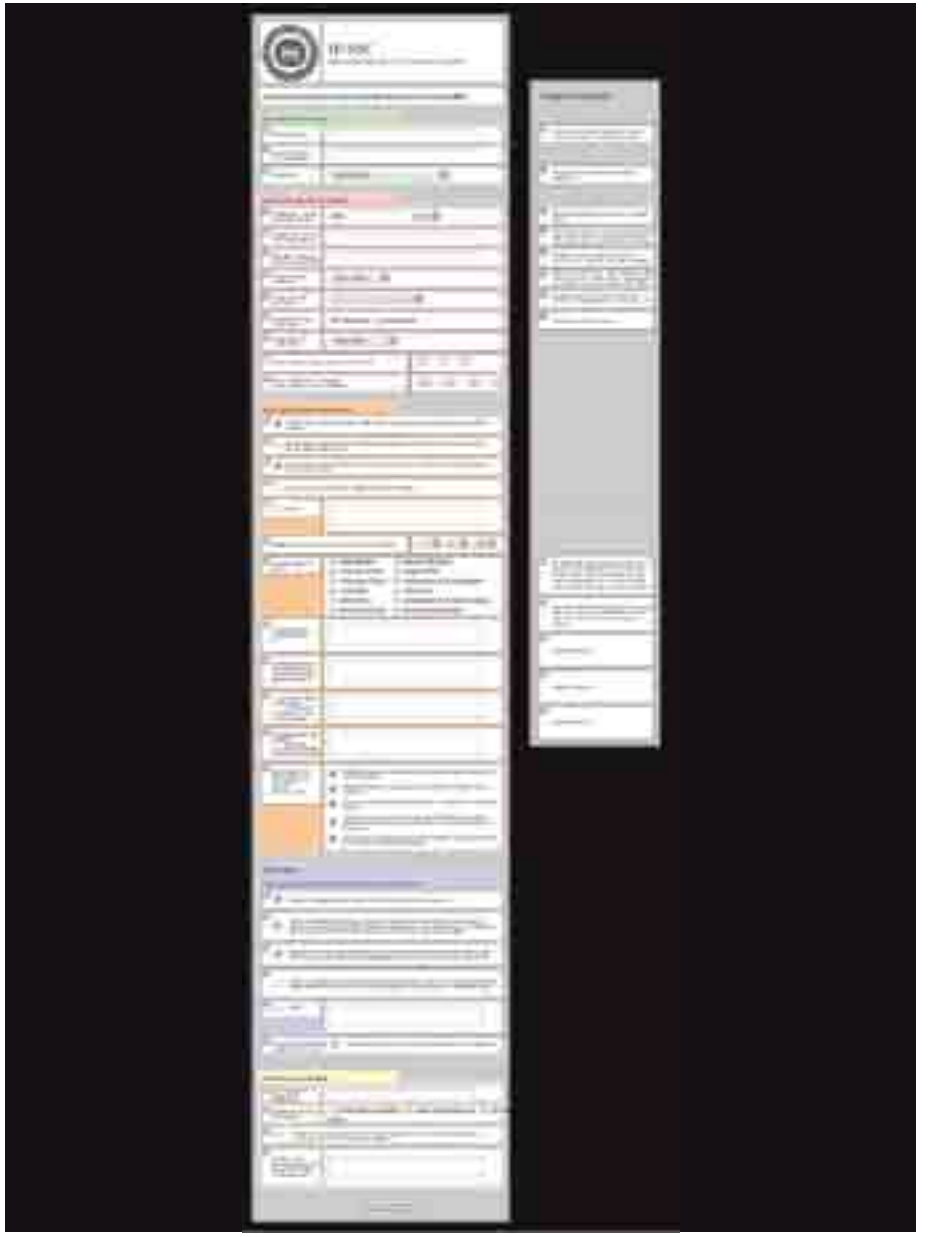

ubermorgen.com, *The Injunktion Generator,* 2001.

http://cryptoforest.blogspot.com
Site de Wilfried Hou Je Bek.

http://www.nettime.org/Lists-Archives/nettime-l-0309/msg00102.html
« Don't Call it Art: Ars Electronica 2003 » par Lev Manovich. Archive du texte posté en septembre 2003 sur la liste de diffusion Nettime.

