WHAT IS INTERFACE AESTHETICS, OR WHAT COULD IT BE (NOT)?

FLORIAN CRAMER

What is interface aesthetics, what could it be, or perhaps: what could it be not? There is, first of all, a colloquial common sense understanding of the term that is best illustrated through screenshots:

- the aesthetics of Windows
- versus the aesthetics of Windows Vista
- versus the aesthetics of Mac OS X
- versus the Macintosh from 1984
- which borrowed from the Xerox Star
- in contrast to the command line interface of Unix
- and its counterpart in 1980s homecomputers such as the Sinclair ZX-81 and ZX Spectrum
- whose aesthetic has been artistically resurrected in the work "All wrongs reversed" by jodi
- . . . artists who had experimented with the aesthetic of operating system and browser user interfaces jodi earlier, for example in their work OSS
- as opposed to the likewise fictitious spinning, blinking and beeping in-your-face 3D operating system user interfaces in Hollywood movies as summed up in the Wikipedia article "Hollywood OS".

While these are powerful, influential and passionately discussed examples of interface aesthetics, they merely represent a part of it that too often gets mistaken for the whole. Interfaces are more than just user interfaces, and aesthetics means more than just look and feel.

WHAT IS AN INTERFACE?

Literally, interface means something in between. The term originated in chemistry where it means, according to Webster's dictionary, "a surface forming a common boundary of two bodies, spaces, phases". In computing, interfaces could be defined as anything that acts as

Date: 9/5/2007.

a common boundary or link in a cybernetic feedback loop either between machine components or between humans and machines. Since the machine called computer differentiates itself into hardware and software, interfaces can thus link:

- hardware to hardware, such as CPU sockets, buses like PCI and USB;
- hardware to software. A good example is the Windows key on PC keyboards, or multimedia controller keys that no longer physically `http://www.pocket-lint.co.uk/news/news.phtml/5940/6964/hollywood-technology-useability-movies-wrong.phtml` control audio hardware, but act as front-ends to software controls, CPU instruction sets such as x86 that no longer correspond to the internal CPU design, but act as software compatibility layer;
- software to hardware; the classical examples are operating system kernels and drivers;
- software to software: plugin interfaces, file formats, protocols, and most generally: application programming interfaces (APIs), i.e. interfaces used between computer programs such as C standard library API, Unix system calls or the DOM/Javascript API in Webbrowsers;
- humans to hardware: keyboards, mice, screen and audio feedback, any controller and feedback device;
- human to software: user interfaces.

Out of these six possible interfacings, media studies tend privilege if not limit themselves to the latter. One could call this phenomenon "the restrained interface", deliberately borrowing from the French literary theoretician Gerard Genette and his notion of "the restrained rhetoric". Genette observes how modern rhetoric and literary studies from Giambattista Vico to structuralism had gradually reduced its notion of rhetoric to merely figures of speech, and finally just to the single figure of metaphor. The similarity to contemporary media studies is all the more striking considering that the restraint of interface to graphical human-to-software-user interfaces is conversely linked to a focus on visual metaphors implemented in those interfaces.[1]

Another problem is that the notion of the user interface is restrained in itself, since it is based on a politics of artificially separating "users" from "developers" based on different priviliges of access to machines

[1] Such as in laurel:interface.

functionality granted by those interfaces. The differentiation of user and programming interface – in programmer lingo: UI and API – is purely conventional, and hence arbitrary. In 8-bit BASIC computers like the SInclair ZX Spectrum, on the Unix command line, in MIT's Lisp machines and even in Alan Kay's initial graphical Smalltalk system, the user interface was also a programming interface and vice versa. Live Coding, such as in Adrian Ward's and Alex McLean's Perl-based musical performances or in jodi's ZX Spectrum video, shows how an API can be employed as a user interface.

