Über Softwarekunst

Unveröffentlichtes Gespräch von Cornelia sollfrank und Florian Cramer

13. August 2003, Celle

C.S. Du beschäftigst dich seit einiger Zeit mit Softwarekunst Softwarekunst Softwarekunst, und das in verschiedenen Funktionen: als Autor, als Programmierer...

F.C.: Als Programmierer würde ich mich nicht bezeichnen, allerhöchstens als schlechten Programmierer.

C.S.: ... als Mitbetreiber und -entwickler der Plattform 'runme.org' - und als Jurymitglied beim ersten Transmediale Softwarekunst-Preis. Was begeistert dich an dem Thema?

F.C.: Der Impuls dazu kam von außen, nämlich von Andreas Broeckmann. Er hat als Netzkunstkurator und Festivalleiter der Transmediale festgestellt, dass es einen Trend gibt in den digitalen Künsten, hin zu Programmierung.

C.S.: Könntest du das bitte genauer erläutern?

F.C.: Es kann ja im Computer ohne die Hilfe von Software nichts entstehen. Ziemlich lange Zeit ist Software in den digitalen Künsten als ein Werkzeug betrachtet worden, und man hat sich relativ wenig Gedanken über dieses Werkzeug gemacht - und welchen Einfluss dieses Werkzeug auf die Kunst hat. Aber je intensiver Künstler mit dem Computer gearbeitet haben, umso problematischer wurde dieses vermeintliche Werkzeug.

C.S.: Fällt dir dazu ein konkretes Beispiel ein?

F.C.: Ja, nehmen wir mal die Entwicklung von jodi als Beispiel. Das fing an mit browserbasierter Netzkunst, die irgendwann angefangen hat, den Browser selbst zu problematisieren und auf unvorhersehbare Weise zu verwenden, zum Beispiel zum Absturz zu bringen oder einen Absturz zu simulieren. Der nächste Schritt war dann, kommerzielle Computerprogramme, nämlich Spiele zu modifizieren, dann schliesslich ganz neu zu schreiben bis sie zu dem Punkt gekommen sind, ganz einfache Computerprogramme, ganz neu in BASIC auf alten Computern zu schreiben und auch diese Programme auszustellen. Die letzte Ausstellung bestand im Prinzip nur noch aus den Quellcodes. [wann, wo, was? Referenz!!]

Das scheint mir durchaus typisch zu sein. Und Andreas machte ähnliche Beobachtungen, nämlich, dass Künstler immer mehr sich mit der Problematik von Software beschäftigt haben und zwar indem sie sie modifiziert oder sogar neu geschrieben haben. Und Medienkunstkuratoren haben einfach bisher zu wenig Aufmerksamkeit darauf gerichtet. So entwickelte er für die Transmediale im Jahr 2001 (die erste die unter seiner Leitung stattfand) die Idee, für den Wettbewerb die Kategorie Software einzuführen. Ich wurde dann von ihm eingeladen in der Jury mitzuarbeiten.

C.S.: Gibt es dazu eine Vorgeschichte? Warum hat er gerade dich angesprochen?

F.C.: Vermutlich weil ich sowohl ein Interesse für Kunst als auch für Software gezeigt hatte. Zum Beispiel hatte ich gerade einen Text geschrieben zur Ästhetik von Freier Software und Open-Source-Entwicklung. [Referenz] Das war zu der Zeit einer der ersten Aufsätze zu dem Thema. Die Konvergenz von Freier Software und künstlerischer Praxis hat mich immer interessiert. Dazu kommt, dass ich mich als Literaturwissenschaftler sehr stark für intertextuelle Poetiken interessiert habe und in der Kunst besonders für plagiatorische und appropriative Verfahren. Diese Art von Ökonomie und dieses Verständnis von Autorschaft und Urheberschaft hat eine ganz breite Parallele und eine ungeheuer starke Kultur im Bereich der Software-Entwicklung generiert. Das war natürlich sehr spannend. Und dann plötzlich zu sehen, wie dieses Feld, diese Software-Kultur wieder zurückgeschwappt ist in den Kunstbetrieb. Ich denke, das kann man mittlerweile unstrittig behaupten. 'runme.org' und 'read-me'-Reader sind ja zweifellos eine Manifestation dieser Konvergenz.

C.S.: Dazu kommen wir gleich noch. Davor würde mich noch interessieren, wie sich eure Juryarbeit gestaltete, 2001 als zum ersten Mal über diese Kategorie entschieden wurde.

F.C.: Ich habe diese Arbeit zusammen mit John Simon und Ulrike Gabriel als äußerst produktiv empfunden. John Simon ist auf dem Gebiet ein Pionier und hatte sehr klare Vorstellungen. Mit ihm habe ich auf der gleichen Wellenlänge gefunkt, d.h. wir haben uns sehr gut verstanden und waren fast immer einer Meinung. Ulrike Gabriel hingegen, die auch Künstlerin ist und mit Software-Programmierung arbeitet, war dann der Unruhepol, was sehr gut war. Wo Simon und ich problemlos übereingekommen waren, hat Ulrike wieder alles über den Haufen geschmissen. (lacht) Das war prima, denn sonst hätten wir uns zu schnell auf Selbstverständ-lichkeiten geeinigt. Die Arbeit an dem ersten Jury-Statement [Referenz] war ebenfalls sehr intensiv und produktiv, und meines

Wissens handelt es sich dabei um einen der ersten Texte über Softwarekunst.

C.S.: Was waren die zentralen Fragen in der Auseinandersetzung?

F.C.: Zuerst galt es zu klären, wie bringt man das Wort überhaupt auf den Begriff. Denn es war zum Beispiel so, dass die Ausschreibung der Transmediale nicht für Softwarekunst lautete. Nicht software art, sondern 'artistic software', also künstlerische Software, d.h. Software, die von Künstlern geschrieben wird. Die Folge war dann, dass wir ganz viele zum Beispiel Musiksequenzer-Programme oder irgendwelche Multimediaautoren-Programme eingereicht bekommen haben. Also künstlerische Software im Sinne von Software für Künstler. Und diesen Unterschied mussten wir erst einmal klarmachen. So entschieden wir uns für den Begriff Softwarekunst.

C.S.: Könnte man dann sagen, dass der Begriff Softwarekunst im Umfeld der Transmediale entstanden ist oder war er bereits vorher irgendwo anders verwendet worden?

F.C.: Ich bin mir diesbezüglich nicht ganz sicher.

C.S.: Ist denn Software als künstlerisches Produkt vorher nie thematisiert worden?

F.C.: Naja, es gab diese Ausstellung, 1970 von Jack Burnham, im Prinzip eine Vorgeschichte in der konzeptuellen Kunst. Leider ist sie nicht dokumentiet worden, aber Andreas arbeitet daran. Er kontaktierte Burnham, evtl. geben sie eine Publikation nachträglich heraus oder es wird versucht, sie zu rekonstruieren.

C.S.: Es war die Ausstellung ‚Software' im Jewish Museum im Brooklyn, richtig?

F.C.: Genau.

Ein Schritt in diese Richtung ist ja schon unternommen worden durch die Publikation der Zeitschrift 'Radical Software'. Das war in den 70er/80er Jahren ein Videokunstmagazin. Ich glaube sie kam aus Kanada - oder USA. Da tauchte der Begriff auch schon auf, wenn auch nicht in diesem engen Sinne von algorithmischer Computersoftware. Ich bin also nicht ganz sicher, ob wir den Begriff Software Art zum erstenmal verwendet haben, es hing irgendwie in der Luft. Von daher ist es gut möglich, dass er auch anderswo bereits benutzt worden war.

C.S.: Gab es dann noch andere Verdichtungen zu dieser Thematik, Auseinandersetzungen?

F.C.: In England gab es eine Gruppe um Adrian Ward, Geoffrey Cox, Alex McLean, die kurz vor uns, im Jahr 2000, die Mailingliste 'eugene' gestartet haben. Sie haben zwar von 'generativer Kunst' gesprochen, aber eigentlich auch das gemeint, (was gemeint?) und auch 'generative.net' geprägt. Und sie schrieben diesen Aufsatz, ich glaube bereits 1999, 'The Aesthetics of generative Code.'

Ausserdem hat Tilman Baumgärtel im Jahr 2000 einen Aufsatz geschrieben über experimentelle Software, wo er verschiedene experimentelle Software-Programmierung von Künstlern vorgestellt hat. Es lag einfach in der Luft, und die Transmediale hat es sozusagen institutionalisiert. Es war sicher das erste Festival, das es ganz explizit als Kategorie ausgeschrieben hat.

C.S.: Mich würde jetzt interessieren, welche Definition ihr damals für Softwarekunst gefunden bzw. festgelegt habt.

