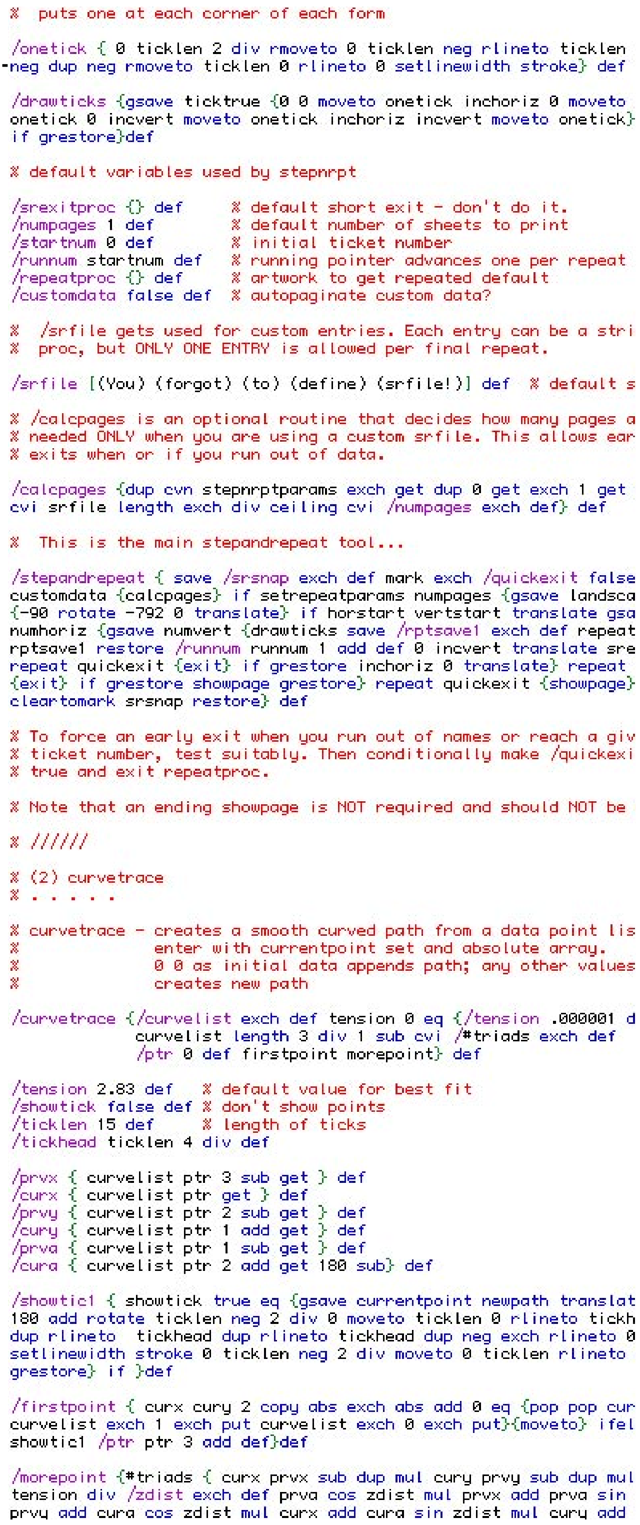
FINAL ESSAY MAY 2007

design paradigm shifts – shift design paradigms

How old media paradigms still rule software, education and process in the digitised era of graphic design...



```
%  puts one at each corner of each form

/onetick { 0 ticklen 2 div rmoveto 0 ticklen neg rlineto ticklen
neg dup neg rmoveto ticklen 0 rlineto 0 setlinewidth stroke} def

/drawticks {gsave ticktrue {0 0 moveto onetick inchoriz 0 moveto
onetick 0 incvert moveto onetick inchoriz incvert moveto onetick}
if grestore}def

% default variables used by stepnrpt

/srexitproc {} def     % default short exit - don't do it.
/numpages 1 def        % default number of sheets to print
/startnum 0 def        % initial ticket number
/runnum startnum def   % running pointer advances one per repeat
/repeatproc {} def     % artwork to get repeated default
/customdata false def  % autopaginate custom data?

%  /srfile gets used for custom entries. Each entry can be a stri
%  proc, but ONLY ONE ENTRY is allowed per final repeat.

/srfile [(You) (forgot) (to) (define) (srfile!)] def  % default s

% /calcpages is an optional routine that decides how many pages a
% needed ONLY when you are using a custom srfile. This allows ear
% exits when or if you run out of data.

/calcpages {dup cvn stepnrptparams exch get dup 0 get exch 1 get
cvi srfile length exch div ceiling cvi /numpages exch def} def

%  This is the main stepandrepeat tool...

/stepandrepeat { save /srsnap exch def mark exch /quickexit false
customdata {calcpages} if setrepeatparams numpages {gsave landsca
{-90 rotate -792 0 translate} if horstart vertstart translate gsa
numhoriz {gsave numvert {drawticks save /rptsave1 exch def repeat
rptsave1 restore /runnum runnum 1 add def 0 incvert translate sre
repeat quickexit {exit} if grestore inchoriz 0 translate} repeat
{exit} if grestore showpage grestore} repeat quickexit {showpage}
cleartomark srsnap restore} def

% To force an early exit when you run out of names or reach a giv
% ticket number, test suitably. Then conditionally make /quickexi
% true and exit repeatproc.

% Note that an ending showpage is NOT required and should NOT be

% //////

% (2) curvetrace
% . . . . . .

% curvetrace - creates a smooth curved path from a data point lis
%              enter with currentpoint set and absolute array.
%              0 0 as initial data appends path; any other values
%              creates new path

/curvetrace {/curvelist exch def tension 0 eq {/tension .000001 d
             curvelist length 3 div 1 sub cvi /#triads exch def
             /ptr 0 def firstpoint morepoint} def

/tension 2.83 def   % default value for best fit
/showtick false def % don't show points
/ticklen 15 def     % length of ticks
/tickhead ticklen 4 div def

/prvx { curvelist ptr 3 sub get } def
/curx { curvelist ptr get } def
/prvy { curvelist ptr 2 sub get } def
/cury { curvelist ptr 1 add get } def
/prva { curvelist ptr 1 sub get } def
/cura { curvelist ptr 2 add get 180 sub} def

/showtic1 { showtick true eq {gsave currentpoint newpath translat
180 add rotate ticklen neg 2 div 0 moveto ticklen 0 rlineto tickh
dup rlineto  tickhead dup rlineto tickhead dup neg exch rlineto 0
setlinewidth stroke 0 ticklen neg 2 div moveto 0 ticklen rlineto
grestore} if }def

/firstpoint { curx cury 2 copy abs exch abs add 0 eq {pop pop cur
curvelist exch 1 exch put curvelist exch 0 exch put}{moveto} ifel
showtic1 /ptr ptr 3 add def}def

/morepoint {#triads { curx prvx sub dup mul cury prvy sub dup mul
tension div /zdist exch def prva cos zdist mul prvx add prva sin
prvy add cura cos zdist mul curx add cura sin zdist mul cury add
```

Piet Zwart Institute
MA Media Design

by Jorrit Sybesma

Foreword & Table of Contents

Shift Design Paradigms / Design Paradigm Shifts

The essay is strongly connected to the conceptual background of my practical graduation project for the MA Media Design department of the Piet Zwart Institute, exploring the concept to a broader extent and construct a theoretical context for the practical work, dealing with the constraints and the role of the computer/ software within the (graphic) design(ers) process, during education and professional practice.

In short, in this essay taking apart the design process is the central issue. During this process of taking apart several aspects related to or influenced by the design process are investigated. The subjects of research are: design education and process, use of software tools and their context, function of computers and the designers mentality.

From my assumption, formulated in the statement and used as the subtitle for this essay: "that old media paradigms still rule software, education and design process in the digitised era of graphic design", the following research questions derived:

[1] Chapter 1: Introduction

Short introduction to subjects, through a descriptive text of the practical work of the final project "Shift Design Paradigms / Design Paradigm Shifts".

The idea of rethinking and reconsidering elements of the graphic design process, started with a thorough look at last years (2006) graduation catalogue, of the MA Media Design department of the Piet Zwart Institute. Immediately I jumped to conclusions, I would have done it differently, that was for sure. Except for known complaints I heard about the design, related to the use of 100 percent magenta coloured pages, my question was whether there was something fundamentally wrong with the design (I like pink) and the process. The design pretends to be more than just a catalogue, I think the designer (01) intended to incorporate parts of the design processes of all graduates. Although much of the work of Teeuwen reflects his craftsmanship and his excellent design skills, in the case of this graduation catalogue he failed in my perception. The result is a kind of 'in between' solution. Most of the catalogues related to an exhibition are 'white squares' with on the left side of the spread a picture and on the right hand side a descriptive text with title. These catalogues can be very effective and on a superficial level very attractive. If carried out very well, perfect as an example of the non outspoken conventions (02) of the generic exhibition catalogue. For the Media Design department however, the booklet is not a traditional catalogue, but it is not succeeding in taking a distance from the "traditionalities" of the catalogue conventions. Being very, very bold: it is faking a non-existing graphic design process. The designer used traditional software tools, in this case Quark XPress, to simulate a factor of coincidence in the design. The design (02) seems to be 'generated' or at least created with a 'random' algorithmic factor that created the overlaying images and texts. Which is a misconception, every page is carefully constructed as the part of catalogue as a whole. I consider that as pretentious, in the context of the course at the Piet Zwart Institute where the definitions which the designer 'plays' with do have a real meaning.

The practical part of the project ‘Shift Design Paradigms / Design Paradigm Shifts’ has derived from the initial idea that the graphic design process should be redefined and redesigned. In the words of the Dutch design critic and curator Max Bruinsma “we do not need to create new forms, but new mentalities” (03). To rephrase that: graphic design should focus on processes and the context it exits in, instead of on new finalised, finished shape. “The design of the process, is more crucial than the design of the actual product.” Although these quotes by C. Thomas Mitchell (04) are made in a broader context of architecture, urban planning, product and interior design, I consider them in relation to the graduation catalogue installation valuable as well. To be reflective on the working process on my previously mentioned final project, the development of the generative graphic design system as a whole has been a more crucial process than dealing with aesthetics on a, micro or page level.

Not centralising aesthetics in a design process, does not necessarily mean that the project ends up as an unappreciated, repulsive object (05), it is just another level of focus: the larger scope of the project as a whole. Using the words of C. Thomas Mitchell again “design should abandon discussions on design itself and become instead a socially oriented process in which we are both spectators and actors” (06). And that is exactly underling the essence of the user generated graduation catalogue. I consider the design process and the ‘final product’ itself as a research process. Meanwhile the project is a process in itself, reflecting working processes of students, while the generative graphic design algorithms that process the production of the ‘product’ are based on the mechanics and processes inside the final projects in progress of all MA Media Design students at the Piet Zwart Institute. Let me just clarify that more in-depth. Due to the dynamic and programmed character of most projects developed during this course and the emphasis of the work done during the process of a project is more important than just the final object, the choice of focussing on the process of the design of the graduation catalogue for that same course seemed a logical approach. None of the projects of the graduating students can be fetched by only a static page design or a single image. The Piet Zwart Institute is not a traditional design school and the background of the students varies from person to person. That background has obviously major influences on the working process of students. Some people are quite trained in going through a formal design process, others find their solution (total) chaos. Eventually in the best scenario, everyone will end up with something finished. Which then forms a facade that hides the crucial struggle that was necessary to get so far.

These struggles could perfectly reflect both process and politics inside the Media Design course — an algorithmically generated design would be appropriate — which ideally should give more insight into the working principles of the course and its students: preventing it from becoming a black box.

As a part of the graphic design process, the use of hardware and software in specific, form an important part of both project and this essay. In the design process and practice of contemporary graphic designers the use of the computer is common and necessary. The use of the computer is limited to a transformation black box for getting ideas from thought on paper to ink on paper. The electronic trajectory adds nothing to the design process, it is purely a matter of productivity, what I consider as a sad waste of exciting possibilities and hardware capacity. The software inside that black box seems to be more constraining than necessary and delivers often only results, which are mainly interesting from the perspective of automatisation, large scale network based workgroups and productivity performance. Within institutions for design education, future graphic designers are taught a single software solution. Although every piece of software has its ‘natural’ constraints and limitations (07), some software is more insisting than other. Students should be made aware of the alternatives for Adobe packages and the wide variety of other available software solutions and should they have the freedom to choose their tool of preference.

These issues as described above will be brought into practice in the graduation catalogue installation (08) at the WORM exhibition in July this year and will be explored in a larger scope and more in-depth in this essay. The emphasis within this essay is on the use of specific software in relation to design education and the process of the graphic design practice.

Jorrit Sybesma
Rotterdam, May 2007

[2] Chapter 2: (re)Design (the) process

How does in reality an educated design process become an unnecessary computer-driven graphic variety festivity?

Graphic designers have a strange relation to both their computer and the software installed on it. The majority of designers uses an Apple Macintosh computer and is convinced of the superiority of both hardware and installed Adobe software. Learned from my own experience, after the struggle of graduation within an average of four years of bachelor studies, the first thing these young professionals run off for to the store is to get their own Apple. A nice, shiny silver one, with a fancy flat screen positioned alongside it. That is the kind of the computer they have been jealously working on for the last four years, at least forty hours a week. After unwrapping the carefully designed box at home, they immediately install their ripped, illegal versions of Adobe's Creative Suite. Everything in place, let the professional practice begin.

The odd thing is that neither the computer nor the software should be the tools to run for, as a graduated professional. None of those two will really help the graphic designer in terms of designing a (great) poster or a (fantastic) logo or (beautiful) stationary. Software and hardware should only exists in the context of graphic design as purely productivity and automatisation tools. Nothing more, nothing less. Graphic designers should care about a good pencil and a piece of blank paper, to sketch on whenever a great idea comes in mind. The designer should master the concept, the idea is more important than the design of the final product. The actual work is done in the head of the designer, even before holding that pencil. Although, that is the through education idealised image of the graphic designer, not dependent of a specific kind of hard- or software. Partly because the majority of the authoritative teachers in design are from the pre-computer era. Now the clash with reality: they should not care about hardware, and in practice they do not really care about software. For most designers their virtual tool is just an oversized flowery-icon which has to be double clicked everyday. At least four times to start Adobe Illustrator, Adobe InDesign, Adobe Photoshop and in addition to that Adobe GoLive. A computer without pre-installed Adobe software, is not considered as a computer by those designers.

The fetishism for physical objects is part of the process of designing and is taught to every student. Feeling the sensation of holding your own designed small, beautiful, printed on quality paper booklet, is something almost every graphic designer is familiar with and can understand. Anyhow, that does not explain the lack of interest in the phase of desktop publishing and the tame acceptance of the lack of influence on that process. A computer without software, would not be a computer at all. The whole design process depends on the trajectory between final print and someone who is inserting ideas into the machine. Through software, design is made visible. The choice for specific software again is something that seems to be so obvious for graphic designers, that they do not even reconsider their choice or think before the act of buying (if that happens anyway). Consider this as misconception number one, further on in this essay this will be explained in a more elaborated way.

