AUGMENTING HUMAN CREATIVITY[1]:

Virtuelle Realitäten als Design-Aufgabe

GEORG TROGEMANN
Kunst- und Medienwissenschaften
Kunsthochschule für Medien Köln
trogemann@khm.de

Zusammenfassung:

„If VR is so great, why are VR entertainment systems so poor?“
Roy W. Latham

Eine einfache Prämisse leitet die hier angestellten Überlegungen: Virtuelle Realitäten sind nicht gegeben, sie müssen erfunden und entworfen werden! Als Design-Aufgabe aber entziehen sie sich der Naturwissenschaft, die sich dem Wissen über natürliche Objekte und Erscheinungen verschrieben hat. Die Herstellung künstlicher Objekte mit geplanten Eigenschaften steht traditionell im Zentrum der Arbeit des Ingenieurs wie auch des Künstlers und Gestalters. In der Informatik zeigt sich allerdings immer deutlicher, dass Virtuelle Realitäten - als komplexe technische Artefakte - nur begrenzt mit den gegenwärtigen ingenieurs-wissenschaftlichen Methoden zu entwerfen und zu realisieren sind. Bis Heute werden Computer primär entwickelt, um vorformulierte Probleme zu lösen, oder Daten nach strikt festgelegten Prozeduren zu bearbeiten. Dieses Verständnis des Computers als Werkzeug zur Unterstützung repetitiver und mechanischer Arbeiten war lange Zeit ein taugliches Paradigma, sowohl für die Entwickler der Maschine als auch ihre Anwender. Viele interessante VR-Applikationen (VR Entertainment, Computerspiele, interaktive künstlerische Installationen, e-Learning Szenarien, etc.) können jedoch nicht zufriedenstellend als „wohldefinierte Probleme“ beschrieben werden, sondern gehören als Design-Aufgaben zum Bereich der „schlecht definierbaren Probleme“.
Der vorliegende Beitrag diskutiert die Begrenzungen heutiger Werkzeuge und VR-Applikationen im Umfeld kreativer Entwurfsprozesse. Als größtes Hindernis gegenwärtiger Autorensysteme erweist sich die Festschreibung ihrer Funktionalität während des gesamten Lebenszyklus und die mangelnde Flexibilität, sich zusammen mit dem Designprozess zu entwickeln. Funktionale Offenheit eines Werkzeugs dagegen würde bedeuten, dass es nicht nötig ist, schon bei der Systementwicklung alle Funktionen zu definieren und zu implementieren. Im Zusammenhang mit funktional-offenen Systemen erlangt zunehmend der Begriff “Emergenz” an Bedeutung, der im Zentrum der hier angestellten Überlegungen steht. Am Beispiel „Interactive Drama“ werden die Herausforderungen für die Entwicklung neuer Autorenwerkzeuge für emergente Erzählformen verdeutlicht.

Schlüsselwörter:
Wohldefinierte & schlecht definierte Probleme, Emergenz, funktional-offene Systeme, Interactive Narratives

[1] In Anlehnung an D.C. Engelbarts ”Augmenting Human Intellect”.

1. Technisch unterstützte Problemlösungskompetenz

> *"... there is little doubt that the worldwide availability of fantasy amplifiers, intellectual toolkits, and interactive electronic communities will change the way people think, learn, and communicate."*
>
> Howard Rheingold, Tools for Thought

Um die Effektivität des individuellen menschlichen Intellekts zu verbessern, entwickelte Douglas C. Engelbart bereits Ende der Fünfziger und Anfang der Sechziger Jahre einen weitreichenden Forschungsansatz. 1962 schreibt er in seinem "Program On Human Effectiveness" (Engelbart 1991):

"Human beings face ever more complex and urgent problems, and their effectiveness in dealing with these problems is a matter that is critical to the stability and continued progress of society. A human is effective not just because he applies to a problem a high degree of native intelligence or physical strength (with a full measure of motivation and purposefulness), but also because he makes use of efficient tools, methods, and strategies."

Die grundlegende Hypothese seines Forschungsprogramms war, dass die Möglichkeit, Symbole in Echtzeit und als Reaktion auf die sich ständig verändernden Denkvorgänge eines Benutzers manipulieren zu können, tiefgehende Auswirkungen auf dessen Problemlösungskompetenz hat. Die Entwicklung neuer Werkzeuge und Interfaces für die Unterstützung komplexer Problemlösungsprozesse stand im Zentrum seines Forschungsprogramms. "Gesteigerte Fähigkeiten" (increased capabilities, Engelbart 1962) sollten einerseits im Hinblick auf die Geschwindigkeit des Problemlösungsprozesses und der Qualität der Lösungen erzielt werden, vor allem aber sollten Lösungen für Problembereiche zugänglich werden, die vorher als unlösbar galten. Douglas Engelbart gehört damit zu den Ersten, die über die Ziele der Computer-Pioniere von Schickard, Pascal und Leibniz bis Turing, Zuse, von Neumann und vielen anderen hinausgingen. Im Fokus der Pioniere stand die Befreiung des Menschen von Routinearbeiten des Geistes. Repetitive geistige Operationen sollten auf Maschinen übertragen werden. Ende der Fünfziger Jahre gab es auch die ersten Ergebnisse auf dem Gebiet der Künstlichen Intelligenz (KI), die in ihrer Zielsetzung ebenfalls über den Anspruch der Pioniere hinausging. Douglas Engelbart unterscheidet sich aber in einem wesentlichen Punkt von der KI, die ihr Forschungsprogramm bereits einige Jahre früher startete. Es geht ihm weder darum, intelligente Maschinen zu bauen, noch den Menschen intelligenter zu machen. Im Fokus stehen technische Systeme, die im Zusammenspiel mit dem Benutzer in der Lage sind, komplexere Probleme zu lösen, als er es ohne diese Unterstützung vermag. Seine Idee ist die einer symbiotischen Mensch-Maschine-Beziehung im Sinne Lickliders (Licklider 1960). Damit ist der Ansatz weder der KI, noch den heutigen ergonomischen Interface-Konzepten zuzuordnen.

"Accepting the term ***„intelligence amplification"*** *does not imply any attempt to increase native human intelligence. The term "intelligence amplification" seams applicable to our goal of augmenting the human intellect in that the entity to be produced will exhibit more of what can be called intelligence than an unaided human could; we will have amplified the intelligence of the human by organizing his intellectual capabilities into higher levels of synergistic structuring."* (Engelbart 1962, p. 79)

Heute ist sein Forschungsprogramm am Stanford Research Institute in Menlo Park vor allem für die Entwicklung der Computer-Maus, Windows, E-mail und Textverarbeitung bekannt. Die von ihm vorgeschlagenen Prinzipien des Zeigens, Auswählens und Manipulierens von Symbolen in Texten, Bildern und virtuellen Räumen haben sich zum Standard der Interaktivität am Computer entwickelt und gelten zunehmend als grundlegende Kulturtechniken.

Mit dem Vorstoß des Computers in neue Anwendungsfelder wurde aber auch deutlich, dass viele der neuen Probleme sich qualitativ von den Problemen der Mathematik unterscheiden. Während mathematisch beschriebene Probleme, hierunter fallen auch die Probleme, wie sie die KI in ihren Anfängen formuliert hatte, uneingeschränkt "wohldefiniert" sind, gehört der gesamte "Kreativ-Bereich", hierzu zählen nicht nur Kunst und Design, sondern auch weite Felder der Wissenschaft, Politik, Wirtschaft, Organisation und Planung, u.v.a., dagegen zum Feld der "schlecht definierten Probleme". In gegenwärtigen Softwaresystemen spiegeln sich dagegen vor allem die Strukturen wohldefinierter Problemlösungsstrategien wieder, d.h. sie zeigen ein relativ rigides funktionales Verhalten. Die Frage, der wir im folgenden nachgehen wollen, lautet deshalb: Sind wohldefinierte Softwaresysteme, d.h. Programme mit fest vorgegebenen Funktionalitäten in der Lage, die Arbeit an schlecht definierten Problemen geeignet zu unterstützen?[2] Oder müssen vielleicht andere Grundlagen geschaffen werden, damit z.B. Virtuelle Welten, Computer Spiele, und nichtlineare Erzählformen nicht ausschließlich Visualisierungen von vorab objektivierten und in ihren Relationen bereits vollständig determinierten Ereignisräumen bleiben.

Die im folgenden angestellten Überlegungen zu virtuellen Realitäten als Entwurfsproblem beziehen sich auf zwei Punkte.

(a) Die Werkzeuge, mit denen komplexe Artefakte wie Virtuelle Realitäten entworfen und hergestellt werden. Zusätzlich zu den existierenden Werkzeugen mit starren Funktionalitäten zur Lösung standardisierbarer Aufgaben, sind für Kreative individuelle Werkzeuge interessant, die funktional-offen (Open-ended) sind und damit in der Lage, sich in Kooperation mit dem Benutzer weiterzuentwickeln und neue Funktionalitäten auszuprägen.

(b) Die Applikationen, hier insbesondere computergenerierte Welten in denen interaktive Erzählformen zum Tragen kommen. In gegenwärtigen VR-Systemen wird davon ausgegangen, dass alle wichtigen Objekte und ihr Interaktionsverhalten festgelegt sind. Darüber hinaus ist der gesamte Verlauf der erzählten Geschichte – trotz aller Interaktivität – vollständig durch den Autor vorstrukturiert. Offene Virtuelle Welten müssten dagegen die Fähigkeit besitzen, sich an die Bedürfnisse des Benutzers und die sich verändernden Situationen anzupassen. Sie sollten emergent sein bezüglich der Geschichte, die sich erst durch die Interaktion mit dem Benutzer entfaltet. Die Frage lautet: Wie können sich System und Benutzer zusammen in unvorhersehbarer Weise entwickeln?