F.C.: Eine richtig gute Definition haben wir eigentlich erst beim ersten 'read-me'-Festival in Moskau entwickelt, eine ganz einfache, nämlich: Softwarekunst ist entweder Kunst, die auf algorithmischen Instruktionen basiert/ die algorithmische Instruktionen als ihr Material hat — das ist schon sehr weit gefasst und schließt z.B. auch noch Handlungsweisungen ein, die nicht direkt für den Computer geschrieben worden sind, wie z.B. Fluxuspartituren; ein gutes Beispiel dafür ist 'dotwalk' von 'socialfiction.org', ein Projekt, bei dem der Stadtraum als begehbarer Computer benutzt wird, mit Gattern, die eben nicht Transistoren sind, sondern Straßenkreuzungen. Und die andere Teildefinition war, Kunst, die sich reflexiv kulturell auf Software bezieht. Ein gutes Beispiel dafür wäre etwa ein Projekt, das beim letzten Browserday in Berlin prämiert worden ist. Dabei handelte es sich einfach um einen Fensterrahmen, und die Idee war, den Fensterrahmen als Browsermetapher zu verwenden. Du läufst mit dem Ding durch die Stadt und browst die Stadt. Damit wird klar, dass der Fensterrahmen mit einer ganz anderen Bedeutung aufgeladen wird, die erst durch die Softwarekultur geprägt worden ist. Da würde ich auch sagen, klar ist das Softwarekunst. Oder wenn jemand auf die Idee käme, ein Ölgemälde von einem Screenshot von Microsoft Word zu machen; da würde ich auch sagen, klar ist das Softwarekunst, und zwar weil es sich kulturell reflexiv auf Software bezieht. Was zum Beispiel bei der letzten Documenta für viele an digitaler Kunst Interessierte eine Enttäuschung war, war die Tatsache, dass sehr viele Arbeiten Internet- und Software-Oberflächen ganz selbstverständlich und unreflektiert integriert haben. Insofern bin ich mit unserer

Definition sehr zufrieden, nämlich einerseits Kunst, die algorithmische Instruktionen als ihr Material verwendet und andererseits Kunst, die kulturell Software reflektiert. Das ist gleichzeitg präzise und weit genug. Bei der ersten Transmediale waren wir noch nicht auf so einem genauen Begriff. Wir sind damals eher ausgegangen von Code, Instruktionscode als Material von Softwarekunst und haben dann aber gesagt, das kann ganz weit sein. Es sollte sich dabei nicht nur um Instruktionscode in einer Programmiersprache handeln, sondern z.b. auch dysfunktionaler Code, oder fiktiver Code, imaginärer Code, und haben einen ganzen Katalog von verschiedenen Code-Arten aufgelistet, z.B. auch Code als Attitude. Das war auch ein Destillat aus der Beobachtung der Kunst, die eingereicht worden war. Wir haben versucht, daraus eine Art Phänomenologie zu entwickeln.

C.S.: In einem deiner Aufsätze hast du geschrieben, Softwarekunst sei ein generativer Begriff (generic term). Was genau hast du damit gemeint?

F.C.: Damit habe ich gemeint, dass Softwarekunst nicht, wie zum Beispiel net.art Ausdruck einer bestimmten Bewegung ist, oder eines speziellen Milieus, dass man sagt, es gibt eine bestimmte Tendenz in der Kunst und die heißt Softwarekunst, und es ist eine bestimmte Gruppe von Leuten, die das betreibt, sondern es ist ein generischer Begriff, wie zum Beispiel Videokunst oder Malerei, d.h., es geht nicht um eine bestimmte Epoche, eine bestimmte Gruppe oder Bewegung; ich glaube, es macht keinen Sinn, von einer Softwarekunst-Bewegung zu sprechen; vielmehr ist es ein Überbegriff für bestimmte künstlerische Praxen.

Und wenn ich an dieser Stelle nochmal auf deine anfängliche Frage zurückkommen darf: Für mich ist die Hauptmotivation, mich mit der Thematik zu befassen, dass wahnsinnig interessante Kunst entsteht, die Softwarekunst ist. Das schließt deine Netzkunstgeneratoren ein, das schließt die Arbeit von jodi ein, die von Juan Leandre. 'dotwalk' finde ich auch ganz fantastisch. Wenn man sich den 'read me'-Reader mal durchliest und guckt was da an künstlerischen Projekten präsentiert wird, dann ist das so reich und so spannend und auch witzig. Und das ist für mich die einzige Motivation, mich mit etwas zu beschäftigen; nicht weil es als Konzept interessant ist, sondern einfach weil interessante Kunst auf dem Gebiet entsteht und für die engagiere ich mich. Und ich versuche nicht, rein deskriptiv damit umzugehen, sondern eben auch theoretische und Begriffsarbeit zu leisten und zu versuchen, daraus Konzepte abzuleiten.

C.S.: Es wurde ja immer wieder diskutiert, ob es überhaupt notwendig und sinnvoll ist, diesen Begriff zu verwenden.

Inzwischen wird ca. drei Jahre damit gearbeitet. Was würdest du sagen, was die Verwendung dieses Begriffes bewirkt oder zum Vorschein gebracht hat?

F.C.: Ich glaube, die Funktion des Begriffes Softwarekunst ist, dass man auf digitale Kunst oder Medienkunst mit einer anderen Perspektive blickt. Ein gutes Beispiel wäre etwa der 'Webstalker' von I|O|D, der immer als ein Klassiker der Netzkunst gegolten hat, der aber auch ein kritisches Softwarekunst-Projekt ist und sich damit auseinandersetzt, wie Software überhaupt einen Zugriff auf bestimmte Informationen leistet, wie Software selbst eine eigene Ästhetik hat, die determiniert, wie man Sachen wahrnimmt. Wenn man sich ansieht, worum es I|O|D ging, was die Überlegungen von Matthew Fuller und Graham Harwood waren, wird man feststellen, dass sie sich vielmehr für Software interessiert haben als für Netze. Das war für sie damals schon viel zentraler als Auseinadersetzung, d.h. wenn man an so einem Punkt dann den Begriff Softwarekunst einführt, dann kann man an so eine Arbeit wie den 'Webstalker' anders herangehen, sie anders angucken und genauer fokussieren, welche Rolle spielt eigentlich die Tatsache, dass es Software ist und man sich mit Software auseinandersetzt? Und ich glaube, dass es eine sehr wichtige Frage ist, mit der man etwas gewinnt und nicht etwas verliert.

C.S.: Das Panel der Transmediale 2003 zum Thema Softwarekunst stellte im Titel die Frage 'Softwarekunst: Neue Kunstrichtung oder kuratorische Fiktion?' Da klingt die kuratorische Fiktion vermeintlich negativ und ist auch so gemeint, als Gegensatz zum Positiven einer neuen Kunstrichtung. Ich würde das aber gar nicht so sehen wollen. Denn eine kuratorische Fiktion kann ja durchaus produktiv wirken. Einfach eine Setzung zu machen und abzu-gleichen.

F.C.: Ja, so sehe ich das auch. Es gibt einige Begriffe, wie Impressionismus, die auch kuratorische Fiktionen sind, die nicht von den Künstlern selbst ins Spiel gebracht, nie verwendet worden sind, sondern von Kritikern konstruiert wurden. Andere Beispiele dafür wären Kubismus oder auch Konzeptkunst. Und genauso verhält es sich mit der Softwarekunst. Das ist kein Terminus, der von Künstlern offensiv nach vorne gebracht worden wäre, oder unter dem sich Gruppenidentitäten formiert hätten, sondern das ist eine Kreation von Kritikern und Kuratoren. Das finde ich aber keineswegs problematisch.

C.S.: Jetzt haben wir schon mehrmals 'read-me' und 'runme.org' erwähnt. Zuerst zu den beiden 'read-me'-Festivals. Ich würde gern mehr über die Hintergründe erfahren und natürlich gern alles über die geheime Vorgeschichte wissen.

F.C.: Es gab bisher zwei 'read me'-Festivals, das erste fand in Moskau, das zweite in Helsinki statt. Zur geheimen Vorgeschichte gehört das Festival 'digital is not analog' in Bologna, das von 0100101110101101.ORG zum Thema 'Kunst und Hacking' veranstaltet wurde. Durch das Thema war zwangsläufig vorgegeben, dass Software eine wichtige Rolle spielte. Die Arbeit von Juan Leandre habe ich dort zum erstenmal gesehen und fand sie ganz phantastisch. Ausserdem waren dabei: Adrian Ward, nullpointer.org aus England, ein guter Programmierer, der experimentelle Software schreibt, eine Vertreterin von Netochka Nezvanova, Amy Alexander, die Yes Men, snafu, Negativland. Wir waren untergebracht in einer Villa in den Hügeln vor Bologna, wo sich immer zwei Personen ein Zimmer teilten. Ich war mit Alexei Shulgin untergebracht. Wir waren eines Tages gezwungen einen längeren gemeinsamen Spaziergang zu unternehmen, weil unser Abholservice nicht geklappt hatte und es kein Essen und Trinken im Haus gab. Also beschlossen wir, zusammen den Hügel runterzulaufen, um ein Frühstück aufzutreiben. Dabei haben wir uns etwas verlaufen und es zog sich. Das war im Juli 2001, also bereits nach der Transmediale. Und ich war Feuer und Flamme für Softwarekunst, immer noch sehr euphorisiert von der Juryerfahrung. Alexei hingegen war sehr skeptisch. Auf diesem Spaziergang also diskutierten wir über Softwarekunst, und Alexei wollte von mir wissen, warum wir das so vertreten würden. Ich habe ihm das damals so auseinandergesetzt, wie ich es gesehen habe, und das muss bei ihm gearbeitet haben. Und ich glaube das ganze Festival und die Leute die da gewesen waren hat uns allen Spaß gemacht und ihm eben auch. Diejenigen, die in der Jury des ersten Festivals waren, nämlich rtmark, Amy Alexander, Alexei, Olga und ich, das waren alles Teilnehmer von dem Bologna-Festival. Eigentlich ist es ein kryptisches Destillat aus dem Bologna-Festival. Alexei war damals sehr frustriert darüber, was aus der Netzkunst geworden war. Und er dachte darüber nach, wie es weitergehen könnte. Net.art hatte sich weitgehend aufgelöst und offensichtlich war für ihn Software der nächste Punkt. Und er initiierte ein Softwarekunst-Festival. 'read_me 1.2' fand dann im Mai 2002 in Moskau statt. Und es war sehr gut, dass es kein anderes Thema als Software gab, also nicht nur eine Kategorie in einem allgemeinen Festival. Ausserdem war es toll, dass es von unten kam, also von den Künstlern selber kreiert war. Bei diesem ersten Festival hatten wir einen Juryprozess, der durchaus noch konventionell war.

C.S.: Schon wieder Jury? Gab es da auch einen Wettbewerb und eine Preisverleihung?

F.C.: Ja, es gab einen Preis. Es war von dem Macros Center und dem russischen Kulturministerium gesponsort worden. Es gab

insgesamt drei Preisträger, drei gleiche Preise, Geldpreise: 1. Deskswap, 2. Experimenteller Wordprocessor, ein animiertes dreidimensionales Schreibprogramm, 3. ??.