Apart from the association with specific hard- and software, the actual (graphic) design process itself is by outsiders (meaning: non-designers) often seen as something that is mystical and unclear (a closed black box). And at the end of a process there is a relief when something visible and physical is produced. Hundreds of books have been written that all promise that they will give insights and inside information on design processes and will explain how designers think and function. Some amusing, especially those written from the perspective of the hypothetical client (10), others unnecessary or obsolete. Theoretically a design process consists of a few phases, which are generally speaking in every trajectory quite similar and therefore not really interesting to mention here, since I plead here for abandon these traditional assumptions. For the matter of completion, a quick overview of the generic design process. Roughly, from orientation, analysis, synthesis, to concept, testing, deciding, designing to printing, will be a quick overview of the process as a whole.

Actually, only a small part of these phases in the design process is relevant for this essay on shifting design paradigms by designing paradigm shifts. Lets ‘ditch’ all of them and just concentrate on the phases that are really relevant. Let me point out the first two main segments of the graphic design process, apart from the mindset that should be different when it comes to changing the design process as a whole. The mind storm before inserting ideas into the computer and the act of using the computer to make the ideas concrete are the two main parts of the process that could be further merged in order to shift design paradigms. Contemporary graphic design software and use of computers do not lend themselves for this idea merging thinking and designing in a process.

Boldly said, the computer is in the old conception of the work field of graphic — except for productivity benefits — only useful for "the purpose of variety" (11). In other words, creating versions of one single graphic design or concept. The computer does not generate ideas or designs by itself, it is a 'dead' machine and the software can not be left alone for a second to do some work by itself. Graphic designer Karel Martens, an old typographer from the pre-computer era, witnessed the rise of the computer in his field of profession. The consequences of software that provides infinite functions and likewise options for the user, are endless number of possible variations on a single design principle. According to Martens that does not necessarily mean that the quality of the actual design is levelled to a new standard. He argues that the wide range of options and the overkill of variety has negative influences on the designers' capacity to make the right decision. "Every design is perfect, the computer rules out any possible technical failure." (11) In combination with perfect printing technologies, the results are "soulless" designs, that lack the "strive for an impossible perfection" (11). Which was the case when printing techniques did not lead automatically to uniformity, with traditional analog methods and mechanics. The consequence is over-saturation, one gets addicted to the idea that after creating a variation, there is always the possibility to create yet another. Martens signals the danger of speeding too fast through a design process, since every step is now only a matter of minutes. I respect the ideas of Martens and his work as well, but I consider — although his statements describe some urgent issues in contemporary graphic design — his assumptions of the work field as traditionally, 'old media' based. In the traditional conception of what a graphic designer should be, he is right. Martens is not at all adjusted to the contemporary world or and the changes profession of a graphic designer, where the Internet, network collaboration and databases fulfil an important role of influence. Nowadays being a graphic designer implies one is — or should be at least — also a system, information, media and process designer. Martens exactly points out the failure of the majority of contemporary graphic designers, which do consider the computer only as a machine for creating varieties of a theme.

And although the failure of graphic design and therefore the designer is still based on the quality of the concept, misleading, impressive software applications can make it for the spectator still difficult to judge whether the designer purely masters the technique or the idea. Involving spectator in the process and transform his role into the one of an actor, will undo that obscurity and reveal the real responsibility of the designer.

The designer should take the responsibility — as far as I am concerned — to open up that black box and provide insight into the design process. Not only to let the spectators act in a social process, but also for the purpose of the quality of his own work. An open graphic design process should reveal the systematic choices of the designer, it is more or less a research into the functioning of rules in space instead of a “discussions on taste and aesthetics” (12), which are only clear to an educated few. That does not mean that one excludes the other though, think of the ‘golden ratio’ (13), were rules and space determined the aesthetic most pleasing result. According to both C. Thomas Mitchell as well as graphic media designer Luna Maurer (14), designs that are purely based on decision made related to aesthetic values, which are not transparent to many and are difficult to understand or to communicate, are not interesting to both designer and public. That process of how to cope with these issues, is part of the academic trajectory of education, in reality other factors play a role in the design process, without putting the aesthetic discussion aside as being not relevant or purely superficial. I think both Mitchell and Maurer have a similar perspective to design Maurer’s work is in specific interesting here, because of the design process and her graphical work (15). She considers discussions on aesthetics and the sublime design or typeface as not honest and not interesting to appeal to a larger public than just colleague designers. Of course that is a bit bold, but in essence her statement in the context of her work is interesting. Her focus in her profession as a graphic designer has always been on the development of a system that creates — or generates — forms or shapes, based on rules and space. Although that are always the factors which a graphic designer is able to ‘play with’, not often these are defined as parameters in a partly automatised generative graphic design process. Maurer wants to provide insight into the systems that create designs, by separating designer responsibility and influence from the actual design work. She defines rules, the ‘system’ of scripted software creates the design in that same process. In essence these principles are similar to the principles my final project is based on, with the difference that in the GradCatGen (16) the choices which do have influence on the generated design, are made by the visitor. Transforming the spectator into an actor. With the intention to provide real insight into the graphic design process and show what the consequences of specific rules and choices on the actual direct printed output are. Opening up the black box, beyond the ‘shrink-wrapped’ design process. Meaning using tools out of the box, without any adjustments, without any criticism. There is no reason to think that the standard out of the box software package should fit all graphic design processes, although often it is accepted as such.

If so the graphic design process degrades — more or less — to a tool for professional amateurs, like a document set-up wizard in for instance software applications like Microsoft Office or OpenOffice as the derived equivalent. One is able to choose multiple non-options to end up with a variation of an original, much in line with what typographer Martens considers the most important task of both computer and software: creating variations of a theme. For the wizards it is mainly the lack of competent creative control that makes these things horrible to use. What if these wizards would create variations of a master document created by a well known graphic designer, like for instance Martens? Is that something to prefer above amateurs using the standard graphic design wizards of today? I really feel very connected to the ideas of Martens and his anxiousness towards the furthermore computerising design landscape. On the other hand, I consider the solution as now being part of his problem set. He considers using the computer as a problem, since — according to Martens — in essence the use of this machine adds conceptually nothing to the graphic design or process. I prefer to approach of not declining the use of these machines, but change the use of it, to search for the justification of using the computer within a design process. The software on and the computer itself form both problem and solution. In addition to that — and actually it is the key issue here — it is a change of mindset, mentality of the designer to change the 'Lego'-bricks of the process so that the tools become part of the design process, although that has not been educated as such.

[3] Chapter 3: Neutrality of the tool

Is there — and if so — what is the influence of software in terms of the use it through the insistent interface and the provided specific functionality, on the graphic design process and design aesthetics?

The use of specific software by graphic designers is not very often questioned or reconsidered and is an obvious choice and regular phase of the design process. But what are the effects of the interface of for instance Adobe software on the design process? And what are the effects of the predefined workflow of such software packages on the aesthetics of the final product that is created with it?

As soon as one starts for instance Adobe's Illustrator, the software provides a free, white, clean, undefined, beautiful, empty canvas (17). At least, that is what the user thinks. Of course one should start with an empty canvas. Well, no. The software is misleading, the open emptiness is fake. Because this white canvas is not white (it is transparent, it does not exist) and it is not empty. It has all kinds of predefined characteristics attached to it, without even notifying or asking the designer. Who is actually in control, the software or the designer? The software automatically determines that you have to use four basic colours (CMYK), you certainly need 300 pixels printed per inch at the end, the scale and the borders are already defined and you definitely need an A4-sized paper which is positioned in portrait mode. Besides that, you will have to use a workspace that measures in millimetres, instead of in picas or points. Probably even a lot more has already been defined, before a graphic designer can set a single brush stroke on the virtual canvas. In essence the software is already steering you towards standards, rules, printing regulations and conventions.

I consider it as the wrong way around anyway. Instead of opening up a world of endless possibilities on a virtual infinite canvas, the Adobe approach could be described as the 'limiting down' (18) method. After launching, the software immediately starts limiting down the user options. Step by step — sometimes by confronting the user with all the impossibilities — one is steered towards a specific working method, a pattern. Is this "user centred or system centred design" (19)? Should the software care about the fact that A4 is the regular paper size that is fed in almost everyone's printer and that it is common to use CMYK-colours, even before I got started with designing at all? Is the software foreseeing or influencing my choices (20)?

Why not start with an 'open' approach and limit down the possibilities afterwards? Well, the reasons are obvious: graphic design is a real profession and time is money and with that reasoning, limitations in the beginning, deliver less (compatibility) problems at the end when sending the data to the printshop. Again a build in, purely based on productivity enhancement paradigm.

After the first struggle with this 'wizard-like' step-by-step (21) (which can actually be switched off - the user is in that case not notified about the 'configuration' of the canvas) limitation of the possibilities, the actual work should start. But not before the most eye catching interface and system elements of the contemporary graphic design packages are thrown in our face: the palettes. Photoshop and Illustrator have numerous, from which only a select few are ever used on a regular basis and from which at least five are a total mystery or have never been seen before. The palette is the small window which provides multiple options to choose from, often these options are related to one specific task. The most natural one is of course the colour palette. The actual physical 'object' used by a painter, where the virtual has been derived from. And although physical and visual qualities do not match reality within the user interface of the software, the metaphor is easily understood. The mistake Adobe has once made is the conception that for all other options yet to be selected by the user of the software, the same metaphor is appropriate to use as well. Since when were we able to choose typefaces from a palette in real life in the profession of a typographer in times the computer did not exist yet? Adobe (and other vendors as well) makes it even worse when they put tools — actually a whole toolbox — in a palette. The idea of the palette is even more misleading when considered as being derived from the painters physical equivalent. The idea of the painters palette is that the painter can actually choose the contents of the palette. It is empty at start and the painter can start filling the dents with paint, different colours, you know the drill. When feeling happy with the arrangement of the palette, the painter is capable of mixing the substances of each dent in a random way with the contents of the other dents and decide on a random moment to use a brush to set the first strike on the canvas. But the — to the virtual world transformed — palette is a lot less advanced. Colours can be dragged — one point for Adobe — but can not be mixed in the same natural way as the analogue equivalent. A small mistake by Adobe, but they continue to make palette mistakes by putting all the software features in the palette as well as in the menu-bar of the software package. Like: brushes in multiple sizes, pencils, fonts, squares, gradients, circles, starts, triangles, graphs, diagrams, lines, dots, arrows, paintbrushes, anchors, hands, scissors, magnifying glasses, knives, magic wizard wands and what not more. Since most of these options have more parameters to set than just ‘on’ or ‘off’, more information had to be added to the palette of choices. The consequence is that the simple metaphor of the palette is destroyed by small circular buttons, a little larger square buttons, pop-ups with arrows and fields a user can type in as well. Where it goes horribly wrong is when comes consistency. A palette option (lets say the ‘brush’) with equal visual appearance in different Adobe applications, can have different operational qualities. The brush of the painter in each application acts differently, that would have been a surprise for the painter in analogue times. To make it even worse: these palettes and their options, have a really unique feature, the user can tab them in order to have more tabs than actually, physically possible on a square centimetre on the screen. Although this is a horrible concept, other options like layering information or using dialogue boxes do not make the working process easier (22). “A mental model does not have to be true or accurate, but it enables the user to work effectively with the modelled process”, according to Gareth Jones (23) from the Dublin City University of the department of Information Technology, on the Photoshop colour selection palette paradigm. But is this really the ideal way of adding new functionality to the software, or should Adobe abandon the idea that the palette metaphor can be used fore more than just select (and mix) colours regardless mental models and inaccurate virtual representations of analogue principles? Well, is there an alternative or a better solution? No, or is there…

All the elements fulfil my expectations — learned through education and practice — and the software is not surprising the user or does not behave unexpected, this turns the tool into a productivity enhancer. As soon as I would like to incorporate dynamic aspects or if I want to escape from the palette paradigm and adjust my user expectations: the software denies that. The use of an alternative can be interesting here: Auto-Illustrator (24). It uses the paradigms, conventions, templates and wizards of the traditional software and ‘mislead’ the user. Although it uses the same paradigms, as soon as clicked the designer starts using the software, it becomes clear that these paradigms were only implemented to argue against the use of them. The envision of the developers has had a totally different starting point then Adobe had. A traditional user of Adobe software is expecting no dynamic behaviour of the software, after clicking on one of the palette options in the toolbox (a palette in a toolbox, how come?). On the other hand it is addressing issues related to the limited functionality of contemporary graphic design software. Why should the designer be limited to the use of static elements in the design process only?

To use the words of the author — Adrian Ward — of the software package "Discover how easy it is to produce complex designs in an exciting and challenging environment that questions how contemporary software should behave". The use of the tool can be confusing, but overall it is a fun thing and it has limited functionality. The developer is a software artist specialised in developing interactive and generative software, mainly for audiovisual and graphical purposes. Besides the Auto-Illustrator package, Ward also developed the Auto-Shop software application. Which is could be considered as a 'generative version' of Adobe's Photoshop. Both packages "insist on the possibility of a radical change in the way people interact with digital technology and software in particular, focusing on an ironical reformulation of the copyright issues and in the extension of aesthetical subjectivity to the code itself". (25)

The fun aspect — although that is how I consider it and how I have experienced it — is that the software uses the same palette paradigms as the traditional proprietary and non-proprietary software and instead of being in control of that application — as one is used to — the software takes control over the design process. The decisions the 'designer' can make is limited to either choose or not choose a specific tool, from clicking on the process becomes a generative one. The good thing is the fact that the interface and palettes show so much similarities between the ones a graphic designer normally comes across in both Photoshop and Illustrator, but in this 'modified' software package these functions "act as semi-autonomous design tools, that are able to take their own decisions on how the work should proceed" (26). I consider this tool mainly as a piece of software that makes users of traditional software packages question their use of it, seeing Auto-Illustrator as a replacement tool is somewhat excessively expressed.

A further developed and more useful tool is 'Scriptographer', developed by the Swiss graphic designer Jürg Lehni (26). The software is not a separate package, that functions as a complete application, it is a fully functional, very well (in the palettes and toolbox) integrated, plug-in. Exclusively developed for Adobe Illustrator. For the rest it can be compared with Auto-Illustrator, although it is not mentioned by the developer, it has clearly been the source of their inspiration. Both software pieces can fulfil the same role as graphic designer Luna Maurer describes software should have in the graphic design process. Being a sustainable part of the 'creative' process of graphic design, instead of rather a dull, purely productivity tool, where 'responsibility' is 'shared' by both designer and software application.

To return to the conventional software again: not only the separate parts of the interface of the software by Adobe (or equivalents) is forcing the user to design in a certain way. Even the product positioning and the continuous integration of multidisciplinary functions is confusing the user. Except for the fact of the 'out-of-focus' approach Adobe seems to have, the questions comes whether we need new versions of existing software anyway. Since version 1.0 of for instance the application Illustrator, nothing has changed fundamentally (27). Just to refer briefly to my previously written essay on the "sad loss of visual abstraction in graphical user interfaces", in which a paragraph was devoted to the 'Aqua'-look of Apple's Operating System Mac OS X. Fundamentally, for the user nothing has really changed in the desktop metaphor experience since 1984, when the original Macintosh was introduced. Since then as (average) computer users — for those who use Windows, you have had the experience roughly since ten years later — are familiar with dragging, dropping and clicking on icons. The same argument can be used when Adobe introduces a new suite of expensive software: what has really changed since Aldus? Except for all kinds of productivity improvements and technical workflow issues. "Software can not wear, it can only get older" (28), which in essence does not really matter, as author and artist Peter Mertens ones wrote in a column in the magazine Items on the new version of InDesign. Why updating or upgrading anyway? Adobe is selling us the necessity of spending another budget on that, for them it is simply a measurement to keep their economical position; it is not about the functionality nor the tool itself. Take for instance Photoshop, that application has been developed originally for the purpose of image-editing and scanning. From version 1.0 up to 3.0 the development was essentially mainly about creating and finishing the core functionality of the application. Since then, Adobe has added a lot of features, plug ins, add-ons and filters (should we pay for lens-flare effects, or bubble-makers, graphic designers should demand for a 'pay-per-menu-option'-implementation (29)), that do not support or enhance the core functionality of the software application. The ideal Photoshop-user should even design their web site with that application, since it has an integrated HTML export functionality. It can draw vector-based graphics and the type engine is as advanced as the one used in the other software of the suite, Illustrator and InDesign. Should we abandon the other applications and start using Photoshop to do all the design work with or should the graphic designer drop all WYSIWYG applications?