Dix hypothèses au sujet de l'art logiciel[1]

Florian Cramer

1. De quoi ne s'agit-il pas ?

De l'art logiciel, tel qu'il est défini dans l'encyclopédie gratuite sur Internet Wikipédia (version de septembre 2003) :

> « L'art logiciel est le terme utilisé pour désigner la conception graphique des éléments visuels contenus dans un logiciel, par exemple une interface graphique utilisateur, des icônes, etc. » [2]

2. De quoi s'agit-il ?

D'artistes qui utilisent des logiciels informatiques pour produire des œuvres qui sont elles-mêmes – ce que seuls les écrivains avaient fait avant eux – des créations numériques fabriquées à partir de symboles, à l'aide d'un ensemble d'outils également

1. L'expression « Software Art » a été traduite dans ce texte par « art logiciel » à la demande de Florian Cramer, qui de ce fait entend aborder l'« art logiciel » non pas en tant que mouvement mais en tant qu'appellation d'ordre général servant à décrire cette pratique (NdT).

2. « Software art is a term for the graphic design of visual elements contained in software, eg. GUI (Graphic User Interface), Icons, etc. », http://www.wikipedia.org/wiki/Software_art (cette définition est celle qui apparaissait en septembre 2003, elle a été modifiée depuis lors).

constitués entièrement de symboles. Nul écrivain ne peut utiliser le langage comme un simple « remplissage » en vue de composer une œuvre qui elle-même ne serait pas du langage. Ainsi la littérature, à l'image d'une boucle récursive, fabrique ses propres outils. De la même manière les 0 et les 1 de l'art numérique sont étroitement liés aux 0 et 1 de l'instrument qui les a non seulement produits, mais également affichés et recopiés.

3. Sans logiciel, pas d'art numérique

Il serait naïf de croire que l'écriture, l'image, le son et le réseau sur un ordinateur existent en tant que tels ou en combinaison « multimédia », dans la mesure où aucune de ces formes de données ne peut exister sans le programme informatique qui les produit. Une vérité qui s'applique non seulement à leur conception et à leur traitement (grâce aux logiciels de traitement de texte, de création graphique ou de création musicale, par exemple), mais aussi à leur simple affichage (dans les navigateurs, les logiciels de visualisation d'image et les lecteurs) et leur reproduction (via les logiciels de réseau et systèmes d'exploitation). De même, toute œuvre numérique qui n'est pas elle-même un programme informatique ne peut exister qu'au sein du cadre que lui aura assigné un logiciel préconçu.
Chaque œuvre d'art numérique participe donc de l'art logiciel au sens large, au moins dans la mesure où il relève d'une forme d'art assisté par logiciel. Elle participe de l'art logiciel au sens strict, me semble-t-il, lorsqu'elle intègre le logiciel non comme aide extérieure, mais comme une partie intégrante de son esthétique.

4. L'art logiciel n'est pas necéssairement numérique ou électronique

Un programme informatique est une série d'instructions formelles (algorithmiques) dont l'exécution peut, mais ne doit pas nécessairement, être effectuée par une machine. Prenons l'exemple suivant :

```
// Classic.walk
Repeat
{
1 st street left
2 nd street right
2 nd street left
}
```

Il s'agit d'un exemple du programme *.Walk*, que l'on retrouve sur le site *Socialfiction* (http://cryptoforest.blogspot.com)[3]. D'après ses créateurs, *.Walk* fonctionne comme un « ordinateur psychogéographique », dans lequel les grilles d'un transistor sont remplacées par

3. Wilfried Hou Je Bek, *.Walk for dummies*, consultable sur *Socialfiction*, http://www.socialfiction.org/dotwalk/dummies.html

les rues d'une grande ville et les électrons par des piétons circulant dans ses artères. Le programme renvoie ainsi à deux antécédents historiques : d'abord à l'art conceptuel et au mouvement Fluxus avec leurs événements para-algorithmiques, minimalistes (tels que ceux orchestrés par George Brecht, La Monte Young et Sol LeWitt selon un paradigme défini par John Cage) ; puis à l'histoire de l'ordinateur moderne dans son incarnation première, à savoir rien de plus que cet appareil imaginaire et théorique que constituait la machine de Turing.