[2] Im Hinblick auf das Phänomen "Kreativität" konzentriert sich unser Beitrag auf einen Ausschnitt, den Aspekt der so genannten "schlecht definierten Probleme". Zwar sind diese Probleme sehr viel häufiger als man gemeinhin erwartet - auch in der Wirtschaft und den Ingenieurswissenschaften gibt es weite Bereiche, die als schlecht definierbare Probleme eingestuft werden müssen – dennoch können Kreativität und schlecht definierte Probleme nicht gleichgesetzt werden. Die Arbeit von Kreativen hat nur begrenzt mit Problemlösungsprozessen zu tun. Kreativprozesse können deshalb nicht einfach nur als Spezialfälle von Problemlösungsstrategien betrachtet werden. Die Problematik kann auch durch die Frage verdeutlicht werden: Welche Probleme lösen Künstler?

2. Wohldefinierte Probleme

> *Ein "Problem" entsteht z.B. dann, wenn ein Lebewesen ein Ziel hat und nicht "weiss", wie es dieses Ziel erreichen soll. Wo immer der gegebene Zustand sich nicht durch bloßes Handeln (Ausführen selbstverständlicher Operationen) in den erstrebten Zustand überführen lässt, wird das Denken auf den Plan gerufen. Ihm liegt es ob, ein vermittelndes Handeln allererst zu konzipieren.*
>
> Duncker, 1935

Computer und Medien im allgemeinen werden unter einer klaren Ziel- und Zweckorientierung entworfen. Ein wirtschaftlich bedeutsamer Zweck von Computern ist die Unterstützung des Benutzers (oder einer kollaborierenden Gruppe von Benutzern) bei Problemlösungsprozessen. Die Probleme, die Engelbart als Kandidaten für seine ersten Untersuchungen zur Steigerung der intellektuellen Effektivität für geeignet hielt, sollten sich nach seiner eigenen Charakterisierung durch eine gewisse Handhabbarkeit auszeichnen (Engelbart 1991). Es sollte insbesondere ein begrenzter Informationsbereich und begrenzte Interaktion mit der externen Welt möglich sein. Dies deutet darauf hin, dass er die qualitativen Unterschiede in der Komplexität formal beschreibbarer Probleme bereits erkannt hatte.

Die Wissenschaft unterscheidet zwei grundlegende Typen von Problemen (Dunbar 1998): wohldefinierte (well-defined) und schlecht definierte (ill-defined). In diesem und im nächsten Abschnitt wollen wir diese auch historisch wichtigen Gruppen von Problemen näher betrachten. Wohldefinierte Probleme sind eng verbunden mit der Theorie der Berechenbarkeit. Die wesentliche Forderung, die an wohldefinierte Probleme gestellt wird, ist die Existenz einer systematischen Methode, um zu entscheiden, ob eine akzeptable Lösung für ein Problem vorliegt. Es reicht insbesondere nicht aus zu fordern, dass ein Problem formal eindeutig beschreibbar ist. Es konnte gezeigt werden, dass eine Reihe klar definierbarer Fragestellungen existieren, für die keine berechenbare Lösung existiert.[3] Wohldefinierte Probleme beinhalten Abstraktionen, die in der Regel sicher stellen, dass verschiedene Lösungsmethoden existieren und somit können Optimierungen im Hinblick auf die besten Strategien und effizientesten Algorithmen vorgenommen werden. Obgleich auch hier festgehalten werden muss, dass bereits für verschiedene einfach formulierbare Entscheidungsprobleme (Probleme mit ja oder nein Antworten) keine effizienten Algorithmen bekannt sind. Da die meisten dieser NP-vollständigen Probleme aufeinander abgebildet werden können, muss festgestellt werden, dass es Probleme gibt, die für praktisch relevante Eingabegrößen nicht mit (gegenwärtigen) Computern bearbeitet werden können.

Im mathematischen Sinne bestehen Probleme aus 4 Komponenten (Dunbar 1998): 1. einem Anfangszustand, der den Stand des Wissens zu Beginn des Problems repräsentiert, 2. einer Menge möglicher Aktionen, 3. einem Endzustand, das Ziel, das erreicht werden soll und 4. einem Kontext, in dem das Problem gelöst werden soll. Der Kontext besteht aus Merkmalen der

[3] Um den Begriff der Berechenbarkeit genauer zu fassen, wurden verschiedene Formalismen (Turing-Maschinen, Markov-Algorithmen, rekursive Funktionen, Grammatiken, etc.) entwickelt, die sich alle in einem gewissen Sinn als äquivalent erwiesen haben. Alle Untersuchungen haben so auf denselben Berechenbarkeitsbegriff hingeführt. Diese Erkenntnis findet ihren Niederschlag in der berühmten Church'schen These, die in einer Variante lautet: Jede berechenbare Funktion lässt sich programmieren. Wichtig ist aber hier vor allem die Umkehrung. Es wurde gezeigt, dass es wichtige, wohldefinierte Probleme gibt, die sich weder entscheiden noch programmieren lassen.

Umgebung, die direkt oder indirekt mögliche Lösungen beschränken oder bestimmte Lösungswege nahe legen. Bei wohldefinierten Problemen sind die ersten drei Komponenten bekannt. Die vierte Komponente muss zur Lösung des Problems in der Regel nicht berücksichtigt werden. Für die Problembearbeitung mit Unterstützung des Computers, gibt es unterschiedliche Möglichkeiten: die automatische (KI) und die manuelle Lösung. Beide Formen sollen im folgenden kurz betrachtet werden.

Die KI hatte sich als Aufgabe gestellt, Probleme eigenständig durch den Computer lösen zu lassen. Die frühen KI-Forschungen gingen aber keineswegs von formal schwer fassbaren Problemstellungen aus, sondern von wohldefinierten Problemen, d.h. Probleme bei denen Anfangszustand, Zielzustand und mögliche Aktionen bekannt sind. In "Steps Towards Artificial Intelligence", einem vielbeachteten Artikel zur KI-Forschung, schreibt (Minsky 1963): "*When we talk of problem solving in what follows, we will usually suppose that all the problems to be solved are initially well-defined. By this we mean that with each problem we are given some systematic way to decide when a proposed solution is acceptable.*" Die Methoden, die Minsky zur Lösung wohldefinierter Probleme der KI untersucht, fasst er unter dem Begriff "heuristisch" zusammen. Zu den "heuristischen" Problemlösungsmethoden durch den Computer gehören für (Minsky 1963) die fünf Kategorien: Suche, Mustererkennung, Lernen, Planung und Induktion. Wohldefinierte Probleme zeichnen sich nun - wie bereits erläutert - dadurch aus, dass ein klares Kriterium existiert, wann eine Lösung für das Problem vorliegt.

Um wohldefinierte Probleme mit dem Computer bearbeiten zu können, sie also heuristischen Methoden zugänglich zu machen, müssen die ersten drei Komponenten (siehe oben Dunbar: Anfangszustand, Aktionen und Endzustand) weiter präzisiert werden. Formal zu beschreiben sind:

- Anfangszustand
- Menge möglicher Aktionen (Operatoren)
  - Nachfolger-Funktion oder Zustandsübergangsfunktion
  - Zustandsraum: Menge aller Zustände, die vom Anfangszustand durch eine Folge von Aktionen erreichbar sind
  - Pfad: Folge von Aktionen
- Ziel-Test
- Pfad-Kosten-Funktion

Die Berechnung einer akzeptablen Problemlösung bedeutet, unter Anwendung einer bestimmten Strategie (der Lösungsmethode) einen Pfad zu finden, der sowohl den Anfangszustand als auch einen Zielzustand beinhaltet. Über die Effizienz des Suchprozesses gibt die Pfad-Kosten-Funktion Auskunft. In gewissem Sinn sind diese Probleme trivial, da die Lösung im Prinzip immer gefunden werden kann, indem der Zustandsraum des Problems mehr oder weniger blind durchsucht wird. In der Regel ist es auch nicht schwer ein Verfahren anzugeben, das den Zustandsraum systematisch durchläuft. Der entscheidende Punkt ist, dass für praxisrelevante Probleme aufgrund der Größe der Zustandsräume das Verfahren nicht effektiv, d.h. nicht durchführbar ist. Die Analyse des Zustandsraumes muss also immer unvollständig bleiben. Heuristische Algorithmen sind gerade jene Techniken, die auch bei unvollständiger Analyse des Zustandsraumes effiziente Lösungen liefern.

Während die automatische Lösung wohldefinierter Probleme verbunden ist mit den Zielsetzungen der KI, sind manuelle Problemlösungsprozesse eng verbunden mit interaktiven Prozessen. Erst durch die Möglichkeit der wechselseitigen Aktivität von Benutzer und System werden manuelle Problemlösungsprozesse zu einem interessanten Forschungsfeld der Softwareentwicklung.

Die manuelle Lösung wohldefinierter Probleme besteht aus vier Schritten:

- Anfangszustand
- Menge möglicher Aktionen (Operatoren)
- Die Auswahl (und/oder Kontrolle) einer Aktion durch den Benutzer
- Die Ausführung der Aktion durch den Computer
- Zielkriterium

Manuelle Problemlösungsprozesse sind gekennzeichnet durch eine klare Aufgabenteilung zwischen System und Benutzer. Das System stellt eine bestimmte Anzahl von Funktionen zur Verfügung und der Benutzer wählt während des Bearbeitungsprozesses nacheinander einzelne Funktionen aus und bringt sie zur Ausführung. Dem Benutzer obliegt dabei die Aufgabe, das Ziel zu erreichen, indem er die Reihenfolge, Parametrisierung und Kontrolle der einzelnen Funktionen und Arbeitsschritte entsprechend plant und überwacht. Nahezu jede zielgerichtete Arbeit mit WIMP-orientierten Systemen kann auf diese Weise als manueller Problemlösungsprozess aufgefasst werden. Usability-Überlegungen im Hinblick auf die Gestaltung der Interfaces liefern hier zufriedenstellende Ergebnisse, da sowohl die Aufgabe, als auch die Lösung der Aufgabe kein prinzipielles Problem darstellen. Sind die Funktionen des Systems einmal festgelegt, dann ist lediglich komfortable Benutzerführung verlangt, um die bereits vorgezeichneten Arbeitsschritte möglichst reibungsfrei zu erledigen. Das Verschwinden der Schnittstelle ist hier die eigentliche Zielsetzung des Interfacedesigns.