What could be that different approach to the use of software in a graphic design process, taken the fact into account that the interface of the conventional software and the paradigms influence the design process too much. The software should be a sustainable part of the graphic design process and not so much as a dumb tool to make design visible or purely something that enables the user to be more productive instead of more 'creative' in general. As stated in a previous paragraph: merge design phase and publishing phase by enriching and enhancing the software.

Very shortly my vision of how this graphic design software should function in its essence. I see the future of graphic design software much more in a package that functions in a similar way as some contemporary web design software suites do. To refer positively to Adobe's GoLive for instance. That piece of software — although it is meant for web developments — has reached an almost perfect between frontend and backend. The software provides options to edit both the source code (whether that is html, php or javascript) and on the other hand the software provides the opportunity to edit the content visually. For graphic design purposes, this kind of 'hybrid' software — where the source code is not treated separately from the visual layer — does not exist. Perhaps a project to start working on, after finishing my course here at the Piet Zwart Institute.

Traditional software packages that rely heavily upon the use of the application through a Graphical User Interface (GUI), are problematic when it comes to using this software in a production line of multiple software packages or as a back end solution. Since all these software packages make extensive use of the What You See Is What You Get paradigm (WYSIWYG), they are limited in terms of controlling these applications externally , other then by — of course — the user. Which is in control of the system, although — as I explained in a previous paragraph — that is what the WYSIWYG paradigm makes you believe. From my previous essay on the icon as a part of the desktop metaphor, it is clear that the concept of the user interface has not been changed since the introduction of the original Macintosh in 1984, and with that the concept of the WYSIWYG paradigm driven software applications like PageMaker (now InDesign) has not changed either. Graphic designers are able to have visual feedback during their design process on the computer and are able to see the consequences of their actions, which can easily be undone if necessary. All beautiful inheritances of the original conception that operating a computer to use for graphic design purposes, should be straight forward and self explanatory.

Unfortunately the automatisation options of these GUI driven applications are limited to those implemented and predefined by its vendor. All of these automatisation options are based on the idea that software is used to produce large publications and should be fed with data from XML files to create large yellow pages or similar publications. In order to script these applications to do different things — related or similar to what my intentions are with my final project — using GUI scripting software (30) seems the only solution that would provide automatisation to a certain level. GUI scripting is actually nothing more than just using a programming or scripting language on top of the user interface of the application one would like to automatise or give commands to. The consequence is that the whole workflow becomes rather top-heavy, since on top of a continuously active GUI driven application like for instance Adobe's Illustrator — from which a user only actively uses 10% of the functions represented via the GUI — another application runs in order to give commands or to pass data. The standard software never becomes a proper back end application, since the active application has to be on front, because the GUI scripting acts just with WYSIWYG principles and is nothing more than an automatised human, with all its limitations attached to it. For automatising a graphic design process, this seems not the tool it adds just another layer to the software system.

Thus, for the purpose realising the technical backend of my final project (GradCatGen) and for the purpose of getting simply to the core of modern day printing technologies and graphic design processes, dropping all conventional GUI software packages, was a necessity. What is the core material a graphic design consists of, when being still on the computer? The answer — of course — is know already: PostScript. All the applications like Adobe's Illustrator (can) produce postscript files as an output for printing purposes. In order not to derive too much into describing the technological equivalents, or the historical development, I have to stick to postscript, since that is also the programming language and page description language used to create the graduation catalogue 2007 with. Postscript can simply be written as plain text files, later interpreted as postscript by either convertor software, printer or ripper. The advantage of using plain text is that it can be generated with almost every software application on a normal computer, using GUI applications or non GUI applications. Another advantage is that the document application (31) paradigm is thrown overboard as well. Switching to text mode drops all the previously mentioned paradigms and problems during the design process, but of course non WYSIWYG software or production methods deliver a different problem set.

First the justification for setting aside the traditional GUI based applications, apart from the already mentioned arguments. A new argument could be that a visual oriented design process is more influenced by a visual oriented design tool, like the GUI driven, drag and drop, point and click software packages. Let me clarify that, I consider non WYSIWYG tools as more neutral, more honest. The user is during the process of designing — a visual process in essence — not distracted or seduced to the use of specific visual elements, which are almost adverted constantly, by the software using the palettes in the interface of for instance Illustrator. These elements are sometimes screaming for attention 'use me', 'click here'! That is of course a bit exaggerated for the sake of my argument, but by the lack of the visual presence of these elements, the user can not be seduced to use them. There is no exact scientific proof for this argument, although it is likely that visual presence has its influence on the usage of a tool. To turn the world upside down and state it very bold: hiding a function somewhere far away in a submenu in the menu bar, would result in less usage of the tool. Apart from this all, with most WYSIWYG applications, the user does not get what he or she wanted, especially when it comes to software developed for design purposes, like creating sites for the web.

To continue with the web comparison, many professional web developers do not use WYSIWYG (32) based applications to create their php, html, xml or other web materials with. They simply use a plain text editor, with or without tag completion or syntax recognition. In essence the trajectory for web and graphic design is similar, a process that finally delivers a visual output for respectively screen and print. The downside of not using WYSIWYG software here, is the extra step of conversion that is necessary during the design process, to see actually what one is busy with. The visual feedback provided by a non WYSIWYG design process is lacking, which could be considered a problem, but it is actually something to get use to as well. On the other hand, for graphic designers this should not really be an issue, since they have been educated to work only on a computer as soon as they have a finalised idea in their head and on paper first. In that case, the hard postscript coding would just be something to visualise their ideas with. Unfortunately that would mean that the computer — and its software — would become again a stupid tool for publishing purposes only.

The idea that working with the source code of graphic design would be purely a masochistic trajectory without any advantages, is wrong (33). The advantage of using plain text, or at least plain text that can later on in the design process be converted to postscript, is the possibility that of generating text files or creating them using scripting or programming languages, like Python, (Java's) Processing, bash scripting, or even AppleScript. The graphic design process can be done in the background. This is a much easier way to use an application as a back end, speaking of real, sustainable performance and productivity enhancements as well. The process can receive data or input from the user, information streams over a network or the Internet, or data piped from a third party piece of software. In addition to that text editors like TeX/LaTeX or Troff/Groff can create postscript files (converted), based on preformatted text files. Using these non WYSIWYG applications adds as a downside another layer of translation to the process of creating the final print file, since these Groff files have to be converted into postscript again.

Except for the fact that plain text files can easily be created using small, external applications or programmed scripts, learning to deal effectively with the source code of graphic design should enable the graphic designer to get a better and more in depth understanding of the actual (technical) process itself. Thus, for educational purposes it would be very profitable for students to learn to use either raw postscript — although that is difficult and the programming language has a steep learning curve — or to use traditional typesetting and document processing WYSIWYAF software (34), like Groff or even TeX. Using this software lets the graphic designer not get distracted by features and nonsense, simply because there are none, and functions have to be found out through researching the documentation. The focus is completely on learning the basics of a technology and let them getting in contact again with the original traditions of their profession, manual typesetting and such. Creating a design with a few clicks belongs to the past.

Not only the educational aspect and the fact that dynamics can be incorporated into the process should be arguments for graphic designers to look further than the traditional — proprietary and open source — software packages. The designer is able to really influence their process, simply because he or she is able to create except for their own graphic design, also their own programmed tools. Something Luna Maurer would like to, but is not capable of doing. In essence that is the only way a graphic designer could really get on top of the process and control it by both mastering technique and concept, or give the control to the software or an external factor

— the user — as a part of transforming design into “a socially oriented process”; while becoming a facilitator of the tool and a curator of their own designed and programmed process. Designers create the process and leave the variability over to the user.

There is only one final argument to make. I do not want to state here that graphic design and the process of designing should be made difficult again. What the contemporary generation of graphic designers should take into consideration is the fact that through these simple software packages and easy to use wizard supported, design templates enriched applications, their profession and their importance is minimised. The transition from graphic design as a rather conservative (35) — old craftsmanship — to a more dynamic one would be a logical step in its evolution. Implied by the further democratisation of all kinds of processes as a result of the consumer shifting from passive spectator, to an active actor or participator.

That transition should be set in as soon as possible, since the rise of the professional amateur (36) is pressing on the work field of the traditional graphic designer. Which did not renew its profession since modernism. Nowadays consumers have become part of the production system and design processes are open and systems shared. Designers should “develop systems for participation” (37), where designers become also the “producer of the tool” (38) itself and therefore focussing on the process. A strange and exiting new role for the graphic designer, in an era where everyone is — or at least considers themselves as — a designer? The designer develops the tool and thinks of the parameters. Creates the wizard, designs the process, which the user can create variety with.

In that sense graphic design has actually changed under influence of computer and software and especially the Internet, although that has not been realised yet by the majority of the graphic designers. There are quite some similarities between what web designers do and the graphic designer should do. Develop processes and possibilities, instead of finalised fixed end products. Web developers accept the fact that the source of the material they work with is code, that is the state of their work since the beginning. Contemporary graphic designers should start accepting that as well. The shift in the kind of software they are using as well as a shift in the appliance of the software within the design process, should be educated. Beginning at the academies for arts and design.

[4] Chapter 4: Tool for the market / market for the tool

What is — and what should be — the role of design education in an environment where single software vendors have too much influence on their future market?

What could prevent graphic designers from getting involved in a design process where 'processuality' — or the focus on the process — is more than or at least as important as the creation of a finalised product? Like for instance the previously described design process of my project, designing the graduation catalogue for the MA Media Design department of the Piet Zwart Institute. Part of the problem is related to education, to what has been taught to them as students and what they have learned through practice. In strong relation to the educational trajectory is the use of specific software tools. The use of specific software has major influence on the design process. Not to exaggerate too much, but even the most incapable designer can create decent graphic designs with good predefined templates or well-wizard-equipped software. The choice of that specific piece of software is again influenced by the educational system, which is on its turn under influence of the economics of the market.

With a simple, straight forward calculation it becomes clear why Adobe's software licence policy is so profitable. The recent developments related to overtaking the Macromedia company illustrates why the design business in terms of the use of software is almost entirely depending on the products of a single manufacturer. Inevitably the question whether software will be created for a market, or the market is created by or for the software becomes relevant in the case of Adobe's market position. Adobe has all the reasons the make designers addicts of their products and keep them addicted. Their only serious single left opponent is the Quark Inc., with their — still a bit outdated — XPress desktop publishing (39) program, which has a continuously shrinking market share. The attack of Adobe in the late nineties on the market share of Quark was (after initial problems with the first version) a great success and soon a large part of the market switched to XPress' competitor InDesign. Presumably after half of the professional market switched (40), the design courses followed. In 2002 some of the Dutch academies made the transition in education. I was surprised of the transition speed at experienced at the Willem de Kooning Academy. After version 2.0 of InDesign was launched, it was almost immediately adopted (2002).

The mechanics of the market economy have their effect on the structure of the design education, since most of the academies are now ‘beroepsopleidingen’, preparing students for a profession.

Adobe could probably have killed Quark a long time ago, but instead of buying their competitor Adobe decided to enlarge their market share by acquire Macromedia, to ensure their position on the market of web development software. The strategy of Adobe is worrying, especially if one realises that within an educational environment the choice of tools is limited to Adobe, Adobe and Adobe. The software of Adobe transforms from possible solution, to a design methodology. No competition is — as we have learned from Quark versus Adobe — not good for the development and innovation of new products, the result will be that Adobe becomes as lazy as Quark Inc. was in the mid-nineties. The idea of having one large supplier of design software is scaring and the comparison with industry giant Microsoft is easily made and Adobe has even plans “to keep beating Microsoft” on all kinds of technologies. In historical perspective, Adobe and Microsoft fought on eBook (book screen readers (41)) standards, Portable Document Standards, digital type management conventions (and type smooth technologies). Microsoft has had the strategy to try to enter and concur the Adobe’s markets, in which they have failed. Even for the PostScript standard, Microsoft came up with an alternative, another total failure is for instance the application PhotoDraw, something that should have competed with Adobe’s equivalent Photoshop. With their very recently introduced Silverlight technology, Microsoft is attacking the market position of Adobe’s the Flash-player, this time with a product which is actually better than the Macromedia equivalent. acquired by Adobe.

To return to the economical model of Adobe’s business strategy. Their strategy starts with selling software to students, through an educational environment. During studies in art and design, students are made familiar with this software, in general already during their first year in the course. In perspective of the trajectory of ‘getting Adobe addict’ it is probably the first educational failure, since in my opinion a design institution or an academy of arts should never use a single tool strategy, but actually show the variety of options a designer has or can explore. From my own experience within the Dutch educational system, the choice is often made to use a single software package to train students in using. Beyond the educational mistake, an economical factor is presumably more decisive for these institutions in their choice for the one solution policy.

That economical factor has simply to do with the costs of the software and the savings and discounts these institutions can receive by choosing for one reseller only or using software from a single manufacturer by volume licensing.

In the end commercial parties, businesses and companies determine the artists and designers tools and influence their choices and decisions by its interface and features. Still, nothing to really have great concerns about, unless the users are non-critical educated using this specific software. The student — thus future graphic designer — should be made aware of the software's influence on their practice and should be able to make decisions onto which degree they let their work or design process be influenced by the software tool. Precisely that critical reflection on the user of software is often lacking in BA design courses, at least in the Netherlands. The lack of that critical reflection is related to the speed of the transition from a traditional design course to a computerised educational environment. Which has had more influence within design academies than in other educational institutions, mainly because teaching staff had problems to adopt to these new technologies at the same speed as their students. That warning was stated in an quality report, written by Hugues Boekraad in the mid-ninetees (October 1996) (42). Information-technolgy in design education was seen as a creative impulse, and has been adopted by institutions without any critical debate on the meaning of these new tools and their influences on the craftsmanship as the "core part of art- and design disciplines". A critical media theory (43) was lacking and the classical core definitions of aesthetics, form, content and technique should be reconsidered. They will probably get another meaning, when "implementing" these new forms of digital technology. Since the date of that report in 1996, the design education has not changed dramatically, alongside the traditional graphic design courses, special sub courses on the use of computers have been added to the core curricula. Making the subtle but important line that differentiates graphic design from desktop publishing very thin in contemporary design education.

Stating that Adobe is the party to blame is too simplistic and to claim that everything they have produced and have achieved in the recent history of graphic design is evil, would be inappropriate and pertinent incorrect. Although their (volume) licensing policy and their software suite bundling strategy gives educational institutions not much choice in terms of choosing a wider variety of proprietary software tools. Ending up with a homogeneous climate, although designers often want to believe that their tools are 'neutral'. That their behaviour, choices or practice is not influenced by the use of specific tools.