5. L'art logiciel n'est pas synonyme d'art conceptuel

.Walk se démarque d'un événement comme l'instruction binaire « on. off. » du *Three Lamp Events* de George Brecht en 1961[4], dans la mesure où cette œuvre reflète une pratique culturelle répétée : l'utilisation des ordinateurs, des logiciels et de leur programmation. Alors que le *Lamp Event* pourrait être considéré comme un préambule à la programmation logicielle artistique via une logique formelle, *.Walk* revendique – par son titre faisant écho au .NET de Microsoft –, son appartenance à une culture logicielle. Par conséquent, dans cette œuvre, ce n'est pas l'art conceptuel qui fait référence au logiciel, mais l'inverse : le logiciel renvoie aux performances et interventions d'ordre conceptuel des années 1960 (qui incluait également la psychogéographie de l'Internationale Situationniste), qu'il revisite en tant que logiciel. Pourtant, cette allusion n'a en soi plus rien de conceptuel ; elle est historique, ironique, œuvre de collage.
Et c'est précisément là que l'art logiciel d'aujourd'hui contredit cette équation d'art et de logiciel telle qu'elle fut présentée en 1970, lors de l'exposition d'art conceptuel *Software* de Jack Burnham au Jewish Museum de New York et dans le premier numéro du magazine d'art vidéo *Radical Software*[5]. Trente ans plus tard, le logiciel n'apparaît plus comme une conception de laboratoire et un paradigme de purification conceptualiste, mais se présente plutôt – et ce depuis la démocratisation des PC et d'Internet – comme du code erroné, comme le responsable de plantages, d'incompatibilités, de virus, exprimant ainsi la contingence et non plus la rigueur des symboles.
Le *net.art* de Jodi, Alexei Shulgin, Vuk Ćosić ou encore I/O/D ayant permis d'esthétiser ces contingences et donc de libérer l'art numérique de son apparent poli académique et industriel, il n'est pas étonnant de retrouver dans la récente histoire de l'art logiciel, qui s'inscrit dans la continuité discursive de l'art en ligne des années 1990, certains noms familiers.
Considérer l'évolution des travaux artistiques de Jodi de 1996 à nos jours permet de voir

4. On retrouve cette instruction sous forme de petite partition dans George Brecht, *Water-Yam/George Brecht*, Éd. Lebeer Hossmann, Bruxelles, 1986 (la version originale date de 1963).

5. Pour l'exposition, voir Edward A. Shanken, « The House That Jack Built: Jack Burnham's Concept of "Software" as a Metaphor for Art », dans *Leonardo Electronic Almanac*, vol. 6, n° 10, novembre 1998, http://www.leoalmanac.org/journal/Vol_6/lea_v6_n10.txt. Également consultable sur http://www.artexetra.com/House.pdf ; *Radical Software* est consultable depuis peu sur le site http://www.radicalsoftware.org

idéalement comment les expériences en *net.art* avec graphiques écran et communications en réseau ont tout d'abord constitué une œuvre dénonçant les contraintes de son environnement logiciel (voir la manipulation d'un navigateur intitulée *OSS* à l'adresse http://oss.jodi.org), pour évoluer ensuite vers la reprogrammation de logiciels (voir l'œuvre *Untitled Game*, basée sur le jeu vidéo *Quake*, à l'adresse http://www.untitled-game.org) et aboutir enfin à un texte source BASIC par la réduction de l'objet visible (dans la toute dernière œuvre, intitulée *10 Programs written in BASIC ©1984*[6]).

