# Interfaces and Errors

G. Trogemann[a], J. Viehoff[a], A. Roch[a]

[a]Academy for Media Arts, Peter-Welter-Platz 2, D-50676 Köln, Germany.

December 22, 2000

**Abstract**

In computer science many strategies have been developed to avoid nearly all types of faulty behaviour in computing systems. Fault tolerant operating systems are part of present computer technologies and the improvement of reliable hardware components is still going on. The quality of modern software engineering is defined in terms of usability, performance and reliability. The computer acts as a tool for solving sophisticated, well defined (numerical) problems.

On the other hand, one should recognize already popular trends to reimplement sources of errors into the deterministic machines, e.g. in the field of genetic algorithms and genetic coding. In the corresponding applications randomness and enforced errors widen the functionality of the computing system.

Once the error has reconquered the code level of the machines, this paper aims to define the subsequent questions: to which extend do we need errors also in the human-computer interaction (HCI), in advanced computer interfaces? What are the creative skills of errors, malfunctions and noise in current and future interface techniques?

For a first sketchy approach to the subject we highlight three completely different aspects of interaction and errors. Firstly, the well known concept of "trial and error" is briefly summarized with respect to the different levels of learning in computer science. Secondly, we flush the basics of our bugging theory. Whereas genetic coding uses random numbers for the mutation of the code population and well defined fitness functions (selection), in the bugging theory both the mutation and the selection is driven by human-computer interaction. The running program is dynamically disturbed and rearranged according to the user interaction. The third aspect of the paper deals with the basics of quantum interactions. In quantum mechanics and quantum field theories errors in terms of perturbations and arbitrary quantum fluctuations are naturally involved in all basic interactions. Moreover, the concept of quantum interaction defines a non-local interface technique that might be adopted in future human-computer interactions.

## 1 Introduction

In the theory and history of computing as well as in computer science the interface plays more and more the most important role. Whenever we treat the computer not only as a symbolic machine, e.g. as a machine that solves differential equations by numerical methods or that performs calculations, but if we take the computer as a machine that communicates, in the sense of a cultural and mass phenomenon, the interface plays a much more important role than the hardware or the logic of current computers.

One of the co-founders of the industrial company AutoDesk, Inc., namely John Walker, did suggest to write the history of computers not in terms of logic, nor in terms of hardware, but in terms of a multiplicity of interfaces [1], [2]. He suggests that the development of the computer is not a linear progress or a genealogy of hardware, lets say by the five hardware generations that usually classify the computer: computers build with relays, vacuum tubes, transistors, integrated circuits and VLSI-circuits... The appearance of the computer in a cultural field would be much more a non-linear history and a question of heterogenity in interfaces. Five distinctions in interface technology are made by John Walker: knobs and dials, batch-processing, time-sharing and graphical menues as interfaces. In other words: the history of computing as a cultural and media history can be written as a matrix of interfaces, which are neither logic nor materialistic.

The science of the 'human-computer interface' has been pursued from very different perspectives. An early point of view of the man-machine interface was that of two entities, the user and the computer. The interface is the medium between the two. "The interface is the stuff that goes between the faces (i.e., senses) of the human and the machine". From this point of view the interface is everything that happens between the user and the machine. The user and the computer are treated as counterparts. The interface problem is in this case reduced to a search for suitable communication styles between two given autonomous systems. The main question then is: 'To which degree will one side adapt itself to the other?' While in the early years of computing the user was the slave of the machine, who had to learn it's cryptic command languages, we now want the machines to mimic us. We try to adapt the machines and to teach the interfaces human communication skills. Interface design has to support the user and to afford the functionality of the system.

A newer and different image of the whole man-machine interface subject is that of "acting through the machine with something else, which is not the machine". "The fact that the person is trying to do something means its really human-work interaction with the computer as an intermediary. So I think for me the focus isn't on interacting with the computer, but interacting through the computer" [3]. The user with his goals and some sort of objective content are now the two counterparts. In this image the computer is no longer the given static opposite, but it becomes the interface itself. The computing system as a whole has to be taken as a flexible medium that can be formed and has to be designed appropriate. The whole existing computer science and its commercial products are seen as an invention of scientists and engineers. They are historical products without unchangeable fix-points and which should be kept in a move.