3. Design – Ein schlecht definiertes Problem

> *"... how does the artist use visualization? ... The answer is that the artist rarely has a specific goal in mind when he begins work, so that the process of visualization is actually the search for a goal rather than the attainment of one."*
>
> S.D. Katz

Schlecht definierte Probleme zeichnen sich dadurch aus, dass weder das Ziel, die möglichen Aktionen, noch der gegenwärtige Zustand des Systems exakt benennbar sind. Wohldefinierte Probleme sind in gewisser Hinsicht statisch, d.h. mit der Problembeschreibung, die ganz am Anfang steht, ist auch der Lösungsraum determiniert. Die Lösungsfindung ist eine kontrollierte Bewegung (z.B. Stichwort Hill-Glimbing) durch den fixen Zustandsraum. Schlecht definierte Probleme zeichnen sich gerade dadurch aus, dass der Zustandsraum der das Problem beschreibt, sich während der Bearbeitung verändert.

Die wichtigsten Charakteristika schlecht definierter Probleme sind:

- Moving Targets – Die Probleme sind nicht vollständig formalisierbar

Eines der Schlüsselelemente jeder computergestützten Problemlösung ist die formale Repräsentation des Problems. Bei schlecht definierten Problemen ist anfänglich nur eine unvollständige und ungenaue Beschreibung der Ausgangssituation sowie des Ziels und der zulässigen Operationen möglich. Den Zustandsraum des Problems bereits zu Beginn durch ein formales Modell festzuschreiben, würde den Horizont der Aufgabe und die Möglichkeiten der Weiterentwicklung der Aufgabenstellung zu sehr einschränken. Wir müssen sicherstellen, dass der Zustandsraum, der das Problem beschreibt, sich während der Bearbeitung weiterentwickeln kann. Das Problem und das Ziel verändert sich, während daran gearbeitet wird (Moving Target). Problemabgrenzung und Problemlösung sind nicht zwei zeitlich vollständig getrennte Phasen, sondern ein simultaner und iterativ fortschreitender Prozess.

- Ein Großteil der Anstrengung muss für die "Problemfindung" verwendet werden

Im Zusammenhang mit so genannten kreativen Arbeitsprozessen ist gerade die Entwicklung einer Fragestellung die wesentliche kreative Leistung. Auch in Alltagssituationen braucht es oft gerade Kreativität, um ein Problem überhaupt erst zu schaffen, es zu erkennen und es zu formulieren. Die Problemfindung und Eingrenzung wird bei den meisten Problemlösungsprozessen unterbewertet. Wie sehen die Algorithmen der Problemerzeugung aus, als Ergänzung zu den Algorithmen der Problemlösung? Welche Methoden existieren, um entstandene Probleme in dynamischen Prozessen zu erkennen? Die selbst bestimmte Definition von Zielen, etwa als der Entwurf von neuen Horizonten (Kontexten, Rahmenbedingungen) ist deshalb neben Problemlösungsverfahren und Bewertungsstrategien eine der großen Herausforderungen für die Formalisierung schlecht definierter Probleme. Eine zentrale Frage lautet: Wie können wir den Benutzer bei der Suche nach Problemen unterstützen? Mit Poincaré: "The question is not, 'what is the answer?' The question is, 'what is the question?'"

- Die Grenzen für die relevante Information sind vage

Es gibt keine eindeutigen Kriterien, welche Informationen bei der Problembearbeitung berücksichtigt werden müssen und welche zu vernachlässigen sind. Ein Zitat aus dem Bereich Design, einem kanonischen Feld schlecht definierter Probleme, soll diesen Punkt verdeutlichen:

"A particular challenge for context-aware applications in design domains is that context emerges throughout the design process, and that a determination of whether some information or action should be considered as relevant or irrelevant context can be determined only during the design process,not beforehand." (Fischer 2001)

- Die Methoden zur Lösung sind nicht genau bekannt

Die Lösungsmethoden liegen nicht in Form standardisierter, hinreichend untersuchter und gut verstandener Algorithmen vor. Im Falle der kooperativen Bearbeitung müssen die zur Lösung zu Verfügung stehenden Ansätze und Vorgehensweisen erst durch einen mühsamen Kommunikationsprozess zwischen den unterschiedlichen Gruppen jeweils neu gewonnen werden. Allerdings gibt es auch hier bewährte Lösungen für wiederkehrende Probleme. Diesen Gesichtspunkt versuchen z.B. "Pattern Languages" zu operationalisieren (siehe Borchers 2000, 2001).

- Es gibt keine definitiven Kriterien, um “Lösungen” zu beurteilen

Schlecht definierte Probleme gelten als offen, weil es keine eindeutige oder korrekte Lösung für ein gegebenes Problem gibt, sondern viele mögliche Lösungen. Im Gegensatz zu mathematischen Problemen gibt es keinen Zeitpunkt, zu dem das Problem als endgültig gelöst betrachtet werden kann. Schlecht definierte Probleme werden nicht “gelöst” in der gleichen Art wie wohldefinierte Probleme es werden. Es wird vielmehr an einem zufriedenstellenden Punkt des Prozesses der erreichte Zustand als Lösung betrachtet. Die Entscheidung, wann ein Zustand zufriedenstellend ist, hängt dabei stark von der persönlichen Sicht des Bearbeiters (oder einer kollaborierenden Gruppe) ab.

- Die Lösung trägt die Handschrift des Bearbeiters

Im Verlaufe des Arbeitsprozesses entstehen individuelle Problemspezifikationen, die eng mit subjektiven Einschätzungen und Sichtweisen verknüpft sind. Unterschiedliche Personen, denen die gleiche Aufgabe gestellt wird, werden zu höchst unterschiedlichen Problemdarstellungen, Ideen und Lösungsvorschlägen kommen. Problemdenken in kreativen Prozessen bezieht die Persönlichkeit des Analysierenden mit ein, ist also ein hochgradig selbstreferenzieller Prozess.

· Kommunikation ist primär

Bei vielen praxisrelevanten schlecht definierten Problemen müssen oft große Gruppen mit unterschiedlichen Erfahrungen und Wissen zusammenarbeiten. Da die Lösung nur arbeitsteilig erreichbar ist, müssen Fachleute aus verschiedenen Disziplinen an einem koordinierten Kommunikationsprozess teilnehmen. Die Kommunikationsmittel, d.h. geeignete “Sprachen”, die auch von Anwendern und Praktikern gleichwohl verstanden werden, stehen im Zentrum verteilter Problemlösungsprozesse.

Beispiel 1: Softwareentwicklung

Softwareentwicklung kann natürlich ebenfalls als Problemlösungsprozess betrachtet werden. Wir können also fragen, zu welcher Art von Problemen Softwareentwicklung gehört. Im nächsten Abschnitt wird beschrieben, in welchem Sinne Software als wohldefiniert gelten kann. Für viele Anwendungsgebiete sind Softwaresysteme mit wohldefiniertem Verhalten nicht nur ausreichend, sondern als Zielvorgabe absolut notwendig. Aber auch, wenn als Produkt ein wohldefiniertes System entstehen soll, ist der Prozess der Softwareentwicklung selbst dennoch schlecht definiert. Während in der Vergangenheit bei der Ausbildung von Softwareentwicklern großer Wert auf mathematisch-formale Kenntnisse gelegt wurde, wird zunehmend erkannt, dass die Fähigkeiten eines koordinierenden und kommunizierenden Ingenieurs viel wichtiger sind, als die eines streng formalen Mathematikers. Im Gegensatz zur Mathematik ist im Bereich der Softwareentwicklung 1. die Problemstellung nie vollständig, sondern lässt Ermessensspielraum der ausgehandelt werden muss, 2. müssen die zur Lösung zu Verfügung stehenden Bausteine teilweise erst durch einen mühsamen Kommunikationsprozess gewonnen werden, und 3. ist die Lösung nur arbeitsteilig erreichbar. Primärer Kenntnisbedarf sind also nicht formale Methoden, sondern Kommunikationsmittel (Trogemann 2001).

Beispiel 2: Design

Designprobleme sind schlecht definiert (Simon 1996) und erfordern zu ihrer Lösung die Beteiligung verschiedener Experten aus ganz unterschiedlichen Disziplinen. Die Offenheit und Komplexität von Designproblemen resultiert aus mehreren Anforderungen. Es müssen nicht nur unterschiedliche Perspektiven vereinigt werden, sondern es sind auch sehr große Mengen von

Informationen zu verwalten, die für den Designprozess relevant sind. *"For most design problems the knowledge to understand, frame, and solve these problems does not exist, but is constructed and evolved during the process of solving them. From this perspective, access to existing information and knowledge (often seen as the major advance of new media) is a very limiting concept. Many social and technological innovations are limited to provide primarily better access, rather than supporting informed participation by allowing learners to incrementally acquire ownership in problems and contribute actively to their solution. ....*
Design problems, being ill-defined and unique, require informed participation by all stakeholders. Openness and complexity in design arises from the need to synthesize different perspectives of a problem, the management of large amounts of information relevant to a design task, and understanding the design decisions that have determined the long-term evolution of a designed artifact. The social interaction among stakeholders in design can be characterized by a ‚symmetry of ignorance' or an ‚asymmetry of knowledge'." (Fischer 2000)

4. Wohldefinierte Software

Nur die Fragen, die prinzipiell unentscheidbar sind, können wir entscheiden.
Heinz von Foerster

Der Softwareentwicklungszyklus kann grob unterteilt werden in eine Analysephase (z.B. Anforderungsanalyse, Benutzeranalyse und Taskanalyse) und die sich anschließende Implementierungsphase (Strukturdesign, Prototyp, Usability Tests, Redesign, etc.). Auf der Basis dieses zyklischen Prozesses entstehen Entwürfe und Prototypen, die bewertet und weiter verbessert werden, bis ein zufriedenstellendes Softwareprodukt entsteht. Die Forschungen konzentrieren sich darauf, die Schwächen dieses Prozesses zu verbessern, z.B. die so genannte "design gap" zu schließen, d.h. die Lücke zwischen den Ergebnissen der Anforderungsanalyse und ihrer direkten Umsetzung in der Benutzungsoberfläche. (Kovitz 1998) weist darauf hin, dass die Ergebnisse der Analysephase, d.h. die Anforderungen (requirements), die an ein Programm gestellt werden, zu trennen sind von den Anforderungen an das Interfacedesign.

