Perspectives on Aesthetic Computing

Paul Fishwick, Stephan Diehl, Jane Prophet, Jonas Löwgren

***Abstract*— We present an introduction to the new interdisciplinary area of *aesthetic computing* and then define this area with examples from each of our own disciplines, practices, and research. While several decades of publication and work have resulted in significant advancements in art, as instrumented with technology, less emphasis has been placed on studying the converse issue of art affecting computing, or "aesthetic computing." We present our individual work in this area, and then critique each others' work to surface different perspectives of the area. By approaching the topic of aesthetic computing in this manner, the paper serves as an introduction, survey, and analysis of the field.**

***Index Terms*— Aesthetics, Art, Semiotics, Computing, Programming, Interaction Design**

I. Introduction

IN collaboration with Leonardo, we participated in a workshop on *Aesthetic Computing* in the hills of southwestern Germany in July 2002. The workshop [1], organized by Paul Fishwick, Roger Malina, and Christa Sommerer, was one week in duration, and was attended by over 30 representatives of the following disciplines: art, design, computer science, and mathematics. The purpose of the workshop was to define the area, and to try to surface key aspects of the field from a variety of perspectives. A manifesto of approximately one page was recently published in Leonardo [2]. Aesthetic Computing is the theory and application of art to computing. Given this brief definition, several questions come to mind and we shall address these questions, and then proceed with the purpose of this paper, which is to cover sample projects in the area and the different ways in which the authors proceed to accomplish their singular crafts.

A question regarding the name "Aesthetic Computing" may well be one requiring us to justify its existence. After all, isn't all computing within the realm of aesthetics, and isn't there already a significant number of projects that capture some combination of computing and art? Let's begin with the first question regarding aesthetics. We will define aesthetics broadly as a combination of cognitive and sensory modes of experience, which is generally associated within the definition of the philosophy of art. However, if we treat only the purely cognitive aspect of aesthetics, we find evidence of this within mathematics [3] and computing. One speaks of an elegant proof for a theorem, or a beautiful representation. With such qualifiers, the mathematician is usually referring to cognitively-grounded aesthetics. When we speak of aesthetics in this article, we take it to mean the range and variety encompassing cognitive and sensual aesthetics characteristic of art. As Knuth points out in his discussion of Metafont [4], underlying his TeX typesetting system, "Type design can be hazardous to your other interests. Once you get hooked, you will develop intense feelings about letterforms." Taken more generally, Knuth is directly addressing the issue of aesthetics as being more than the purely cognitive, even within computing, which grew out of mathematics. A textual section of a computer program will have both denotative as well as connotative signifiers, and it is easy to imagine that the program might align itself with the goals of art, thereby stretching the traditional boundaries of what may be considered to be a usable computer program representation. The second question regarding the combination of science with art is addressed by first noticing that, unlike computer or digital art, aesthetic computing implies that art is affecting—and reflecting—some aspect of computing. This results in quite a different agenda than that found in many other fields of digital art. It may be that taken to its logical conclusion, art affects computing and that this, in turn, affects art, closing the loop: art creates art. However, we are presently a long way from that proposition.

We acknowledge our interest of aesthetics in computing to expand human-computer interaction and representation to reach into numerous areas of computer science from the primary operating systems interface to the interface used by computer scientists to create programs, and by scientists to create models of geometry and dynamics. This leads to two observations, partially justifying the move toward aesthetic computing: (1) aesthetics in computing are broader than the purely cognitive dimension, and (2) the art-science confluence embedded within the discipline of interaction design is broader than the primary "desktop" interface. The first observation is one of both ontology and epistemology—that while we might leverage existing aesthetic principles toward the sensory. Consider the case of "software patterns" as one

Manuscript received X, 2003.

P. A. Fishwick is with the Computer and Information Science Department at the University of Florida (corresponding author), e-mail: fishwick@cise.ufl.edu.

Stephan Diehl is with the Computer Science Department at the Catholic University Eichstätt, Germany , email: diehl@acm.org

Jonas Löwgren is with the School of Arts and Communication, Malmö University, SE-205 06 Malmö, Sweden, email: jonas.lowgren@k3.mah.se.

Jane Prophet is Director of The Centre for Arts Research Technology and Education (CARTE), University of Westminster, London, email jane@carte.org.uk

example. One would surely state that these patterns reflect abstractions of software structures, and that they are found lurking in numerous software applications. There are two ways at looking at patterns, reflecting two ways of defining aesthetics: cognitive versus material. What if the patterns were surfaced in a way that was more attuned to material embodiment? This would build upon the existing pattern literature and extend it. The "factory" method espoused by Gamma et al. [5] could be the basis for a 2D or 3D scene which looked, and operated, like a factory. The look and feel of the factory would improve with artistic influence and guidance, and provide strong metaphorical cues. This sensory dimension seems lacking from the pattern literature since one generally finds that representation is limited to rectangles and arcs. And yet, it is not only in the pattern literature that it is lacking, since the visual minimalism of program structures, and the mathematical structures underlying them, is fairly common in computing. The second observation is that interaction design needs to concern itself with all interaction and presentation found in computing, including that of how to represent data and program structures, for instance. The emerging areas of Information Visualization [6] and Software Visualization [7,8] render themselves toward this goal, and yet the representations of information and software could do with greater emphasis on a wider range of artistic expression, without sacrificing utility. So, it is not that the area of design does not presently concern itself with incorporating aesthetics, but that the current level and degree of this incorporation needs to be expanded beyond that of the typical *user interface* at the operating systems level.

We now have a working idea of aesthetic computing, and we need to explore different approaches. The structure of the paper begins with four statements on aesthetic computing by each of its authors, followed by a discussion of each others' views and statements.

II. Formal representations (Fishwick)

A. Scope

In applying aesthetics to computing, we need to confine ourselves to some aspect of computing, or one of its subfields such as HCI, visualization, or discrete structures, to name a few. For the RUBE Project [9-12], we have focused primarily on representations, informed through an artistic sensibility, in mathematics and computing notation, from the notation of algebraic and differential equations to that of program and data structures. Our basic philosophy is that we want to build a system that allows a multiplicity of different notations to be constructed so that one may see and hear the same underlying formalism in numerous ways. Not only do different people and cultural entities enjoy working with a formalism using different metaphors, but also the same person or group can benefit from being exposed to diverse presentations.