In the classical interface theory (the computers we use daily) interfaces grew out of the paradigm of industrial problems. Interfaces have been designed with different criteria that have nothing to do with the problem of symbolic calculation. The discipline that is concerned with the design, evaluation and implementation of interactive computing systems for human use is called 'Human-computer interaction' (HCI). The challenges of HCI are to improve the safety, utility (functionality), effectiveness, efficiency, and usability of systems. Usability, i.e. making systems easy to learn and easy to use, is the key concept in this list of goals. The starting-point for all considerations of interface questions is the intention to organize some work, or get some insight, or search for information. We first of all do not want to deal with computers. The problem is to build up an analogy to the industrial and ergonomic design of a working place. Although the idea to take the whole computer as a malleable interface is very powerful, some characteristic features of the resulting systems are still very poor. The weak goals of HCI lead to:

- mass products and exchangeability of users
- standardization of interaction and communication
- systems that make the user getting what he expects, nothing more
- sciences that do not connect the theory of interfaces to the theory of computation

We try to design one interface for everybody. That is in terms of culture and in terms of the first industrialization: the computer is a monoculture and the computer is used as a tool that is used to produce products. The resulting tools are mass products that expect some sort of average user for it's optimal functioning. This leads to a standardization of interaction and communication styles. Since the whole system is user centered, the computer gets a rather passive part. The user has a goal in mind and the system just has to guide the user safely and quickly to settle his work. The key areas of research to improve the human-machine interfaces are therefore multi-modality and multi-coding aspects. But these areas are not connected to the theory of computation.

In sciences the error is normally seen as something negative and should be avoided. Many strategies to avoid errors in calculations have been developed in the past, e.g. better materials and design concepts which allow the improvement of quality, or fault tolerant systems, where even in the case of errors the expected overall performance is maintained. A different strategy is to recognize errors as something that cannot be avoided. If we once accept that certain events are inevitable, we can concentrate on the reaction. An insurance model for example is a system that accepts the unforeseeable event and tries to make sure that some sort of compensation for the bad effect is guaranteed. The compensation does neither avoid nor undo the error. It just launches a second event that is positive in nature and therefore capable to smooth the consequences of the first negative event. In computer science we also begin to accept that people and machines both do err. Under this assumption interfaces have to provide reactions that help the user managing the situation [3], [4].

But if errors are anyhow unavoidable and people and machines do err all the time, then science and technology should not only take this well-known fact into account and provide appropriate reactions. If errors belong to the nature of computation we could go one step further and ask how we can reflect the fact of errors within the basic structure of our theories and machines ? Numerous researchers have recently stressed the critical but positive role of breakdowns especially in design and art activities. Errors in art and design are considered as something positive or even necessary to achieve certain results and behaviour. Genetic Algorithms and Computational Evolution are technical examples for complex solution strategies that are built on a positive evaluation of errors.

# 2 The Error in Computer Science

As a first approximation, an error can be seen as the unexpected that forces us to change our plans and actions. The unexpected is the difference between our assumptions or standards (models, theories, believes, assessments, protocols and standard specifications) and our actual observations. Terms like randomness, confusion or breakdown are different forms of describing unexpected events. A first thesis on errors could go: errors are not objective, they always depend on an observer or a community of observers and their interpretations and general rules. The border between what should be seen as an error and what not is blurry and changing in time. What for one is still tolerable or even fine might for somebody else be totally unacceptable. Situations which I judged as flawless yesterday, might today being perceived as intolerable faults. In many cases of course we do not have problems to draw the distinction whether an event is an error or not. It is the background shared by a community that creates a general accepted error. If the community is intact we normally are in agreement whether something is afflicted by an error or not. In the context of computing systems different types of errors can be distinguished.

## 2.1 Malfunction of Hardware and Software

'Malfunction' means that something does not fulfill its intended or formally described behaviour. The incorrect functioning of computing systems can originate from two different types of sources: physical breakdowns or logical faults. While software errors are always programming, i.e. logical faults; hardware errors can arise from design faults or material breakdowns, i.e. ongoing physical and chemical processes. Material faults are usually time dependent, e.g. spontaneous breakdowns of parts of a system that up to this point worked perfectly. The speed of these unavoidable processes depends on different sources like environmental and working conditions, material structure, component design and physical composition of the system. This occurrence of misbehaviour is to a certain degree statistically describable by probability laws for the live time of an object. Design errors depend completely on the conceptions and ideas of the system developers. They can slip into a system at different phases during the development of hardware or software modules. They can arise during the conception of the system, which means the specification of the system is already defective. But they can also be built in during the programming phase. Implementation errors mark the difference between the intention of the developers and the translation into a piece of software. Design errors can even be caused by some sort of meta-errors if certain parts of the development process are supported by another computing system. An incorrect compiler is for example able to produce incorrect machine code from correct higher level programs. In this case the error was made one step earlier during the design phase of the compiler.