C.S.: Wie und wann ist dann die Plattform 'runme.org' entstanden?

F.C.: Insgesamt waren wir nicht so zufrieden mit dem Juryprozess. Bei einer Bootsfahrt auf der Moskwa - auf einer inoffiziellen Fähre - saßen wir auf dem Deck und überlegten, was wir anders und wie wir es besser machen könnten. Dabei enstand die Idee, anstatt des klassischen Juryprozesses, eine Plattform im Internet zu schaffen, eine offene Plattform, eine Netzdatenbank für Softwarekunst, wo wir sowohl als Jury und Experten eigene Projekte gezielt platzieren, aber auch Künstler selbst ihre eigenen Projekte einbringen können. Es sollte alles gesammelt werden und dann in einem zweiten Schritt gewissermaßen die Crème abschöpfen. Über das, was in der Datenbank steht, wollten wir dann Features schreiben und die Arbeiten, die wir am besten finden, sollten bei dem nächsten Festival gezeigt und evtl. auch die Künstler eingeladen werden.

C.S.: Das heißt, die offene Plattform sollte eure Selektion nicht vollständig ersetzen!?

F.C.: Der Plan war, erst in die Breite zu gehen, die offene Plattform bereitzustellen und dann aber aus der Plattform zu filtern.

C.S.: Ich bin mir nicht sicher, ob ich die Praxis der Selektion durch Experten, des Wettbewerbes und der offenen Kokurrenz als besonders produktiv empfinde. Wieso haltet ihr daran fest - in einem selbstorganisierten Zusammenhang?

F.C.: Ich finde es gut und sinnvoll und nicht prinzipiell schlecht, Preise zu vergeben.

C.S.: Aber steht das nicht grundsätzlich im Widerspruch zu der Open-Source-Kultur, auf die du verweist? Da geht es doch auch nicht um einzelne 'Autoren', die etwas Beonderes geleistet haben, sondern mehr um das gemeinsame Arbeiten an Projekten.

F.C.: Ja eben, es geht um Projekte. Und unsere Preise gingen nie an Einzelpersonen oder Autoren, sondern immer an Projekte. Wenn es nur einen Macher gibt, dann ist es eben EIN Autor, der den Preis bekommt, aber es sind durchaus auch schon Gruppen ausgezeichnet worden.

C.S.: Wieso findest du das Prinzip mit der Preisvergabe sinnvoll?

F.C.: Ich finde es sinnvoll, weil die Festivals ein Budget haben, und es sollte dafür verwendet werden, Künstler zu finanzieren. Das ist eine Einnahmequelle für Künstler. Als Literatur-wissenschaftler kenne ich das aus dem Literaturbetrieb. Lyriker zum Beispiel finanzieren sich fast ausschließlich über Stipendien und Preise, und so ähnlich ist es ja auch im Medienkunstbetrieb. Wenn so ein Preis anständig dotiert ist, dann kann ein Künstler ein halbes Jahr davon leben und Projekte finanzieren. Und ich finde es besser, den Künstlern das Geld zu geben, als es in Festivalorganisation zu stecken.

Wenn du jetzt auf Freie Software ansprichst und Hackerkultur: die wünschen sich das. Es wird auf 'slashdot' immer mal wieder diskutiert, warum es keine öffentlichen Preise oder Fördergelder für freie Entwickler gibt. Gerade nachdem die Dotcom-Bubble geplatzt ist, können kaum noch Leute in Firmen Freie Software schreiben. Das heißt, die Ökonomie des Kunstbetriebes könnte auf diese Weise positiv in die Ökonomie der freien Software einfließen – und nicht nur der Kunstbetrieb von der freien Software lernen. Das System von Stipendien und Preisen wäre durchaus ein sehr gutes System für Freie-Software-Entwicklung. Die Leute die das machen, sind im Prinzip freie Künstler. Nur wenige Stars wie Linus Thorvalds sind angestellt; die breite Masse kämpft dafür, ohne bezahlt zu werden.

C.S.: Als Künstlerin, die viele Jahre von Stipendien etc. gelebt hat, sehe ich das natürlich einerseits ein, was du sagst. Andererseits finde ich, dass die Preise und Stipendien als wichtiger Bestandteil einer Künstlerbiografie erheblich dazu beitragen, ein Kunstsystem zu kultivieren, das Konkurrenz fördert zwischen den Künstlern und damit Neid und Eifersucht als äußerst kontraproduktive Faktoren, anstatt zum Beispiel die Idee von Kollaboration, was für mich die Basis einer anderen Ökonomie ist. Wie geht das für dich zusammen?

F.C.: Die Frage ist nur, wie könnte man es anders machen? Wie schüttet man dieses Geld aus? Es einfach nach dem Gießkannenprinzip zu machen, bringt es auch nicht. Als jemand der die Kritiker- oder Rezipientenposition einnimmt, bin ich durchaus der Meinung, dass zur Vermittlung auch Auswahl gehört. Ich will klarmachen, welche Arbeiten ich gut finde, besser als andere.

Ich kann es verstehen, dass es bei der Netzkunst, die sich als Soziotop generiert hat, eine Problem war, andererseits hatte ich aber auch mit der Netzkunst immer Probleme sofern sie sich als dieser egalitäre Soziotop definiert hat. Und ich finde es nicht verkehrt, jetzt retrospektiv zu sagen, *die* Sachen waren

interessant und *die* waren weniger interessant. Was mich damals schon eher abgeschreckt hat und was ich auch heute noch eher als kuriose Fußnote dieser Geschichte sehe, sind zum Beispiel die digitalen Städte oder internationalen Städte. Die sind zwar kulturhistorisch interessant, weil da ganz irrsinnige Utopien fortgeschrieben worden sind, wie etwa freimauerische oder rosenkreuzerische Utopien von André oder Campanella aus dem 17.Jahrhundert. Die sind quasi nochmal neu erfunden worden, aber auch mit demselben Irrsinn. Auch mit derselben Nichthaltbarkeit.

C.S.: Aber diese digitalen Städte haben sich doch nicht als Kunst-Projekte verstanden?

F.C.: Doch, ich glaube schon. Zumindest hatten sie was Verblasenes.(lacht)

Was mir noch zu dem Preissystem einfällt: Ich glaube nicht, dass das System als solches verkehrt ist, vielleicht ist es nur manchmal problematisch, wer diese Preise vergibt. Für mich ist ein positives Gegenbeispiel der Literaturbetrieb. Vielleicht ist bei der digitalen Kunst einfach das Problem, dass es zu wenig Personen und Institutionen gibt, die das entscheiden, während es sich im Literaturbetrieb eingebürgert hat - gerade in Deutschland ein Aspekt, der mir das Land sympathisch macht - dass eigentlich jede Kleinstadt einen Literaturpreis und ein Literaturstipendium hat. Es gibt Stadtschreiberstellen und wahnsinnig viele Literaturhäuser. Und in der Summe ist das gut. Die Entscheidungen und die Auswahl die da getroffen wird, die ist gut. Und es hilft gerade Schriftstellern, die nicht Mainstream sind. Es ist ein Fördersystem und ein Auswahlsystem, dass es gerade in Deutschland experimentellen Dichtern ermöglicht, zu überleben. Leute wie Oskar Pastior leben nur davon. In den USA ist es so gereglt, dass solche Schriftsteller meistens an Universitäten lehren. Ja, und wenn jede Kleinstadt ihren Netzkunst- und/oder Softwarekunstpreis hätte, dann hätten wir auch die Probleme nicht. Es gibt einfach nur drei, vier Festivals, die solche Preise vergeben, und das ist zu wenig und mag die Entscheidungen verzerren.

C.S.: Lass uns zurückkommen zu den 'read me'-Festivals. Die Plattform 'runme.org' ist also zwischen den beiden Festivals eingerichtet worden und hatte welche Funktion?

C.F.: Ja, genau. Nach dem ersten Festival kam noch Alex McLean als weiteres Mitglied in die Expertengruppe, und der hat die Plattform auch programmiert. Die Plattform ist von der Funktionalität und von der Ästhetik sehr vergleichbar mit Plattformen der freien Softwarekultur. Wir haben uns damit - ganz

bewusst - an diese Kultur angelehnt. Die Plattform war dann mit einem Vorlauf von ca. 9 Monaten vor dem zweiten Festival fertig. Das, was auf 'runme' eingestellt worden war, sowohl von uns als Kuratoren als auch von den Künstlern selbst, bildete das Ausgangsmaterial für das zweite Festival. Dieses Material haben wir gefiltert und die Projekte, die wir gut fanden, haben wir gefeatured, d.h. Texte darüber geschrieben, die in dem Reader erschienen sind. Der Reader ist ein Destillat aus der Website, ein direktes.

C.S.: Hast du einen Überblick darüber, wie viele Projekte sich jetzt auf der Plattform befinden? Und wie viele davon habt ihr hervorgehoben?

F.C.: Es sind auf jeden Fall über 100 Projekte auf der Plattform und herausgezogen haben wir ca. 30.

C.S.: Projekte auf die Seite hochzuladen ist vollkommen unzensiert möglich?

F.C.: Ja, und es hat gut geklappt. Was es allerdings gibt, ist eine hierarchische Unterscheidung zwischen den Projekten, die einfach drauf gestellt sind von den Mitgliedern und denjenigen, die gefeatured werden.

C.S.: Inwiefern lehnt ihr euch nun an die freie Softwareszene an?