The consequences of a 'homogeneous software climate' is that educational institutions, as long as they do not teach critical and reflective use of tools, work in an 'unexpressed' collaboration with in this model Adobe, to create their economical market. Indirectly by using financial resources from the government, while in the Netherlands the national government is strictly against sponsored education, universities owned by businesses, in any form or model.

How profitable is the strategy of Adobe anyway? Well, the hard numbers tell that an academic student licence for the average Adobe software package — like Adobe Creative Suite 3 — is about $299,- VATS excluded. But, for institutionalised organisations like a design academy, Adobe will offer volume licensing, 30-70% off academic prices of single software copies (44). It is actually not the financial profit that makes the policy of Adobe that interesting. In addition to the financial part, it is also the update and product launch strategy by Adobe that makes it necessarily to update your Adobe package on a regular basis. In four years Adobe has updated their software three times, transition from separate packages, to CS1 to CS3. As consumers we are made believe that these updates are an absolute necessity and we can not design for the future if we do not own them. And Adobe makes sure we have to update, upgrade, patch or install their newest version, because of the lack of backwards compatibility.

Once learned to operate the Adobe software during the time at the academy, Adobe has presumably customers for life, because after obtaining their BA degree at the design academy they will start their professional practice. Since one has to work for 40 years here in the Netherlands, in order to profit from a well deserved pension, every Adobe user has to go at least through thirteen major software updates during his or her practice. Considering using an average of three updates per four years.

Minus the start up time in which young professionals do not pay for their licences, makes eleven updates over 35 years of practice for Adobe. Although students and volume licenses cost only a fraction of the ‘professional’ software versions, and although at almost 60% of the design studios there are licences ‘issues’, related to unlawful installations, according to an estimation of the Business Software Alliance (45), Adobe is still able to make great profits in their business of developing unnecessary updates. Designers become addicts for life (46), expensively educated by the design academies, with only a little investment — in the form of a volume licence discount and a student licence option — by Adobe. The ideal business model and the Adobe influence is even so addictive that being a graphic designer is — without questions asked — being associated immediately with using Adobe products. Using Adobe becomes a way of working or living, at least for some and even non designers are slowly incorporated by Adobe. Since every image editing action is in even the Dutch language now named after an Adobe application, Photoshop: ‘Photoshopping’ (47).

To return at last to the idea that design institutions do not have the freedom to choose a software package based on non-economical arguments. Although the industry and the market do pretend there are no alternatives to the palette paradigm driven (48) and unacceptable expensive software of Adobe, there are other parties which have developed equivalent suites with similar options. The main argument of these educational institutions for not using this software is either they were not aware of the existence — which is a non argument actually — or that the workflow of these applications (49) as a suite is not as slick and smooth as the one Adobe is providing. Which makes teaching it more difficult. An economical argument has not been used, and would not have been a valid one, since none of the ‘replacements’ would have been an expensive alternative, because the costs of these open source suites are zero… •

[5] Chapter 5: Conclusion

Will the designer disappear or evolve differently in an era where everyone is a pro-am?

Recapitulating: the following three main issues have been discussed in relation to the paradigms of old media paradigms that still rule the design process (chapter 1), software (chapter 2), and education (chapter 2) in the digitised era of graphic design.

The work field of graphic design has not renewed itself over the last forty years. The traditions of the analogue printing methods are still the foundation of the graphic design process of the digitised era. The transition to a computerised landscape has not lead to a fundamental change in neither graphic design nor the design process. The advantages of using the computer nowadays is only limited to mainly productivity enhancements, the design process is therefore — economically speaking — in a good shape. Computers have mainly been deployed to easily create multiple variations of a single theme and not so much as a tool for thought.
The limitations of current software packages and the ongoing changes in the market shares of large vendors, do not work in favour of changing the traditional work field.

Education should deal with the emerging process of breaking with the traditional printing, work field and software paradigms. As a result — and under influence of the changing world regarding the rise of the Internet and other networked environments — the graphic design work field should emerge towards a profession, more closer to that of a web designer. Focussing on the creation of a system, a tool that can be used by others. Since everyone consider themselves as a designer, the relevance of the professional becomes unclear. Unless the professional is able to make the required transition and keep ahead of the professional amateur. The graphic designer can become the facilitator of the 'wizard' for this continuously growing group of users.

Not only conceptually a shift in the mindset has to be made. Technically speaking, graphic designers should accept that — since their work has transferred to mainly working on the computer — they have to deal with a source code. Again, a similarity with the web developer. The source code of graphic design is PostScript, since the beginning of the so called Desktop Publishing revolution. Until now, no graphic designer uses tools that both enable him to work on the visual, as well as on a code level. Changes in using specific software — that yet has to be made — enables options for the user that go far beyond purely productivity enhancement. The graphic designer will become a programmer of a process, fully using all the number crunching capabilities of what the computer has to offer. How a product becomes a process, difficult to be taught, easy to learn…

The graduation catalogue installation forms the practical beta test of the ideas, as explored and explained in this essay. And as soon as I leave the Piet Zwart Institute, the first real project practising the theories, will be for the redevelopment of the graphic identity of WORM. Whether the ‘unwrapped’ design system functions, will be clear in the course of the next year, 2008. The follow-up for that project will be the development of a piece of software — facilitating what a graphic design process is needing and breaking with the existing paradigms — as described in the chapter 3. An evaluation will follow.

Jorrit Sybesma
Rotterdam, May 2007

[ ] References, Images & notes

(01) The designer is Roger Teeuwen, my former teacher at the faculty of graphic, information and communication design at Willem de Kooning Academy in Rotterdam.

(02) A typical exhibition catalogue, a generic, anonymously design consisting out of a 'white square' with perfect pictures on one side of the paper, guided by texts in typographical sublimely set typefaces on the other half of the spread. Shown here are the exhibition catalogues 2006 for the Gerrit Rietveld Academy, Amsterdam. For the "DesignLab"-faculty and the writers academy department. Designer: unknown, format A2 folded to A3-size, printed on newspaper paper. See for enclosed images, page 35.

(03) For enclosed images of the referred work by Roger Teeuwen, see page: 43-44. Graduation catalogue 2006 for the MA Media Design department of the Piet Zwart Institute, Rotterdam. Designed by graphic designer and teacher at the Willem de Kooning Academy, Roger Teeuwen.

(04) "We do not need to create new forms, but new mentalities" quote from Dutch design critic Max Bruinsma in an article in the Dutch design journal Items, published in September 1995 based on a lecture given by Bruinsma at 'IDEM' the Industrial Design Educational Meeting in September 1995 in Alden-Biesen, Belgium.

(05) Thomas C. Mitchell is professor in design methods and environmental design at the University of Indiana and author of various titles on the subject of redefining design in diverse contexts.

(06) Over the years I bought at least six personal digital assistants, these small pocket size computers, useful to store thoughts with a stylus or a keyboard with ultra small keys. Although these small computers were perfectly designed objects, which were irresistable because of their outer beauty. Inside these objects were ugly and hard to use. And as a part of a working process they were utterly useless.

(07) C. Thomas Mitchell, 'The product as illusion, in Design after Modernism' published by Wiley in 1988.

(08) A hammer is not limiting, unless someone asks you to use it to saw with.

(09) When mentioning the graduation catalogue installation, I refer to the production counter as a spatial object in an environment — meant for the exhibition at WORM — where visitors can make use of it and are able to function as 'spectators' and 'actors' in the generative, print and production process of the graduation catalogue.

(10) Some quotes, for the sake of relaxation: "design is the window on your business reputation", "head in the clouds, feet on the ground", "learn how designers think", "designers design to be number one", "designers always aim to achieve a predefined goal". From: 'Understanding Design' (BIS Publishers, Amsterdam, 2006) by author and professor in design education at the Technical University at Deflt, Kees Dorst and 'Design voor opdrachtgevers' (translation: 'design for clients'; published by BIS Publishers, Amsterdam, 2000) by author, designer and "creative strategy director" Jos van der Zwaal.

(11) Quotes from graphic designer and typographer Karel Martens based on an interview published in the Dutch design journal Items (#6/Y5), in October 1996 (page 12-19).

(12) Thomas C. Mitchell is professor in design methods and environmental design at the University of Indiana and author of various titles on the subject of redefining design in diverse contexts.

(13) Golden ration is a mathematical system based on natural proportions. Using these proportions in for instance architecture, created pleasant and aesthetically appealing shapes and forms of building, with of course the 'correct' proportions. Since the Renaissance, architects and artist using the proportions of the golden ration to base their works on. Source: http://en.wikipedia.org/wiki/Golden_ratio and http://nl.wikipedia.org/wiki/Gulden_snede and http://de.wikipedia.org/wiki/Goldener_Schnitt

(14) Luna Maurer is a German — based in the Netherlands — graphic designer, who graduated at the Sandberg Institute (Amsterdam) in 2002 after having completed a full course at the Rietveld Academy as well, where she is active as a tutor 'Interactive Media' since 2005 (source: design journal Items, issue #4 - October 2005).

(15) For enclosed images of the referred work by Luna Maurer, see page: 45-46. Work by graphic designer Luna Maurer, I referred to her theory and work within the essay, in addition to that image materials reflecting her working process the best. Showed is a poster and invitation for the 'Nest — Design for the Interior' exhibition, at the Stedelijk Museum in Amsterdam. Luna Maurer developed it in collaboration with programmer Jonathan Puckey. He developed the "programmatic design system" called Paper Plus Plus. "The design process is divided in two: first the system is set out in a set of chance based rules, then these rules are executed by hand. The designer functions both as the designer of the program and the executor of the program." Relevant note: Maurer did not develop or design the program here, she is only the executor of the program.

(16) When referring to 'GradCatGen', I refer to my practical part of the final project: the Graduation Catalogue Generator. A piece of software that generates this years graduation catalogue for the MA Media Design department of the Piet Zwart Institute. In short, the GradCatGen is using graphic design algorithms to generate the catalogue's pages, which are based on the students working processes or mechanisms, dynamics or concepts of their projects.

(17) Why is for instance Adobe Illustrator providing me a limited canvas? One of the advantages of the computer, is that can create 'unlimited' virtual spaces (whether that are 2D or 3D is not relevant here), and the first thing the software does is limit my space. That is contradictory. It should be the other way around, I just start working on my virtual unlimited canvas and I myself determine the scale why working and the size of the canvas when I am done. The next paradigm issue is related to the fashion in the software industry of a few years ago, to relate every process to 'the Internet'. Since that fashion has also influenced my Adobe's favourite Illustrator, that is something I bump into more than often. If I would like to export my design as a JPEG-file (yes, Illustrator is almost Photoshop as well) I have to export it as 'ready for the web'. Even if I do not want to have it made web-ready. The consequence of that action is that the self-preprogrammed thinking of Illustrator kicks in and blocks my action. I am not allowed to make large images for the web, while I do not even want to save this image for the web! The software denies to export is as JPEG! Is it a tool for me, or do they create a user for the tool?

(18) Of course every piece of software is packed with paradigms that limit the functionality of the software. Every software has its 'natural' boundaries that exactly determines the purpose of the software. A software is not an operating system and the limitations are therefore to ensure that the user is able to work with the software, at least to a certain level. Are the constraints of for instance Adobe more influential or more problematic than those of for instance Groff? These limitations become only too much of a constraint as soon as these paradigms turn — in context of this essay — the graphic designer into a user of the software.

(19) When referring to "user-centered" design and the term "system-centred" design, I refer to the words of Jef Raskin, who used these terms in relation to the development of operating systems for personal computers. He was convinced that the user could never be blamed for not understanding a system or function and that the software should be adopted to the needs of the user, instead of vice versa. (Source: http://jef.raskincenter.org/humane_interface/index.html)

(20) Adobe's Palette Paradigm. Immediately after double clicking the flowery icon of one of the Adobe applications, the software starts limiting our possibilities of our virtual white canvas. How empty is the immaculate, white canvas in reality? For the images, see page 37-38.

(21) Is this the graphic design process of the future? Adobe's software is only a few gradual steps away from the feared "Wizard" (Microsoft Windows terminology), an on-screen visual assistant that lets the user choose between non-options in order to complete either a set-up for a software package or hardware installation or a formatting assistant for 'designing' graphical documents for the purpose of presentations, creating stationery, faxing paper or business cards. When formatting documents using these 'Wizards' the user is using the computer according to its qualities (referring back to typographer Martens), creating many variations of an original. The influence of the user is limited and only related to visual, superficial aspects of the so called 'design', since he or she is only (in most software applications) in the ability to choose from a standard set of 'dtp'-parameters, images and texts. Shown here is the Document Wizard implemented in the OpenOffice variant NeoOffice. For enclosed images, see page 37.

(22) 'The Personal Computing Paradigm: Look and Feel', 1999/2000 - by author Michael Tsai published as an online article at the website 'About This Particular Macintosh' (source: http://www.atpm.com/5.05/paradigm.shtml)

(23) Dr. Gareth Jones, School of Computing, Dublin City University. Quoted from a lecture on User Interface Development in relation to design paradigms, found at source: http://www.computing.dcu.ie/˜gjones/

(24) Auto-Illustrator is a graphic design software package developed by Signwave that enables the user to experience the possibilities of generative systems within "their own graphic designs". More important is the fact that with the development of this software package, the intention was to question the behaviour of contemporary graphic design software suites; which is referred to as such within the essay. URL for downloading a limited version of the Auto-Illustrator 1.2: http://www.auto-illustrator.com/ For enclosed images, see page: 40.

(25) Quoted from a text on the site "The Best of Adrian Ward", about his award winning software: http://www.d-i-n-a.net/2002/en/metagallery/autoill.html

(26) Screenshot of the interface of Illustrator, with the Scriptographer tool installed and very well integrated in the toolboxes and palette paradigms of Adobe. Scriptographer is an excellent example of the conception that designers become more an more a provider of a tool than just the creator of a product. For enclosed images, see page 39.

(27) Enclosed a list of changes in the Photoshop-software, developed by Adobe, these are only the so called significant changes over time, since the very first version of that software package. Enclosed as well an image of the user interface of the very first Photoshop version. (Source: http://en.wikipedia.org/wiki/Adobe_Photoshop). For enclosed images, see page 36 (interface Photoshop 1.0) and page 41 (version history list of Adobe Photoshop).

(28) Referring to artist and theorist Peter Mertens, who wrote columns — related to software and graphic designers — for the design journal Items for several years in the past. Directly quoting from or referring to issue #5 November 2000 'Mac OS X, A New Start' (page 61), issue #1 April 2000 'Shopping or Adobe Design' (page 66), issue #6 January 2001 'Photoshop: for 1.000 things and more...' (page 61).

(29) A typical example of unnecessary implementation of functions in an application like Adobe Illustrator, which leads only to visual noise and is purely based on the result of visual effects. The functionality is known as "twirl" (envelope with mesh or shape) in Illustrator and opens up the possibility to shape typographical objects (outlined typography) to self-assigned vector based shapes or prefab objects. The results are astonishing. Since the first introduction of such a vector-based editing tool, a lot others have followed, not as plug-ins, but standard build-in (unfortunately one can not 'uninstall' those). For enclosed images, see page 42.

(30) GUI scripting in the case of the Macintosh Operating System (Mac OS) is part of the AppleScript programming scripting language that enables the user to activate so called 'System Events' that can address menu options, menu-bars, buttons, keyboard and other user interface elements. Although AppleScript is a proper, powerful and useful scripting language (it has its downsides as being too human and very verbose), in the role of a GUI scripting system it is close to worthless as being a serious workflow.