"The fundamental reason for carefully distinguishing interface design from requirements is that requirements are designed in response to an open-ended problem, but interfaces are defined in response to a well defined problem. That is, there are no rigid criteria for evaluating requirements; we simply make a decison to build software to bring about certain effects. An interface, on the other hand, derives from a well-defined problem – the one provided by the requirements. You can prove deductively that an interface meets requirements. But you can't validate requirements by such a deductive method; you, the customer, simply judge them as good or bad effects to achieve.

In diesem Sinne ist das obige Zitat von Heinz von Foerster zu verstehen. Die Fragen, die mit Hilfe formaler Methoden und Berechenbarkeitsüberlegungen prinzipiell unentscheidbar sind, können, bzw. müssen wir entscheiden. Die Anforderungsanalyse (im Sinne Kovitz' der Kunde) entscheidet über die unentscheidbaren Fragen. Nachdem aber diese Fragen entschieden sind, ist der Rest ein weitestgehend wohldefiniertes Problem. Am Ende des Software-Entwicklungsprozesses steht ein Produkt, das wohldefinierte Eigenschaften und Funktionalitäten besitzt. Heutige Software kann insofern als wohldefiniert bezeichnet werden, als ihr gesamtes

Repertoire an Funktionen, zulässigen Interaktionsformen, Menüpunkten und dynamischen Verhalten bereits vollständig in der Anforderungsbeschreibung bzw. dem bei Auslieferung des Produktes beiliegendem Manual beschrieben sind[4]. Schlecht definierte Probleme sind aber gerade jene Probleme, für die keine finale Anforderungsliste geschrieben werden kann. Sie zeichnen sich dadurch aus, dass sich die Anforderungsliste durch die Arbeit am konkreten Problem verändert, und zwar bis kurz vor Abschluss des Projektes.

Aus den Merkmalen schlecht definierter Probleme lassen sich verschiedene Richtungen ableiten, in die heutige wohldefinierte Software erweitert werden kann, damit sie zur Bearbeitung dieser Aufgaben besser geeigneter ist.[5]

a) Der Benutzer sollte die Möglichkeit haben, die inneren Strukturen (Algorithmen) seiner Werkzeuge zu verändern, ohne dazu programmieren zu müssen. (Das heißt, das Interface ist das Programm.)
b) Die Werkzeuge sollten individualisierbar sein, d.h. die kreative Handschrift des Benutzers sollte sich über einen längeren Zeitraum in der Struktur des Werkzeugs einprägen.
c) Das Material (Objekte, Informationen, etc), mit denen der Benutzer arbeitet sollte nicht nur einen Standpunkt der Interpretation zulassen, sondern verschiedene Sichtweisen unterstützen und diese gegebenenfalls sogar generieren.
d) Das System sollte selbstmodifizierende Qualitäten besitzen, um flexibel gegenüber den sich ändernden Bedingungen kreativer Designprozesses zu sein, ohne dem Benutzer die Kontrolle über diese Veränderungen zu entziehen.

Entwickler von Softwaresystemen stehen grundsätzlich vor dem Problem, dass sie Systeme für Tausende oder gar Hunderttausende entwerfen, die aber so funktionieren sollen, als wären sie für den Einzelnen gemacht. Ein Ansatz, der diesem Missverhältnis Rechnung trägt und versucht die Individualität des Benutzers auf andere Weise in die Systementwicklung zu integrieren, sind "adaptive Systeme".[6] In (Kobsa 1993) sind verschiedene wünschenswerte Anpassungsleistungen adaptiver Systemen beschrieben. Der im nächsten Abschnitt behandelte Begriff „Emergenz" geht über die Zielsetzung adaptiver Systeme hinaus. Die Lernfähigkeit adaptiver Systeme bezieht sich auf automatisch ablaufende Anpassungen der beteiligten Daten und Prozessparameter, kurz der Operanden des Systems. Bei emergenten Prozessen unterliegen dagegen auch die Operatoren, d.h. die Algorithmen und Methoden, also der Prozess selbst der Veränderung. Dieser Wechsel hat tiefgreifende Auswirkungen auf die zugrunde liegenden Formalisierungen.

[4] Die Unentscheidbarkeit der Anforderungen bei kreativen Softwareentwicklungsprozessen spiegelt sich auch in der Tatsache wieder, dass Software im künstlerischen Bereich nie als "fertig" deklariert wird, sie befindet sich vielmehr immer in einem (lauffähigen) Zwischenzustand, der aber nur für die gegenwärtige Fragestellung zufriedenstellende Ergebnisse liefert. Es geht bei medienkünstlerischen Prozessen in der Regel nicht darum, ein verkaufbares Produkt mit definierten Funktionen abzuliefern, sondern der Prozess der Softwareentwicklung und die dabei gemachten Erfahrungen haben große Bedeutung für zukünftige künstlerische Entscheidungen. Die derzeit mächtigsten adaptiven und funktional emergenten Systeme für künstlerische Fragestellungen sind der Computer zusammen mit dem programmierenden Künstler!

[5] Auf die einzelnen Punkte wird in Abschnitt 5, unter Polaritäten, noch mal ausführlicher eingegangen.

[6] Unter adaptiven Systemen soll hier nicht benutzerinitiierte oder benutzerselektierte Adaption verstanden werden, die in der Regel durch Profildateien oder Präferenzmenüs gesteuert wird, sondern Anpassungen, die automatisch während des laufenden Betriebs der Software aufgrund eines Benutzermodells erfolgen.

5. Funktional-offene Systeme und Emergenz

> *Open-ended systems are about what godhood is about. An open-ended system is one where you begin something and the longer that system runs, the bigger and bigger [its creation gets]; it's making a bigger space into which the next version of itself can move.*
>
> Stewart Brand

Der Begriff "Emergenz" spielt im Hinblick auf die Entwicklung kreativitätsunterstützender Systeme eine wichtige Rolle. Eine landläufige Definition von Emergenz lautet: Das Ganze ist mehr als die Summe seiner Teile. Aus der Sicht kreativer Entwurfsprozesse liegt die Anforderung an emergente Systeme jedoch näher an der Frage Ross Ashbys: "Can a mechanical chess-player outplay it's designer." ... "to what extend is a machine restricted by the limitations of it's designer." (Ashby 1952, zitiert nach Cariani 1993, S. 119).

Es ist zu beobachten, dass der Begriff Emergenz in den letzten Jahren in ganz verschiedenen Bereichen benutzt wurde, um qualitativ vollkommen unterschiedliche Phänomene zu beschreiben. In (Cariani 1991) wird der Versuch unternommen, den Begriff in seiner vielfältigen Verwendung für verschiedenste Phänomene näher zu fassen. Cariani unterscheidet drei Definitionen von Emergenz: "Computational Emergence", "Thermodynamic Emergence" und "Emergence relative to a model". Insbesondere die aus der Kybernetik stammende dritte Charakterisierung erweist sich für die hier angestellten Überlegungen als wichtig.

Computational Emergence
Zu den emergenten Berechnungsmodellen die aus der Informatik heraus, d.h. im Hinblick auf die Verwendung in Computern entwickelt wurden, zählen zelluläre Automaten, Artificial Life Modelle und Neuronale Netze. Das interessante globale Verhalten dieser Systeme ergibt sich aus der Interaktion lokaler Regeln. Das globale Verhalten lässt sich nicht direkt aus den lokalen Regeln ablesen oder vorhersagen. Erst in der Ausführung der lokalen Regeln, d.h. wenn das System läuft, zeigen sich dem externen Beobachter globale Strukturen. Das System selbst weiß nichts von seinen globalen Strukturen, es braucht einen Beobachter außerhalb des Systems, der das globale Verhalten "sieht". Eine der Hauptfragen im Zusammenhang mit emergenten Berechnungsmodellen ist, wie verteilte Architekturen, in denen eine große Anzahl von Interaktionen und daraus resultierend unvorhersehbare Effekte auftreten, effektiv für den Entwurf von Computersysteme genutzt werden können. Da der Entwurf von Computern im Hinblick auf einen Zweck erfolgt, werden emergente Berechnungsmodelle untersucht, um z.B. neue Lernsysteme zu entwickeln, Computernetze zu organisieren und robuste Verhaltensstrategien in komplexen Umgebungen zu erlangen. Für die Lösung derartiger Probleme zeigen emergente Berechnungsmodelle aber auch verschiedene Schwächen.

Wesentliche Nachteile sind:
- Es gibt keine Anleitung, wie das System zu konstruieren ist, damit das gewünschte Verhalten entsteht
- Wir erhalten keine Einsicht, wie und warum sich das globale Verhalten einstellt
- Es gibt keine Rückkopplung von der höheren Ebene auf die niedrigere Ebene. Die höhere Ebene beeinflusst nicht die lokalen Regeln der niedrigen Ebene, wie dies beispielsweise in biologischen Systemen der Fall ist

Thermodynamic Emergence
Thermodynamische Emergenz beschreibt, wie Ordnung aus Unordnung entsteht. Zum Beispiel können Prozesse, die auf Mikroebene nur stochastisch beschrieben werden können, auf Makroebene durchaus diskretes deterministisches Verhalten zeigen. Ein häufig zitiertes Beispiel in diesem Zusammenhang ist "ideales Gas". Stochastische Bewegungen der Atome und Moleküle auf Mikroebene erzeugen stabile Eigenschaften des Drucks, der Temperatur und des Volumens auf höherer Ebene. Im Bereich Artificial Life wird nun ebenfalls seit längerem das Verhalten komplexer adaptiver Systeme simuliert, um zu erklären, wie stabile, komplexe Strukturen weit weg vom Gleichgewicht eines Systems entstehen. Die Antworten auf diese Fragen können uns helfen, eines der grundlegenden Probleme kreativer Prozesse zu verstehen: Wie können wir unwahrscheinliche aber nützliche und wirksame Lösungen für Entwurfsprobleme finden?