F.C.: Eine der Websites, an die wir uns angelehnt haben, ist 'freshmeat.net', die wahnsinnig populär ist, das Generalverzeichnis von Freier Software überhaupt. Vergleicht man 'runme.org' mit 'freshmeat.net' wird man feststellen, dass die Oberflächen ganz ähnlich sind, mit einer weblog-ähnlichen Oberfläche, wo die neuen Projekte immer ganz oben auf der Homepage erscheinen und ansonsten in der Datenbank liegen, die man durchsuchen kann, und es gibt Kategorien und Stichwörter, nach denen man auch suchen kann und unter denen die Projekte auftauchen. Die Kategorien können bei 'runme.org' jederzeit modifiziert werden. Jeder, der ein Projekt einträgt, kann eine neue Unter- oder Hauptkategorie generieren. Und es gibt auch so eine typografische Wolke von Kategorien, die in Beziehung zueinander gesetzt sind. Dann gab es noch eine kleinere Website, die auch gar nicht so bekannt ist, die uns sehr stark inspiriert hat: 'sweetcode.org'. Die haben wir auch verlinkt.

'sweetcode.org' wird von jemandem gemacht, der sich daran gestoßen hat, dass oft gesagt wird, Freie Software sei nicht originell und oft nur eine Reimplementation von Konzepten, die bereits vorher proprietär entwickelt worden sind, wie Linux und

GNU als freie Implementationen von UNIX. Und UNIX ist eben doch letztendlich ein proprietäres Betriebssystem. Deswegen hat er eine Website für Softeware gemacht, die originell ist, d.h. freie Software, die aber in irgend einer Form konzeptuell interessant ist, überraschend. Man kann diesen Service auch als Mailingliste abonnieren, so dass man über die jeweils neuen Projekte informiert wird. Es sind sehr wenig Projekte auf der Website und es kommt nur ca. eins pro Monat neu hinzu, aber wirklich tolle Sachen, die wir dann prompt für 'runme' und 'readme' abgeschöpft haben.

C.S.: Das klingt vielversprechend. Könntest du ein Beispiel von da vorstellen?

F.C.: Ja, da gibt es eine Software, die auf normalen Röhrenmonitoren Muster generiert. Röhrenmonitore strahlen ja und senden damit in einem elektromagnetischen Frequenzbereich, u.a. im UKW-Bereich. Das Programm macht quasi einen UKW-Sender mit Hilfe von Bildschirmgrafiken, d.h. wenn man ein Radio direkt vor den Bildschirm stellt, spielt es diese als Melodien ab. Ein tolles Projekt; man hat erst abstrakte Muster, monochrome Streifen, die sich bewegen auf dem Bildschirm und das Radio davor spielt eine minimalistische Melodie wie "Alle meine Entchen oder so". [Wie heisst diese Software? C.S.]

C.S.: Ist das nun ein intendiertes Kunstprojekt, oder habt ihr als Kuratoren es in den Status erhoben?

F.C.: Ein Grund, warum ich es sehr wichtig finde, Kuratoren zu haben, ist genau der: Wir können Projekte auf die Plattform ziehen, von Programmierern zum Beispiel, die das, was sie tun selbst nie als Kunst betrachten würden. Wenn man nur Projekte hätte, die von den Autoren selbst auf die Plattform gestellt würden, würden all diese tollen Projekte fehlen. Viele der Projekte kommen aus dem Kontext von Hackerkultur oder der freien Software, und wir fanden sie aber als Softwarekunst interessant und haben sie integriert, wir z.B. diese beschriebene UKW-Projekt.

C.S.: Mit einigem Vorlauf der Plattform fand dann das zweite 'readme'-Festival im Mai 2003 in Helsinki statt. Inwiefern unterschied es sich vom ersten? Inwiefern hatte die Plattform dazu beigetragen, die Qualität zu verändern?

F.C.: Die Funktion des Festivals war hautpsächlich die, einen öffentlichen Diskurs zu dem Thema zu generieren. Es gab klassische Panels mit Vorträgen. Insofern hat es sich nicht so stark vom ersten Festival unterschieden. Der wichtige Unterschied

bestand in den künstlerischen Arbeiten, die fokussiert worden sind. Da war 'runme.org' der Katalysator oder das Auswahlmedium, das uns die Arbeiten geliefert hat, die dann präsentiert worden sind. Zusätzlich gab es eine kleine Ausstellung, die Alexei Shulgin gemacht hatte, im Vorraum des Konferenzsaales. Sie bestand aus sehr witzigen Projekten, war insgesamt sehr stimmig und war eine der wenigen guten digital- /Computerkunst-Ausstellungen, die ich bisher gesehen habe. Das Festival fand statt in einer Kunsthochschule in Helsinki, und er hat einfach alle möglichen Computer genommen, die es gab in dieser Schule. Das ergab nicht den üblichen Einheitslook. Und er hatte sehr einfache Arbeiten ausgewählt, die sofort verständlich sind, die etwas witziges, Buntes hatten. Das war sehr erfolgreich.

Man muss auch dazu sagen, dass das Festival eher eine Kreation von Alexei und Olga ist. Die Festivalgestaltung, auch die ganze Organisation haben wirklich die beiden gemacht, die Auswahl der Panels. Was das Expertenteam gemacht hat, ist eben der Filterprozess auf runme.org'.

C.S.: Jetzt war schon öfter die Rede von einem Expertenteam. Wird das immer aus den gleichen Personen bestehen bleiben, oder ist es vorgesehen, dass die Experten auch rotieren?

F.C.: Nein, das ist eine Mafia. (lacht.) Es können natürlich andere Leute hinzukommen. So sind zum Beispiel Pit Schultz und Thomax hinzugekommen im letzten Jahr; die haben aber nicht viel gemacht. Thomax hat ein Feature über das GNU-Projekt geschrieben, aber das war's auch schon. Eigentlich hoffen wir schon noch andere Leute zu finden, die sich aktiv einbringen.

C.S.: Nochmal zur Plattform: Das Modell ist angelehnt an die Kultur der Freien Software. Wie wichtig ist diese Infrastruktur, die ja ein Modell für Selbstorganisation ist? Sie bündelt und zentralisiert ohne zu institutionalisieren? Bei dieser Frage habe ich im Hinterkopf die Netzkunst, bei der es ja genau das nicht gab: eine gebündelte, aber offene Plattform zum Austausch.

F.C.: Der Punkt ist ein ganz anderer, nämlich, dass die Softwarekunst eben keine Bewegung ist, sondern tatsächlich eine Kuratoren- und Kritikerkonstruktion. Und genau das ist 'runme.org' auch. Es handelt sich dabei um Softwareprojekte, von denen einige als Kunst gelabelt werden und andere nicht. Und wir ziehen das alles zusammen. Es handelt sich dabei einfach um eine Plattform von unten, weil 'readme' und runme' sehr kleine, selbstorganisierte Initiativen sind, und keine großen Festivals wie zum Beispiel 'ars electronica'. In unserem Expertenstab sitzen auch keine Großkuratoren drin. Aber es hat eben einen

analytischen und keinen synthetischen Zugriff. Es geht nicht um die Konstruktion einer Identität 'Softwarekunst', sondern es geht darum, zu sehen, was als Softwarekunst interessant ist und das zu bündeln und zugänglich zu machen.

Im Hintergrund ist tatsächlich auch immer noch die Auseinandersetung mit dem Juryprozess. Das erste 'readme'-Festival in Moskau mit einem noch relativ traditionellen Juryprozess hat sich von der Transmediale dadurch unterschieden, dass alle eingereichten Projekte auf einer Website publiziert wurden. Damit war es auch sehr transparant für das Publikum.

Bei der ersten Transmediale, bei der ich als Jurymitglied mitgearbeitet habe, gab es immer wieder Fragen danach, warum das eine auf die Shortlist gesetzt wurde und jenes nicht. Es wäre den Leuten viel klarer gewesen, wenn wir damals auch so eine Website kreiert hätten. Aber wir sind einfach nicht auf die Idee gekommen. Man hätte ganz klar erkennen können, dass wir alles, was damals halbwegs interessant war, auf die Shortlist gesetzt hatten. Der Grund für die wenigen guten Einreichungen war sicherlich, dass das Feld noch so undefiniert war und wir einfach sehr viele Arbeiten auf Kunsthochschuleniveau hatten, die einfach nicht interessant waren.

C.S.: Man kann eure Entwicklung also so zusammenfassen, dass ihr im ersten Schritt, alles veröffentlich habt, was eingereicht wurde, dann die Plattform geschaffen habt, als offenes Archiv.

F.C.: Es war uns wichtig, auch die Entscheidungsprozesse auf die Website zu verlegen und zusätzlich klarzumachen, dass wir selbst auch noch Projekte featuren, die nicht eingereicht worden sind.

C.S.: Du betontest immer wieder, dass 'runme' angesiedelt ist in einem Bereich, in dem sich Kunst und Freie Software überlappen, dass es sich aus beiden speist. Dazu habe ich noch viele Fragen. Eine wäre, warum sollte es für Programmierer von Interesse sein, sich in diesen Kontext zu begeben? Was haben sie davon, plötzlich als Kunst gelabelt zu werden und einen netten Text über ihre Arbeit zu lesen?

Der Kunstkontext neigt ja sehr dazu, sich immer wieder neue Bereiche ausserhalb zu suchen, aus denen er sich speist und am Leben hält, die er sich einverleibt. Das ist in der Regel aber immer eine einseitige Geschichte, eine Oneway-Kommunikation. Oder fließt auch etwas zurück?

F.C.: Dazu kann ich nur berichten, was ich von Alexei weiss, der derjenige ist, der mit den Programmierern in Kontakt tritt. Die

Haltung ist die, gerade wenn es sich um Freie Software handelt, dass die Leute sagen, macht, was ihr wollt, solange ihr euch an die GPL haltet. (lacht) Das ist ja mit eine Codifizierung von Freier Software, dass der Kontext nicht determiniert wird. Du kannst sie für jeden Zweck einsetzen. Du kannst freie Software auch für eine Atomraketensteuerung einsetzen. Das ist ganz explizit so vergesehen. Würde man eine Klausel einfügen, die z.B. den Einsatz der Software für Atomraketensteuerung verbieten würde, wäre es nach der Open-Source-Definition oder den Debian-Free-Software-Guidelines keine freie Software mehr und würde dann in der Destribution von Debian in den Non-Free-Bereich abwandern. Das heißt, dass aus diesem Grund Freie Software immer in den Kunstkontext reingenommen werden kann. Das ist ein Begriff von Freiheit, der in dem Terminus 'Freie Software' steckt. Deswegen haben die kein Problem damit. 'read-me' und 'ruunme' wurde auch schon bei 'slashdot' positiv erwähnt und verlinkt. Das wird gar nicht negativ gesehen. Ein alter Wunschtraum von mir ist, auch die Freien Softwareprojekte mit netzkünstlerischen und softwarekünstlerischen Praxen in Verbindung zu bringen.