B. Implementation

At the University of Florida, we have constructed a software system called RUBE, which allows for different representations to be applied to a select number of formal dynamic model specifications. Using RUBE, it is possible to change the way that formal models look and sound. By formal models, I am referring to a large class of models used for specifying systems that incorporate time for analysis and simulation: finite state machines, Petri networks, Markov models, queuing models, System Dynamics graphs, as well as ordinary and partial differential equations. RUBE uses XML (eXtensible Markup Language), which separates content from presentation. In XML parlance, *content* refers to an abstract specification defined as a document tree, and *presentation* refers to the way that the tree is presented to the user, the way it looks and sounds. Thus, using RUBE and guided by the XML philosophy, one may specify an equation, but then choose to present the equation as linear text, a network, or a three dimensional structure. Choices as to which presentation to employ can be guided by XML style sheets and their associated transformations.

RUBE's architecture is based open source software, and begins with authoring toolkits: SodiPodi for 2D vector drawing, and Blender for 3D modeling. Let's consider the 3D pipeline beginning with Blender. The artist creates a 3D model in Blender, and then uses a Python scripting interface allowing attributions to be made regarding semantics. For example, one might point to an object and say that this object is a *state* or a *function*. After the semantic assignment, an X3D (eXtensible 3D) file is created for the presentation, and a special XML file is created for specifying the formal model. After some XML transformations, this XML file is translated into Javascript or Java, whereby it can be reincorporated into the VRML file, resulting in an interactive VRML world. The 2D transformations are similar, except that SVG (Scalable Vector Graphics) is used for presentation.

Let's begin with a formal definition of a Finite State Machine (FSM) M [13]. These machines have states that are connected to each other through transitions that are made active when an input to the machine is of a particular value. Here is a formal definition for M:

$$M = < Q, I, O, \delta, \lambda >$$
$$Q = \{S1, S2, S3\}, \delta : Q \times I \rightarrow Q$$
$$\delta(S1,0) = S1; \delta(S2,0) = S2; \delta(S3,0) = S3;$$
$$\delta(S1,1) = S2; \delta(S2,1) = S3; \delta(S3,1) = S2$$
$$I = \{0,1\}, \lambda : Q \rightarrow O$$

Even though this text might seem to be the formal specification for M, it is actually one of many ways to look at the underlying formalism which would is encoded in XML. The above text is one type of presentation among many. In general, all presentations require additional natural language semantics if we are to make sense of them. Q is the state set for M; I the input set, O the output set, δ the transition function from one state to another, and λ the output function. Figure 1 illustrates our second presentation for the FSM. It has

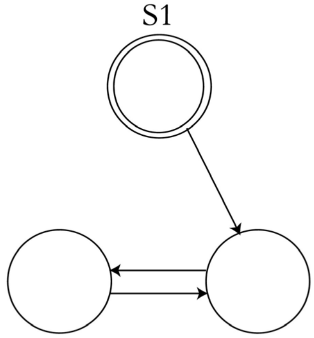

Fig 1: A 2D static snapshot of an interactive diagrammatic FSM interface

iconic properties so that when the machine is in state S2, the presentation of a circle for S2 encodes the concept of a *boundary*, and that which is *inside* it. That is, the graphical depiction of S2 is consistent with the underlying metaphors of set theory, whereas the purely textual presentation does not capture these metaphors. Moreover, Fig. 1 is incomplete on a non-interactive medium, such as paper, since the additional information encoded in the text representation is equally present during interaction with the figure. Similarly, the arrows capture the notion of transitioning from one state to another, since anyone who has seen an arrow fly knows that it is aimed toward a target. The metaphors of the figure dramatically improve the semantics of the machine, and so one is led to wonder whether employing presentations with alternative aesthetics might further improve the impact of the metaphor. The underlying assumption is that material aspects of levels of representation are based largely on what is available for a society, and what is affordable and materially efficient. Consider Figure 2 as a representation that has only recently become possible through computer graphics, and the ability to employ 3D components. The metaphor of the circle as a boundary has been replaced by a small gazebo-like structure. The arrow in Fig. 1 is now shown as a red-clothed woman walking from one state to another along a lamp-lit walkway.

There are a host of philosophical issues that come into play here. Isn't there a need to enforce visual minimalism within this sort of structure? What are the cultural barriers imposed that might prevent the adoption of models like Fig. 2 being made possible for science and engineering? With respect to the issue of minimalism, we should note that is quite possible to maintain abstraction without requiring visual minimalism. Within the context of the art community, this can be seen when comparing and contrasting the genres of Abstract Expressionism and Surrealism. Both of these genres contain a wide variety of works that employ symbolism, iconography, and the richness of semiotics even though the visual

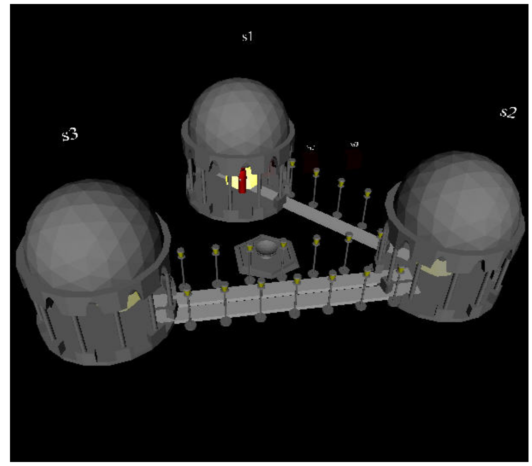

Fig 2: A 2D agent-based presentation of the FSM using the Virtual Reality Modeling Language (VRML).

presentations are strikingly different. Deriving the idea of an abstract state in an FSM, for example, need not imply that the state be presented visually in a minimalist fashion. The key thing is to strengthen the metaphor underlying what it means *to be a state*, and the corresponding elements of boundary that go along with it.

The second question about cultural barriers may be at the heart of the aesthetic computing challenge. Computer scientists have been educated with minimalist figures and text, and so it may come as a shock to realize that our representations for formal objects are not as constrained as we may have thought. Until the era of computer graphics and fast computers, we had little need to inquire about, what initially appeared to be, exotic ways to encode formal knowledge. However, it is not only a challenge for computer scientists, but also for artists, since artists should be encouraged to consider the computer, and computing practices, as *subject material* in addition to raw material. This suggestion on formal structures as raw or subject material may strike some artists as a modernist era agenda; however, there the computer, and its mathematical foundations, creates significantly higher complexity as a tool, or as a subject, than paint, palette knife, or chisel ever could.

III. SOFTWARE VISUALIZATION (DIEHL)

Software is neither matter nor energy. It is just a kind of information. Matter and energy are media which carry information and thus software. To develop new or understand existing software it has to be projected into a human readable form – the program text. Note that the program text is not the software, but just a representation thereof. The program text is written in an artificial language with a strict syntax and a more or less well-defined semantics. Trying to understand a real software system by reading its millions of lines of program text is a vain task. As a consequence many tools have been developed to support software understanding. These tools rely on analysis and visualization techniques. Per se, software is invisible. In a famous paper [14], Turing awardee Fredrick Brooks even stated that “software is invisible and unvisualizable” because each kind of visualization only addresses “one dimension of the intricately interlocked software elephant.” These dimensions include the static structure of the software, its dynamics and its evolution. Or to put it in other words, different kinds of visualizations show how software is encoded, how it behaves, and how it is developed. In the following we will present and discuss examples of visualizations for each of these dimensions.