Emergenz in Bezug auf ein Modell
Festzuhalten ist, dass die meisten Modelle, die zur "Computational Emergence" zählen, aus der Sicht des Programmierers nicht als emergent gelten können. Sie sind insbesondere nicht "emergent in Bezug auf ein Modell" (Rosen 1985). Der Systememtwickler hat ein vollständiges Modell vom Geschehen, dieses vollständige Modell ist das Programm selbst. Auch die Einführung von Zufallselementen ändert nichts an diesem Sachverhalt. Auf der Ebene der Programmierung lässt sich das Verhalten des Computers immer als vollkommen deterministisch beschreiben. Insbesondere die Definition der "Computational Emergence", die unter Emergenz globales Verhalten als Folge der Interaktion lokaler Regeln versteht, ist nicht emergent in Bezug auf das Modell, d.h. das Computerprogramm, zu bezeichnen. Rosen (siehe Cariani 1989) definiert diese Form der Emergenz als Abweichung des Verhaltens eines physikalischen Systems vom Modell, das ein externer Beobachter von dem System (sich gemacht) hat. Diese Form der Emergenz beinhaltet eine Änderung der Beziehung zwischen dem beobachteten System und dem Beobachter. Wenn wir ein System beobachten, das seine interne Struktur verändert und damit sein Verhalten ändert, dann müssen wir als Beobachter, um das Verhalten des Systems weiterhin erfolgreich vorhersagen zu können, unser Modell vom System ändern. Cariani unterscheidet des weiteren zwei Formen adaptiven Verhaltens: syntaktische und semantische Emergenz. Syntaktische Emergenz bezieht sich auf die Software, d.h. auf Veränderungen in den Algorithmen bzw. den Prozessen die auf einem System laufen. Ein laufendes System kann neue algorithmische Funktionalitäten ausbilden, die vom Systementwickler nicht spezifiziert wurden. Vom Standpunkt des Beobachters ist damit neues Verhalten "emergiert". Semantische Emergenz bezieht sich auf die Hardware. Das Entstehen neuer Sensoren und Effektoren in einem adaptiven System ändert ebenfalls das Verhalten dieses Systems. Die Emergenz der Sensoren, die neue Informationen liefern, ist durch Aspekte begründet, die außerhalb des gegenwärtigen Beobachtungsbereiches der Umgebung liegen.

Die Bedeutung der Emergenz für Kreativprozesse
Emergente Systeme werden unverzichtbar, wenn wir schlecht definierte Probleme bearbeiten wollen, also Situationen, in denen wir uns nicht über die Form einigen können, wie das Problem zu beschreiben ist. Kreatives Design ist ein derartiger emergenter Prozess, bei dem das erzeugte Verhalten und die erzeugten Strukturen nicht auf die anfängliche allgemein akzeptierte Spezifikation reduziert werden können. Ziel ist deshalb die Entwicklung emergenter Prozesse, die in der Lage sind, die sich ändernden Bedingungen anzupassen und neue Potentiale in Problemlösungsprozessen zu erfassen. Dazu gehört, dass das System mit Situationen konfrontiert

wird, die während der Entwicklung des Systems nicht vorhersehbar waren. Cariani beschreibt diesen sehr wichtigen Punkt wie folgt:

"To build devices which find new observational primitives for us, they must be made epistemically autonomous relative to us, capable of searching realms for which we have no inkling."

Emergente Systeme reorganisieren ihre innere Struktur um ihre eigene Leistung in einer realen Umgebung zu verbessern. Wohldefinierte Softwaresysteme können dagegen nur innerhalb der funktionalen Zustände operieren, die vom Entwickler vorgesehen sind. Eng verbunden mit dem Begriff der Emergenz ist – wie Cariani deutlich macht - der Begriff Autonomie[7]. Doch was ist ein autonomes System, wie ist es zu "benutzen" und was bedeutet es für seinen Benutzer. Eine autonome Einheit ist in der Lage, ihre eigenen semantischen Kategorien zu entwickeln, jenseits der Einschränkungen durch die semantischen Kategorien eines Beobachters. Doch wie ist ein derartiges Gerät dann sinnvoll einzusetzen, wenn eine der wesentlichen Bedingungen für Nützlichkeit gemeinsame semantische Kategorien zwischen Benutzer und System sind?

Die derzeit erfolgreichsten Methoden, entwicklungsfähige Systeme zu realisieren, basieren auf dem Prinzip der "Interaktiven Evolution". Der computerzentrierte Designprozess der interaktiven Evolution unterscheidet sich grundlegend von herkömmlichen Designprozessen. Während im klassischen Designprozess die Weiterentwicklung des Status Quo durch einen hochgradig reflexiven Prozess erfolgt, konzentrieren sich evolutive Verfahren auf die Erzeugung einer großen Anzahl von Alternativen, von denen nur wenige Verbesserungen darstellen und der wesentliche Schritt darin besteht, die Interessantesten für die weitere Entwicklung auszuwählen.

Die Herausforderung emergenter Systeme geht aber sehr viel weiter. Das System darf nicht - wie Interaktive Genetische Algorithmen - an den Kategorien, mit denen es zu Beginn gestartet wurde, festhalten. Im Gegenteil, das strikte Festhalten an existierenden Kategorien würde das System als "unkreativ" disqualifizieren. Einige Fragen lauten dann: Wie können solche Systeme trainiert werden? Wie kann ein zwischen Benutzer und System koordinierter Fortschritt innerhalb einer bestimmten Designaufgabe sichergestellt werden? Können wir bestimmte Qualitätsstandards von den Ergebnissen des Systems erwarten? Wie kann die Kontrolle des System aussehen? Ist die einzige Möglichkeit, das System zu kontrollieren, die Messung und Bewertung der Ergebnisse. Wie kann das System seine innere Struktur offen legen? Können wir die Struktur dann überhaupt verstehen?

Die Entwicklung emergenter Systeme zur Unterstützung kreativer Entwurfsprozesse erfordert die Beschäftigung mit verschiedenen Polaritäten. Diese sind jedoch nicht als Attribute von Systemen zu verstehen sondern als Reflexionsbestimmungen. Ein System hat nicht einfach das Attribut "offen" ("geschlossen"), "analog" (digital") usw., sondern "offen", "digital" ist wie "oben",

[7] Der Autonomie-Begriff ist in der Informatik und den Ingenieurswissenschaften umstritten. Eine wichtige Forderung, die an autonome Systeme - wie wir sie hier diskutieren - gestellt werden muss, ist die Fähigkeit der „Selbsterzeugung von Wahlmöglichkeiten" im Sinne Gotthard Günthers. Heutige Systeme treffen dagegen nur Pseudo-Entscheidungen. "Pseudo-decisions are characterized by the fact that their alternatives always lie within the conceptual range of the programmer. This, of course, does not exclude that the programmer is completely taken by surprise when faced with the decision a computer has made. (...) What is important in this case is that the possible choices were implicitly generated outside the computing system." (Günther 1970) „Echte Entscheidungen" dagegen zeigen eine „gewisse Unabhängigkeit" vom Programmierer.

"links" eine standpunktabhängige Bestimmung, die jederzeit bei entsprechendem Standpunktwechsel in ihr polares Gegenteil umkippen kann. Es gilt, diese Widersprüche in ihrem Wechselspiel zu thematisieren und aufzulösen.

Polaritäten

- Leichte Bedienbarkeit vs. Universalität

Alan Kay, der Anfang der 70-er Jahre am Palo Alto Research Center (PARC) der Xerox Corp. im Silicon Valley arbeitete, ist einer der Pioniere des Usability-Konzeptes. Kay entwickelte die Vision eines dynamischen, interaktiven elektronischen Buches, das von jedermann, einschließlich Kindern, einfach zu bedienen sein sollte. Er folgte dabei, wie schon Douglas Engelbart, dem bedeutenden Aufsatz des Computerpioniers Vannevar Bush, der kurz nach Ende des Zweiten Weltkrieges unter dem Titel "As we may think" im US-Magazin »The Atlantic Monthly« seine Maschine namens "Memex" vorstellte. Alan Kay entwickelte aus diesem Ansatz das WIMP-System (Windows-Icons-Menus-PointingDevices). Anderthalb Jahrzehnte später wurde das Konzept durch den Apple-Gründer Steven Jobs im "Macintosh" umgesetzt. Damit wurde die Trennung zwischen “Usability” und “Universalität”[8] des Computers vollzogen. Die Ebenen der Kommandozeilen und der Programmierung wurden zu Bereichen, die nur noch dem Experten zugänglich sind. Die Universalmaschine Computer wird zum Spezialwerkzeug. Eine der ungelösten Fragen lautet seither: Ist es möglich, einfache Bedienbarkeit zu erreichen, ohne das volle Potential der Maschine - die Universalität im Sinne der Berechenbarkeit- zu verlieren?

- Subjekt vs. Objekt

Kreative Entwurfsprozesse sind eng verbunden mit individuellen Sichtweisen, mit der Fähigkeit, einen Standpunkt einzunehmen und mit permanenten Standpunktwechseln. Insbesondere, wenn in kollaborativen Systemen verschiedene Perspektiven zusammen kommen, unabhängig ob als Benutzer, Entwickler oder realisiert in Systemobjekten, müssen Mechanismen bereit gestellt werden, die in der Lage sind, diese überlagerten Ansichten eines einzigen Objektes zu beschreiben. Die wichtige Fähigkeit des Standpunktwechsel ist in den gegenwärtigen Standardprogrammiersprachen nicht abgebildet. Objekte der Realität, die aus einer bestimmten Perspektive als Einheit wahrgenommen werden, sind aus einer anderen Perspektive eine Vielheit. Programm-Objekte sind mit sich selbst identisch. (Smith 1996) fordert deshalb eine eigene Ontologie der Informatik, d.h. die Behandlung der Frage der Identität eines Objektes als eigenständige technische Fragestellung. In unseren gegenwärtigen Entwicklungsumgebungen wird dagegen versucht, den einzigen Standpunkt der Interpretation und der Manipulation sicher zu stellen. Es gibt den “global Observer” und die Kontrolle der Objekte und ihre Organisation sind streng hierarchisiert. Wichtige Fragen im Zusammenhang mit kreativitätsunterstützenden Systemen sind deshalb: Wie können wir unseren globalen Standpunkt der Interpretation und Manipulation überwinden? Wie können wir heterarchische Systeme bauen, als Alternative zu den bisherigen Organisationsformen der Hierarchisierung und Modularisierung komplexer Systeme? Da die Weiterentwicklung des System im Austausch mit dem Benutzer erfolgen soll und sich z.B. individuelle Handschriften nur über einen langen Zeitraum herausbilden, kann auch die Arbeit mit den Systemen nicht auf Kurzfristigkeit und Austauschbarkeit des Benutzers hin angelegt sein. Eine weitere Frage ist deshalb: Wie können Langzeitbeziehungen zwischen System und Benutzer aussehen?