C.S.: Warum? Was versprichst du dir davon?

F.C.: Ich bin der Meinung, dass Freie Software ästhetische und künstlerische Defizite hat, dort wo sie meint, ästhetisch und künstlerisch zu sein. Darüber müssen wir nicht lange reden. (lacht.)

C.S.: Doch, ich hätte noch gern ein Beispiel für das, was du da behauptest.

F.C.: Naja, nehmen wir mal die 'Art&Beauty'-Workshops des CCC. Wenn die Leute, die das betreiben, mal 'runme' angucken würden, könnten sie eine Menge lernen. Oder ein anderes Beispiel, wo künstlerische und ästhetische Entscheidungen eine große Rolle spielen, ist die Kreation freier Desktops, wie KDE (?) und gnom, die nun wirklich eine Enttäuschung sind. Ich finde Freie Software hat eine ganz fantastische Ästhetik, dort, wo es um ganz klassiche UNIX-Kommandozeilen geht. Mein eigener Computer besteht aus einem schwarzen Bildschirm mit einem weißen Kommandozeilenprompt drauf. Sonst nichts. Das ist toll und sehr weit entwickelt. Da, wo dann aber versucht wird, fast im Sinne von Bauhaus, eine Alltagsästhetik zu entwickeln, freie Software benutzbar zu machen für viele Leute, da wird dann plötzlich auf diese abgestandenen Microsoft- und Apple-Paradigmen zurückgegriffen. Das finde ich sehr enttäuschend. An dieser Stelle könnte ich mir eine Menge produktiver Interferenz vorstellen. Ich hatte mal die Idee, des 'jodi'-Desktops. Also einen Linux-Kernel mit einem 'jodi'-Desktop drauf. Das ist eine

alte Utopie von mir, die ich bereits 1999 auf einem Panel bei der WOS geäußert habe. Das ist natürlich eine Utopie: Die Sprachen, die sich Netzkunst/digitale Kunst in den letzten Jahren erarbeitet hat, auch mal in die freie Software-Entwicklung einfließen und sich dort produktiv bemerkbar zu machen.

C.S.: Ich glaube auch, dass das eine Utopie bleiben wird, weil sich - nach meiner Erfahrung - die Entwickler Freier Software oft als omnipotent betrachten und absolut nicht das Gefühl haben, an irgend einer Stelle, z.B. den ästhetischen Qualitäten, defizitär zu sein.

F.C.: Es gibt schon so Bereiche, also zum Beispiel... Bleiben wir bei den Desktops. Ich glaube, das ist eine Entwicklung, die erst einmal gründlich auf die Schnauze fallen muss, bevor sie sich eines Besseren besinnt.

C.S.: Aber was soll das heißen, auf die Schnauze fallen? Die User schlucken doch ganz brav alles, was ihnen vorgesetzt wird. Ich denke da auch an die Oberfläche von Mac OSX. Die schlimmste Zumutung, die ich überhaupt kenne. Sie bereitet mir körperliches Unwohlsein.

F.C.: Es gibt auch Gegenbeispiele, die vielleicht etwas esoterisch sind, z.B. das Squeak-System. Das geht zurück auf den Ur-Desktop, den Alain Kaye damals bei Xerox Park entwickelt hat. Es basiert auf Smalltalk, dieser objekt-orientierten Programmiersprache. Das gibt es mittlerweile auch als Freie Software unter einer freien Lizenz. Es unterscheidet sich von anderen Desktops dadurch, dass es frei programmierbar ist. Du bekommst keine fertigen Anwendungen, sondern kannst dir, ähnlich wie unter der UNIX-Kommandozeile, nach dem Baukastenprinzip, aus Modulen eigene Anwendungen zusammenstellen. Ich weiss, dass das eine große Dynamik hat und viele Netzkünstler mit und an dem System arbeiten. Walter van der Cruijsen zum Beispiel ist jemand, der seit Jahren mit Squeak arbeitet. Ich war in diesem Jahr bei der E-Poetry-Konferenz in den USA. Ein Veteran auf dem Feld ist Jim Rosenberg. Er macht Gedichte in Form von Diagrammen und hat schon in den 70er und 80er Jahren für John Cage gearbeitet und dessen Computerprogramme entwickelt. Rosenberg arbeitet auch mit Squeak. Bei vielen Künstlern, die schon in der Programmierung drin sind und die immer ein Werkzeug gesucht haben, mit dem sie sich ihre eigenen Oberflächen bauen können, hat sich eine wahnsinnige Dynamik entwickelt. Und gleichzeitig hat es eine gewisse Dynamik in der Freien Sofwareszene. Ich könnte mir vorstellen, dass so in fünf Jahren Squeak abhebt und KDE und Gnom verdrängt. Auch weil es konzeptuell besser ist. Da hätte man schon etwas, woran Künster substantiell mitgearbeitet haben, und

was auch eine ganz andere Kultur und Ästhetik hat, als das was wir bisher als Desktop-Metapher kennen.

C.S.: Ich zitiere Matthew Fuller, der in dem Reader darüber schreibt, dass sich in Software gesellschaftliche Verhältnisse manifestieren und dass Software diese Verhältnisse auch reproduziert. Bei der Freien Software ist eine Art Community auszumachen, die das betreibt. Nun frage ich mich, kann man das auf die Softwarekunst übertragen? Bildet 'runme.org' eine Community und aus welchen Leuten besteht sie? Was sind ihre Motivationen?

F.C.: Es handelt sich dabei einfach nicht um eine homogene Szene. Diejenigen, die als Künstler und Programmierer von sich aus Projekte auf runme stellen, kommen fast alle aus dem Netzkunstkontext. Und das sind bekannte Namen, die wir alle kennen. Und die anderen Projekte ziehen wir als Experten aus der Hacker oder Freien Softwareszene. Es ist einfach ein Spezifikum der 'runme'-Betreiber, dass wir gern mit Software rumspielen. Ich selbst bin Freier-Software-Fan und lese alle Ankündigungen auf 'freshmeat.net'. Wenn mich da was interessiert, lade ich mir das runter und kompiliere das und probiere es aus. Und Amy Alexander ist auch so jemand. Aber ich vermute, du willst darauf hinaus, dass es sich um den männlichen Hackertyp handelt, den man auch bei CCC findet. Das ist sicher richtig.

C.S.: Für euch ist es wichtig, Vermittlunsgarbeit zu leisten. Gerade wenn ihr Texte schreibt über Projekte, die nicht aus der Kunstszene kommen, hat das eine stark vermittelnde Komponente. Was ist eure Motivation?

F.C.: Die Motivation ist, dass solche Sachen überhaupt wahrgenommen und nicht vergessen werden. Ich glaube, das ist auch eine wichtige Funktion, die der Kunstbetrieb einnehmen kann: Vieles, was an phantastischen und witzigen Arbeiten gemacht wird, würde sonst völlig unter den Tisch fallen. Die Sachen sind für einen Distributor, der eine Linux- oder BSD-Distribution zusammenstellt, die vorallem funktional sein soll, nicht interessant. Wir mögen diese Sachen einfach und finden sie sollten irgendwo präsentiert werden. Eine Rolle spielt auch, dass es diese Art von Ästhetik auch in der Netzkunst gegeben hat.

C.S.: Worüber wir noch gar nicht gesprochen haben, sind jetzt einzelne Bereiche der Softwarekunst. Ganz besonders aktiv bist du ja zum Beispiel bei den Codeworks. Könntest du darauf etwas eingehen?

F.C.: Ja, dabei handelt es sich um eine Art E-Mail-Netzkunst, die mit Fragmenten von Programmiersprachen und Netzwerkprotokollen, visueller Poesie und englischer Sprache arbeitet. Es war mir immer besonders wichtig, diese Arbeiten auch in den Softwarekunst-Kontext hineinzuziehen. Netzkunst hat schon immer ganz stark Hacker-Ästhetik appropriiert: sei es ASCII Art, Spielemodifikation oder Viren. Was wir dann gesehen haben war, dass es in der Hackerkultur Sachen gibt, die sind genauso gut. Oder vielleicht sogar besser. Mein Lieblingsbeispiel, über das ich mich schon immer gern aufgeregt habe, ist das ASCII Art Ensemble. Es hat sich gut im Kunstkontext vermarktet. Gleich-zeitig gab es das AA Project, das aber im Kunstkontext niemand kannte. Das sind tschechische Hacker, die eine Bibliothek entwickelt haben, die eine Echtzeittransformation macht von Bitmap-Grafiken in ASCII Art. Und zwat extrem gut. Mit Dithering, Halbtonrendering, Anti-Aliasing, so dass man 1:1 Computergrafiken in ASCII Art umwandeln kann. Und es gibt sehr viel Freie Software, die diese Bibliothek nutzt. Zum Beispiel kann man in GIMP (dem Standard-Grafikprogramm für Linux und BSD) Grafiken als ASCII speichern, weil es auf diese Bibliothek zugreift. Du kannst mit den Standard-Media-Playern, also Xine und M-Player, mit denen man sämtliche Videodateien, aber auch DVDs und Video-CDs abspielen kann, als Option zum Ausspielen ASCII einstellen. Das heißt, du musst gar nicht erst, wie beim ASCII Art Ensemble einen Film, in dem Fall 'Deep Throat', in mühsamer Arbeit umcodieren und dann als Java-Applet laufen lassen, sondern man kann eben, wenn man die Deep Throat DVD hat, sie in Echtzeit als ASCII Art betrachten. Oder zum Beispiel gibt es ASCII Art Webcam Software. Damit kann man direkt auf einen HTTP-Server streamen. Oder wenn man eine TV-Tunerkarte im PC hat, kann man in ASCII fernsehen. Dafür gibt es zwei Programme TTV und AATV. Und es gibt auch Grafikbetrachter in ASCII. Die haben sogar eine reale Funktion. Ich könnte mich jetzt zum Beispiel über deinen Rechner per SSH auf meinen Uni-Rechner einloggen und mir dann z.B. Viedeodateien über eine SSH-Terminalverbindung oder eine Telnet-Verbindung in ASCII ansehen. Auch wenn man keine grafische Oberfläche hat, sondern nur die Kommandozeile, kann man eben auch auf der Kommandozeile Grafiken ansehen. Das ist gar nicht so unpraktisch. Es gibt auch einen, mit dem man zoomen kann, in ASCII reinzoomen, das ist toll. Da besteht ein Reichtum an ASCII Art Software. Und mit dem, was das AA Project gemacht hat, haben sie das ASCII Art Ensemble locker abgehängt. Im Englischen würde man sagen, sie haben die gesmoked, d.h. das ASCII Art Ensemble hat nur noch den Auspuff gesehen, von dem was das AA Project zustande gebracht hat(lacht.)