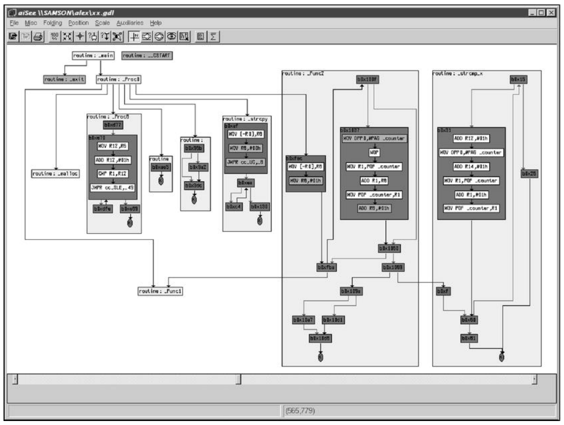

Fig. 3: Program structure with analysis results.

In Figure 3 a graphical representation of a program, its control-flow graph, is shown. In addition the graph contains some information computed by a program analysis. With the help of this visualization developers can detect certain kinds of errors, so-called stack overflows, in their programs [15].

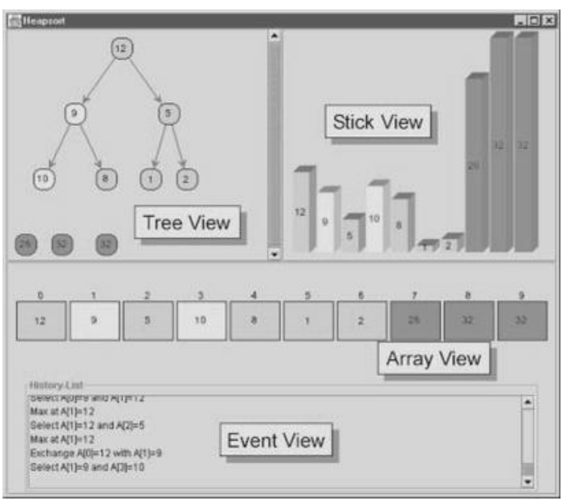

Fig. 4: Animation of the HeapSort algorithm

In Figure 4 a snapshot of the animated execution of a sorting algorithm is shown. The window contains several representations or views of the data sorted. Algorithm animations are typically used in education. Most of the animation techniques do not scale for real software systems.

Finally, Figure 5 is a pixel map. In this example the color of the pixel at position (x,y) represents the number of times file f_x and f_y have been changed together relative to the total number of times file f_x has been changed. From this figure the developer can see how strong different files are coupled. We call this kind of coupling evolutionary [16], because it is based on the change history of files, to distinguish it from the logical coupling usually used in software engineering. As the files are sorted by the containing directory the pixels form blocks. These blocks indicate that files within a directory are coupled, i.e. often changed together. Software developers are mainly interested in the outliers. These are those pixels representing couplings between files in different directories, like those labeled “Patches” in Figure 5. Outliers can be a sign of a bad system architecture.

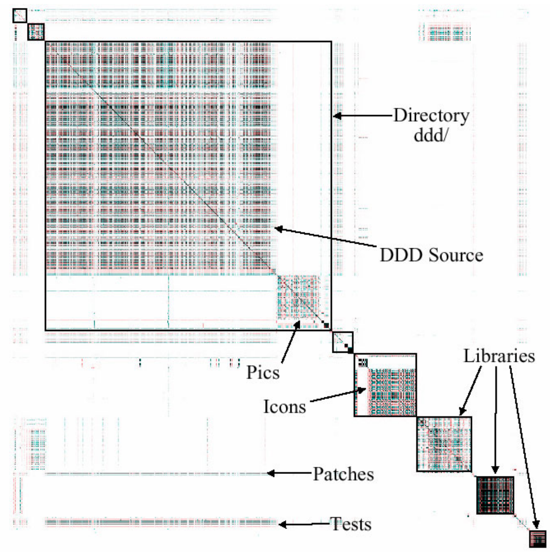

Fig. 5: Number of simultaneously changed files

We have now seen three very different kinds of visualizations as they are used for understanding software and its development process.

Now what about the aesthetics of these visualizations? It may come as a surprise to the artistic eye of the reader, but there have been actually some aesthetic criteria involved when computing these visualizations:

In Figure 3 the number of edge crossings and bends has been reduced and directed edges are mostly drawn downwards. In Figure 4 color is used consistently in the different views. While the algorithm actually performs discrete changes, the animation of these changes is done smoothly. In Figure 5 color coding based on the heat metaphor is used: The color red shows the highest coupling, blue low coupling and white now coupling at all. In other words, hot files are those that are often changed together.

The sole purpose of the above mentioned aesthetic criteria is to produce visualizations that convey the information as clearly and effectively as possible. As a consequence of concentrating on the automatic generation and usability of visualizations the current research on software visualization is rich when it comes to the different properties of software that have been visualized, but poor when it comes to the spectrum of visual metaphors used: boxes, circles, lines and color.

IV. Interdisciplinarity and Aesthetic Difference (Prophet)

I will consider the interdisciplinary area of aesthetic computing through a discussion of the interdisciplinary project, Cell. Cell is a collaboration between an artist (the author), liver pathologist (Neil Theise), mathematician (Mark d'Inverno), computer scientist (Rob Saunders), and curator/producer (Peter Ride). The Cell project explores new approaches to cell behaviour and its representation and uses mathematics to bridge the gap between scientific theory and computer visualisation. Our premise is that artists can 'imagine' scientific and mathematical theories and thereby influence the development of scientific, mathematical, and computer science research and their associated aesthetics.