(31) The idea that an application should create application specific output or code as a document, instead of system wide, cross platform, 'inter-applicational' documents.
'The Personal Computing Paradigm: WYSIWYG: Is it What You Want', 1998 - by author Tom Iovino published as an online article at the website 'About This Particular Macintosh' (source: http://www.atpm.com/4.12/page7.shtml)

(32) WYSIWYG means What You See Is What You Get. In relation to software it means that what the user sees on the screen, is actually what the document will like in the end, either coming out of the printer or being published on the web. In terms of using software to create documents using the WYSIWYG-method, it means that the design process is purely based on a visual working procedure. The user can point out objects using a mouse, drag and drop without having to remember "layout commands". (Source: http://en.wikipedia.org/wiki/WYSIWYG)

(33) It is a bit masochistic, since postscript is little more difficult to program with in compared to HTML for instance.

(34) WYSIWYAF means What You See Is What You Asked For (in reference to programs such as those used for manual typesetting such as TeX or troff, that what is retrieved from the system is what the user specified - in essence, a statement of GIGO; sometimes also YAFIYGI: You Asked For It, You Got It) from source: http://en.wikipedia.org/wiki/WYSIWYG

(35) What has changed in graphic design over the past 100 years? An overview of works by Swiss typographer and graphic designer Josef Müller-Brockmann (poster for 'Internationale', silkscreen 90.5 x 128 cm - 1963), the Amsterdam-based people of Experimental Jetset (poster for the exhibition 'Public Address System' in London, digital print A0 - 2003) and the young and wild Rotterdam-based design duo Almost Modern (posters for 'user exhibition', silkscreen various sizes A3/A1 - 2006). Interesting to see though is that where the work of Experimental Jetset (what is in the name) is created using high technology and a revival of analogue production methods in the graphic design process is part of the work of Almost Modern (what is in the name), using silkscreen printing technologies (again). I can not imagine that over forty years, we still see these kinds of (typo)graphic designs. Forty years ago, the use of the Helvetica was renewed, nowadays almost 'retro', although some consider it still as refreshing. For encloses images, see page 36.

(36) Director of the Dutch Design Platform, the Premsela Stichting and graphic designer Dingeman Kuilman in an duo interview with publicist (innovation advisor for the British Labour Party and editor for the Financial Times) Charles Leadbeater by Marc Vlemmings for the Dutch design journal Items, issue #2 - April 2007. (page 36-50).

(37) Loosely based on an interview with graphic designer (autodidact) Mieke Gerritzen, part of a series of conversations ('Aanstormend treft Arrivé') between recently graduated graphic designers and those who are already arrived in the work field. Published in December 2004 in the design Journal Items, issue #5 (page 46-51).

(38) "The designer functions both as the designer of the program and the executor of the program". Work done by the developer of previously mentioned Maurers Paper Plus Plus graphic design scripting application, Jonathan Puckey. Who graduated recently (July 2006) at the Rietveld Academy in Amsterdam, having Maurer as both teacher and friend. Again, he also developed a scripting environment as an additional plug-in for the Adobe Illustrator software package, called Tile Tool. The work shown here has been created using that tool in specific (source: http://www.jonathanpuckey.com/), it does not really create new forms but it is a result of a changed mentality that questions traditional graphic design processes. For encloses images, see page 33.

(39) DTP or dtp means DeskTop Publishing, the act of using a specific software application on a computer to create the actual graphic design with.

(40) There are no exact figures known of the market share of Adobe InDesign versus Quark XPress. Quark Inc. is still advertising that they have an 80% market share, but declare that they have not updated these figures since Quark 4. We are now at version 7. Other sources declare (CNN Money - Mark Borden) in an article of September 2005, that at least 75% of the publishing market now uses InDesign instead of Quark XPress and that the market share of Quark Inc. is still shrinking. (Sources: http://quarkvsindesign.com/history-of-the-war/ and http://srh.typepad.com/blog/2006/12/indesign_or_qua.html and http://money.cnn.com/magazines/business2/business2_archive/2005/09/01/8356481/index.htm)

(41) "...to keep beating Microsoft..." based on an article ("How Adobe plans to keep beating Microsoft") published on CNETnews.com by the Warthon School of the University of Pennsylvania at April 17, 2004. (URL http://news.com.com/2030-1046_3-5190097.html)

(42) Critical media theory meaning in this context: there was not a clear and consistent policy towards the use of hardware and software by those who were in the position to make the decision, which is nowadays often an issue as well.

(43) Referring to an article by design critic, curator, researcher, author and teacher in design theory Hugues Boekraad on contemporary design education in the Netherlands, published in the Dutch design journal Items (#6/Y5), in October 1996 (page 48-51).

(44) Because of the lack of real insight into the process here at the Willem de Kooning Academy, only a rough estimation can be given here.

(45) These figures by the Business Software Alliance (BSA) should not be taken too serious, since the BSA is not an independent organisation. The BSA is a tool of and therefore financially supported by Apple, Microsoft, Adobe and other large software vendors. (Source: http://www.bsa.org/)

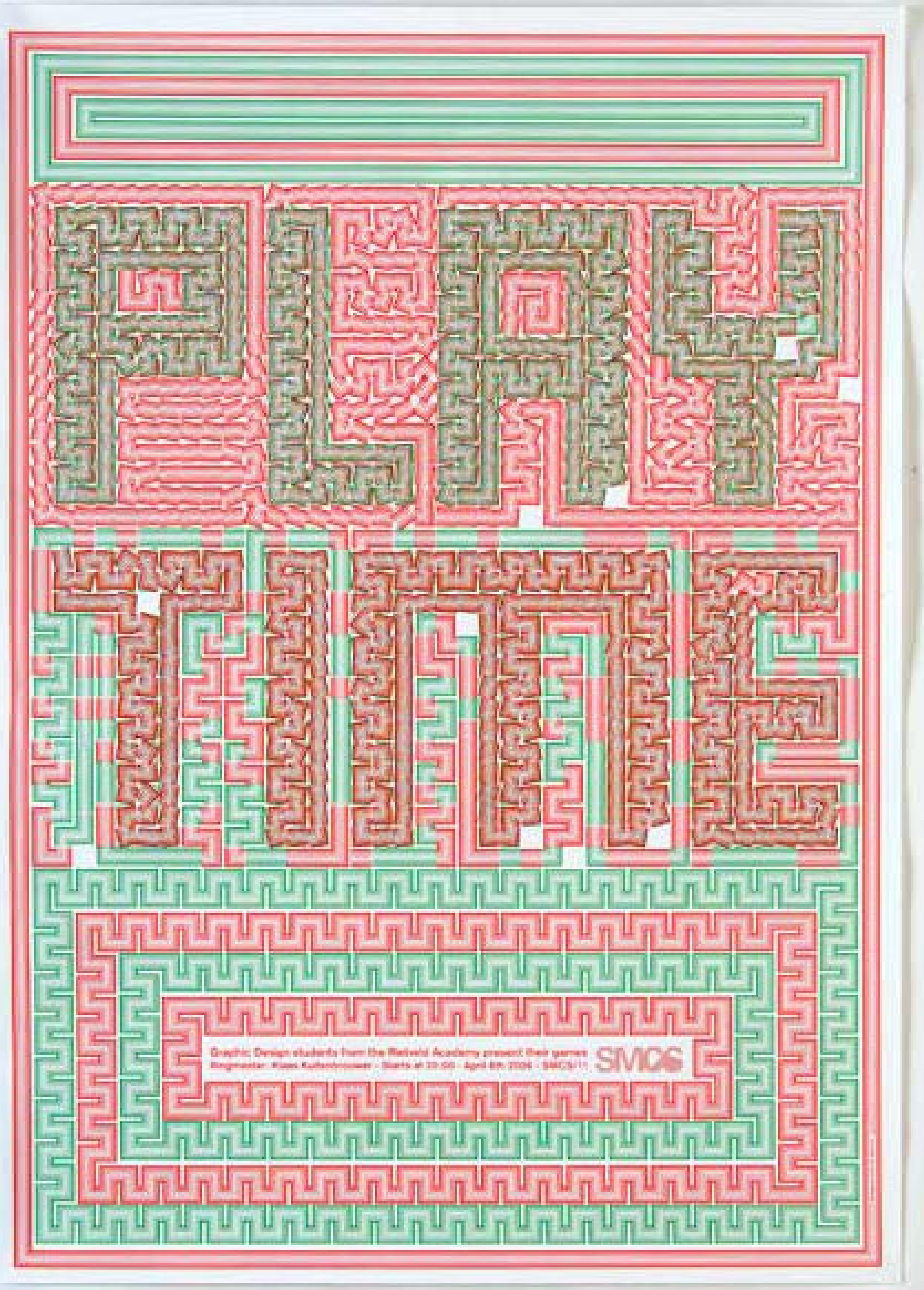
(46) Customers for life is of course a normal business strategy for these software vendors. Presumably Quark Inc. has had the same strategy as Adobe, in relation to keep their customer theirs. Quark failed, in theory Adobe can fail as well. The only issue here is that the failure by Quark was made in an upcoming market of a brand new operating system for the Macintosh. Mac OS X is considered to be the main platform for publishers and Adobe has benefit from Quark failing to make that transition and they had a much more up to date software package, with for instance implementing solutions for — their in house developed technologies — PDF file format. Since Adobe is by far the largest vendor in the business, there is no real competition. Unless Microsoft has a new product, which has actual advantages compared to their Adobe equivalents. Or do the open source alternatives have the future?

(47) ‘Photoshopping’ is the neologism for image editing in general, it does not even have to involve the actual software application Photoshop by Adobe. Just as searching over the Internet is to Google and making a photocopy in the US is to Xerox something. Adobe was not very pleased with it and advised to use different terms, such as “chopped”, “chop” and “chopping”. (Source: http://en.wikipedia.org/wiki/Adobe_Photoshop)

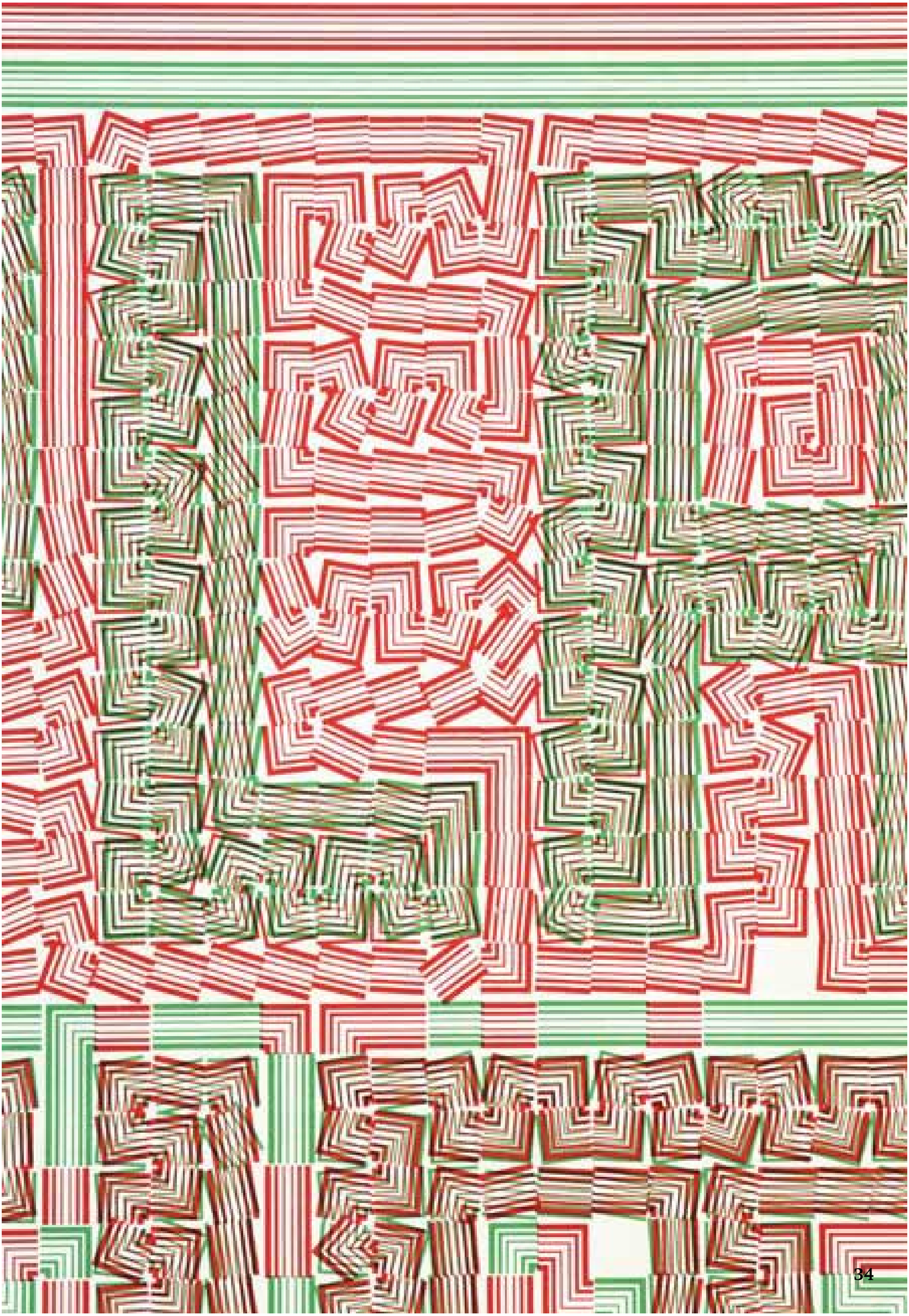
(48) Palette paradigm driven: referring to the palette paradigm as explained in chapter 4.

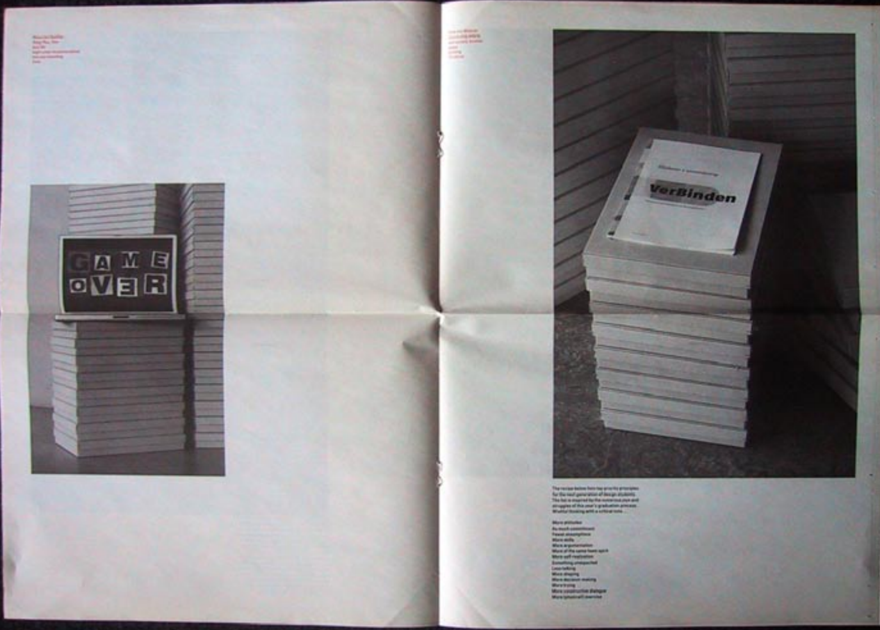
(49) Alternative software packages for existing Adobe applications: ‘The Gimp’ as a ‘Photoshop’ killer and ‘Scribus’ as an alternative for Adobe’s ‘Illustrator’.