Il est vrai que l'art logiciel, lorsqu'il se fait minimaliste, présente une certaine similarité avec la pratique plus ancienne de l'art conceptuel ; ce rapprochement reste toutefois antinomique dans la mesure où il n'intervient pas dans l'esprit de cette dématérialisation de l'œuvre artistique typique de la période 1966-1971, telle qu'elle est décrite par Lucy Lippard dans son livre *Six Years*. Au contraire, dans l'art logiciel, le logiciel est considéré comme un élément matériel – un postulat qui constitue également une condition préalable aux *codeworks* (« œuvres code ») d'artistes tels que Jodi, antiorp, mez, Alan Sondheim, Johan Meskens et Lanny Quarles[7], mêlant éléments syntaxiques empruntés aux langages de programmation, aux protocoles réseau, aux messages système, et expressions argotiques caractéristiques de la culture informatique, comme le montre l'e-mail suivant de l'artiste française Pascale Gustin :

```
L'_eN(g)Rage \ment politi][~isch][K et l' _art is T(od)
][ref lex][1/O.ns 10verses NOT es][
--------\B(L)ien-sUr 2 que/S\tions f.Ond(ent)
-----------------------------------A:
--------------------------][menta les_sel][l] a tenement) T nem T
-tout d_abord--------1/O(f.ne
1 of 1 deletions
1 deletion done
apply: Command attempted to use minibuffer while in minibuffer
```

6. Exposé lors d'*Electrohype* à Malmö en 2003.

7. Voir aussi notamment Alan Sondheim, « Introduction: Codework », dans *American Book Review*, vol. 22, n° 6, University of Houston-Victoria, Victoria, septembre 2001, p. 1-4, http://www.litline.org/ABR/issues/Volume22/Issue6/sondheim.pdf ; McKenzie Wark, « Essay: Codework », dans *American Book Review*, vol. 22, n° 6, septembre 2001, p. 1- 5.

6. L'art logiciel n'est pas synonyme d'art algorithmique

Si le logiciel, défini de façon générale, est composé d'algorithmes, peut-on assimiler l'art logiciel à l'art algorithmique ou à l'art génératif – dont Philip Galanter a fourni une définition bien utile :

> « L'art génératif renvoie à toute pratique artistique dans laquelle l'artiste crée un processus, comme une série de règles en langage naturel, un programme informatique, une machine ou tout autre mécanisme, mis en œuvre ensuite avec un certain degré d'autonomie et contribuant ou aboutissant à une œuvre artistique achevée. »[8]

Certes, il se peut que l'art logiciel implique une certaine autonomie dans l'enchaînement des événements, telle que l'a décrite Jack Burnham dans des essais fortement marqués par la cybernétique et la théorie générale des systèmes nées au cours des années 1960[9]. Une autonomie que l'on retrouve par exemple dans le cas d'un code fonctionnel prenant l'apparence d'une application de PC classique[10], ou encore dans le cas d'instructions formelles univoques comme dans *.Walk*. Cependant, si l'on considère certains des sous-genres d'art logiciel les plus prisés, tels que les modifications de jeux vidéo[11] et les navigateurs expérimentaux[12], il ne s'agit plus de l'autonomie esthétique de processus algorithmiques, mais de leur interruption intempestive par l'action combinée du logiciel, de l'homme et des données en réseau. En outre, d'après la définition de Galanter, dans l'art génératif le logiciel n'est qu'un moyen parmi d'autres, qui ne constituera pas en lui-même une œuvre d'art mais se contentera plutôt d'y « contribuer », à l'image de nombreuses formes d'art assisté par ordinateur (notamment la musique électronique) dans lesquelles le logiciel n'est pas considéré comme faisant partie de l'esthétique de l'œuvre, mais comme élément agissant à l'arrière-plan.

L'art logiciel, pour sa part, ne satisfait pas aux critères de l'art génératif, ou du moins n'est en mesure d'y satisfaire que dans un sens métaphorique et non technique, c'est-à-dire lorsqu'il produit un logiciel dysfonctionnel et imaginaire. C'est le cas des *codeworks*, par exemple.