A. *Context*

Our practical research context is a triangle of different experimental research environments: Theise's medical laboratory; d'Inverno's and Saunders' respective mathematical and computer science labs; and Prophet's artist's studio. Each provides a different context for the work and has particular embedded methodologies, from the hypothesis driven ethos of the medical research lab, to reflexive practice in the art studio, and the empirically driven environment of mathematics. The aesthetic context of each discipline is equally diverse. Visualisation in the laboratory differs from visualisation in the art studio, aesthetics are important to both. In the medical laboratory representation is usually taken literally, leading to scientific illustration. In the art studio 'representation' is a term and process framed by debates in cultural theory and numerous theories of representation (for example, an image, sound, object can signify something without actually sounding or looking anything like it). In Cell we are taking account of a wide range of aesthetic traditions, including the aesthetics of mathematical equations to produce a range of practical outcomes (including animated 3D illustrations of cells, mathematical models, Zen gardens, sound pieces, and robotic artworks). In addition we document, develop and evaluate the interdisciplinary collaborative process itself. Through a series of studio and laboratory visits, Theise and I became immersed in each other's working culture and identified significant cultural differences. For example, in cell biology the 'photographs' of tissue slides have a truth status, and are accepted as 'proof' of experiments and hypotheses within papers. The beauty of these representations (see Fig.6) produced as part of his laboratory research, is important to Theise and there appears to be a correlation between aesthetic quality (specifically how 'beautiful' a representation is) and the publication rate of associated papers. Fig. 6 is one of Theise's images, and represents skin tissue from a female mouse who received a bone marrow transplant from a male mouse. Blue nuclei of hair follicle lining cells surround the orange, autofluorscent hair shaft (large arrow). Two of these nuclei contain fluorescently labeled Y-chromosomes (small arrows) indicating that they derive from the donated male bone marrow, not from the female's own original cells. Thus, bone marrow stem cells have given rise to skin-type lining cells.

By contrast, in contemporary art practice photographic representations have no automatic truth-status (quite the contrary) and are assumed to be subjective rather than objective. Artificial life (Alife) systems that have a graphical and/or sound output pose other challenges to notions of representation. Their outputs are autonomous, not controlled or 'made' by the artist or illustrator. They are time-based, not still. They are not constant or predictable as visual or aural outputs are produced in real-time to represent the software running beneath them, which is itself constantly changing as a complex system of interactions between entities takes place. They are not 'top-down' but 'bottom-up' and potentially emergent: unexpected behaviours represented as images or sounds emerge from many interactions based on simple rules.

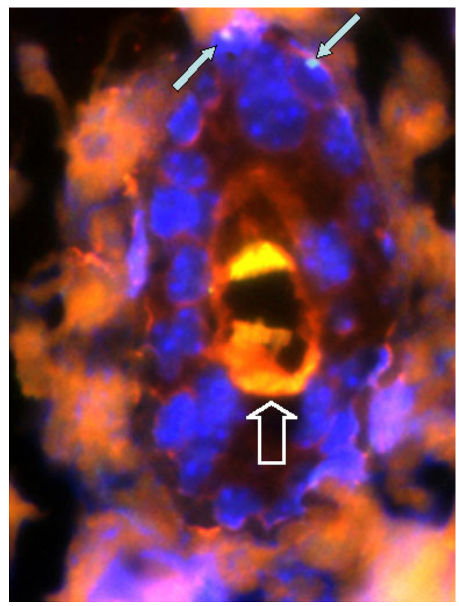

Fig. 6: Hair follicle with lining cells derived from bone marrow

B. *Converging: focussing on behaviour across time*

Having orientated ourselves to each other's research environments we identified key concepts and contexts on which to focus, and converged and focussed on those areas. Theise's research examines the plasticity of adult stem cells and their function. To do this he uses processes based on hypothesis and hypothesis-testing including repeatable laboratory experiments and the analysis of specimens of cell tissue. Because the tissue is dead at the time it is analysed it represents a frozen moment in time, from which researchers understand another aspect of stem cell behaviour, and extrapolate further hypotheses to test. My experience as an artist working in time-based media and Alife suggests a different approach to assessing stem cell behaviour. We therefore decided to develop an Alife engine to enable the scientist to look at *simulated stem cell behaviour as it happens* within the complex system of a wider community of cell types and enzymes. This output draws on the aesthetics of medical illustration and expands it by making the connection between changing image and underlying behaviour accessible (click on a moving cell and see Fig.7, a data read-out).

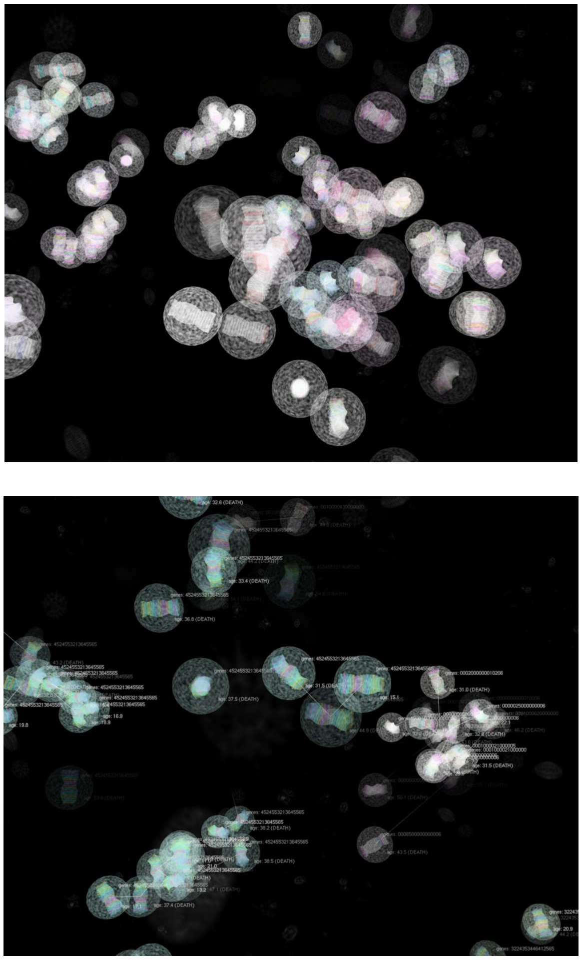

Fig 7: Cell 3D Java alife representations

C. Collaborative process, aesthetic impact

The collaboration expanded to include a mathematician with expertise in agent-based systems (d'Inverno) to determine the mathematical rules that describe adult stem cell behaviour as proposed by Theise (that stem cells can evolve into mature cells of other organs such as skeletal muscle, bone, and brain with unexpected plasticity). A computer scientist (Saunders) interprets the mathematical model and produces a real time graphical display. Interface features enable scientists (such as Theise and d'Inverno) to make changes to stem cells and see the results; map changes through generations of stem cells; see behaviour displayed as real-time animation and graphs. The aesthetic is that of medical and scientific illustration, and in common with medical illustration, a team has produced this scientific visualisation. Group/multiple authorship is an accepted part of scientific publication. This is not the case in art practice. Standing against the canon of the artwork as a product of a sole 'genius' artist, are conceptual and digital artworks where collaboration is more common. Public acknowledgement of collaborators is a significant part of the aesthetic framework of such art practice and is in marked contrast to the unacknowledged appropriation of "assistants'" work that has traditionally dogged the fine arts.