C.S.: Aber das AA Project ist nie im Kunstkontext aufgetaucht.

F.C.: Genau. Und es war zeitgleich oder früher als das ASCII Art Ensemle. Wenn die so klug gewesen wären und sich im Netzkunst/Medienkunst-Kontext verkauft hätten, hätten sie eine riesige Karriere machen können. Und das spielt auch eine Rolle für uns. Wir wollen gute Projekte, die im Hacker- und Freie Software-Umfeld entstehen einfach der Kunst zuführen. Die müssen einfach gefeatured werden.

C.S.: Zu deiner Arbeit mit diesem Thema gehört ja auch der 'unstable digest', den du für 'nettime' zusammenstellst. Erklär das doch bitte mal genau, was du da unter welchen Gesichtspunkten zusammenstellst. Ich bin mir sicher, dass es für viele Leser recht kryptisch ist, was da wöchentlich in ihrer Mailbox landet.

F.C.: Codeworks ist eine Disziplin von Netzkunst, die sich zu einer eigenen Disziplin oder einem Subgenre verselbständigt hat. Es fing an mit Leuten wie Vuk Cosic oder auch jodi, die angefangen haben, Code-Irritationen nicht nur auf Websites zu betreiben, sondern auch in E-mails zu posten. Netochka Nezvanova hat auch eine ganz eigene Ästhetik entwickelt. Es war von Anfang an ein Teil der Netzkultur, die sich auf Mailinglisten manifestiert hat. Und es war von Anfang an auch immer ein Zankapfel: Toleriert man es auf diesen Listen, oder nicht, wenn jodi mal wieder so ein Code-Störfeuer veranstalten. 'nettime' ist unter anderem deswegen moderiert worden. Die 7/11-Mailingliste ist ein Ergebnis dessen; diese Liste sollte genau solchen Experimenten eine Forum bieten. Wenn man sich die Liste in ihren Anfängen ansieht, merkt man, wie Künstler eine Ästhetik da herausgebildet haben. Bei m.e.z. zum Beispiel, der australischen Netzkünstlerin, kann man ganz genau sehen, wie sie sich eine immer differenziertere Privatsprache gebaut hat. Dazu benutzte sie teilweise Codes aus dem IRC und allen möglichen Programmiersprachen. Daraus hat sie eine fast Lewis-Carrollsche Privatsprache entwickelt. Es gibt aber auch andere, wie Alan Sondheim, der, wie ich jetzt erst erfahren habe, das schon in den 70ern gemacht hat. Er hat mir gerade einen Stapel Arbeiten aus den 70ern geschickt, in denen er bereits mit Interferenzen von experimenteller Poesie, Konzeptkunst und Computerprogrammierung experimentiert. Mich interessiert das auch als Literaturwissenschaftler sehr stark, weil es auch eine Art ist mit Sprache zu arbeiten.

C.S.: Kann man das irgendwie nachvollziehen oder entziffern, was in den Privatsprachen formuliert wird?

F.C.: Die kann man dann entziffern, wenn man sich mit Computern gut auskennt. Es hängt aber auch sehr davon ab, über welche Künstler wir da sprechen. Die Ästhetiken und Vorgehensweisen sind

ganz unterschiedlich. Antiorp arbeit mit einem Filter, der englische Sprache einfach umcodiert, z.B. wird aus einem i eine Ausrufezeichen (!). Zu Beginn ist dieses Genre als ASCII Art verhandelt worden. Es war aber nicht wirklich ASCII Art. ASCII Art ist klassischerweise eine figurative Darstellung im Medium von ASCII Code, also im Prinzip ein Typogramm so wie man es auch aus der konkreten Poesie und aus der Figurendichtung seit der Antike kennt. Das war es ja nicht, weil es nicht nur darum ging, etwas wie eine visuelle Buchstaben-Poesie zu machen, sondern das Spezielle war, dass man nicht wusste, ob es sich dabei um eine Maschinenkreation oder eine menschliche Kreation handelt. Auch bei Antiorp gab es immer die Spekulation, dass es ein Softwareprogramm sei, ein Bot, der losgeschickt wird. Damit spielen sie alle. Und auch auf der symbolischen Ebene, dass nämlich Codes aus Programmiersprachen, aus Netzwerkprotokollen verwendet werden, recycelt werden. Und das erscheint mir relativ neu. Natürlich gab es dafür auch Vorläufer, etwa die Perl Poetry oder die Dichtung in der Programmiersprache Algol, wie sie von Oulipo geschrieben worden ist, einer französischen Dichtergruppe, die sich schon früh mit der Konvergenz von Algorithmen, Computer und Dichtung auseinandergesetzt hat. Aber es war weder ASCII Art, noch Perl Poetry, und ich habe mich gefragt, was eigentlich das Neue daran ist. Und ich glaube das Neue ist, dass da Computercodes als Material genommen werden. Sie werden nicht mehr im Labor konstruiert, wie das noch in der Algol-Dichtung von Oulipo oder der Perl Poetry der Fall war, sondern es werden bereits existierende Quellcodes oder Netzwerkprotokollcodes collagiert. Die kulturelle Voraussetzung dafür ist der Personal Computer und die Vernetzung des Personal Computer durch das Internet. Deswegen ist es auch kein Zufall, dass diese Kunstform genau in der Mitte der 90er Jahre entstanden ist und eine solche Virulenz erfahren hat.

C.S.: Eine wichtige Komponente dieser Kunst ist sicher auch, dass sie den User verwirren und verunsichern will. Bekomme ich nun deinen 'unstable digest' erwarte ich das geradezu, d.h. dieses Moment der Verunsicherung geht verloren.

F.C.: Das mag ein Problem sein.

C.S.: Was ist die natürliche Umgebung dieser Kunst?

F.C.: Der Irritationsfaktor kam natürlich dadurch zustande, dass auf Listen wie 'nettime' gepostet wurde und eine Interferenz hergestellt hat zu dem Diskurs, der da normalerweise herrscht und den auch gesprengt und in seinen Codierungen in Frage gestellt hat. Das funktionierte ab dem Punkt nicht mehr, ab dem es dann geblockt und moderiert wurde, und Mailinglisten wie 7/11 waren

dann noch die Ghettos, in denen diese Kultur gepflegt wurde. Wo es sich dann zwar auch ausdifferenziert und weiterentwickelt hat, indem die Leute miteinander und ihren Codes gespielt haben, aber es war eben auch ghettoisiert.

C.S.: Warum hast du den 'unstable digest' begonnen?

F.C.: Das kam zustande in einer Korrespondenz mit mez, die sich darüber beklagt hat, dass ihre Postings nie auf nettime und regulären Mailinglisten durchkommen. Ich schlug ihr verschiedene Lösungen dafür vor, z.B. einen anderen Moderationsfeed für nettime zu kreieren. Die andere Idee war, diese künstlerischen Arbeiten und Interventionen an meine eigenen nettime-Beiträge, die sich auch durchwegs an das nettime-Diskursformat halten, ein Posting von mez als trojanisches Pferd anzuhängen. Ich finde es übrigens wichtig, dass 'nettime' moderiert wird. 'nettime' hat eine sehr wichtige Funktion als Diskursforum, und es wäre in dieser Qualität nicht existent, wenn es nicht moderiert werden würde. Wir sehen das gerade an der 'rohrpost' was passiert, wenn die Ökonomie gestört ist. - Das klappte auch. Und dann war die dritte Idee, einen richtigen Digest zu machen, der das Format des 'nettime'-Announcers aufnimmt. Ich habe das vorher mit den Moderatoren besprochen, aber da ich selbst kein Moderator von nettime bin. D.h. es ist kein offizielles nettime-Projekt, aber es wird geduldet. Eigentlich finden es alle gut.

C.S.: Ja, wir finden alle gut, was wir machen (lacht). Woher beziehst du das Material für den Digest?

F.C.: Zu einem geringen Teil beziehe ich es aus gebouncten 'nettime'-Postings. Es kommt größtenteils von Mailinglisten, die dezidiert Foren sind für diese Codeworks.

C.S.: Woher stammt eigentlich der Begriff 'codework'?

F.C.: Den hat Alan Sondheim eingeführt, und ich finde ihn sehr gut. Die erste Codifizierung dieser Codeworks gab es in einem Special Issue der American Bookreview, das Alan Sondheim als Herausgeber gestaltet hat. Verschiedene andere Autoren haben dazu auch noch beigetragen.