And of course, aesthetic notions exist just as well for software-to-software interfaces, hardware-to-software, software-to-hardware and hardware-to-hardware interfaces. The search term "ugly API" currently yields [...] Google search result, "beautiful API" [...]. Similar results can be found when researching Internet flamewars on the beauty or ugliness of other technology, in the sense of: beauty and ugliness of the internal, logical design. This brings us into the field of mathematical, scientific and engineering aesthetics that has existed since Pythagoras equated the musical octave in its beauty to the division of numbers by two. Beauty and ugliness, as judgments of taste, however bring us into the realm of aesthetics. [Fludd, Monochord]

WHAT IS AESTHETICS?

Judgments of taste of course involve more than just beauty and ugliness and, to come back to my initial screenshots, more than just look and feel. Ever since the Alexander Gottlieb Baumgarten coined the term aesthetics in 1735, the concept has been philosophically understood in a much farther reaching anthropological sense as:

(1) theory of perception
(2) theory of judgments of taste. Classicist aesthetics focused on beauty – which its more sophisticated theorists from Winckelmann to Walter Pater always understood as a inherent tension between consonance and dissonance –, other theories of art from Pseudo-Longinus to romanticism and postmodernism tended to focus on such aesthetic categories of the sublime, the ugly, the disgusting, the obscene.
(3) theory of art, notably since Hegel, and thus encompassing older-than-18th century philosophies of beauty and art, such as Pythagorean philosophy and Neoplatonism.

In other words, not much is gained if one broaden ones focus from the restrained interface to a more comprehensive technical and cultural understanding of computer interfaces if one ends up discussing what is an ugly or a beautiful API, thus buying into the post-Pythagorean fallacy of art as conceptual beauty of technical design. It is just another restrained concept of interface aesthetics because it reduces of art to beauty and of aesthetics to theory of beauty, with the implication that beauty is good and ugliness is bad.

Such equations are turned upside down in concepts of the "hack" as an ugly, yet efficient and ingenious fix, and in art like jodi's which honors the ugliness of software [pic]. Alexei Shulgin's projects 386 DX and Fuck-U-Fuck Me `http://www.mediaartnet.org/works/fuck-u-fuck-me`, for example, point out and value ugliness and obscenity of hardware-to-user interfaces, too. Edmund Burke's 18th century "Investigation of Our Notions of the Sublime and Beautiful" names pleasure and pain, terror, obscruity, suddenness among the registers of aesthetic perception, and here they also turn into pleasure and pain of the hardware interface, terror of the desktop, obscurity of the API, and so on. In other words, the more sophisticated strands of digital art supplemented the post-Pythagorean engineering aesthetic of the beautiful interface that had likewise governed the institutional mainstream of electronic arts, with a reflection of the computational sublime. Which of course seems to perfectly match Jean-François Lyotard's identification of the sublime experience to a postmodern "incommensurability of reality", a condition for which he all the more accounts computational information technology.

A problem, however, is that the mere notion of computer interface aesthetics contradicts textbook aesthetic theory. It is completely outside the conceptual framework of, for example, Kant's Critique of Judgement and Hegel's aesthetics since it neither involves perception of nature, nor of art, being instead about the perception and judgment of determinist logical systems. As such, it conflates the aesthetic with what Kant considered to be its opposite when he writes in the Critique of Judgement:

> "That which is purely subjective in the representation of an object, i.e., what constitutes its reference to the subject, not to the object, is its aesthetic quality. On the other hand, that which in such a representation serves, or is available, for the determination of the

> object (for or purpose of knowledge), is its logical validity."

Technological aesthetics, thus short-circuits, or reconciles Kant's opposites of knowledge and judgements of taste, respectively of logic and aesthetics. That used to be its hack and punk appeal.

But what remains when that appeal has worn off? Even an interface aesthetics that reflects user interface *(*and) API, hardware *and* software, beauty *and* ugliness might be unsatisfying as a more comprehensive theory of art. In the worst case, it boils down to glorification of new technology as the better contemporary art, be it in iPod lifestyle, Web 2.0 hype or media theoretical glorification of CPU instruction sets. None of this is provocative anymore like Marinetti's saying that a racing car is more beautiful than the Nike of Samothrake used be, but is in mainstream accordance with the ideology of "creative industries" supplanting autonomous arts.