D. Aesthetics and Conceptual art

The notion of 'an' aesthetic is contentious in Cell. Like much conceptual art, the idea behind Cell (namely modelling the behaviour of stem cells) and the means of producing it (via interdisciplinary collaboration) are more important than the finished work or its (fixed) appearance:

"In conceptual art the idea or concept is the most important aspect of the work . . . all planning and decisions are made beforehand and the execution is a perfunctory affair. The idea becomes the machine that makes the art." [17]

Conceptual art can be defined as the "appreciation for a work of art because of its meaning, in which the presentation of shape colour and materials have no value with out the intentions of the work." [18] If conceptual art has an aesthetic then it is the dematerialisation of the art-object; the object only has value as a materialisation of the idea, not in and of itself. Mathematics and computing science can both operate without materiality and can describe the immaterial, which is one reason why there may be a mutual attraction between computer scientists, mathematicians and artists working conceptually using digital media. The robot artwork produced as part of Cell will comprise of objects and their functionality and appearance is important, but the idea behind the artwork is of equal, or more importance. The idea itself can be seen as part of the developing aesthetic of digital and other contemporary arts: the development of an aesthetic of virtuality and engagement.

E. Scale

Aesthetic debates concerning scale are central to the Cell projects. I have been interested in the 'sublime' in contemporary culture for a number of years, in particular, fractal mathematics and an apparent cultural shift to a sublime of the very small and detailed. This develops ideas of the 'natural' or 'religious' sublime [19], based on our experience of the human body in landscapes so large and overwhelming that they prompt a sense of awe and momentary terror. I suggest that there is now a sublime of the micro, nano and virtual, a similar awe and terror as we try to grasp an inner landscape of a scale too small in relation to the human body for most of us to comprehend. Theise's theory challenges the paradigm of the progressively differentiating adult stem cell (and of cells being one of the body's smallest 'building blocks'). He draws attention to the role of technology (the microscope) in assigning high status to the unit of the cell (bounded by the cell wall made visible for the first time via microscopes). Determining the bounded cell as a key 'unit' is central to our thinking of scale and reinforces a reductive model of the human body in medicine. Theise notes that if a different imaging technology had been invented (instead of the microscope) that, for example, showed patterns of energy or fluid movement through the body, then we would be less inclined to reductionism. Any subsequent development of the microscope would have qualified the 'fluid' model to note that fluids sometimes moved across boundaries (i.e. across cell walls).

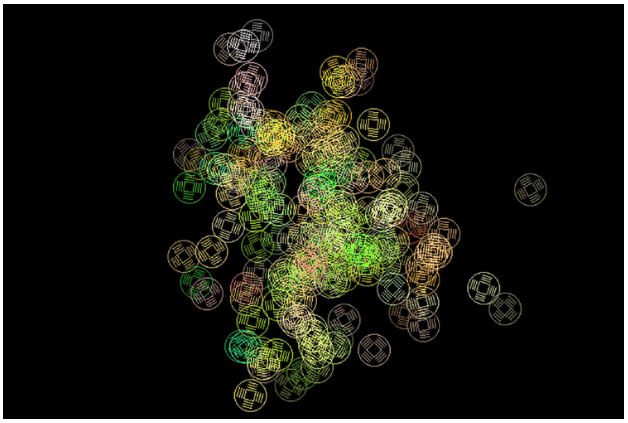

Fig 8: Cell 2D Cell alife java representation

We have discussed the modeling of stem cell behaviour and the role of mathematics in exploring and representing shifts in scale. In choosing to 'manifest' the mathematical model by developing software written in java, Saunders kept the project sensitive to scale at the level of the computer code itself. Java language was suggested by Saunders because of its scalability, providing the project with a tool that could be used on many different types of computer, and on computers with a variety of speeds etc. (see Fig 8). Scalability has become central to the ethos used to design the Cell software. Discipline-specific theories of 'scale' converge in Cell to become a key theme and aesthetic.

F. *Aesthetics and Visualization*

To date, the real-time graphic representation arising from Cell's complex adaptive system is not what I would consider a piece of art. It is a scientific visualisation, informed by the aesthetic framework of my art practice. As detailed below, conceptual art has influenced our approach to making Cell. In developing the look and feel of the scientific visualisation, I have emphasised to the medical and computer scientists that I work with that as a contemporary artist I have been educated to resist *beautifying* the graphics. As far as possible, I want the graphic look and feel to reflect the underlying software, to draw attention to essence of the idea or concept with as little frippery and decoration as possible. From this standpoint the java version is a more satisfying outcome than the 3D version. The 3D version has been influenced the aesthetics of medical illustration and its goal of explaining via precise observation of *the appearance of things*: this is at odds with my emphasis on *the behaviour of things* (in this case stem cells). What is key for me as an artist working on the project is to transmit a sense of 'stem cell-ness'. What is captivating about our contemporary understanding of these cells seems to be the way that they behave rather than the way they look. The aesthetic framework in flight and gaming simulations has also contaminated the 3D version of Cell, in particular the characteristic focus on surface rendering, depth and transparency reflects a relentless drive towards photo realism. This is not what I believe to be of most interest as an artist working with software. Photorealism is informed by the medium, material and aesthetic of photography. I am interested in engaging with the medium of code and software and in trying to use materials in keeping with that, and to develop a look and feel that contribute to a computer aesthetic, for example that take account of the lack of depth of field and infinite focal range of virtual space. Saunders and I have discussed these issues at length and he is interested in developing a 3D version of Cell that avoids the criticisms I outline above. We are committed to developing a 3D version because that is closer to our model of stem cell behaviour than the 2D java version. We will attempt to reflect the characteristics of virtual space (no depth of field, infinite focal range etc.) as we do this.

V. INTERACTION DESIGN & AESTHETICS (LÖWGREN)

Interaction design is concerned with shaping the use qualities of digital artefacts. Another way of putting it would be to say that interaction design is design of the digital materials, where the word design is used in the strong sense of exploring all aspects of a possible future: aesthetical and ethical aspects as well as structural and functional.

It may seem odd, and in fact it is odd, to talk about aesthetic computing as if it were something new and hitherto unexplored. All computing is aesthetic in the sense that all use of digital artefacts entails aesthetic reactions. To be sure, many contemporary digital artefacts tend to elicit aesthetic reactions along the lines of frustration, indifference or boredom, but these are aesthetic reactions nevertheless. And as designers, we are free to aim for other kinds of reactions if we like.

Of course, there are historical reasons for the existence of such blind spots. The academic roots of interaction design stem from the field of human-computer interaction, where the main focus has always been on task-oriented use of digital artefacts and in particular its efficiency and absence of errors. It is not surprising that the image of computer use is sometimes simplified to the extent that useful, efficient, and error-free use is seen as the whole picture. The disregarded parts of the picture, including aesthetic qualities of the use, will then appear as new when they are brought into consideration.