## 2.2 Coding and Calculation Errors

Computers are normally seen as systems for storing, transmitting and processing data. Coding includes all methods to efficiently represent the data. If we want to bring anything into a digital computing system we have to decide about appropriate coding of the according phenomena. If we for example want to deal with real world objects we permanently have to make choices about their representations. Since digital computers can only deal with discrete quantities, the coding method is crucial for the accuracy of the results. Three different types of errors are connected to the coding process: quantification errors, sampling errors, and calculation errors. Quantification and sampling errors are produced by the coding of continuous signals for digital machines, e.g by digital representation of images, sounds, space, color or real numbers. The sampling rate determines the number of sample points, while the quantification is responsible for the number of bits per sample point. Calculation errors are a consequence of digitization. They occur because the word length of digital computers is restricted. If we for example multiply two numbers with eight digits the result is a number with sixteen digits. When we have two numbers with sixteen digits the result is a number with thirty-two digits. But it does not matter how many digits the machine uses to represent numbers. Sooner or later the maximum will be reached by repeated multiplication. And the situation is even worse for division, since the result can normally not be expressed with a finite number of digits, e.g. the result of 1 divided by 3 is a periodical number. Any digital computer is afflicted with this rounding problem caused by repeated arithmetical operations.

## 2.3 Uncontrollable Fluctuation / Noise

Computer systems can be devided in two classes: digital and analog machines. Today the analog principle has totally vanished from the market as well as from science. The critical question of analog machines was: how big is the uncontrollable fluctuation of the mechanism in relation to the significant signal? There is no analog machine, which really can calculate the product of two numbers. What we get is the product plus a small but unknown size, which represents the so called noise. This random value is the characteristic of the involved physical process. The ubiquity of uncontrollable fluctuations in any physical mechanism is the main reason why digital computing gained victory over analog computing. In digital computing on the other hand one also has to deal with errors, namely the above mentioned coding and calculation errors. So the difference between analog and digital errors is not of qualitative but of quantitative nature.

## 2.4 Operation Errors

While malfunction, coding and noise are characteristics of the machines that can be observed by the engineer, an operation error means misbehaviour of the user. An operation error can be a wrong sequence of interactions caused by carelessness or because the user is not familiar with the system. An operation error can also be a misjudgement when an experienced user is confronted with a complex situation that is asking to much of him. If the number of alternatives is small, trial and error is a good strategy of solving such problems by trying various ways and inspecting the results. But trial and error is also an important strategy for adaptation and development of systems, which will be discussed in the next section.

## 2.5 Trial and Error

Adaptability to new surroundings and new situations are important features of any intelligent, creative or autonomous system. Learning, as a change in performance can be seen as a function of errors. Improvement of performance usually means a reduction of errors. When you try to get rid of an error, you can choose different strategies. First you will try to solve a problem by applying only small variation in behaviour. You will cling to any strategy as long as possible and only modify some parameters within the scope of your current way of behaviour. Only if the situation becomes completely hopeless you will be willing to change an attitude and strategy or give up the whole theory. In computational learning processes errors can also have consequences for the system on different scales. The cybernetic description of learning processes (Bateson) distinguishes between several levels of learning: zero learning, 1*st* order learning, 2*nd* order learning, etc... [5] Zero Learning means, only hard wired or soldered links between stimulus and response are implemented. 'Trial and Error' is explicitly excluded. The system always gives the same response whenever it is confronted with the same question. We even speak of zero learning, when the user marks the error and says what has to be changed (e.g. delta rule in neuronal nets). If a system gives a different and better answer when it is asked again and if this improvement is caused by a external teacher, we are still in the scope of zero learning. Zero learning in adaptive systems means the user structures everything, he is even responsible for correcting errors.

Only at the next level of learning the system itself uses trial and error to improve its performance. 1*st* order learning is the usual form of learning and is also known as learning in structured environments (genetic algorithms, neuronal nets with Hebb rule). Here the context of the system and the characteristic algorithms remain unchanged. The distinction between the system and its environment exists only for a global observer. So the system does not recognize when it has to change its own identity. It does not know anything about its boundary and about its own function. 1*st* order learning in adaptive systems means the user chooses the learning algorithms and the system applies the algorithms to the data that is automatically gathered during the usage of the system.

2*nd* order learning - sometimes called learning of learning - modifies not only data (operands), but also the rules of learning (operators). It changes to whole identity of the system and the context, in which the system lives and behaves. The system is able to draw a distinction between itself, the environment and the user, who also draws distinctions between itself and the environment. The levels of learning can be thought further by repeating the basic principle. $N + 1th$ order learning is the change of the process in $nth$ order learning. One of the great challenges for research in adaptive systems is the invention of algorithms that realize second order trial and error strategies (in difference to ordinary trial and error, e.g. adaptation of data in 1*st* order learning processes).