[8] Der Terminus der Universalmaschine beruht auf der Churchschen These, die postuliert, dass jede formal algorithmisierbare Tätigkeit durch einen Computer ausgeführt werden kann.

- Kontrolle vs. Autonomie

Eine wichtige Paradoxie selbstmodifizierender Systeme lautet: Wenn System und Benutzer sich zusammen in unvorhersehbarer Weise entwickeln können, wie kann dann der Benutzer sicher sein, dass das Ergebnis die gewünschte Qualität besitzt? Hieraus leitet sich die Frage ab, wie in selbstmodifizierenden Systemen die Kontrolle zwischen dem Designer des Systems (Autor, Ingenieur, etc.), dem Benutzer und dem System selbst verteilt werden muss. Wie können wir Systeme entwickeln, die zur Selbst-Entwicklung fähig sind, ohne die Kontrolle zu verlieren? Etwas anders formuliert lautet die Frage: Wie können wir Zuverlässigkeit und Determiniertheit mit Unbestimmtheit und Überraschung verbinden? Es gibt eine grundlegende Polarität zwischen Kreativität und Autonomie auf der einen Seite und Wohldefiniertheit, Spezifizierbarkeit und Zuverlässigkeit auf der anderen Seite.

- Hardware vs. Software

"Evolvable Machines" zählen sicherlich zu den wichtigsten emergenten Technologien der Gegenwart. Bei der Anwendung des Evolutionsprinzips auf die Hardware-Entwicklung können zwei unterschiedliche Methoden unterschieden werden: die "on-line" und die "off-line" Methode (Sanchez und Tomassini 1996). Bei der off-line Methode wird der evolutionäre Entwurf als Software-Simulation ausgeführt. Nur die interessantesten Lösungen werden auch in Hardware realisiert. Im on-line Fall dagegen ist jede genetische Einheit auch eine autonome physikalische Einheit, die fähig ist, sich selbst zu modifizieren. Entscheidend ist, dass die Resultate der on-line Methode prinzipiell nicht mehr in Software simuliert werden können, da analoge Effekte am Ergebnis beteiligt sind. Auf diese Weise gewinnt die Materie, die durch den Funktionalismus aus der Informatik verbannt wurde, wieder an Bedeutung. (Cariani 1989) geht davon aus, dass nur Systeme, die ihre eigene Hardware manipulieren können, auch neue funktionale Kategorien (semantische Emergenz) ausbilden können. Nur neue Sensoren können neue Verbindungen mit der externen Welt herstellen, indem sie neue Observablen, d.h. neue beobachtbare und messbare physikalische Größen kreieren. Auf der Ausgabeseite des Systems können nur neue Effektoren neue Aktionen, d.h. neue Formen des Eingriffs in die Umgebung ermöglichen. Der Beweis dafür, dass auf Softwarebasis semantische Emergenz nicht möglich ist, steht noch aus. Viel wichtiger ist aber die Tatsache, dass wir solche Systeme bereits haben. Der Entwickler/Programmierer zusammen mit dem Computer ist ein semantisch emergentes System! Wir müssen lediglich die experimentelle Untersuchung neuer symbiotischer Muster für ein besseres Zusammenspiel zwischen System und Anwender verstärken.

- Geschlossenheit vs. Offenheit

Zur Unterstützung kreativer Entwurfsprozesse fordern wir funktional-offene Systeme. Darunter verstehen wir die Fähigkeit eines Systems, sich während seines Betriebes in der Interaktion mit dem Benutzer weiterzuentwickeln und neue Funktionen auszubilden. Aus der Biologie wissen wir allerdings, dass die organisatorische Geschlossenheit eine wesentliche Voraussetzung für autonome Systeme ist. Dies macht u.a. die "Closure Thesis" von (Varela 1979) deutlich: "Every autonomous system is organizationally closed. ... organizational closure is to describe a system with no input and no output." (Goldammer und Paul 1996) verdeutlichen die Konsequenzen dieser Forderung für technische Systeme: "Während alle bis heute bekannten Modelle der Neuroinformatik ausschließlich klassische Input/Output-Systeme – also offene Systeme – beschreiben, stellen die von der ‚Kybernetik 2. Ordnung' geforderten Modelle biologisch

kognitiver Netzwerke geschlossene Systeme dar. Hier besteht ganz offensichtlich ein unvereinbarer Widerspruch zwischen offenen und geschlossenen Systemen, Netzwerken oder Modellen der Beschreibung. Entscheidend ist dabei die Erkenntnis, dass nur geschlossene Systeme eine Umgebung besitzen können, während offene Systeme prinzipiell keine Umgebung besitzen." Die Umgebung muss vom Standpunkt des Systems aus existieren, nicht nur vom Standpunkt des Beobachters eines Systems, wie es heute bei technischen Systemen üblicherweise der Fall ist. Von einer "echten" Umgebung fordern wir zudem, dass prinzipiell Unvorhersehbares passieren kann, das also nicht schon beim Systementwurf berücksichtigt werden konnte.

6. „Interactive Drama" - Emergenz in virtuellen Realitäten

> *Aber die neuen Technologien, die überall auf der Welt aus dem Boden schießen, werden uns weiterhelfen. Im Kino müssen wir uns erst einmal von der Tyrannei befreien, und es sind deren vier: Die Tyrannei des Texts, die Tyrannei des Schauspielers, die Tyrannei des Bildausschnitts, und, am allerwichtigsten, die Tyrannei der Kamera ... Picasso sagte: Ich male nicht, was ich sehe, sondern was ich denke. Bisher haben wir nur Filme gemacht über das, was wir sehen. Die neuen Technologien werden uns die Freiheit geben, Filme über das zu machen, was wir denken.*
>
> Peter Greenaway

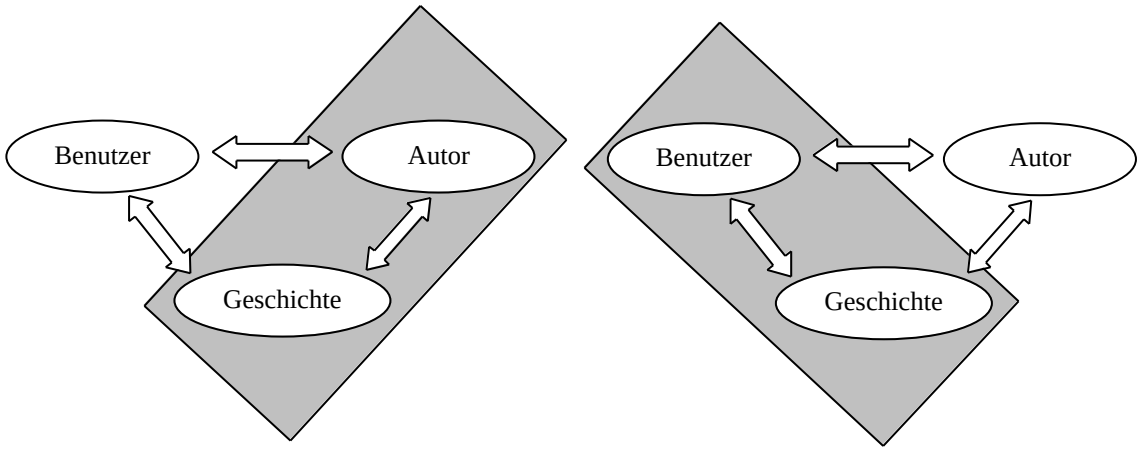
Der Begriff "Interactive Drama" wurde von Joseph Bates und seinen Kollegen in den neunziger Jahren eingeführt, um ihr Oz Projekt an der Carnegie-Mellon Universität zu beschreiben (Ryan 2001). Inzwischen wird der Begriff auf alle VR-Applikationen angewendet, die zu Unterhaltungszwecken entworfen werden (Computerspiele, Interaktive Filme, etc.) und dabei Elemente klassischer Erzählstrukturen (Märchen, Theater, Film, Novelle, etc.) verwenden.[9] Präzisierungen der Begriffe „story", „narrative" und „drama" helfen, das Gebiet besser zu fassen:

"The story, which is easily confused with the idea of narrative, will refer to the succession of actions that happen in the world represented by the narrative. The story "lives by itself", including actions that are not explicitly told, while the narrative is only what is told, and implies the notion of an explicit or implicit narrator. … By drama, we mean a special kind of narrative, where the actions are directly represented to the spectator. A novel is not a drama, because it involves the written language, which is an indirect representation (the arbitrary nature of the linguistic sign). A play, a Movie are drama, as well as a video games. Note that drama is not necessary visual: radio plays are kinds of drama. … Interactive drama refers to this specific kind of drama where the audience can modify the course of actions in the drama, thus having an active role." (Szilas 1999)

Der Begriff „Narration" umfasst damit alle Elemente und Techniken die eingesetzt werden, um den Informationsfluss vom Autor zum Publikum zu kontrollieren. Ein Problem, das in diesem Zusammenhang häufig diskutiert wird, ist der scheinbare Widerspruch zwischen "Narration" und "Interaktivität", bzw. das Dilemma „Plot vs. Interaktivität". Interaktivität erlaubt dem Benutzer

[9] Begriffe wie "Digital Storytelling", "Nonlinear Narratives", "Interactive Fiction", „Multiform Plot", etc. thematisieren die gleiche Problematik und sind meist als Synonyme aufzufassen.

den Verlauf der Geschichte zu beeinflussen. Die Handlung (Plot) ist aber, im engeren Sinne verstanden, eine vordefinierte Sequenz von Ereignissen. So gesehen zerstört Interaktivität die Handlung. Dies gilt aber nur, wenn eine sehr enge Definition des Begriffs „Plot“ angelegt wird, die sich an den Vorstellungen des Drehbuches orientiert. Um eine Geschichte zu erzählen, müssen sicher die möglichen Aktionen eingeschränkt werden, es sollte nur zugelassen werden, was „dramaturgisch sinnvoll“ ist. Andererseits ist das, was wir als den Inhalt einer Geschichte wahrnehmen, abstrakter als die Details der Handlung. Cris Crawford vergleicht eine Geschichte mit einem Haus mit vielen Fenstern; man kann sich die Geschichte von vielen verschiedenen Blickwinkeln aus ansehen, sie dennoch verstehen und als dieselbe Geschichte identifizieren.