How, then, should aesthetic use qualities be dealt with in interaction design? A common fallacy is to equate aesthetics with pleasing visual design. To be sure, there are design situations where the immediate visual impression is an important factor in determining the outcome of the interaction. A good example is a web shop with one-time customers who should ideally spend money on their first and only visit. The visual design of the web shop's front page is crucial in establishing the right combination of desire and credibility to actually make customers enter, shop and pay.

In nearly every other case, however, we need to realize that a digital artefact is constituted primarily not by its static visual design but by its *dynamic Gestalt* – the character of the interaction over time. Digital design materials are temporal in this respect, at least as much as they are spatial. The rest of this section provides three examples of aesthetic qualities in the use of digital artefacts. They serve as illustrations of how the aesthetics of interaction design can be approached and articulated for further debate as well as for practical application in concrete design situations.

Pliability is the quality of plasticity, malleability of the digital artefact under the hands of the user. A set of information is pliable to the user if it feels like a responsive material, a matter of inquiry that can be manipulated and experienced in a tactile sense. Pliability contributes to a highly involved process of exploration where the loop between senses, thought and action is rapid and physical rather than elaborate and mental. I make a small, quick move – the material shapes and responds – I notice something new – I make another move – and so on. Ahlberg and Shneiderman [20] were the first to articulate this quality, under the label of "tight coupling"..

In the interpretation above, pliability concerns the micro-qualities of interaction, the qualities of the surface. Examples of interaction design aiming at pliability include the influential concept of dynamic queries [21] as well as the more recent interaction technique Sens-A-Patch [22]. Another aspect of pliability has to do with the user's possibilities to act freely and shape the material according to the larger situation at hand, such as annotating the margins of a paper form to communicate something outside the rigid boundaries of the form itself. As Henderson and Harris [23] point out, this kind of deep pliability is often unnecessarily lost in the transition from paper to computer systems in, e.g., administrative work. It is straightforward to see how many existing administrative systems could be extended to accommodate free-form annotation and the equivalent of sticky notes.

Fluency as an aesthetic quality of digital artefacts is brought to the fore in relation with the increasingly pervasive digital infrastructure. Use is not necessarily a primary activity at the focus of attention, it is not a binary variable of either using a digital artefact or not. With ubiquitous and mobile computing, it becomes more of a dance among multiple representations and mediations. Streams of information flow between center and periphery as we move through the shifting environments of everyday life and work. Transitions need to be graceful and nondisruptive.

As a simple but conceptually powerful example of fluency, consider the Hazed Windows concept by interaction design students Trine Freiesleben, Miska Knapek and Henrik Moberg at Malmö university [24]. The idea is simply to offer a more lightweight and transient communication channel, for instance between a little girl and her granny who live in different cities. The girl draws or writes with her finger on a display in her home. Her signs are redrawn at granny's display but gradually fade away, as the metaphorical "window" is filled with metaphorical condensation again. If granny happens to see the signs before they are gone, she can reply in the same way. A sign on the display disappears completely in a few hours.

The strength of the Hazed Windows concept is not in its focus on emotional lightweight communication, which has been done thousands of times before, but rather in the elegant questioning of the hidden core assumptions of computer-mediated communication. When we do our email, we devote our full attention to it. We expect messages to be persistent until we choose to file or delete them. In short, we approach the communication situation as a binary task. The Hazed Window hints at the possibility of a new middle ground, a new approach to computer-mediated communication in text and pictures that exhibits a much greater degree of fluency.

Seductivity refers to the captivating qualities of a digital artefact. Following the seminal analysis by Khaslavsky and Shedroff [25], seduction is described analytically as a process of enticement, relationship and fulfillment. Enticement concerns grabbing attention and making an emotional promise. The subsequent relationship is based on making progress with small fulfillments and more promises, possibly lasting for a long time. The fulfillment, or ending, involves making good on the final promises and ending the experience in a positive and memorable way.

It should be clear from the description above that seductivity, in the sense used here, is a quality that crucially depends on the dynamic Gestalt, the temporal qualities of the digital artefact. Sexually explicit pictures on the front page of a web site has nothing to do with it. The example offered by Khaslavsky and Shedroff is the Visual Thesaurus [26] by Plumb Design. It is a web application that duplicates the contents of a traditional thesaurus but takes on entirely new qualities by virtue of its interactive properties. The user explores words, their synonyms and eventually the transient and temporary nature of language itself by navigating a beautifully animated network of words and their interrelations.

Khaslavsky and Shedroff argue that the Visual Thesaurus is seductive in the sense that it offers surprising novelty, goes beyond obvious needs and expectations, and creates an emotional response due to its visual and interactional beauty (enticement). It connects to personal goals through the fascination of words and concepts and promises to fulfill those goals (relationship). The casual viewer may discover deeper meanings of looking up a word in the sense of the multidimensional and dynamic relationships between concepts (fulfillment).

VI. DISCUSSION

Fishwick

Regarding Stephan’s introduction, one of the tenets of aesthetic computing, at least from my perspective, is that we should endeavor to balance body and mind, sense and thought. If software is “invisible,” then by ad absurdum, everything is invisible. For example, I could say that most art is invisible because the artists manifest artwork within their minds prior to physical construction. It is not clear that computing (including software) enjoys a kind of privileged mental status in this regard. I would say the same holds for mathematics. Brooks’ quote seems to re-emphasize this artificial Cartesian mind/body duality. We need to be careful here since if we condone this philosophy, then we place an unnecessary barrier between art and computing. The medium should be an integral part of software and mathematics, and not viewed as dwelling deep in Plato’s cave only to be trumped by ideal mental constructs conveniently positioned outside.

Diehl

In my view, Software is information and thus just another kind of entity, so it is orthogonal to matter and energy. Regarding your reference to “invisible art,” I think this is a good point. The artist manifests her feelings and ideas in a piece of art, but the piece itself is not the feeling and is not the idea. It is just a representation of it. I think that the visualization and the software are two separate things. And each visualization only covers an aspect of the software, in a sense it is an "application" or an "interpretation" of the software, but it is not THE software. Bubble sort remains bubble sort independent of the medium, whether I run it on a computer, let a group of students sort themselves by height or whether I print the program text on the screen.

Löwgren

In section I, defining aesthetic computing as the theory and application of art to computing implies a position within computer science, where the field of art is 'applied' to the extent that it can improves practices and understandings of computing. Paul's and Stephan's sections appear to fall into this category. Is this a position we want, and one we feel comfortable with? Personally, I am not sure. The following comments might help illustrate why I find the initial position a little awkward. The section on the aesthetic qualities of mathematics and programming source code (in section I) is slightly confusing to me. The discussion of 'cognitive aesthetics' appears to be about the beauty of certain representations as perceived by specialists creating and manipulating those representations (mathematicians, programmers) [27]. Then, there are a few paragraphs on wider perspectives -- but sections 2 and 3 are essentially back on the visual 'beauty' of representations.