Trial and error can be seen as the mutual influence and correcting power in the shared drift and development of system and user. In this sense trial and error strategies are important for the following questions: How can systems extend their own functionality ? How can we model context switches in a dialog between the system and a user ? How can a system change its own identity ? How can a system learn to inform the user about its own state and knowledge ?

# 3 Interaction and Errors in Media Studies

Classical media theories such as the Toronto School of Media Theory coined once the term "the media is the message". This slogan was already stated in the 20s and 30s of the last century and was called "fitting the message to the line". It meant to transform the message into a representation that is able to be communicated over a physical communication link. Here the message is the slave of the channel it is to be transmitted over. But if the physics of communication channels are related to the messages they transmit, we can at least in electronical media create meaning by changing the physical properties and states of the communication channel. This implies the question how errors could be useful in changing the message computer.

## 3.1 Malfunction in Cultural Theory

Faults and malfunctions in the theory of knowledge are very prominent. French intellectuals like Georges Canguilhem in 'The Normal and the Pathological' and Michel Foucault in 'Madness and Society' or 'The Order of Things' did show how pure reason even in its institutionalized form is based on madness, because it excludes madness. We only can say what is normal, because we set what is not normal. The german philosopher Martin Heidegger did stress this point in his explication of the 'Zuhandenheit' and 'Vorhandenheit' of things in Being and Time:

"But we discover the unusability not by looking and ascertaining properties, but rather by paying attention to the associations in which we use it. When we discover its unusability, the thing becomes conspicuous ... The modes of conspicuousness, obtrusiveness, and obstinancy have the function of bringing to the fore the character of objective presence in what is at hand. What is at hand is not thereby observed and stared at simply as something objectively present. The character of objective presence making itself known is still bound to the handiness of useful things ... The references themselves are not observed, rather they are 'there' in our heedful adjustment to them. But in a *disturbance of reference* -in being usable for...- the reference becomes explicit. It does not yet become explicit as an ontological structure, but ontically for our circumspection which gets annoyed by the damaged tool. This circumspect noticing of the reference to the particular what-for makes the what-for visible and with it the context of the work, the whole 'workshop' as that in which taking care of things has always already been dwelling. The context of useful things appears not as a totality never seen before, but as a totality that has continually been seen beforehand in our circumspection. But with this totality world makes itself known" [7].

The argument can be put very simply and easily into a optical metaphor: if we wear glasses, and if we -in a reduction of complexity- call these glasses a medium, we are usually not aware of wearing glasses. Only if these glasses fail or get broken, we suddenly become aware of the media environment that we live in. This process immediately turns into a reflection of the media we use to live in the -in terms of Heidegger- 'world' [7]. Terry Winograd and Fernando Flores for the computer science community did apply Heideggers analysis of the 'Vorhandenheit' and 'Zuhandenheit' to computer societies. In the case of using a hammer, the hammer only appears as a tool, if it malfunctions [7], [3]. The question of physical breakdowns are used by Winograd/Flores to question the design of computers. But Winograd and Flores did use breakdowns only to design new systems or better systems. Computer industry is already used to it: hackers are used to find traps and security holes in the system and have the function to make the system much more save. The error is in that case a stability function, not a creative function out of which new meaning is generated.

## 3.2 Noise in the Channel of Communication

In creative environments, like creative software, interactive installations and media art, the interface should follow different needs: the observer or the artist should determine the outcome of the "product"; lets better say the state of the software individually. Subjectivity in most so-called interactive installations is not really possible, if tools are used to manipulate data through standard and mass interfaces. Under interactive art one could understand a state of the creative process that is an answer of the system to each single observer, might it be an artist or a spectator. This answer must be distinguishable to each other person interacting with software, and not only following determined paths. In other words: if a subject uses a so-called "media-art-environment", the most challenging task would be that the code recognizes the subject as different from other observers.

Well, creative processes, returning to the question of the interface, are about to get individuality or personality. The unexpected, the new or unprobable communication situation are important issues to reach that goal. One approach to build new creative environments is reflecting again the error. It is an old trick that the error during the process of creation affects the tool or the outcome, and is turned into creation. This is quite natural, because if a tool is misused, the imagination of the personal user or artist or observer meets the medium during a process of reflection and with this process form and content is interpreted by the subject individually. Misusing tools enriches the circulating products. Let us call that in a general sense "art". Art is an individual interpretation made possible with the artists imagination that observes a transformed relation between content and form. An error can play an important role in the reflection of the artists work, the error can give the user of software or the artist a chance to reflect the "physical reality" of the medium she or he is working in.