8. Une citation que l'on retrouve sur http://www.philipgalanter.com/downloads/ga2003_what_is_genart.pdf ou encore http://www.generative.net/read/definitions

9. Voir aussi la version allemande du *Structure of Art* de Jack Burnham, traduite maladroitement sous le titre *Kunst und Strukturalismus*, DuMont Schauberg, Cologne, 1973.

10. *Auto-Illustrator*, Adrian Ward, 2001, http://www.auto-illustrator.com

11. *Untitled Game* de Jodi (http://www.untitled-game.org/downloads.html) ; *retroyou* de Joan Leandre (http://www.retroyou.org).

12. *Web Stalker* de I/O/D (http://bak.spc.org/iod/iod4.html) ; *Nebula.M81* de Netochka Nezvanova (http://www.mediaartnet.org/works/nebula/) ; *%Wrong Browser* de Jodi (http://www.wrongbrowser.org) ; *Shredder* de Mark Napier (http://www.potatoland.org/shredder/) ; *Discoder* de Kensuke Sembos et Yae Akaivas (http://www.exonemo.com/DISCODER/indexE.html) ; *ZNC Browser* de Peter Luining (http://znc.ctrlaltdel.org/pc_znc2.0.htm).

7. L'art logiciel ne repose pas sur du vide, il fait partie d'une culture logicielle

Si l'art logiciel conçoit moins le logiciel comme un moyen de contrôle des processus génératifs que comme un matériel de jeu, il ne l'interprète plus – à l'inverse des pratiques « classiques » de l'art conceptuel et de l'art génératif – comme de la syntaxe pure. Le logiciel devient alors sémantique, porteur d'une signification esthétique, culturelle et politique[13]. Alors qu'en 1970 la culture logicielle – telle que documentée par l'exposition *Software* de Burnham et l'affrontement entre l'art conceptuel et le développement logiciel dans des laboratoires de recherche – restait circonscrite au domaine universitaire, et que même la culture *hacker* se bornait à de prestigieuses institutions telles que le MIT et Berkeley, c'est aujourd'hui une culture de masse qui s'accompagne d'une esthétique quotidienne du logiciel. De même, comme le montrent notamment les débats autour des logiciels libres, des monopoles, des licences, des logiciels publicitaires (*adware*) et des logiciels espions (*spyware*), le logiciel a désormais de plus en plus souvent une résonance politique. Malgré tout, la critique culturelle des logiciels n'a fait l'objet que de quelques tentatives sporadiques, comme dans les essais de Wolfgang Hagen, de Matthew Fuller ou encore sur la liste de diffusion softwareandculture lancée par Jeremy Hunsinger[14].

8. L'art logiciel n'est pas l'art du programmeur

Historiquement, le fossé entre l'« utilisation » et la « programmation » d'ordinateurs résulte de la création de l'interface utilisateur « iconique[15] » et de sa commercialisation par Apple et Microsoft, qui pour la première fois attribuèrent à chaque mode d'opération un medium différent : des images « iconiques » pour l'« utilisation » et du texte alphanumérique pour la « programmation ». C'est précisément à ce moment que la programmation d'ordinateurs est devenue un art obscur, associé à un savoir que seule possèderait une prétendue élite[16]. Les programmeurs ont bien entendu cultivé ce mythe et repris à leur compte l'héritage idéologique de la fin du XVIIIe siècle, en créant à travers le personnage du *hacker* une réincarnation du génie romantique.

Chaque exposé sur l'art logiciel comporte donc un risque, à savoir celui d'entretenir le culte du génie de la programmation. Un culte que contredisent les logiciels imaginaires, simulés et dysfonctionnels, ainsi que les manipulations de logiciels existants, qui ne nécessitent aucune maîtrise particulière en programmation[17]. Si le logiciel peut être non seulement le matériau de base de l'art logiciel, mais également son objet de réflexion, alors cette réflexion peut s'exprimer à travers des matériaux qui en seraient complètement différents – comme le montre l'œuvre de Julia Guther et Jakob Lehr *n:info*, dévoilée à Berlin lors de l'édition 2001 du festival Browserday. Celle-ci se présente comme un navigateur prenant la forme d'un cadre de fenêtre amovible, et renverse la rhétorique des logiciels de PC « iconiques » en proposant un outil analogique comme métaphore du logiciel numérique, exposant par là même l'application logicielle dite de « navigation Web » comme une technique culturelle, un mode de perception et de pensée[18]. On ne trouverait donc rien à redire à de l'art logiciel qui prendrait la forme d'une image peinte.