If experimental digital art is seen as its mere antithesis, and playful subversion of interfaces, one runs danger of reducing it to the role of the court jester of those industries.

Yet interface aesthetics might have its place as a critical perspective on both media theories and computer science right because of its insistence on the "aisthesis", perceivability of the linking agent in cybernetic feedback. If an API can be as ugly as a GUI, it means that it is also a user interface. Unlike a conventional engineering angle, an aesthetics perspective thus can shed light on the arbitrary political nature of interface conventions. The restraint of the notion of the interface just corresponds to the technical restraints in the interface proper. This stil means that "interface aesthetics" remains a limited subset of aesthetics which will never be sufficient for a critical perspective on electronic arts, but a seductive trap for critiques to get caught in. Perception and subjective judgments of taste as a whole cannot be reduced to cybernetic feedback unless one buys into behaviorism and naive ideologies of artificial intelligence and artificial life.

In short, the issue seems to be that, for someone whose primary interest is aesthetics and the arts, not computing, the study of the beauty, the ugly and the absurd of computers has some hack appeal, but gets dull sooner or later because it's about deterministic systems whose states are finite. This why researching interface aesthetics is not something I would like to do for the rest of my life.

Aside from that, an even more serious issue lurks behind such an approach. Ever since the late Heidegger applied existential philosophy to technology, to the effect that technology became an a priori of the human condition rather than a human product, there is a technological sublime that many media theories have subscribed to. As second nature, technology aesthetically takes the place both of Kant's natural beautiful and his mathematical and dynamic sublime. We are, in other words, no longer overwhelmed by mountain ranges or thunderstorms, but for example by the pervasiveness of computing. If, with Lyotard, the sublime experience of computing thus becomes postmodern experience of incommensurability of reality, it obscures the fact that this particular reality (i.e. the reality of computing) is based on constructed policies and can be hacked.

There are ways out of those dilemmas. One would be a simple shift from "interface aesthetics" to an "aesthetics of interfacing", in other words, towards a perception anthropology of *practices* instead of systems. The 386DX software and hardware interface for example might be dull by itself, its musical output, too, but Alexei Shulgin's rock star performance which interfaces with it isn't.

But this should not mean to subscribe to an aesthetics of social practices either. Theodor Adorno's aesthetic theory remains, in my opinion, an outstanding critical achievement in its refusal to bless either practices or objects – respectively the autonomy or sociality of art –, putting them instead into an aporia of a negative dialectics that produces no synthesis and trying to "break the coercive character of logic with its own means".[2] For Adorno,

> "The dual nature of artworks as autonomous structures and social phenomena results in oscillating criteria: Autonomous works provoke the verdict of social indifference and ultimately of being criminally reactionary; conversely, works that make socially univocal discursive judgments thereby negate art as well as themselves"

Just as this dialectic implies a critique Kant's and Hegel's concepts of natural beauty, it means that technology – including GUIs, APIs, protocols, instruction sets etc. – by itself isn't very exciting from the point of view of a critical aesthetics. Its objecthood, without any artistic

[2] Minima moralia, p. 199

intervention, is "creative industries" territory,[3] lacking the negative dialectical pole of critical autonomy and asociality. In this context discussed here, this critical quality is only achied through the aesthetic interfacing by artists like jodi and Shulgin; a quality which resists recuperation into what an alliance of techno-existentialist media studies and the so-called creative industries might have in mind with"interface aesthetics" or the "aesthetic interface".

[3]The meme of the creative industries should however not be mixed up with Adorno's and Horkheimer's notion of the cultural industries – since it's not about the entertainment industrialization of art, but the idea that industrial product design aesthetically supplants and obsoletes autonomous art.