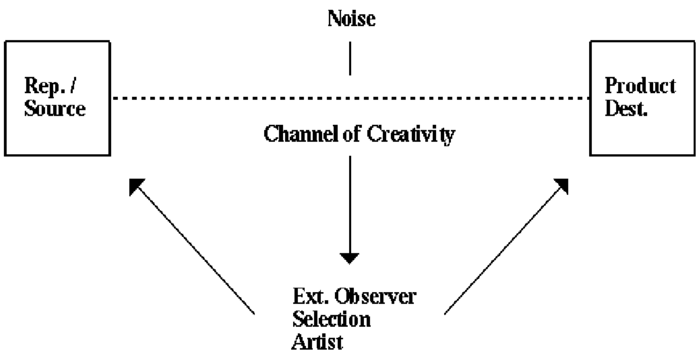

Figure 1: Max Bense: Scheme of Creativity.

Max Bense as a computer scientist in 1969 did make a scheme of positive reactions to errors (see figure 1). Signals $Sig$ are turned into signs and meaning $Z$ through a material function $Fmat$ of the communication channel $Fmat(x, y, z, \ldots, t)$ and are open to the interpretation of the observer as a relation $R$ between the transformable signs $M$, the object $O$ and the interpretation $I$. The process of aesthetical 'semiose' is therefore:

$$Sig \rightarrow Z \equiv Fmat(x, y, z, \ldots, t) \rightarrow R(M, O, I)$$

Here the error is induced through the physical communication channel, interpreted by the observer through a passive process in selecting the noise as a creative and aesthetical expression. But in the scheme of Bense the observer just selects messages. It would be a challenge now to let the user, artist or observer induce errors to the communication channel itself. This can be done in computer science with a theory of bugging in free and open software environments.

## 3.3 Theory of Bugging: Errors induced through the Observer

In a more general view the french philosopher and media theorist Michel Serres developed a theory of the parasite [6]. In his book 'The Parasite' source, destination and noise can rotate in their position (see figure 2).

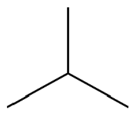

Figure 2: Michel Serres: Parasite Theory.

Only under the assumption of static technical communication the scheme of Max Bense is theoretically useful, so to say in non-dynamical communication environments. In the case of the observer changing the physical state of the communication channel, e.g. programming, the observer creates a message himself. Serres general scheme of the parasite can therefore be expanded in a creative bugging scheme, if we assume that the observer, artist or user induces an error into the medium (see figure 3).

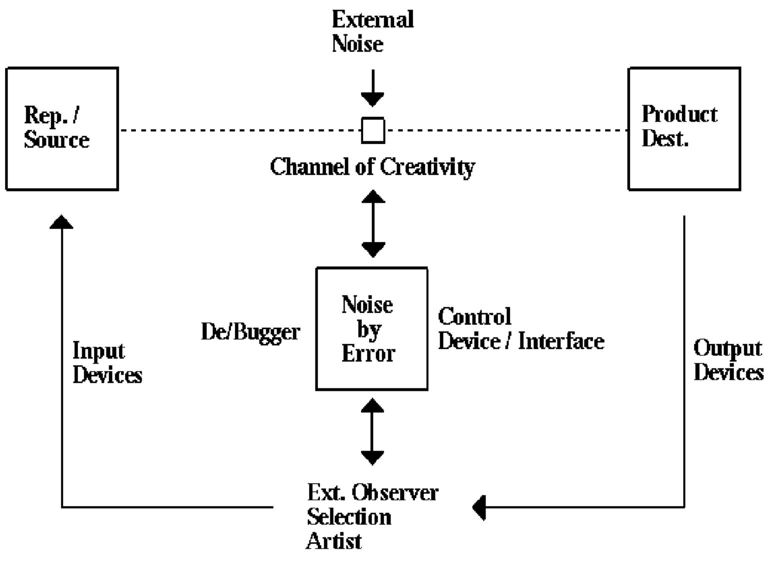

Figure 3: Bugging Theory.

In creative environments there is need for a *theory of bugging* and the practice of *virulent or parasitological coding.* We should distinguish between what we can do with current interfaces (editors, integrated development tools, debuggers) and with advanced interfaces. In practice and with current interfaces a bugging theory would include several steps for the observer:

1. Get the code of a tool which is used to produce art in open source.
2. Re-Compile it with symbolic expressions.
3. Let it run in the Debugger.
4. Re-Code the Program during run-time by injecting virulent errors or re-compile it by injecting virulent codes.
5. Save the errors for reproduction and exhibition.