9. Les clichés de genre pourraient rendre l'art logiciel ennuyeux

Certes, le risque de se cantonner dans des stéréotypes existe aussi dans des formes artistiques qui, comme le mouvement Fluxus, ne se définissent pas via des matériaux spécifiques. Et pourtant, l'art logiciel pourrait bien perdre tout intérêt – aux yeux des critiques, conservateurs et jurys de concours – si son répertoire se limitait à des navigateurs Web expérimentaux, des visualisations de données, des jeux vidéo modifiés et des *cracks*[19] (tels que les virus informatiques et les *fork bombs*[20]). Un autre problème vient du fait que l'on associe souvent l'art logiciel à cet ensemble qu'est l'« art des médias », ce qui a pour conséquence d'empêcher certains programmes présentant un intérêt artistique – tels que ceux qui apparaissent dans le cadre du GNU/Linux et des logiciels libres – d'arriver jusqu'aux concours, festivals et expositions d'art logiciel.

10. En réalité, le débat sur l'art logiciel et sa légitimité en tant qu'art ne concerne pas l'art logiciel lui-même

On s'interroge régulièrement sur la pertinence du mot « art » dans la dénomination « art logiciel », qui deviendrait alors une discipline à part entière. La manière naïve d'envisager la question considère le logiciel comme le simple résultat d'une ingénierie, remettant ainsi en cause sa valeur artistique. À l'inverse une vision plus réfléchie dénonce le fait qu'un mouvement polymorphe se soit vu, une fois de plus et de façon injustifiée, affubler de l'étiquette « art ». Effectivement, tout comme la culture japonaise traditionnelle s'est passée de tout concept d'arts libéraux par opposition aux arts appliqués, on constate

13. Le *Injunction Generator* de Ubermorgen.com (http://www.ipnic.org/intro.html), qui émet automatiquement des injonctions juridiques, ainsi que le serveur proxy *Insert_coin* (http://odem.org/insert_coin/) de Alvar Freude et Dragan Espenschied, qui censure du texte, sont deux exemples convaincants de l'activisme politique de l'art logiciel.

14. Wolfgang Hagen, « Der Stil der Sourcen. Anmerkungen zur Theorie und Geschichte der Programmiersprachen », dans Wolfgang Coy, Georg C. Tholen, Martin Warnke (Sous la dir. de), *Hyperkult*, Stroemfeld, Bâle, 1997, p. 33-68 ; Matthew Fuller, *Behind the Blip. Essays on the Culture of Software*, Autonomedia, New York, 2003 ; softwareandculture, page d'accueil sur http://lists.tmttlt.com/listinfo.cgi/softwareandculture-tmttlt.com et archives sur http://lists.tmttlt.com/private.cgi/softwareandculture-tmttlt.com/

15. Ici Florian Cramer utilise le terme « iconique » au sens d'une relation de ressemblance, en référence à Charles S. Peirce (NdT).

16. Et ce, bien que la programmation dans un langage usuel ne nécessite guère plus qu'une connaissance des variables, des boucles et des instructions de type « si/alors ».

17. Comme par exemple le *ScreenSaver* de Ivan Khimin et Eldar Karhalev (http://runme.org/project/+screensaver/), une configuration de l'économiseur d'écran Windows qui permet d'obtenir un carré volant à la fois suprématiste et hypnotique.

18. *n:info,* http://myhd.org

19. Les *cracks* sont des programmes informatiques qui servent à perturber la marche normale d'un autre logiciel, généralement pour en dépasser les protections ou les restrictions d'utilisation (NdT).