Fishwick

Each of us applies aesthetic computing in a way that tends to amplify our own particular specialty areas. Mathematicians and computer scientists are on familiar terms with beauty as it relates to their work, and so there is nothing wrong with focusing on either cognitive or visual beauty, although I admit that beauty is one of many facets of the aesthetic experience. For the computer scientist, *representation* and *modeling* are paramount, which is why I focus on representation of formal structures, and Stephan focuses on software and program visualization. This is what we do professionally, and so we are seeking ways in which art and the theory of art (aesthetics) can improve our discipline. However, I also think you are right to suggest that aesthetic computing can, and should, be broader than this.

Löwgren

To continue, the discussion of patterns in section I as well as the whole section II appears to argue that 'the application of art to computing amounts to finding more visually interesting representations for some sort of computing 'contents', such as algorithms or data structures. The contents themselves are more or less given, and the job of the 'artist' is rather a cosmetic one (which, to me, seems more appropriate for a graphic designer than a visual artist). The audience for the 'artwork' appears to be computing specialists. To illustrate why I find this perspective rather limited, consider the wide range of art dealing with computing not as a given but as a subject for inquiry. A rather well-known example here could be Adrian Ward's AutoIllustrator [28], which is both a mockery of commercial productivity software and also, more importantly, poses rather important questions on the nature of authorship. When a tool starts to act more independently, who is the originator of the resulting work? The tool user? The programmer/designer of the tool? Is AutoIllustrator an example of aesthetic computing? I certainly think so, perhaps a more interesting example than the ones offered by Paul and Stephan.

Diehl

The examples that are given in section 2 are visualizations of different aspects of software. These visualizations are goal driven. I don’t claim that they are aesthetic computing, but that software visualization can both learn from aesthetic computing and provide material and techniques. When we developed the pixelmap to show file coupling, we didn’t know what the result would be and were very positively surprised when we saw that most couplings appeared along the diagonal. At least in research, we work to some extend like artists. Our material are the visualization techniques and data about software systems. And in a way we form this material by combining the techniques and applying them to the data to get nice or useful pictures and provide new insights into software.

Prophet

For me, computing is less a subject for inquiry and more a medium that brings with it a range of discourses and has a particular set of qualities. As an artist I want to work with the computing medium in ways that reflect, counterpoint and explore the characteristics of the medium and its discourse (I am not really interested in computing as a tool to make beautiful pictures for example). I think AutoIllustrator reflects and comments on software design and the experience of using software like Adobe Illustrator by making software. This is like the way that Modernist painting explored the material of paint by making paintings like Malevich's White on White - the painting wasn't a representation of something else, it *was* a reflection on the material qualities of paint and of canvas. AutoIllustrator stays close to its subject in form and material but it is not a representation of something, it is an idea and comment about something.

Fishwick

Jane – can you inform us in more detail about the aesthetics of scientific visualization with regard to your Cell project? One could argue that any scientific visualization could be part of aesthetic computing; however, I imagine that the process of Cell, incorporating you as an artist, changed the outcome from what Cell would otherwise be if it did not consider aesthetic experience and involvement?

Prophet

Form is in part lead by function and the current 3D version functions well in the context of medicine and reflects a medical illustration aesthetic. It is enough that the current version simulates a highly controversial 'paradigm shift' in theories of stem cell behaviour; for it to also fulfill an artistic agenda would confuse matters and make it less useful as a medical tool. The aesthetic of medical illustration is to avoid any expressive artifice in the image and to attempt to produce something as real-looking as possible. In this case it shares some aesthetic concerns with simulation and gaming. However, medical illustration tends to remove dirt and extraneous objects in order to expose, or focus on, a central mechanical structure or behaviour across time. 'Exploded' models in anatomy drawings and sequential sketches showing for example, cell division, have been replaced by Flash animations and more recently by biomedical illustrations that draw on agent based systems. At the heart of Cell, for all the professionals involved in the collaboration, has been the development of a conceptual model of stem cell behaviour and then a *maths model* that informs the production of a simulation (a real-time complex adaptive system with a graphic user interface and graphic output). This 'stem cell engine' will now be used to drive a number of diverse outputs, each peer reviewed in our particular disciplines. Theise has authored papers for medical journals on the impact of Cell on his thinking about stem cells; d'Inverno will author mathematical papers and produce a musical score and sound piece; Saunders will make web toys (www.robsaunders.net) and Prophet will make future robotics artworks.

Fishwick

I see, so the actual process of doing Cell has resulted in a kind of massive catalytic reaction, involving all the players. This does play well into your notion of conceptual art's focus on process rather than outcome; however, this leads me to one of your earlier points on conceptual art—that it reflects an idea or dematerialization. I would like to suggest that conceptual art reflects the process of making art, and as I demonstrated in my section, processes can be visualized and materialized within a field of artistic influence. This is done by *modeling the process* and thus surfacing it as an abstraction for the phenomenon in question. For this reason, I am happy with viewing "conceptual art as process," but I do not think this necessarily accords with dematerialization. If one records the process, or models it, then the process itself is manifested as an artform.

VII. CONCLUSIONS

One of the key tensions that represented itself in the discussion is the interplay between that which artists produce versus that which computer scientists produce. Artists have an agenda based a wide variety of styles and aesthetics from formalism and cultural exploration to capturing social, political, or economic aspects of phenomena. Computer scientists are primarily after high utility artifacts; if something is useless, then most mathematicians and computing professionals will tend to shy away from it. However, there is a line that stretches from "no use" to "full use" if there can be such unambiguously defined things. To see this, we need to step back to the definition of aesthetic computing: *the application of art practice and theory to computing*.

There is no reason why this application must be targeted on artifacts of high general utility. By exploring the boundaries and interstices of the "use range", we think we can enhance art and computing. Artists will become more familiar with elements of computing such as data structures, programs, architectures, and even core mathematical structures upon which computing is founded. What the artist produces *reflects* a computing essence, whether or not the result is of immediately obvious utility. To the extent that a piece makes one reflect, it is useful in exploration, creativity and education, and so even the term "use" or "usability" becomes circumspect. Thus, aesthetic computing for the artist can range from Software Art (a fairly new movement defined by artists producing their own programs or languages), to representational art where the computing element becomes the subject of the art, rather than the material for the art as in Software Art. It is this ability to weave through the webs of utility and computing that makes aesthetic computing a unique enterprise.

For the Computer Scientist, what the artists produce will be fertile ground for representation, interaction design, and human-computer interfaces in general. As in Software Art, where some artists are becoming computer scientists of a sort, we also have the converse situation where some computer scientists become artists.