By changing, mixing addresses, confusing control variables and/or structures, we are about to decontrol the trajectory of interaction with software. There are existing a lot of errors that are pleasant to look at. Numerical errors in digital image synthesis, errors in computer solid geometry. Out of these syntactical changes new visual forms or signs can be created individually.

Of course, an error based interaction with tools through integrated development environments opens up the question how in advanced interfaces the software could be designed to handle possible errors of the user. It should extend a simple editor to change programmed code. Here, in comparison to genetical algorithms, the interface would have the task to mediate not only the selection procedure to the user, but also the mutation procedure. The observer would not be just a slave of random genetical evolution, but be more a genetical engineer in a kind of code laboratory that mutates and resynthesize the code dynamically. This could imply the question to design new dynamical compiler that learn the errors of the user and display the code itself as a complex and dynamical interface which is open to the user for creative and erratic manipulation.

By inducing errors in addresses, algorithms, we are showing possibilities of the physical properties of the communication channel. Signals $Sig$ are turned into signs and meaning $Z$ through the exchange of erratic codes $C_e$. The material function $Fmat$ of the communication channel $Fmat(x, y, z, \dots, t)$ is just important for the erratic observer $O_e$, if the process of semiose pleases to him. The interpretation of the observer is then not only passive, but also an active relation $R$ that changes the message to be transmitted:

$$Sig \overset{C_e}{\longleftrightarrow} Z \equiv Fmat(x, y, z, \dots, t) \overset{O_e}{\longleftrightarrow} R(M, O, I)$$

In turn: we do not debug like the industry, but bug. We do not find the physical fault in the system, but show the possible faults of the system. By showing the syntactial structures in computers, we see nothing else, but the physical reality of computing: possible functions of the machine. In other words: the medium, not the intended and anticipated messages by the software industry. Changing the codes and illustrating the different visual effects of this process by recoding is nothing less than showing the medium of the computer as a communication channel. It is the interface, not the materiality of the media, that is in question. The artist here would construct or inject errors as if she or he would be a virus in a parasitological environment. If we do not want to use the computer and the software as tools that are like the glasses hidden from an awareness and reflection we could simply show other physical states or being by crashing them. This can be done by bugging open software in current and advanced development environments. Through showing the media by inducing virulent error codes -that is recoding open software- we are in between saving its message and explore the computer -again- as a medium.

# 4 Interaction and Errors in Quantum Physics

In the currently available models of human-computer interaction (HCI) the influences of errors are expelled from the interaction theory. But in the previous chapters of this paper we have discussed the reimplementation of errors into HCI, following the strategy of "trial and error" (section 2) and the bugging theory (section 3). Whenever the user interacts with the computing system, he or she might disturb the computation and changes the state of the machine on data **and** the code level. However, such an interaction model has a prominent counterpart in the world of physics: quantum interactions in quantum mechanics and quantum field theories. Since all computing systems and interface techniques are - at lowest level - designed out of physical constitutents (in rising order: electrons, neutrons, protons, atoms, molecules, cristalls, etc.), we should also consult quantum physics for more general definition of interaction.
The models of physical interactions in the microscopic world **include** errors in a very elementary way. Here the influences of perturbations and arbitrary fluctuations - the sources of errors in physics - are important and well established features in quantum mechanics and quantum field theory [8, 9]. In this part of the paper we will introduce interactions as perturbations by explaining the Stern-Gerlach experiment in some detail.
Quantum theories are non-local theories. Therefore the models of quantum interactions can be interpreted as a global interface technique for the physical system. This aspect is of particular interest for computer science, because currently no HCI theory provide interfaces with non-local functionality. If we could design non-local interfaces even in classical (non-quantum) computing systems, a variety of new computer applications would emerge. As in quantum physics, global interfacing in computer science should also start with interactions as perturbations.

## 4.1 Quantum Interaction and Perturbation

The Stern-Gerlach experiment intuitively illustrates why we need a new understanding of interaction at small distances and it motivates the global dependence of physical observables in terms of commutation relations. Interacting with the quantum world always disturbs unpredictably the quantum system under consideration, in the Stern-Gerlach experiment: a beam of spin $1/2$ electrons. The original, undisturbed quantum state of the system, which can be a superposition of single states, is not completely accessible by measurements or observations. Moreover, measurements and observations destroy information within the system, since - in general - only one particular state of the superposition "survives" the interaction.
On the first view, these "active" interactions in quantum mechanics seem to be restricting with respect to classical interactions and to human-computer interaction. Nevertheless, perturbations and fluctuations are unavoidable but