20. Une *fork bomb* vise à saturer, jusqu'au plantage, la capacité de gestion de processus d'un ordinateur en multipliant rapidement une tâche à accomplir (NdT).

également dans la culture logicielle, aussi bien dans le cas du logiciel libre que dans celui du logiciel commercial, une vision de l'« art » dans le sens de sa racine étymologique latine *ars*, qui signifie « habileté », « savoir-faire ».

Comme l'a démontré un festival organisé par l'artiste Alexeï Shulgin, il peut être possible, grâce à l'imagination et à la mentalité *hacker* des programmeurs de logiciels en accès libre, de rassembler dans le cadre de l'art logiciel les travaux d'artistes et de non-artistes autoproclamés[21]. Toutefois, les objections à la classification de l'art logiciel dans la catégorie « art » ne sont qu'une des différentes façons de reconsidérer le concept d'art en lui-même.

Lev Manovich, dans son compte rendu intitulé « Don't Call it Art: Ars Electronica 2003 »[22], formule une objection personnelle d'une manière plus subtile que les deux précédentes en appelant à ne pas qualifier l'art logiciel d'« art », dans la mesure où cette pratique s'exclurait elle-même du cadre de l'« art contemporain » de par son trop grand intérêt pour un matériau spécifique. Or l'art contemporain, tel qu'on peut le voir dans les galeries, les salons, les musées et leurs expositions, se compose de sous-disciplines témoignant d'une attitude qui est loin d'être neutre à l'égard de leurs matériaux respectifs : on trouve d'un côté des peintures grand format et de l'art photographique pour les collectionneurs privés, et de l'autre des installations artistiques académiques (souvent sur support vidéo), généralement présentées dans des salles subventionnées et produites par des artistes et conservateurs issus des *cultural studies*[23]. Ceci mis à part, l'art logiciel n'est rien d'autre qu'un terme générique, au même titre que l'art pictural, sonore, scriptural ou vidéo. En outre il n'a pas été défini par les artistes eux-mêmes, mais par les critiques et les conservateurs, qui avaient identifié dans l'art numérique contemporain une tendance à utiliser des logiciels comme medium[24].

Il devient donc très facile de justifier l'utilisation de l'expression « art logiciel ». En effet, elle provient simplement du fait qu'aujourd'hui des productions majeures en art contemporain (à l'image de celles décrites dans cet article) voient le jour sous la forme de logiciels. Il devient alors nécessaire de disposer d'une théorie et d'une critique de l'art logiciel.

21. Pour exemples, le programme *WinGluk Builder* développé par un *hacker* et récompensé lors du festival Read_Me 2002 (http://readme.runme.org/1.2/inde6.htm) et le programme *Tempest for Eliza* exposé l'année suivante (http://www.erikyyy.de/tempest/), qui implémente un émetteur radio à ondes courtes par le biais d'un graphique affiché sur des moniteurs.

22. Lev Manovich, « Don't Call It Art: Ars Electronica 2003 », Nettime, septembre 2003, http://amsterdam.nettime.org/Lists-Archives/nettime-l-0309/msg00102.html

23. Née dans les années 1960 en Grande-Bretagne, la discipline des *cultural studies* mêle principalement l'analyse littéraire à la sociologie. Elle ne possède pas d'équivalent institutionnel en France (NdT).

24. C'était le cas notamment de Saul Albert dans « Artware », *Mute*, n° 14, Mute Publishing Ltd, Londres, 1999, p. 63-65, http://twenteenthcentury.com/saul/artware.htm, d'Alexander Galloway dans « Year in Review: State of net.art 99 », dans *Switch*, 1999, http://switch.sjsu.edu/web/v5n3/D-1.html, d'Andreas Broeckmann qui a introduit en 2000 une section « Software » dans le festival Transmediale, et enfin de Tilman Baumgärtel avec l'article « Experimentelle Software. Zu einigen neueren Computerprogrammen von Künstlern », dans *Telepolis*, Heise Zeitschriften Verlag, Hanovre, octobre 2001, http://www.heise.de/tp/r4/artikel/9/9908/1.html