(00) A small remark here. Neither this chapter nor this essay is a ‘manifest against Adobe’, since the effects of software with a very present and binding Graphical User Interface (as Adobe’s software), workflow and fixed set of functionality are all quite similar, notable and visible. This same argument of software that forces the designer to deal with the interface first, before getting to the actual work and where the functionality of specific tools is almost advertised in the GUI — the interface becomes noise, a factor of distortion — is also usable in relation to the use of the open source alternatives. There is an educational solution for this, which has been revealed in the previous chapter of this essay.



PLAY
TIME
Graphic Design students from the Rietveld Academy present their games
Ringmaster: Koen Kuhlenkreuzer · Starts at 20:00 · April 8th 2006 · SMCS!!
SMCS

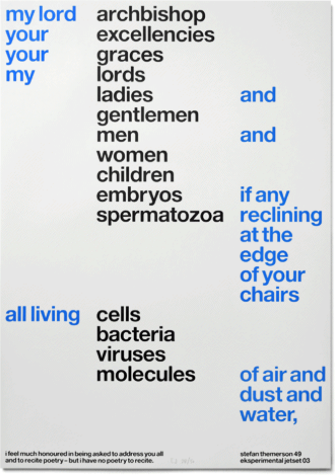



GAME
oV3R
VerBinden

1963
sinfonie-konzerte der tonhalle-gesellschaft
internationale juni-festwochen zürich

1 2 3 4 5 6

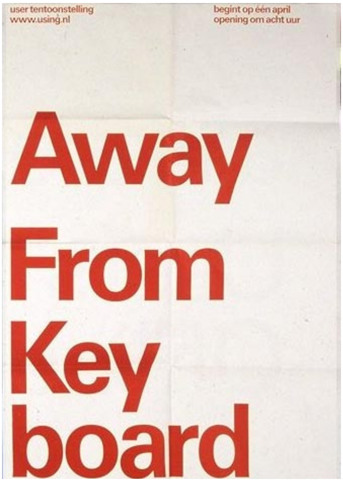
walter berry, van cliburn, margrit conrad, zino
francescatti, ernst häfliger, paul klecki
rudolf kempe, joseph krips, nathan milstein
eugene ormandy, rudolf serkin, maria stader
william steinberg
gemischter chor zürich, stalder-quintett
tonhalle-quartett zürich, tonhalle-orchester



my lord archbishop
your excellencies
your graces
my lords
ladies and
gentlemen
men and
women
children
embryos if any
spermatozoa reclining
at the
edge
of your
chairs
all living cells
bacteria
viruses
molecules of air and
dust and
water,

i feel much honoured in being asked to address you all
and to recite poetry – but i have no poetry to recite.

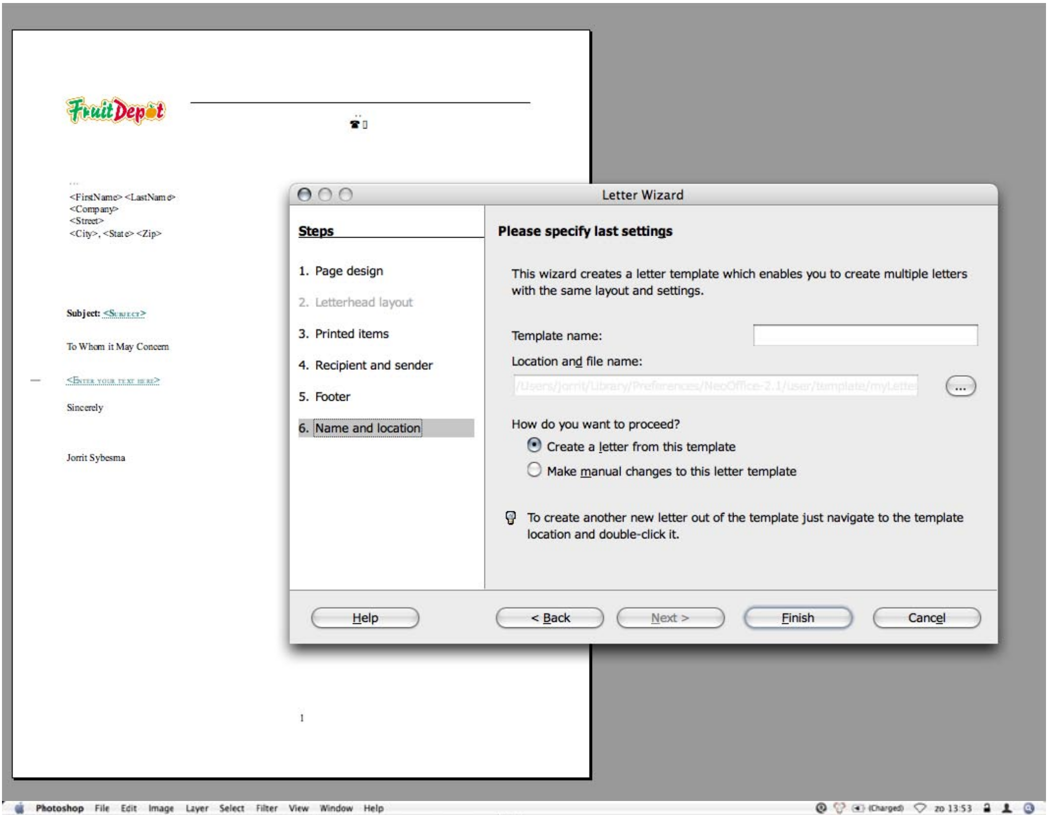
stefan themerson 49
eksperimental jetset 03



user tentoonstelling
www.using.nl

begint op één april
opening om acht uur

Away
From
Key
board



<FirstName> <LastName>
<Company>
<Street>
<City>, <State> <Zip>

Subject: <Subject>

To Whom it May Concern

<Enter your text here>

Sincerely

Jorrit Sybesma

1

Letter Wizard

Steps

1. Page design
2. Letterhead layout
3. Printed items
4. Recipient and sender
5. Footer
6. Name and location

Please specify last settings

This wizard creates a letter template which enables you to create multiple letters with the same layout and settings.

Template name:

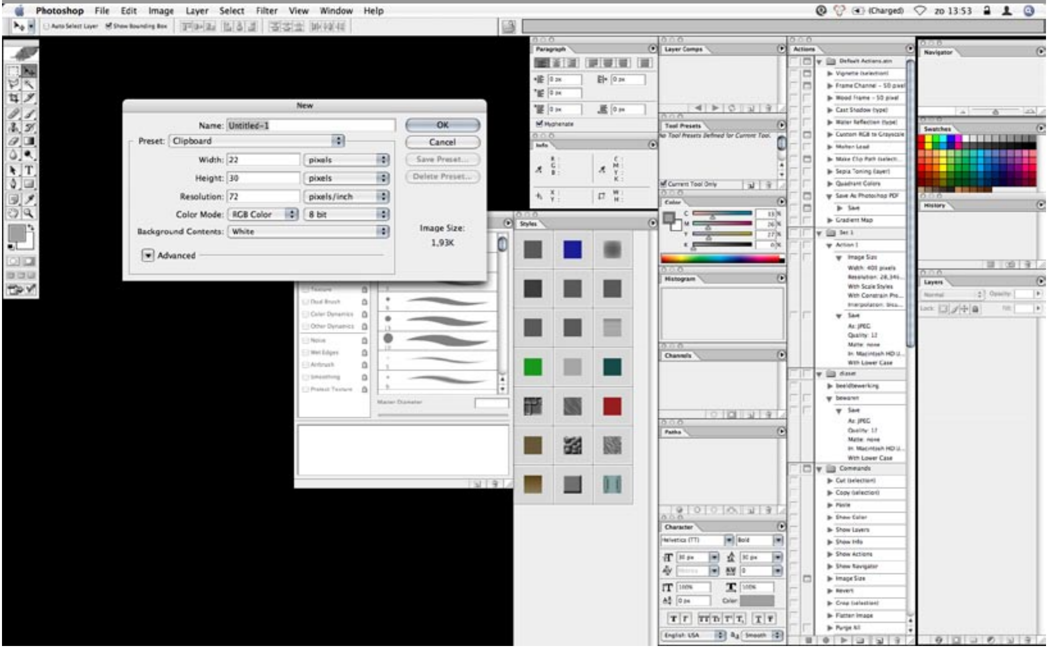
Location and file name: /Users/jorrit/Library/Preferences/NeoOffice-2.1/user/template/myLetter

How do you want to proceed?

● Create a letter from this template

○ Make manual changes to this letter template

To create another new letter out of the template just navigate to the template location and double-click it.



New

Name: Untitled-1

Preset: Clipboard

Width: 22 pixels

Height: 30 pixels

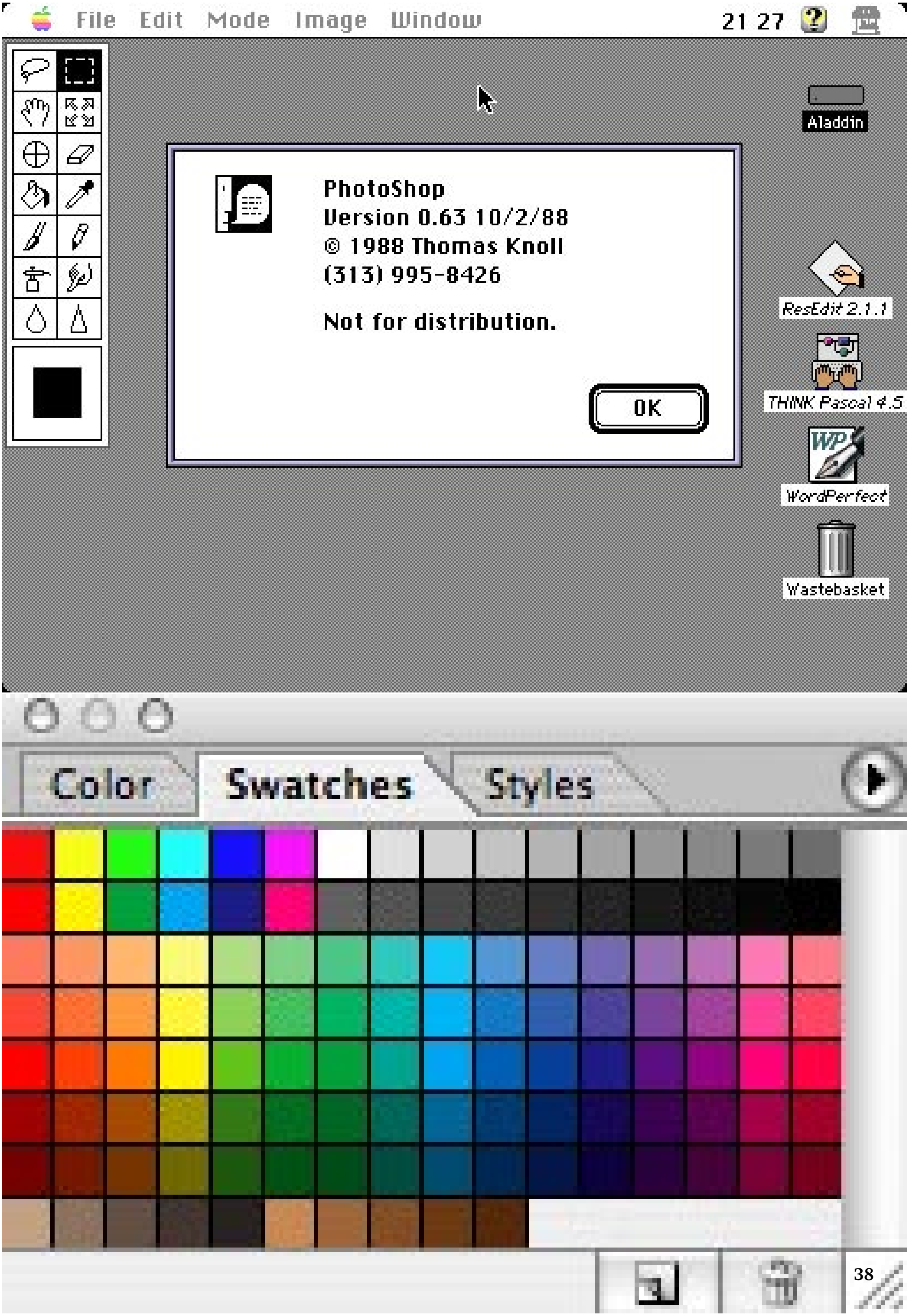
Resolution: 72 pixels/inch

Color Mode: RGB Color 8 bit

Background Contents: White

Advanced

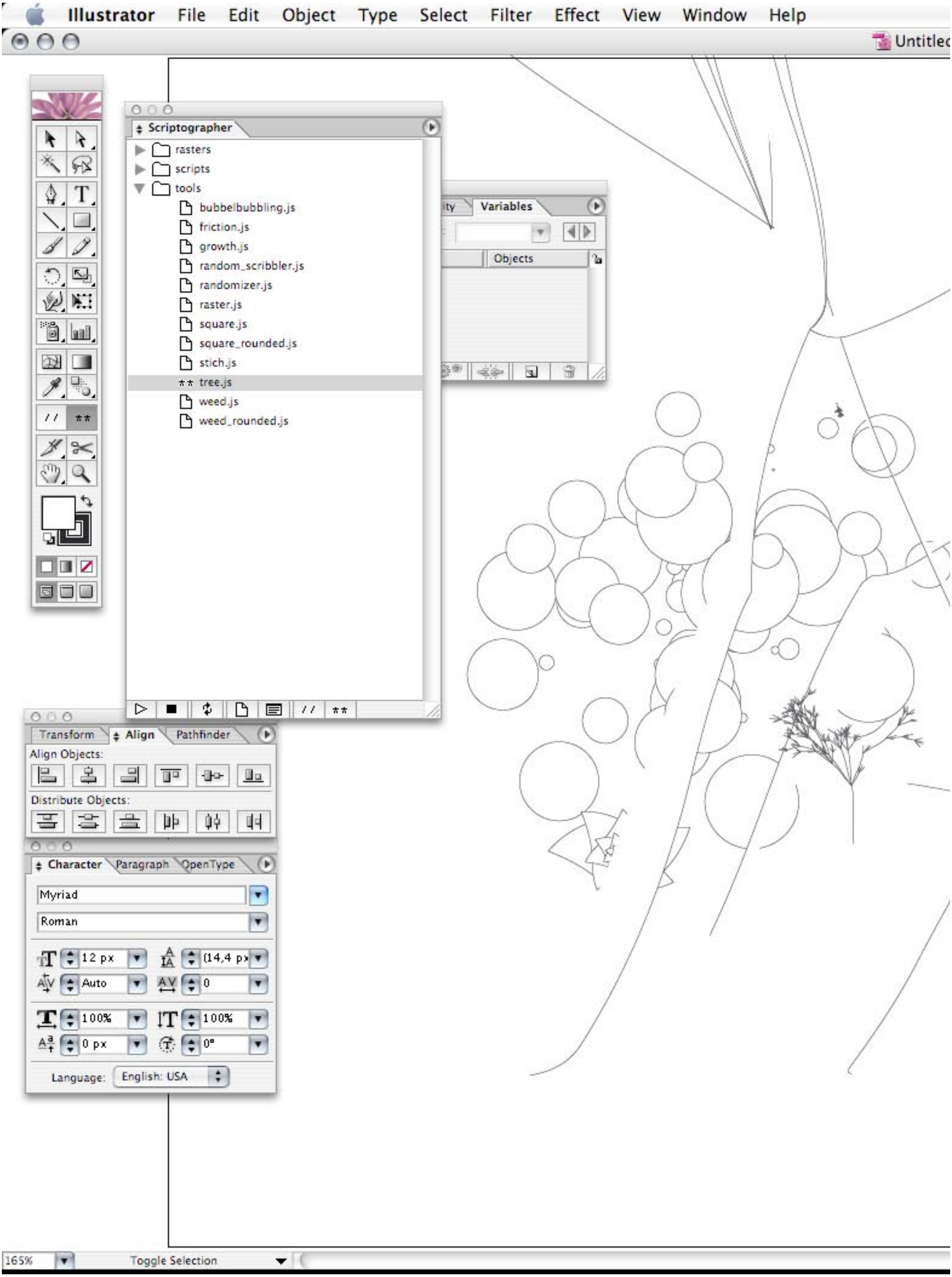
Image Size: 1,93K



PhotoShop
Version 0.63 10/2/88
© 1988 Thomas Knoll
(313) 995-8426

Not for distribution.