Zurück zum Material. Es gibt sehr viele Mailinglisten, die sich dem verschrieben haben. Es gibt nach wie vor 7/11, das war mal zwischendurch in der Agonie, inzwischen läuft es wieder. In der Agonie von 7/11 haben sich verschiedene Mailinglisten neu formiert, zum Beispiel arc.hive von mez, eine Abspaltung der webartery-Liste, die auch ab und zu solche Beiträge hat. Dann gibt es 'wryting' von Alan Sondheim, eine Liste, die mehr aus der

elektronischen Poesie kommt, auf der alle möglichen Leute posten, die im e-Mail-Medium experimentelle Poesie schreiben. Auf der 'syndicate'-Liste gibt es auch viele solche Beiträge, dann 'o-o' von Mindaugas Gapsevicius, einer kleinen Liste mit wenigen, sporadischen Beiträgen... Und das sind sicher noch nicht alle.

C.S.: Das heißt, du abonnierst all diese Listen und musst all das permanent lesen?

F.C.: Ja, ja. (lacht). Dem 'unstable digest' kann man übrigens auch entnehmen, wo die Beiträge ursprünglich hingeschickt worden sind.

C.S.: Könnte man sagen, dass der Digest eine ähnliche Funktion hat wie die 'runme'-Plattform.

F.C.: Ja, durchaus. Der Digest ist zwar rein subjektiv - ich arbeite mit Alan Sondheim daran, und wir verstehen uns blind, d.h. wir haben genau den gleichen Geschmack - und es geht mir darum, abzuschöpfen von den Listen, was wir interessant finden. Ein anderes Kriterium als unseren Geschmack gibt es auch nicht.

Um nochmal auf das zurückzukommen, was du vorhin sagtest: Natürlich haben wir ein Ghetto geschaffen, und die Arbeiten kanalisiert. Es hat ein rigides Format, einmal pro Woche gibt es dieses Posting auf nettime. Das schließt jeglichen Irritationsfaktor aus.

C.S.: Aber warum findet du es trotzdem gut?

F.C.: Wenn es den Digest nicht gäbe, gäbe es dies Art von Postings auf nettime überhaupt nicht. Insofern ist es ein mehr und nicht ein weniger. Und es erreicht dadurch ein Publikum, das diese ganzen anderen Mailinglisten, die ich vorhin aufgezählt habe, niemals lesen würde. Man schafft also für diese Art von künstlerischer Arbeit noch ein erheblich größeres Publikum als alle diese Mailinglisten zusammengenommen. Ausserdem mag ich dieses Format, weil es die verschiedenen Arbeiten nochmal zusammenbringt. Und wir machen ja auch nicht typografisch lange Bindestriche zwischen den einzelnen Arbeiten, sondern sie fließen ineinander über. Das mag ich auch. Ich gebe mir auch immer viel Mühe, nicht nur in der Auswahl, sondern auch das in eine gute Reihenfolge zu bringen, also den Audienceflow. Und das hat für mich nochmal eine Qualität, dass auf diese Weise die Arbeiten nochmal miteinander kommunizieren.

C.S.: Gibt es Feedback auf die Postings? Und wenn ja welcher Art?

F.C.: Es gibt immer wieder Leute, die einfach schreiben, dass sie das mögen. Und wir haben ja nicht nur künstlerische Arbeiten drauf - das ist auch wieder eine Parallele zu 'runme' - sondern auch interessanten Spam. Es gibt manchmal so abgefahrenen Spam, der so unglaublich ist, dass man den auch als Kunst einstufen kann. Sowas kann dann auch mal in dem Digest landen. Meistens sind so 1-2 Spams in einem Digest. Die Grenze zwischen Kunst und Spam zu thematisieren, ist auch interessant und dafür ist ein Digest sehr geeignet. Wir bekommen manchmal auch Spam zugeschickt von Leuten.

C.S. Lacht. Ja, das soll es geben. Kannst du zu eurem Arbeits- oder Auswahlprozess noch etwas sagen?

F.C.: Ja, es gibt den Versuch, transparent zu machen, wie der Digest zustandekommt. Der Digest erschient normalerweise sonntags auf nettime, und donnerstags gibt es immer eine Betaversion oder einen Snapshot auf der Mailingliste arc-hive. Ich lese wie gesagt, all diese Listen, lege mir für jeden neuen Digest einen neuen Ordner an und sichere die einzelnen E-mails den Ordner als einzelne Textdateien. Es gibt ein Perl-Script, das dann aus den Textdateien in ihrer Reihenfolge den Digest generiert und das werfe ich immer am Donnerstag schon mal an und kreiere dadurch die Betaversion, die noch nicht fertig aussortiert und in Form gebracht ist.

Es gibt einfach diese Formatbeschränkung, dass man sagt eine gute Größe sind ungefähr 40KB, mehr würde die Postfächer der abonennten verstopfen. Und es erschient einmal pro Woche. Mit der Beschränkung habe ich aber kein Problem. Das ist sogar eine sportliche Herausforderung für mich. Es gab ja auch bei Oulipo das Konzept des 'Contrainte', also der Restriktion, einer selbstauferlegten Restriktion, zum Beispiel einen Roman ohne den Buchstaben E zu schreiben, wie Georges Perec. Und so sehe ich auch die Form des Digest als eine Contrainte an. Und den muss man eben sinnvoll füllen und daraus was machen.

C.S.: Ihr betreibt das jetzt schon ziemlich lange. Wie lange genau?

F.C.: Seit August 2002.

C.S.. Und wie lange soll oder kann es weitergehen?

F.C.: So lange, wie es interessante Postings gibt; so lange es Künstler gibt, die auf diesem Feld interessante Arbeiten generieren. Und ich sehe da noch kein Ende. Es gibt bei den einzelnen Künstlern große Qualitätsunterschiede. Einige Leute

sind brillant und andere haben halt in der Masse der Dinge, die sie ausspucken mal etwas Geniales dabei. Das wird dann auch abgeschöpft.

C.S.: Wird das Material noch anders archiviert?

F.C.: Es gibt ein Webarchiv auf zwei Websites, nämlich auf dieser Website von Johannes Auer (netzliteratur.net), da gibt es ein Archiv, und dann noch auf der Website News-Apprentice-Guild (?) von August Ryland(?). Die haben auch den Snapshot drauf. Im Moment arbeite ich noch an der Verbesserung des Perl-Scripts, die noch eine Datenbank einbinden soll, die die ganzen Postings verwaltet. Als drittes Frontend zusätzlich zu den E-mail-Digests und den Websites soll dann noch eine pdf.Datei, also ein Druckpublikation hinzukommen. Aus den sämtlichen Digests soll dann ein Buch generiert werden. Man kann dann jede Woche ein neues Buch generieren. Dafür verwende ich ein klassisches UNIX-Schriftsatzsystem, nämlich Troph. Das ist das Programm, das unter UNIX verwendet wird, um man-Pages zu formatieren. Es ist aber ein universelles Textformatierungssystem, mit dem man auch Bücher setzen kann. Es funktionert ähnlich wie HTML: ASCII mit Steuercodes. Das wird dann der 'unstable'-Reader werden, den man sich jede Woche neu runterladen kann.

C.S.: Eigentlich fände ich es schön, noch mehr über die individuellen Arbeiten zu erfahren. Bisher haben wir hauptsächlich über die Struktur gesprochen.

F.C.: Ich schreibe gerade einen Aufsatz über eine Arbeit von mez. Der ist entstanden aus der Jury-Diskussion zum ersten 'read-me'-Festival. Ich hätte damals am liebsten mez für ihre gesammelten Arbeiten, die sie eingereicht hatte, den ersten Preis gegeben. Es gab aber Widerstände von den anderen Jurymitgliedern, die argumentiert haben, das sei nur code-chic und ornamental. Daraufhin nahm ich eine Detailanalyse einer Arbeit von ihr vor: 'Virological Condition' oder 'Virological Conditioning'. Als Wissenschaftler würde ich sagen, das ist als zeitgenössiche Lyrik absolut Top - obwohl sie sich selbst nicht als Lyrikerin versteht.

C.S.: Kannst du bitte nochmal die Codeworks in Beziehung setzen zur Softwarekunst allgemein.

F.C.: Codeworks sind ein Teil von Softwarekunst und zwar aus zwei Gründen: 1. Weil Softwarecode, algorithmischer Code als Material verwendet wird. Und ist dann in der Regel kein running Code, sondern es ist imaginärer Code oder Code as Attitude wie wir es damals in der ersten Tranmediale-Jury definiert haben. 2. Weil es

eine kulturelle Reflexion auf das Phänomen Software ist. Deswegen könnte ich das nie aus der Softwarekunst ausklammern und meine Rolle in den Jurys war auch immer die, den Bereich der Codeworks stark zu machen.

C.S.: Ich habe den Eindruck, dass in der Softwarekunst der Begriff des Handwerks wieder eine große Rolle spielt, also der handwerklichen Fähigkeiten des Künstlers. Ist es notwendig, dass er sozusagen selbst an seinem Material herumschnitzt oder kann man die Idee/das Konzept von der Ausführung abkoppeln?

F.C.: Auf 'runme.org' gibt es eine Kategorie 'Digital Folklore and Artisanship'. Was die Idee der autonomen Kunst, auf die du anspielst, wirklich bedeutet, und dass es sich dabei sehr um ein westliches Konzept handelt, habe ich erst begriffen als ich letztes Jahr in Japan eingeladen war. Dort gibt es traditionellerweise den Begriff der autonomen Kunst wie wir ihn kennen überhaupt nicht. Und auch die Trennung zwischen Kunst und Kunsthandwerk existiert nicht. Es geht genau um die Trennung zwischen Kunst und Handwerk, also zwischen der getöpferten Schale und dem Ready-made von Duchamp. Ich glaube aber, dass es nicht ganz verkehrt ist, mit den Kategorien zu arbeiten. Wenn man zum Beispiel klassische ASCII Art ansieht — die vor der Netzkunst entstand — also die Kühe und Marilyn Monroes etc., die würde ich als Kunsthandwerk ansehen. Als solche sind die auch gut. Wenn man es in den Kunstkontext stellen würde, wäre es naiv. Den Begriff von Artisanship zu verwenden, hilft einfach den Sachen dann auch gerecht zu werden und sie nicht unter den Tisch fallen lassen zu müssen.