(a) Autor/Werk (b) Konsument/Werk

Abbildung 1: Verhältnis Produzent/Konsument/Produkt

Am Beispiel „Interactive Drama“ können wir die in den zurückliegenden Abschnitten behandelten Fragestellungen verdeutlichen. Wie in Abbildung 1 gezeigt muss sowohl das Verhältnis zwischen Autor und Geschichte grundlegend überdacht werden (Abb.1.a), als auch das Verhältnis zwischen Geschichte und Benutzer (Konsument) (Abb.1.b).[10]

(a) Der Autor muss sich auf eine andere Ebene des Erzählens begeben, er kontrolliert die Erzählung nicht mehr direkt, sondern gibt nur noch den Rahmen vor, innerhalb dessen der (aktive) Konsument die Geschichte kreiert. Entscheidend ist, ob es ihm gelingt einen Rahmen zu generieren, innerhalb dessen sich interessante Geschichten entwickeln können. Die grundlegenden Prinzipien des Dramas müssen dabei erhalten bleiben, um die emotionale Wirkung und Identifikation zu garantieren.

(b) Erst, wenn die Arbeit des Autors abgeschlossen ist, tritt der Benutzer auf den Plan. Er soll nun in die Tiefenstruktur der in Echtzeit generierten Geschichte eingreifen können. Die Aktionen sollen den Verlauf in verschiedener Weise modifizieren und das Gefühl vermitteln, frei agieren zu können und nicht einem vorgezeichneten Pfad zu folgen.

[10] Das Dreiecksverhältnis Produzent/Konsument/Produkt ist im Kontext interaktiver medienkünstlerischer Arbeiten generell neu zu überdenken, da hier die Aktionen des Betrachters die Aussage der künstlerischen Arbeit mitbestimmten sollen.

Wir greifen damit die beiden zu Beginn des Beitrags, am Ende des ersten Abschnitts, gestellten Fragen wieder auf. (a) Wie müssen die Werkzeuge zur Erstellung virtueller Welten, insbesondere interaktiver Erzählformen aussehen, und (b) wie können diese VR-Applikationen für den Benutzer „offen“ gestaltet werden. Die beiden Fragen können natürlich nicht unabhängig voneinander betrachtet werden, die Mächtigkeit der Werkzeuge wird wesentlich die Qualität und Offenheit der Erzählung mitbestimmen. (Crawford 1999) formuliert die Anforderung an die Werkzeuge wie folgt:

„Our task shifts from designing the storytelling itself to designing the infrastructures in which the artists will work, and the tools they will need to do so. It's rather like the use of word processors. The designer of the word processor makes no attempt to design the content of the documents to be created with the word processor; instead, the designer defines the mental image that organizes the user's thinking while writing the document."

Derzeit sind die Autoren interaktiver Erzählungen meist noch gezwungen, ihre Vorstellungen zusammen mit Softwareentwicklern zu realisieren. Eine große Herausforderung besteht darin, Werkzeuge bereitzustellen, die den Autor selbst in die Lage versetzen, in einem offenen Entwurfsprozess neue virtuelle Erlebnisräume zu erstellen. Der Autor muss befähigt werden, Welten zu schaffen, die für den Besucher auch nach mehrmaliger Nutzung, d.h. über lange Zeiträume, spannend bleiben. Die Historie der Interaktion muss hierzu in den Geschichtsverlauf eingewoben werden. Die meisten wissenschaftlichen Ansätze versuchen derzeit, KI-Elemente in die Systeme einzubauen. Einen Überblick über die Verbindungen zwischen der KI und nicht-linearen Erzählstrukturen geben (Mateas und Sengers 1999). Spätestens wenn wir den Begriff der Narrativität im engeren Sinne verwenden, d.h. einen Autor voraussetzen und nicht bereits jeden interaktiven Ablauf als Geschichte bezeichnen, treffen wir auf die oben beschriebenen Polaritäten von “Kontrolle vs. Autonomie”. Wie viel Kontrolle übt der Autor aus, wie viel Freiheit hat das System? Nicolaus Szilas hat deutlich gemacht, dass z.B. die Aktionen der Charaktere in erster Linie motiviert sein müssen durch die Beschränkungen der Geschichte und nicht durch emotionale, psychologische oder soziale Gründe (Szilas 1999):

“*Propp's*[11] *notion of function gives an interesting light on the current trend in intelligent characters: it is not necessary, or at least not enough, to put AI in characters for interactive drama, because those characters are strongly influenced by the narrative: they should not only decide by themselves, but according to the narrative. They are accomplishing Propp's functions, rather than psychologically motivated actions. Thus, the right place of AI in interactive drama is not in character, but in narrator*.”

Die Ansätze zur formalen Darstellung und Beschreibung nicht-linearer Geschichten können grob in graph-basierte Methoden und emergente Methoden unterteilt werden. Bei graph-basierten Methoden (finite-state models) wird die Geschichtsstruktur bzw. der Geschichtsverlauf durch ein Zustandsübergangsnetz modelliert. Die Knoten des Netzes repräsentieren einzelne Situationen (Szene, Kameraeinstellung, Umgebungsattribute, etc.) und die Kanten Benutzerinteraktionen. Durch die Entscheidungen des Benutzers wird ein bestimmter Pfad durch den Graphen festgelegt. Im einfachsten Fall sind die einzelnen Knoten in sich abgeschlossene, lineare Filmsequenzen, die nur aktiviert oder deaktiviert werden können. Der erste Fall entspricht dem bekannten Vorgehen,

[11] Bereits 1928 hat Vladimir Propp hundert russische Erzählungen analysiert und daraus eine kanonische Form der Erzählung konstruiert, die auf 31 aufeinander folgenden Regeln besteht.

interaktive Anwendungen und Systeme zur Informationsvermittlung zu gestalten, indem die Informationen in interaktiv ansteuerbare “filmische Häppchen” zerlegt wird (Abbildung 2). In komplexeren Interaktionsmodellen, bzw. den dazugehörigen Graphbeschreibungen kann dann jeder einzelne Knoten wieder ein eigenes vollständiges Netzwerk sein, das einen bestimmten Aspekt der Geschichte oder eine Szene beschreibt.

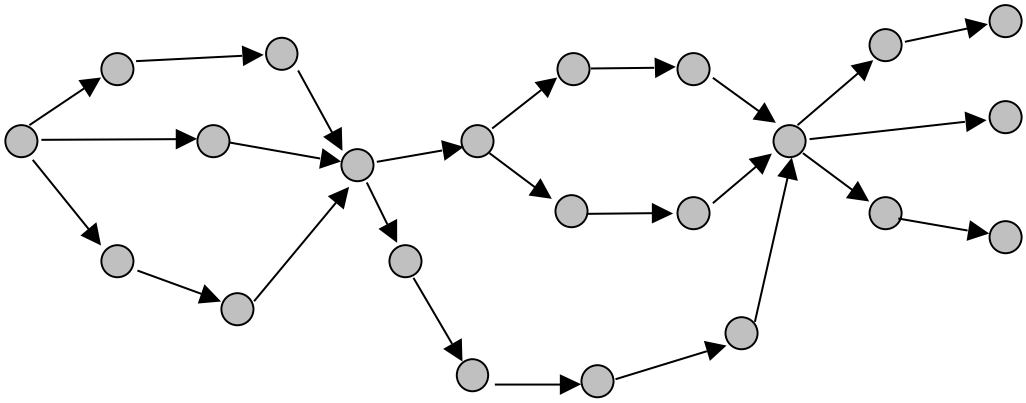


Abbildung 2: Gerichteter Entscheidungsgraph

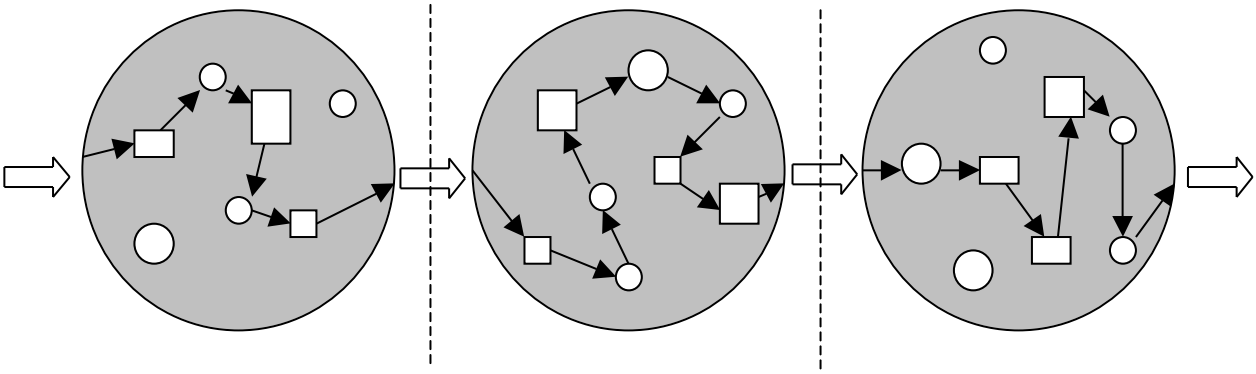


Abbildung 3: “String of Pearls”

Abbildung 3 zeigt ein anderes Prinzip, nach dem interaktive Erzählformen häufig aufgebaut sind, die sogenannte „Perlenkette“. Die linear aufeinanderfolgenden Bereiche bilden nicht die logische Entscheidungsstruktur ab, sondern die räumliche Situation und die Navigationsmöglichkeiten in der 3D-Welt. Der Übergang zwischen den einzelnen Perlen (z.B. Level-Übergang) ist determiniert und in der Regel von „Cutscenes“ begeleitet, die die Geschichte weitererzählen. Innerhalb eines Knotens kann der Benutzer frei navigieren, wobei er bestimmte Probleme lösen muss (In Abb. 3 dargestellt durch Rechtecke), um den nächsten Level zu erreichen. Andere „Erlebnisse“ sind optional, sie machen die Geschichte interessanter, sind aber für ihren Verlauf nicht entscheidend (In Abb. 3 dargestellt durch Kreise). Adventure Games wie MYST sind nach diesem Prinzip aufgebaut.