OK



Myriad

Roman

12 px

(14,4 px

Auto

0

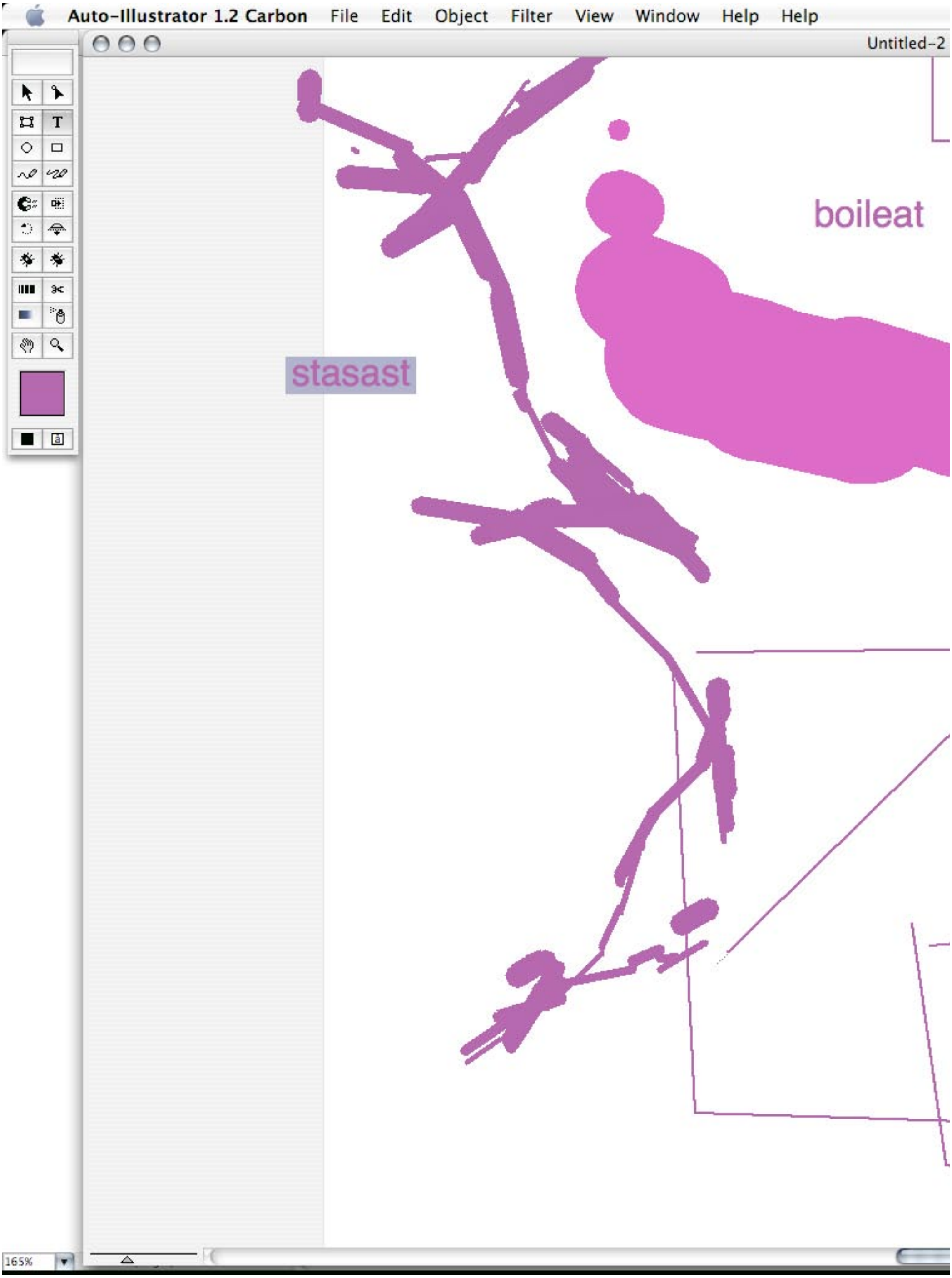
100%

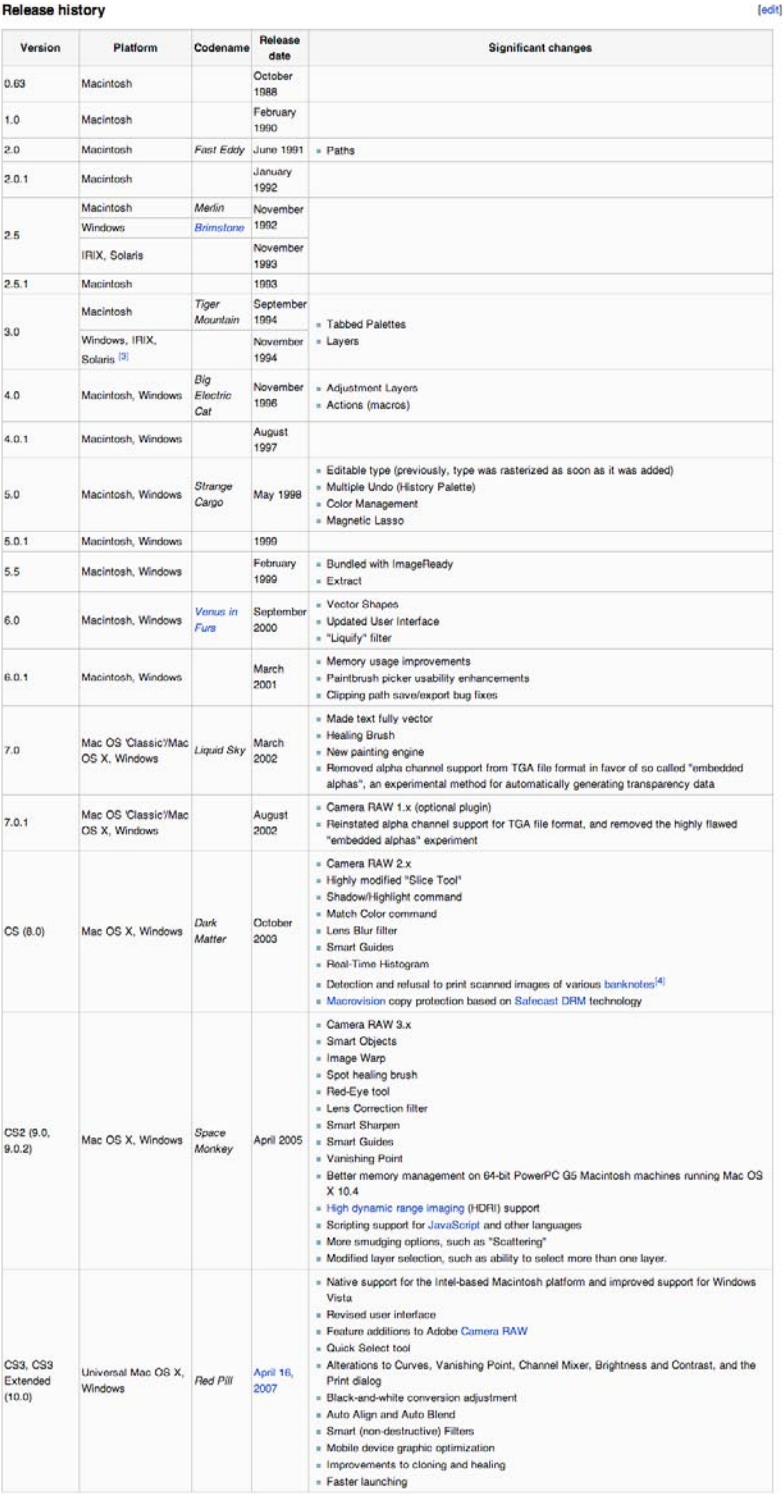
100%

0 px

0°

Language: English: USA


Auto-Illustrator 1.2 Carbon
File
Edit
Object
Filter
View
Window
Help
Help
Untitled-2
boileat
stasast
165%



Release history

[edit]

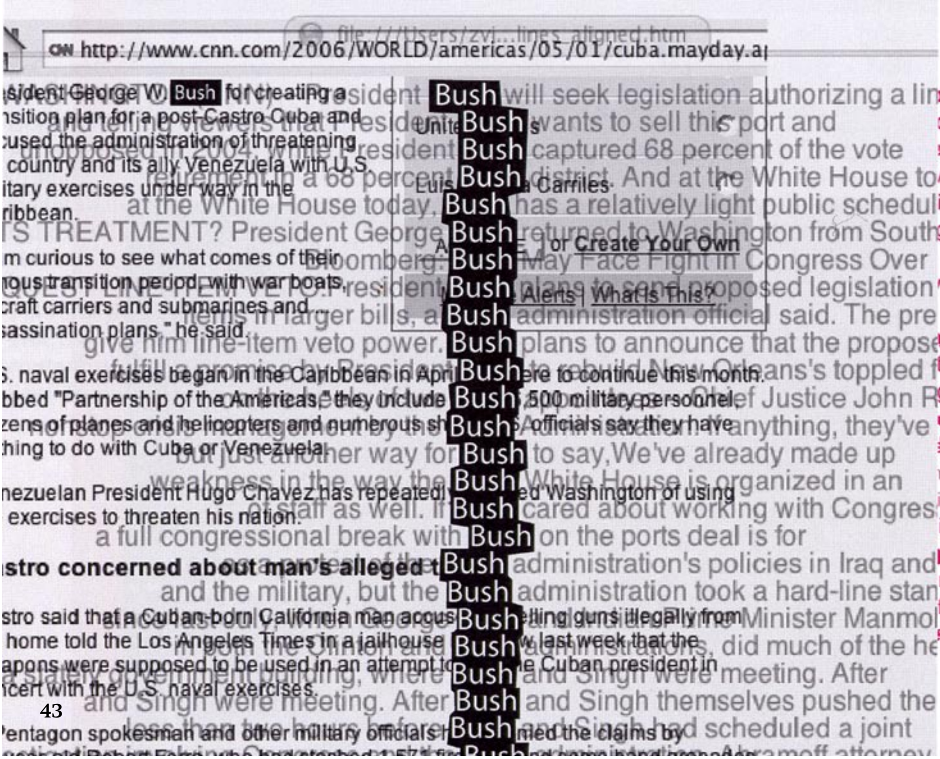
Version	Platform	Codename	Release date	Significant changes
0.63	Macintosh		October 1988	
1.0	Macintosh		February 1990	
2.0	Macintosh	*Fast Eddy*	June 1991	- Paths
2.0.1	Macintosh		January 1992	
2.5	Macintosh	*Merlin*	November 1992	
	Windows	*Brimstone*		
	IRIX, Solaris		November 1993	
2.5.1	Macintosh		1993	
3.0	Macintosh	*Tiger Mountain*	September 1994	- Tabbed Palettes - Layers
	Windows, IRIX, Solaris [3]		November 1994	
4.0	Macintosh, Windows	*Big Electric Cat*	November 1996	- Adjustment Layers - Actions (macros)
4.0.1	Macintosh, Windows		August 1997	
5.0	Macintosh, Windows	*Strange Cargo*	May 1998	- Editable type (previously, type was rasterized as soon as it was added) - Multiple Undo (History Palette) - Color Management - Magnetic Lasso
5.0.1	Macintosh, Windows		1999	
5.5	Macintosh, Windows		February 1999	- Bundled with ImageReady - Extract
6.0	Macintosh, Windows	*Venus in Furs*	September 2000	- Vector Shapes - Updated User Interface - "Liquify" filter
6.0.1	Macintosh, Windows		March 2001	- Memory usage improvements - Paintbrush picker usability enhancements - Clipping path save/export bug fixes
7.0	Mac OS 'Classic'/Mac OS X, Windows	*Liquid Sky*	March 2002	- Made text fully vector - Healing Brush - New painting engine - Removed alpha channel support from TGA file format in favor of so called "embedded alphas", an experimental method for automatically generating transparency data
7.0.1	Mac OS 'Classic'/Mac OS X, Windows		August 2002	- Camera RAW 1.x (optional plugin) - Reinstated alpha channel support for TGA file format, and removed the highly flawed "embedded alphas" experiment
CS (8.0)	Mac OS X, Windows	*Dark Matter*	October 2003	- Camera RAW 2.x - Highly modified "Slice Tool" - Shadow/Highlight command - Match Color command - Lens Blur filter - Smart Guides - Real-Time Histogram - Detection and refusal to print scanned images of various banknotes[4] - Macrovision copy protection based on Safecast DRM technology
CS2 (9.0, 9.0.2)	Mac OS X, Windows	*Space Monkey*	April 2005	- Camera RAW 3.x - Smart Objects - Image Warp - Spot healing brush - Red-Eye tool - Lens Correction filter - Smart Sharpen - Smart Guides - Vanishing Point - Better memory management on 64-bit PowerPC G5 Macintosh machines running Mac OS X 10.4 - High dynamic range imaging (HDRI) support - Scripting support for JavaScript and other languages - More smudging options, such as "Scattering" - Modified layer selection, such as ability to select more than one layer.
CS3, CS3 Extended (10.0)	Universal Mac OS X, Windows	*Red Pill*	April 16, 2007	- Native support for the Intel-based Macintosh platform and improved support for Windows Vista - Revised user interface - Feature additions to Adobe Camera RAW - Quick Select tool - Alterations to Curves, Vanishing Point, Channel Mixer, Brightness and Contrast, and the Print dialog - Black-and-white conversion adjustment - Auto Align and Auto Blend - Smart (non-destructive) Filters - Mobile device graphic optimization - Improvements to cloning and healing - Faster launching

een keuze uit het
eindexamenwerk
1996
13 pagina's mode, grafisch- en produktontwerpen