useful in all parts of quantum physics. For example: perturbation theory in quantum field theory counts the errors systematically. Starting with the basic (classical) interaction we use higher order Feynman diagrams [10] to classify the contributions coming from the vacuum structure of the theory. In Quantumelectrodynamics (QED), the strange theory of light [10], one can calculate the power expansion for the disturbed interaction of an electron with an external magnetic field. Here the resulting theoretical prediction for the anomalous magnetic moment of the electron fits perfectly to the experimental data. Indeed, the predictions of perturbation theory in QED are extremely precise and can be checked in the experiments up to very high accuracy ("The best theory we have!", [11]). In other words: the errors in quantum interactions shift the theoretical values for physical observables towards the experimental numbers. Without the perturbations no matching between theory and experiment can be achieved.
Within the strong interaction[1] in the standard model of elementary particle physics quantum fluctuations are responsible for advanced (dynamical) processes, not present in classical physics, e.g.: the quark confinement [12], the spin of the proton or the mass generation of the $\eta\prime$ particle [13, 14]. Here "active" quantum fluctuations and perturbations, closely related to the complex vacuum state of the quantum field theory, create a flux tube in the static quark antiquark potential [12] or contribute to the proton spin in terms of disconnected diagrams [14]. These particular quantum effects demonstrate creative skills of errors in quantum interactions.

## 4.2 The Stern-Gerlach Experiment

The Stern-Gerlach experiment illustrates in a dramatic manner the completely new features of interaction in quantum mechanics. The experiment was originally conceived by O. Stern in 1921 and carried out by W. Gerlach et al. in 1922. A modern and detailed introduction to the subject and additional references can be found in [8]. The original experiment was done with silver (Ag) atoms. The source for the atom beam was a hot oven. Through a small hole a moderate number of atoms escaped from the oven. For the sake of simplicity we turn directly to ideal experimental settings. In the ideal experiment we have a beam of simple electrons with spin 1/2 escaped from the oven and bundled in a collimating slit (see figure 4). The spin of the electron, $\vec{S} = (S_x, S_y, S_z)$, is an additional discrete degree of freedom[2]: the component $S_i$ of $\vec{S}$ can point in two directions: $S_i = +1/2$ and $S_i = -1/2$. Next, the electron beam is subjected to an inhomogeneous magnetic field in $z$-direction (see figure 4).
Due to the magnetic interaction

[1] Quantumchromodynamics (QCD) is the non-abelian gauge theory of the strong interaction. QCD describes the dynamics of the quarks, the constitutents of protons and neutrons.
[2] Of course, the discreetness of physical observables was the first step towards quantum physics. Here I want to focus on the commutation relations in quantum mechanics.

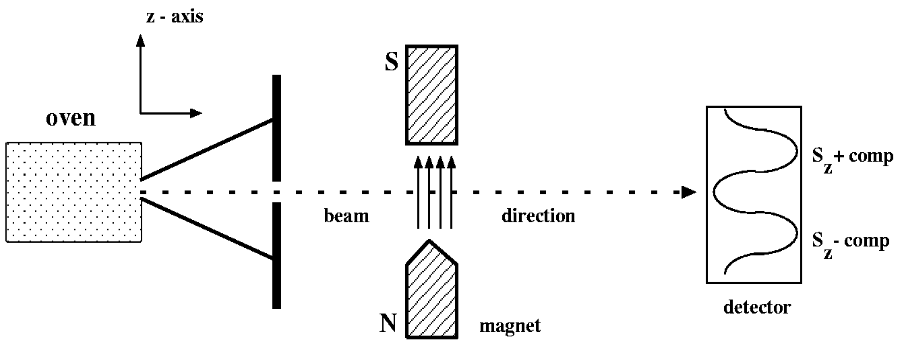

Figure 4: The Stern-Gerlach experiment.

$$F_z = \frac{\partial}{\partial z}(\vec{\mu}\vec{B}) \simeq \mu_z \frac{\partial B_z}{\partial z}, \tag{1}$$