Wenden wir auf die beiden Strukturierungsprinzipien nun Cariani’s Definition der Emergenz in Bezug auf ein Modell an, dann sehen wir, dass graph-basierte Methoden sehr wohl emergent sein

können, aber lediglich für den Betrachter (Spieler), dagegen nicht für den Autor. In Bezug auf den Zuschauer lebt der Erfolg der Geschichte nicht unwesentlich davon, ob es dem Autor gelingt, die klassischen dramaturgischen Elemente zu integrieren und z.B. durch überraschende Wendungen Spannung zu erzeugen. In Bezug auf den Autor ist die Geschichte aber keineswegs emergent. In graph-basierten Modellen sind die Entscheidungen des Benutzers immer Pseudo-Entscheidungen innerhalb einer vorgegebenen Hyperstruktur. Unterschiedliche Geschichtsverläufe sind lediglich verschiedene Trajektorien in einem vom Autor vorstrukturierten Raum. Graph-basierte Methoden zeichnen sich dadurch aus, dass der Autor detaillierten Einfluss auf den Verlauf der Geschichte hat. Der Autor wird die gesamte Geschichte prädeterminieren, indem er alle möglichen Handlungsstränge und ihre Übergänge vorstrukturiert und in Gedanken durchspielt. Dennoch ist diese Vorgehensweise sehr mächtig, und die Möglichkeiten sind bei weitem nicht ausgelotet. Entscheidend ist, dass die Trennung zwischen Autor und Nutzer es erlaubt, den gesamten Erfahrungsschatz und das Wissen des Autors über Erzählstrukturen zu nutzen und mit den Erwartungen des Zuschauers zu spielen. Die Methode ist vor allem für die Produktion virtueller Umgebungen geeignet, die kinoartige Erlebnisse vermitteln. Ein wesentlicher Nachteil besteht darin, dass mit der Zahl der möglichen Entscheidungen und Interaktionen des Benutzers die Struktur der Geschichte - und damit die Zahl der vom Autoren zu denkenden Handlungsstränge - exponentiell anwächst. Vor allem aber fehlen Werkzeuge, die den kreativen Prozess des Entwerfens derartiger nicht-linearer Geschichten unterstützen. Wichtiges Element wäre dabei zum Beispiel eine „preview“-Funktion, die es dem Autor ermöglicht schon während des Entwurfs in die Erzählung einzutauchen, um so wichtigen Feedback zu bekommen, ob komplexe Verzweigungen die beabsichtigten Wirkungen haben.

Systeme, die auch in Bezug auf den Autor als emergent zu bezeichnen sind gehen sehr viel weiter und lösen dessen alleinige Herrschaft auf. Der Autor gibt lediglich einen strukturierten Rahmen vor, innerhalb dessen sich die Geschichte in der Interaktion mit dem Benutzer entwickelt. Es gibt nun zwei generell verschiedene Möglichkeiten, das System nicht zu prädeterminieren, sondern bis zu einem gewissen Grad entwicklungsfähig zu halten. Die erste - im Zusammenhang mit formalen Strukturen übliche Methode - ist die Ausweitung des Zustandsraums. Die formale Beschreibung wird so erweitert, dass es nicht möglich ist, alle Pfade durch den Zustandsraum der Geschichte zu gehen. Der “Schöpfer” des Systems definiert die Regeln, er kann aber nicht die Zustände vorhersagen, die das System einnehmen wird. Nach Festschreibung des Zustandsraumes bedeutet die Erzeugung eines konkreten Geschichtsverlauf die Navigation durch den aufgespannten Zustandsraum. Emergenz dagegen würde verlangen, dass das System konzipiert wird im Hinblick auf die Weiterentwicklung des Zustandsraums. Der Zustandsraum wäre in diesem Fall sehr viel kleiner, würde sich aber stetig verändern. Zu diesem Ansatz sind, im Gegensatz zu graph-basierten Methoden, bisher keine grundlagentheoretischen Arbeiten in der Informatik bekannt. Um die Felder Imagination und Informatik einander anzunähern, müssen wir nach unserer Überzeugung mit den Forschungen genau an dieser Stelle ansetzen.

Literatur

Ashby, W.R. (1952): Can a mechanical chess-player outplay its designer? British J. for the Philosophy of Science. 3, pp. 44-57

Borchers, J. (2000): Interaction Design Patterns: Twelve Theses. Position Paper, Workshop on Pattern Languages for Interaction Design: Building Momentum, CHI 2000, The Hague, Netherlands, April 2–3

Borchers, J. (2001): A Pattern Approach to Interaction Design. AI & Society Journal of Human-Centred Systems and Machine Intelligence, Springer-Verlag, London

Brooks, K. M. (1996): Do Story Agents Use Rocking Chairs? [http://brooks.www.media.mit.edu/people/brooks/papers/RockingChairs.html]

Cariani, P. (1989): On the design of devices with emergent semantic functions. Ph.D. Thesis, Binghamton, New York: State University of New York at Binghamton

Cariani, P. (1991). Emergence and Artificial Life. In: Artificial Life II, ed. C. G. Langton, C. Taylor, J. D. Farmer & S. Rasmussen, Addison-Wesley

Cariani, P. (1993): To evolve an ear: epistemological implications of Gordon Pask's electrochemical devices. Systems Research 1993; 10 (3):pp. 19-33

Crawford, C. (1999): Assumptions underlying the erasmatron interactive storytelling engine. In: Proceedings of the AAAI Fall Symposium on Narrative Intelligence. [http://www-2.cs.cmu.edu/afs/cs/user/michaelm/www/nidocs/Crawford.pdf]

Dodsworth, C. (1998): Digital Illusion: Entertaining the Future with High Technology, New York: ACM Press, Siggraph Series

Dunbar, K. (1998):Problem Solving. In: Bechtel, W.; Graham, G. (Eds.): A Companion to Cognitive Science. London, England: Blackwell, pp. 289 - 298

Engelbart, D.C. (1962): Augmenting Human Intellect: A Conceptual Framework. Summary Report prepared for the Airforce Office of Scientific Research, Washington 25, D.C. In: R. Packer and K. Jordan, (ed.): Multimedia, From Wagner to Virtual Reality. New York: W.W.Norton & Company, pp. 64 – 90.

Engelbart, D.C. (1991): Program on Human Effectiveness. In: J. M. Nyce, P. Kahn, Hrsg.: From Memex to Hypertext. Vannevar Bush and the Mind's Machine. San Diego: Academic Press, S. 237 - 244

Fischer, G. (2000): Shared Understanding, Informed Participation, and Social Creativity - Objectives for the Next Generation of Collaborative Systems. In: Proceedings of COOP'2000, Sophia Antipolis, France, May 2000

Fischer, G. (2001): Articulating the Task at Hand and Making Information Relevant to It. In: Human-Computer Interaction Journal, Special Issue on Context-Aware Computing, (in press). [http://www.cs.colorado.edu/~gerhard/papers/hci2001.pdf]

Goldammer, E.; Paul, J. (1996): Autonomie in Biologie und Technik. Kognitive Netzwerke – Artificial Life – Robotik. In: Ziemke, A.; Kaehr, R.: Selbstorganisation, Jahrbuch für Komplexität in den Natur-, Sozial- und Geisteswissenschaften. Berlin: Duncker & Humblot, S. 277 – 298

Günther, G. (1970): Proposal for the Continuation of an Investigation of a MATHEMATICAL SYSTEM FOR DECISION-MAKING MACHINES Under Grant: AF-AFOSR 68-1391. Department of Electrical Engineering, University of Illinois, Urbana

Kobsa, A. (1993): Adaptivität und Benutzermodellierung in interaktiven Softwaresystemen. In: O. Herzog, Th. Christaller und D. Schütt, Hrsg.: 17. Fachtagung KI. Berlin: Springer Verlag

Kovitz, B. L. (1998): Practical Software Requirements, A Manual of Content and Style. Manning Publications Co.: Greenwich, Connecticut

Licklider, J.C.R. (1960): Man-Computer Symbiosis. In: R. Packer and K. Jordan, (ed.): Multimedia, From Wagner to Virtual Reality. New York: Norton & Company, pp. 55 – 63.

Mateas, M.; Sengers, P. (1999): Introduction to the Narrative Intelligence Symposium. AAAI 1999 Fall Symposium on Narrative Intelligence. Cape Cod, MA, November 1999

Minsky, M. (1963): Steps towards artificial intelligence. In: E. A. Feigenbaum and J. Feldman, (eds.), Computers and Thought. New York: McGraw-Hill, pp. 406-450.

Trogemann, G. (2001): Computer und Kreativität. In: H. Oberquelle, R. Oppermann, J. Krause, (Hrsg.): Mensch & Computer 2001. Stuttgart u.a.: Teubner Verlag, S. 25 – 35
Rosen, R. (1985): Anticipatory Systems. Pergamon Press, New York.
Ryan, M.-L. (2001): Narrative as Virtual Reality. Immersion and Interactivity in Literature and Electronic Media. Baltimore: John Hopkins University Press
Sanchez, E.; Tomassini, M.(Eds.) (1996): Towards Evolvable Hardware, Springer-Verlag, 1996
Saunders, R. and Gero, J. S. (2000): The Importance of Being Emergent, Artificial Intelligence in: Design '00
Simon, H. A. (1996): The Sciences of the Artificial. Third ed.; The MIT Press: Cambridge, MA
Smith, B. C. (1996): On the Origin of Objects, Cambridge: MIT Press
Szilas, N. (1999): Interactive drama on computer: beyond linear narrative. Symposium. AAAI 1999 Fall Symposium on Narrative Intelligence. Cape Cod, MA, November 1999
Varela, F. (1979): Principles of Biological Autonomy. In: Klir, G. (ed.): General Systems Research. Vol II, Amsterdam: North Holland Publ.