When Designs Flicker

Florian Cramer

3



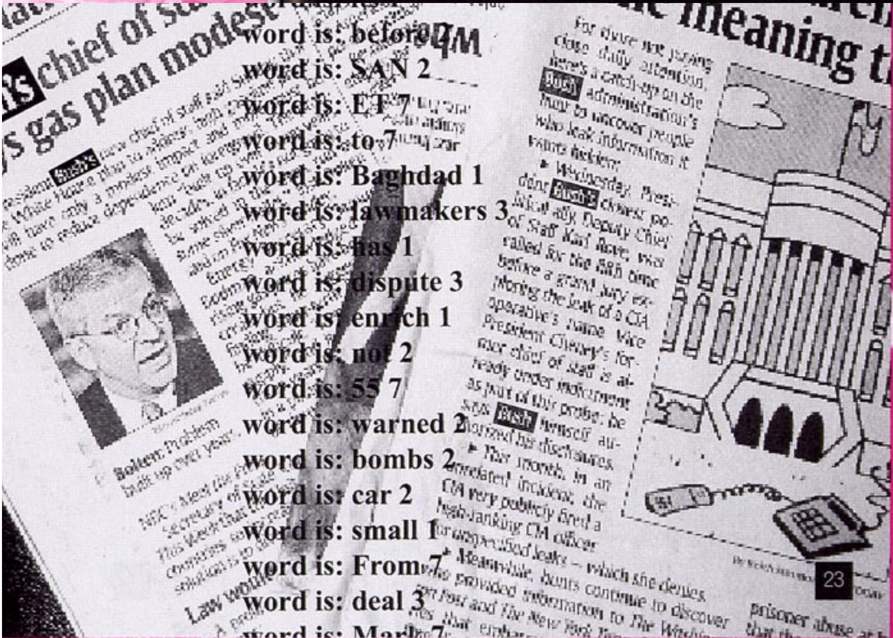
rs from 2004-2006, Dragana Antic,
ins, Kristina Andersen, Matthew

irk, Femke Snelting, Mongrel,

ADECIMAL(2), HTML(3), CSS(3), PYTHON(4),
RCUIT BOARD(1), WIRES(1), PBASIC(1),
TON(4), SOFTWARE(4), FASHION(4), INTERNET(5),
STEMOLOGY(4), RATE THIS ITEM(3),
4), REMOTE(2), LOCAL(5), TANGIBLE(5), GIFT (5),
REPLACE(2), REMOVE(1), ADD(3), CREATE(5),
OKS(3), PRINT(1), DISTRIBUTION(2), WEBSITE(1),
2), MATHEW FULLER(5), PRODUCTION(5),
AD(1), WEBCAM(0), CYBER SEX(1),
AD(4), DOWNLOAD(2), HOUSING(2),

nwardrobe.org

Dragana Antic
Sounds of Qualia



```
word is: before 7
word is: SAN 2
word is: ET 7
word is: to 7
word is: Baghdad 1
word is: lawmakers 3
word is: has 1
word is: dispute 3
word is: enrich 1
word is: not 2
word is: 55 7
word is: warned 2
word is: bombs 2
word is: car 2
word is: small 1
word is: From 7
word is: deal 3
word is: Mark 7
```

• you.

be

fuck?

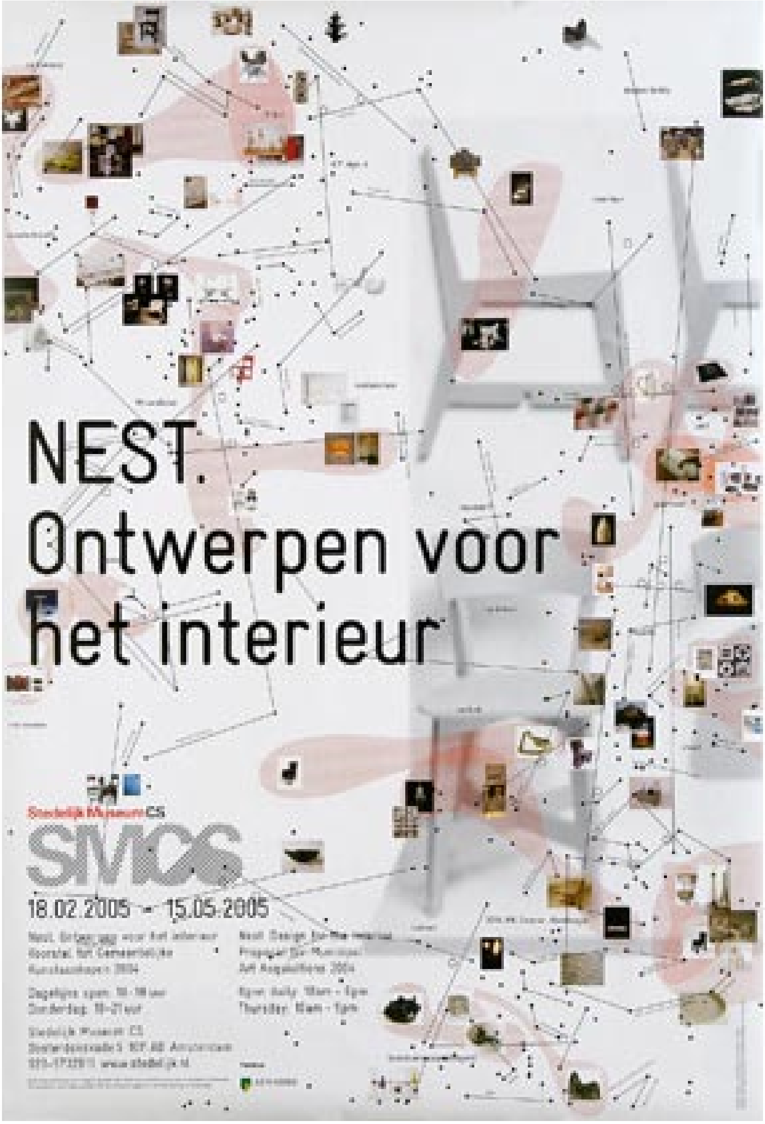
azy.

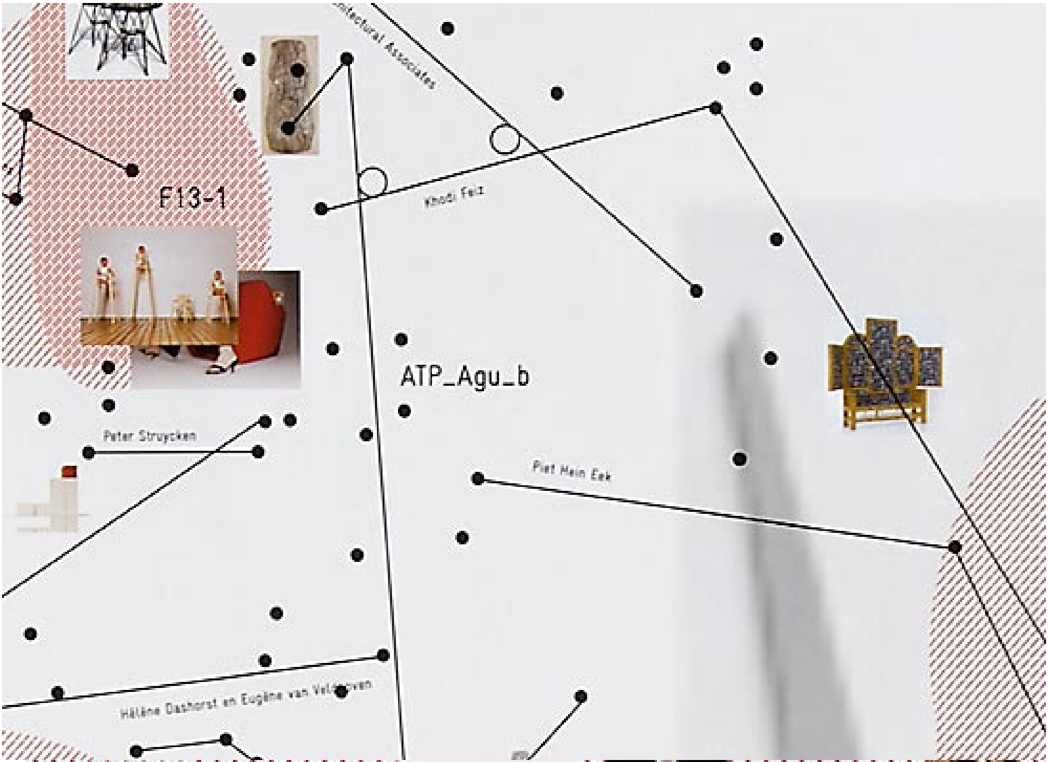
heart

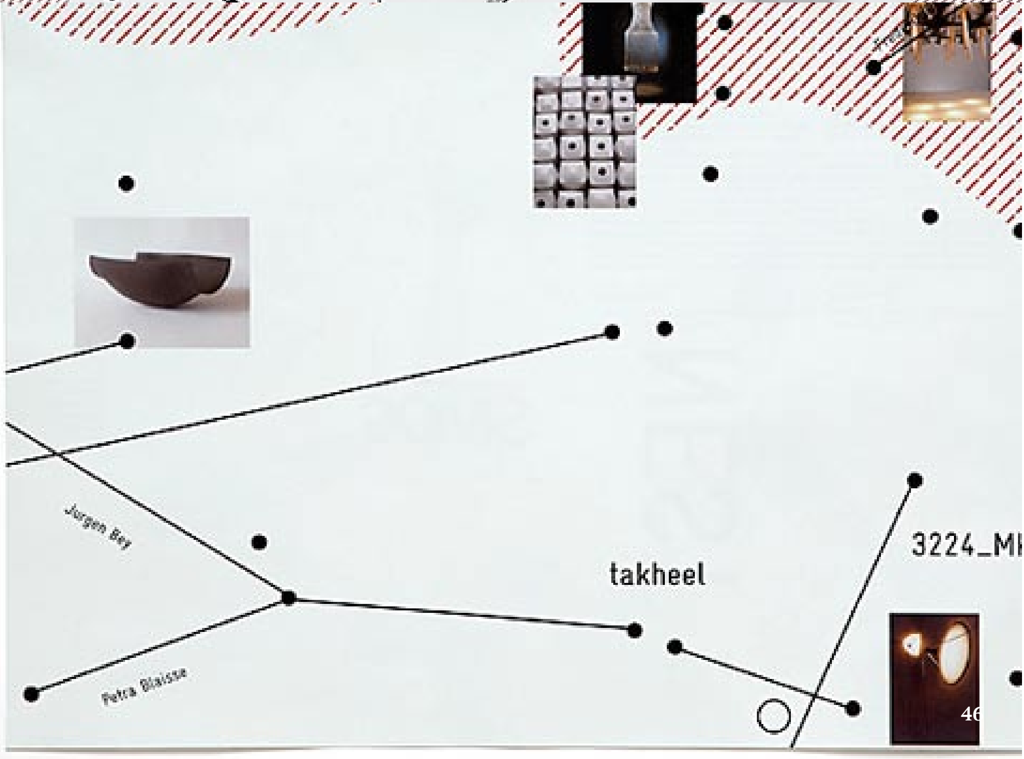
nely.

be

o sad.


NEST.
Ontwerpen voor
het interieur
Stedelijk Museum CS
SMCS
18.02.2005 - 15.05.2005
Nest. Ontwerpen voor het interieur
Voorstel tot Gemeentelijke
Kunstaankopen 2004
Nest. Design for the interior
Proposal for Municipal
Art Acquisitions 2004
Dagelijks open 10-18 uur
Donderdag 10-21 uur
Open daily 10am - 6pm
Thursday 10am - 9pm
Stedelijk Museum CS
Oosterdokskade 5 1011 AD Amsterdam
020-5732911 www.stedelijk.nl


hitectural Associates
F13-1
Khodi Feiz
ATP_Agu_b
Peter Struycken
Piet Hein Eek
Hélène Dashorst en Eugène van Veldhoven


Jurgen Bey
takheel
3224_MK
Petra Blaisse

FINAL ESSAY MAY 2007

design paradigm shifts – shift design paradigms

How old media paradigms still rule software, education and process in the digitised era of graphic design...

Piet Zwart Institute
MA Media Design

by Jorrit Sybesma

```
% puts one at each corner of each form

/onetick { 0 ticklen 2 div rmoveto 0 ticklen neg rlineto ticklen
neg dup neg rmoveto ticklen 0 rlineto 0 setlinewidth stroke} def

/drawticks {gsave ticktrue {0 0 moveto onetick inchoriz 0 moveto
onetick 0 invert moveto onetick inchoriz invert moveto onetick}
if grestore}def

% default variables used by stepnrpt

/srexitproc {} def      % default short exit - don't do it.
/numpages 1 def         % default number of sheets to print
/startnum 0 def         % initial ticket number
/runnum startnum def    % running pointer advances one per repeat
/repeatproc {} def      % artwork to get repeated default
/customdata false def   % autopaginate custom data?

%  /srfile gets used for custom entries. Each entry can be a stri
%  proc, but ONLY ONE ENTRY is allowed per final repeat.

/srfile [(You) (forgot) (to) (define) (srfile!)] def  % default s

% /calcpages is an optional routine that decides how many pages a
% needed ONLY when you are using a custom srfile. This allows ear
% exits when or if you run out of data.

/calcpages {dup cvn stepnrptparams exch get dup 0 get exch 1 get
cvi srfile length exch div ceiling cvi /numpages exch def} def

%  This is the main stepandrepeat tool...

/stepandrepeat { save /srsnap exch def mark exch /quickexit false
customdata {calcpages} if setrepeatparams numpages {gsave landsca
{-90 rotate -792 0 translate} if horstart vertstart translate gsa
numhoriz {gsave numvert {drawticks save /rptsavel exch def repeat
rptsavel restore /runnum runnum 1 add def 0 invert translate sre
repeat quickexit {exit} if grestore inchoriz 0 translate} repeat
{exit} if grestore showpage grestore} repeat quickexit {showpage}
cleartomark srsnap restore} def

% To force an early exit when you run out of names or reach a giv
% ticket number, test suitably. Then conditionally make /quickexi
% true and exit repeatproc.

% Note that an ending showpage is NOT required and should NOT be

% //////

% (2) curvetrace
% . . . . . .

% curvetrace - creates a smooth curved path from a data point lis
%              enter with currentpoint set and absolute array.
%              0 0 as initial data appends path; any other values
%              creates new path

/curvetrace {/curvelist exch def tension 0 eq {/tension .000001 d
           curvelist length 3 div 1 sub cvi /#triads exch def
           /ptr 0 def firstpoint morepoint} def

/tension 2.83 def   % default value for best fit
/showtick false def % don't show points
/ticklen 15 def     % length of ticks
/tickhead ticklen 4 div def

/prvx { curvelist ptr 3 sub get } def
/curx { curvelist ptr get } def
/prvy { curvelist ptr 2 sub get } def
/cury { curvelist ptr 1 add get } def
/prva { curvelist ptr 1 sub get } def
/cura { curvelist ptr 2 add get 180 sub} def

/showtic { showtick true eq {gsave currentpoint newpath translat
180 add rotate ticklen neg 2 div 0 moveto 0 ticklen rlineto tickh
dup rlineto dup tickhead  rlineto tickhead dup neg exch rlineto 0
setlinewidth stroke 0 ticklen neg 2 div moveto 0 ticklen rlineto
grestore} if }def

/firstpoint { curx cury 2 copy abs exch abs add 0 eq {pop pop cur
curvelist exch 1 exch put curvelist exch 0 exch put}{moveto} ifel
showtic /ptr ptr 3 add def}def

/morepoint {#triads { curx prvx sub dup mul cury prvy sub dup mul
tension div /zdist exch def prva cos zdist mul prvx add prva sin
prvy add cura cos zdist mul curx add cura sin zdist mul cury add
```