On a more general level, the encounter between art and computing may by studied in terms of disciplinary relations. Jantsch [29], who is the originator of concepts such as multi- and interdisciplinarity, views disciplinary integration as an evolutionary hierarchy. If a traditional disciplinary approach is specialization in isolation, then *multidisciplinarity* simply refers to adding different disciplines without any direct cooperation between them.

The next step up the evolutionary ladder is *crossdisciplinarity*, where one discipline supports the other within the other's own discipline. Our discussion above, and the identification of fields such as Software Art and Aesthetic Computing, might suggest that this is more or less our current state of progress. An illustrative example is our introduction of the dimension of utility, as a way to capture one of the ways in which the fundamental values of the disciplines or art and computing differ.

The most advanced integrative step, according to Jantsch, is *interdisciplinarity*. It involves direct cooperation in both directions where the outcomes could typically not be achieved entirely without any of the disciplines involved. It also entails the formation of new concepts, practices and values transgressing the traditional boundaries of the disciplines involved. It is our firm conviction that the encounter between art and computing holds the potential for interdisciplinarity in this strong sense, and our ongoing dialogue might represent a possible step in that direction.

ACKNOWLEDGEMENTS

The authors are indebted to Dagstuhl, in Germany, where many of the ideas in this paper flourished in a synergistic week-long session. We also would like to acknowledge the Leonardo organization for their co-sponsorship of the event. Paul Fishwick would like to thank his graduate students in the RUBE Project (Minho Park, Jinho Lee, and Hyunju Shim), the National Science Foundation under grant EIA-0119532 and the Air Force Research Laboratory under grant F30602-01-1-05920119532. Jane Prophet's contribution is part of the Cell collaboration and has been written following discussion with: Mark d'Inverno, Peter Ride, Rob Saunders, Neil Theise and Katrina Jungnickel. Cell research has been conducted with awards from The Wellcome Trust sciart; Shinkansen Future Physical 'BioTech' and The Quintin Hogg Trust.

REFERENCES

[1] Aesthetic Computing Dagstuhl Seminar 02291, July 2002, http://www.dagstuhl.de/02291

[2] Aesthetic Computing Manifesto, Leonardo 36(4), MIT Press, 2003, pp. 255-256.

[3] Hadamard, Jacques, *The Psychology of Invention in the Mathematical Field*, Princeton Univ. Press, 1945.

[4] Knuth, Donald, *The METAFont Book*, Addison-Wesley Publishing Co., 1986.

[5] Gamma, Erich, Helm, Richard, Johnson, Ralph, and Vlissides, John, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley Publishing Co., 1995.

[6] Card, Stuart, Mackinlay, Jock, and Shneiderman, Ben, Editors, *Readings In Information Visualization: Using Vision to Think*, Morgan Kaufman Publishers, 1999.

[7] Diehl, Stephan, Editor, *Software Visualization*, Springer Verlag, LNCS 2269, 2001.

[8] Stasko, John, Domingue, John, Brown, Marc, and Price, Blaine, Editors, *Software Visualization: Programming as a Multimedia Experience*, MIT Press, 1998.

[9] Kim, Taewoo, Lee, Jinho, and Fishwick, Paul "A Two-Stage Modeling and Simulation Process for Web-Based Modeling and Simulation", ACM Trans. on Modeling and Simulation, 12 (3), July 2002, pp. 230-248.

[10] Fishwick, Paul, "Aesthetic Programming: Crafting Personalized Software", Leonardo, MIT Press, 35 (4), 2002, pp. 383-390.

[11] Hopkins, John and Fishwick, Paul, "Synthetic Human Agents for Modeling and Simulation", Proceedings of the IEEE, 89 (2), Feb. 2001, pp. 131-147.

[12] Hopkins, John and Fishwick, Paul, "Exploiting the Agent-Based Metaphor in Software Visualization using the RUBE System", Journal of Visual Languages and Computing, 14 (1), Feb. 2003, pp. 97-117.

[13] Fishwick, Paul, Simulation Model Design and Execution: Building Digital Worlds, Prentice Hall, 1995.

[14] Brooks, Frederick P., "No Silver Bullet", IEEE Computer, Apr. 1987, pp 10-19.

[15] aiCall StackAnalyzer, AbsInt GmbH, http://www.aicall.de

[16] Thomas Zimmermann, Stephan Diehl and Andreas Zeller, "How History justifies System Architecture", in Proceedings of International Workshop on Principles of Software Evolution IWPSE'2003, IEEE Computer Society Press, September 1-2, Helsinki, Finland, 2003.

[17] LeWitt, Sol, "Paragraphs on Conceptual Art', Artforum, Summer Issue, 1967.

[18] Kosuth, Joseph; Guercio, Gabriele (Editor); Lyotard, Jean-Francois, Art After Philosophy and After: Collected Writing, 1966-1990, MIT Press,1991

[19] Burke, Edmund, Phillips, Adam, (Editor) A Philosophical Enquiry into the Origin of Our Ideas of the Sublime and Beautiful, Oxford University Press, 1998.

[20] Ahlberg, C., Shneiderman, B. (1994). Visual information seeking: Tight coupling of dynamic query filters with starfield displays. Human Factors in Computing Systems (CHI '94 Proceedings), pp. 313-317. New York: ACM Press.

[21] Ahlberg, C., Williamson, C., Shneiderman, B. (1992). Dynamic queries for information exploration: An implementation and evaluation. Human Factors in Computing Systems (CHI '92 Proceedings), pp. 619-626. New York: ACM Press.

[22] Löwgren, J. (2001). Sens-A-Patch: Interactive visualization of label spaces. In Banissi, E. et al. (eds.) Proc. Fifth Int. Conf. Information Visualization (IV2001), pp. 7-12. Los Alamitos, CA: IEEE Computer Society, web: http://webzone.k3.mah.se/k3jolo/Sens-A-Patch

[23] Henderson, A., Harris, J. (2000). Beyond formalism: The art and science of designing pliant systems. In Kaasgard, K. (ed.) Software design and usability. Copenhagen: CBS Press.

[24] Freiesleben, T., Knapek, M., Moberg, H. (2001). Hazed windows, http://webzone.k3.mah.se/kid01016/projekt_forskningstema.htm

[25] Khaslavsky, J., Shedroff, N. (1999) Understanding the seductive experience. Communications of the ACM 42 (5): 45-49

[26] Visual Thesaurus: http://www.plumbdesign.com/thesaurus

[27] Gelernter, David, (1998) Machine Beauty: Elegance and the Heart of Technology, Basic Books, New York.

[28] AutoIllustrator: http://www.auto-illustrator.com and http://www.adeward.com/swai.html
[29] Jantsch, E. (1972). Technological planning and social futures. London: Associated Business Programmes Ltd.