C.S.: Ich stelle die Frage nochmal anders: Wie wichtig ist es für Softwarekunst, dass der Künstler ein guter Handwerker/ Programmierer ist?

F.C.: Für Softwarekunst ist das überhaupt nicht wichtig. Er/sie kann ein/e hundsmieserable/r ProgrammiererIn sein und trotzdem tolle Softwarekunst machen. Dafür gibt es genug Beispiele, etwa 'dotwalk' oder das Browserfenster. Im Gegenteil, das könnte sogar eine Chance sein, wenn ein Künstler überhaupt nicht programmieren kann und Softwarekunst macht, weil das dann sogar eine Möglichekit bietet, noch eine schärfere kulturelle Reflexion auf Software und ihre Ästhetik anzustellen. Man hat dann einfach einen unverstellteren Blick als jemand, der als Hacker involviert ist und womöglich ein sehr fragwürdiges Verständnis von Softwarekunst als schönen Code hat.

C.S.: Das interessiert mich sehr, der schöne Code. In Hackerkreisen wird man ständig damit konfrontiert, und es

bereitet mir immer großes Unwohlsein, weil die Idee von Schönheit in der bildenden Kunst im 20.Jahrhundert mehr als in Frage gestellt wurde.

F.C.: Wir haben uns in der Jury auch sehr daran abgearbeitet. Der Begriff kommt eigentlich von Donald Knuth, der der Begründer der Informatik oder wie es im Englischen heißt, der Computer Science ist. Es gibt von ihm das mehrbändige Werk 'The Art of Computer Programming'. Die Software TEX hat er nur geschrieben, um dieses Buch setzen zu können. Und er sagt ganz bewusst, als Informatiker, Programmieren ist eine Kunst. Frag irgend einen Programmierer, nicht nur CCC-ler oder Hacker, auch Microsoftprogrammierer, ob Programmieren eine Kunst ist, und die Wahrscheinlichkeit, dass er ja sagt, ist ziemlich hoch. Sie werden dir sagen, ein Computerprogramm zu schreiben, sei im Prinzip so wie einen Roman zu schreiben. Das Bewusstsein, dass Software und Kunst etwas miteinander zu tun haben, damit hat kein Programmierer ein Problem. Sie wissen, was es ist und können die Komplexität und Subjektivität von Programmen einschätzen. Aber was es in den naturwissenschaftlich/technisch geprägten Kulturen gibt, ist ein Verständnis von Kunst als Schönheit, das klassizistisch, oder noch nicht einmal klassizistisch ist. Es ist ein sehr einseitiges Verständnis von Schönheit, nämlich Schönheit als Symmetrie und klarer Fügung. Wobei selbst der Klassizismus, Winckelmannscher Klassizismus von Spannungselementen ausgeht. Sieht man sich Laokoon als ästhetisches Paradigma eines auf Schönheit fixierten Klassizismus des 18. Jahrhundert an, dann ist der Punkt des Laokoon eben nicht flach und spannungslos, sondern dass es da auch widerständige, hässliche, verzerrte Elemente gibt, die sich aber in einer Symmetrie auflösen. Es gibt eine Formulierung, die auf Heraklit zurückgeht: "Harmonia est Discordia Concors". Das war ein Leitbegriff auch der musikalischen Harmonielehre der Renaissance von Franchino Gaffurio, einem Hauptmusiktheoretiker der Renaissance, dass schöne Musik auch aus Dissonanzen besteht, einer Auflösung aus Konsonanzen und Dissonanzen. In den Informatikerkulturen herrscht ein Verständnis von Schönheit, das nicht einmal ein dissonatens Moment beinhaltet, sondern wirklich von der reinen Konsonanz ausgeht.

C.S.: Auf welcher Ebene?

F.C.: Auf der Ebene des Codes.

C.S.: Ja klar, aber der hat ja selbst zwei Ebenen, die Zeichenketten und zum anderen deren Funktionalität.

F.C.: Es geht um die Klarheit einer mathematischen Formel. Der Code soll in sich eine Gefügtheit haben, nicht in dem Sinn, dass der Quellcode visuell schön aussieht, sondern dass er konzeptuell rein ist. Dass man einen eleganten Algorithmus drin hat. Es ist nichts Überflüssiges drin. Es gibt keinen Fehler in dem Programm, weil alles gut gefügt ist. Das ist ein ganz anderes Verständnis von Schönheit als es selbst im Ästhetizismus/Klassizismus vorherrscht. Das genau meint aber Knuth mit seiner "Art of Computer Programming", ein missverstandenes Renaissance-Verständnis einer extrem hohen Craftsmenship, als extrem hohen Kunsthandwerkes, das so gut ist, dass es dann eben künstlerisch ist. Das ist nicht mein Verständnis von Softwarekunst. Das kann allerhöchstens ein Teilbereich von Softwarekunst sein, das auf jeden Fall. Ein genialer Quellcode, ein toller Algorithmus, dem kann man eine konzeptell-künstlerische Qualität zugestehen. Aber das ist eben nicht alles und deswegen ist Craftmenship auch nicht alles.

C.S.: Das bedeutet letztendlich die Kriterien sind da bei Softwarekunst nicht anders als bei anderer Kunst auch.

F.C.: Ja. Wenn wir uns über Kunst im allgemeinen unterhalten, dann gibt es auch Beispiele, von einem extrem hohen Kunsthandwerk, das eine künstlerische Qualität hat, nehemn wir z.b. Tilman Riemenschneider oder gotische Kunst mit der unglaublichen Detailspräzision oder Dürer. Oder, um nicht nur Beispiele aus vergangen Jahrhunderten zu nehmen, Nam June Paik, dessen Arbeiten eine große handwerkliche Qualität aufweisen. Ich will das alles nicht aus einer genie-ästhetisch/romantischen Sicht schlecht machen, aber man schlägt damit nicht alles tot. Software ist ein Material, insofern stellen sich dieselben Kriterien von Materialbehrrschung wie in allen anderen Künsten auch.

C.S.: Wie ist deine Prognose für die Softwarekunst? Wird sie vom Kunstbetrieb angenommen/ einverleibt werden?

F.C.: Meine Vermutung ist die, dass es sich genauso verhalten wird wie mit anderer Medienkunst, und gearde auch so wie mit Netzkunst. Ich würde sagen, wir sind mittlerweile an einem Punkt, wo Netzkunst kaum noch spezifisch ist als Netzkunst, sondern allgemein es Kunst gibt, die mit dem Netz als Medium arbeiet und eben einfach Kunst ist. Genauso wird es Kunst geben, die mit Software arbeitet. Und die wird auch im Kunstbetrieb festetzen. Ähnlich ist es ja auch mit Videokunst; letzte Documenta als bestes Beispiel. Es läuft gar nicht mehr als Videokunst, sondern schlicht und einfach als Kunst. Dennoch ist Videokunst als Genre und Subkunst nicht tot. Es gibt nach wie vor eine Videokunst, die

sich ganz speziell mit der Materialität des Mediums Video auseinandersetzt.

C.S.: Aber ist Softwarekunst als solche, als spezifische Fokussierung des Umgangs mit einem kulturellen Phänomen und einem Material nicht doch eine Insidergeschichte? Der Zugang war ja bei der Netzkunst schon sehr schwierig, obwohl sich diese zum großen Teil noch an den sichtbaren Oberflächen des Netzes abspielte. Meines Erachtens bedarf es einer enormen Vermittlungsarbeit im Bereich der Softwarekunst.

F.C.: Es ist klar, dass Leute, die selbst einen technischen Hintergrund haben, damit gut klarkommen, die können Softwarekunst gut rezipieren, und wer das nicht hat, steht vor massiven Problemen. Das ist mir so klar geworden, als ich den in Berlin lebenden Fluxus-Künstler Thomas Schmitt in die jodi-Ausstellung geschickt habe, und der nichts anderes sehen konnte, als "bunte Scheiße".

C.S.: Es würde auch an ein Wunder grenzen, wenn er, Jahrgang 1943, ohne jeglichen Zugang zu Computern, das alles verstehen könnte.

F.C.: Aber das ist mit jeder Kunst so. Es gibt einfach keine Kunst, die transparent ist, die sich mittelteilen würde ohne kulturellen Hintergrund. Das ist eine absolute Illusion. Es mag zwar Kunsthistoriker geben, die daran glauben und Michelangelo als Beipiel dafür bringen, dass es universelle Kunst gibt, die jeder versteht. Das ist Quatsch. Das versteht auch niemand, der keine Ahnung hat von Emblematik und christlicher Ikonografie hat. Es gibt keine Kunst, die ohne kulturelles Spezialistenwissen funktioniert. Das ist bei einem Goethe-Roman nicht anders als bei Softwarekunst. Die Frage könnte dann nur noch sein, welches kulturelle Wissen universeller als ein anderes ist. Und da muss man zugeben, dass ein kulturelles Wissen von Software heute noch relativ speziell ist.

C.S.: Wobei sich der Umgang mit Computern bis in alle Ecken ausgedehnt hat. Man hat Legionen von Menschen, die täglich mit Software zu tun haben, in ihren Büros zum Beispiel.

F.C.: Ja, und die Leute die täglich mit Windows, Word, Excel etc. arbeiten, die lieben alle jodi. Die verstehen das sofort, wie sie genau diese Frustration mit den Interfaces kennen, die Abstürze kennen, die ganze Absurdität, die da hineinprogrammiert ist. Es gibt das Gerücht, dass es eine Zeit gabe, in der Sekretärinnen sich geseitig weiterempfohlen haben, jodi.org als Home des Browers einzustellen. Das teilt sich einfach auf einer bestimmten

Ebene sofort mit. Einfachheit ist oft ein Merkmal guter Kunst - ohne daraus jetzt einen Gemeinplatz machen zu wollen