als Dilettanten agieren müssen. Hilfreich wäre bei diesem desolaten Stand der Dinge immerhin der Austausch nicht nur über die ›fertigen‹ Programme, sondern auch über die Erfahrungen, die wir jeweils aktuell bei der Entwicklung wie der Erprobung und Anwendung dieser Programme machen. MoveParser etwa ist aufgrund der teilweise sehr spezifischen Kritik meiner Studenten an einzelnen Programmkomponenten im Laufe eines Semesters durch immerhin vier Versionen gegangen – ein Fortschritt, der ohne die Durchführung dieses (mitunter desillusionierenden) Praxistests so nicht möglich gewesen wäre.

Ein derartiger fortlaufender Erfahrungsaustausch über »work in progress« (etwa über eine spezielle Diskussionsliste zum Thema »Philologisches Programmieren«) könnte schließlich in die Organisation von Workshops münden, in denen die Relevanz und Zugänglichkeit verschiedener Programmiersprachen und -konzepte für den Philologen praktisch erprobt werden kann. Und wer weiß: Vielleicht fände sich für solche Veranstaltungen unter den Entwicklern von Programmiersprachen auch ein interessierter Sponsor.

LAGADONISCHE SCHLUSSBEMERKUNG

Welche Sprache der Computer spricht, das ist also durchaus keine Frage, die in der Computerphilologie nur studentische Hilfskräfte interessieren sollte. Mit der Ignoranz gegenüber den Funktionsprinzipien des Mediums würde vielleicht die größte Chance vertan, die der Computer der Philologie bietet. Ich will damit nicht dafür plädieren, aus Philologen dilletierende Informatiker zu machen: Der Erkenntnisgewinn, den die Einsicht in die Funktionsweise der Technik als solcher für uns birgt, ist in der Tat begrenzt. Von eminentem Interesse dagegen wäre eine kritische Reflektion der Bedingungen und Prozesse, die den Wechsel von der traditionell ›analogen‹ Perspektive auf das symbolische System »Text« beziehungsweise »Literatur« zu einer ›digitalen‹ kennzeichnen. Der Reduktionismus und die methodische Rigidität, welche die Digitalisierung literaturwissenschaftlicher Verfahren notwendig begleiten, schaffen neue experimentelle Bedingungen, unter denen wir erstmals kontrolliert menschliche Verstehensakte in Bezug auf Texte in einem anderen als dem symbolischen Medium einer natürlichen Sprache modellieren können. Hierin liegt die eigentliche Chance des Mediums: in einem methodologischen Verfremdungseffekt, wie ihn die Textwissenschaften des 20. Jahrhunderts ansatzweise bereits im Formalismus und Strukturalismus erfahren haben. Die Unterwerfung literarischer Texte und Verstehensprozesse unter das digitale Prinzip setzt vorübergehend die hermeneutische Orientierung der Philologie außer Kraft und zwingt zur Kärnerarbeit der präzisen Übersetzung und Modellierung jener unerhörten Bedeutungsvielfalt, die ästhetischen Produkten eigen ist. Dieses Vorhaben gelingt allenfalls näherungsweise – und eröffnet eben darum im Vergleich zu den bewährten hermeneutischen und sozialwissenschaftlichen Verfahren

eine neue differentielle Perspektive auf den Gegenstand Literatur. An der Computerphilologie als rechnergestützter Modellierung literarischer Produktions- und Rezeptionsprozesse ist mithin zweierlei interessant: der voranschreitende Präzisionsgrad derartiger Modellkonstruktionen, mehr noch aber der Erkenntnisgewinn, den wir aus den vielfältigen Instanzen fruchtbaren Scheiterns ziehen können, die den Weg dahin markieren. Auch in dieser Hinsicht scheint mir Swifts Satire für unseren Zusammenhang relevant, weil sie an dem absurden Beispiel einer mechanischen Sinnproduktion aufzeigt, wie ungeachtet des voreiligen Objektivitätsanspruchs einer computer- beziehungsweise algorithmenbasierten Literaturwissenschaft mit Notwendigkeit an entscheidender Stelle wieder hermeneutische Akte in den Verfahrensprozeß einfließen müssen. Computerphilologische Forschung zu betreiben heißt also, einen Parallel- und Seitenweg zu den hermeneutischen Verfahren zu beschreiten, und nicht, neuerlich der Schimäre objektivierbaren Sinns nachzujagen. Gerade deshalb muß, wer den Computer auf intelligente Weise als philologisches Forschungsinstrument einsetzen will, früher oder später seine Sprache, und das heißt programmieren lernen. Sonst, so fürchte ich, wird uns die lagadonische Schulstube zum computerphilologischen Dauerquartier: Die einen drehen die Hebel, die anderen – stiften den Sinn.

---

[1] Klassenbegriffe, Berufsbezeichnungen, diesen zugeordnete Pronomina et cetera im Masculinum bezeichnen hier wie im Folgenden, wo immer sinngemäß zulässig, beide Geschlechter.

[2] Jonathan Swift: Travels into Several Remote Nations of the World. By Lemuel Gulliver [...], Köln 1995, S. 190 ff.

[3] Systemanforderungen: Windows 3.1 oder 95; Maus; VGA oder SVGA; ca. 900 KB freier Speicher auf der Festplatte.

[4] Mein besonderer Dank gilt den Studentinnen und Studenten des im Wintersemester 1996/97 an der Universität Hamburg veranstalteten Seminars *Goethe digital [...]*, die durch rege Kritik und vielfältige Verbesserungsvorschläge entscheidend zur Weiterentwicklung des MoveParser-Programms beigetragen haben.

[5] Jan Christoph Meister: Alle Erklärung muß fort und nur Beschreibung an ihre Stelle treten. Vor-Überlegungen zu einer computergestützten empirischen Analyse fiktionaler Handlungen. In: Internationales Jahrbuch für Germanistik 2 (1994), S. 30-58. – Ders.: Weltbegebenheiten und Privatgeschichten in Goethes *Unterhaltungen Deutscher Ausgewanderten*. Zur Methodologie einer computergestützten Analyse von Handlungsstrukturen in literarischen Texten. In: SPIEL (= Siegener Periodikum für Internationale Empirische Literaturwissenschaft) 1 (1994), S. 125-142.

[6] Thomas Pavel: The Poetics of Plot. The Case of the English Renaissance Drama. Manchester 1985.

[7] Siehe hierzu: A. J. Greimas: On Meaning. Selected Writings in Semiotic Theory. Minneapolis 1987. – Ders.: Elemente einer narrativen Grammatik. In: H. Blumensath (Hg.): Strukturalismus in der Literaturwissenschaft. Köln 1972, S.47-67.

[8] Als Textbasis eignen sich verständlicherweise vorwiegend realistische Geschehensschilderungen, in denen weder das szenische Kolorit, noch die kommentierenden oder gar autothematischen Anteile das Übergewicht gewinnen (obwohl prinzipiell – wenn auch unter erheblichen Schwierigkeiten – dieses ›Erzählgeschehen‹ ebenfalls als Move-Verkettung beschreibbar wäre.)

[9] Siehe hierzu: Jan Christoph Meister: Consensus ex machina? Consensus qua machina! In: Literary and Linguistic Computing 10 (1995), S. 263-270.