the electrons in the beams with $S_z = +1/2$ (spin up, $\mu_z < 0$) experience an upward force whereas electrons with $S_z = -1/2$ (spin down, $\mu_z > 0$) feel an downward force. Hence, the beam splits into two parts corresponding to $S_z = +1/2$ and $S_z = -1/2$ (see detector in figure 4). Our Stern-Gerlach apparatus thus measures the z-component of $\vec{S}$ and we call this machine $SG_z$. By rotating the direction of the inhomogeneous magnetic field we design the machines $SG_x$ and $SG_y$ for measuring $S_x$ and $S_y$, respectively.
We now consider the combination of Stern-Gerlach machines, the sequential Stern-Gerlach experiment (figure 5). First, the beam is subjected to $SG_z$ and we get two distinct beams with $S_z+$ and $S_z-$ (see figure 5 (a)). We block the $S_z-$ beam and take the $S_z+$ beam as the input of the second Stern-Gerlach machine $SG_z$. Obviously, this time we do not see the $S_z-$ component since we have blocked this component in the previous apparatus. In the next experiment we start again with the $SG_z$ apparatus and we block the spin down component $S_z-$. The $S_z+$ beam enters the second apparatus that is of type $SG_x$ and measures the $x$ component of $\vec{S}$ (see figure 5 (b)). It is not surprising that we see both components $S_x+$ and $S_x-$ because the electron spin $\vec{S}$ has initially a random orientation when escaping.
In the last experiment we use three Stern-Gerlach machines and we step into the strange world of quantum physics. We start again with $SG_z$ and we block the resulting $S_z-$ beam. The $S_z+$ component enters the second machine, $SG_x$, and we block the $S_x-$ beam. Next, we use the $S_x+$ component as input for the third machine, another $SG_z$ apparatus (see figure 5 (c)). Since we have blocked the $S_z-$ component in the first apparatus, of course we would expect only the $S_z+$ beam. But, surprisingly, we observe the $S_z+$ beam **and** the $S_z-$

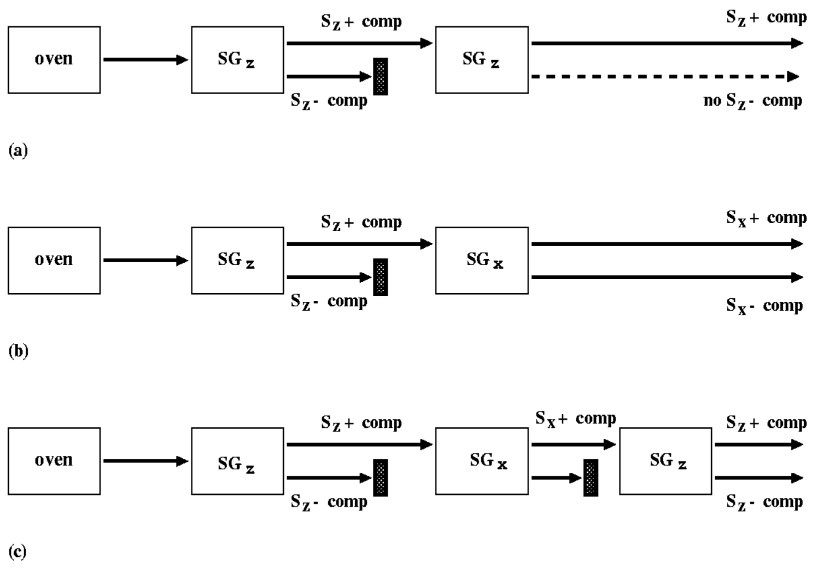

Figure 5: Sequential Stern-Gerlach experiment

beam. The $S_z-$ component, originally surpressed by the first $SG_z$ apparatus, has reappeared!

The findings from sequential Stern-Gerlach experiments have no equivalent in classical physics. The interpretation of the results is as follows: measuring the $S_x$ component of the electron spin destroys the information previously found for the $S_z$ component. The measurement of $S_x$ invokes a perturbation of the quantum state and the informations on the $S_z$ component are lost. The three components of $\vec{S}$ can not be estimated at the same time. In the languange of quantum mechanics we say that the components $S_x$, $S_y$ and $S_z$ do not commute:

$$[S_i, S_j] = S_i S_j - S_j S_i = i\epsilon_{ijk}\hbar S_k \neq 0. \tag{2}$$

The same is true for other "classical" observables, e.g. the location $\vec{x}$ and the momentum $\vec{p}$ of the particle. In this particular case we are faced with the uncertainty relation of Heisenberg: we can not measure $\vec{x}$ and $\vec{p}$ at the time $t$ up to all precisions. Or, in other words, measuring $\vec{x}$ destroys informations on $\vec{p}$ and vice versa.

In classical physics all dynamics are determined by the equations of motion. The classical variables, for example $\vec{S}$, $\vec{x}$ or $\vec{p}$ are connected only via the equations of motion. In quantum physics we have an additional constraint for physical variables: the commutation relations. Interacting with one component of $\vec{S}$ has some impact on the other components beyond the range of the equation of motion, in quantum mechanics: the Schrödinger equation or the Heisenberg equation. Via the commutation relation for $\vec{S}$ (eq. 2) an interaction with $S_z$ "disturbs" the components $S_x$ and $S_y$. In this sense quantum interactions are global perturbations. From quantum mechanics we can learn that interacting with quantum systems evokes errors and the state of the system is changed unpredictably. The interaction reduces the amount of information in the system and it destroys former knowledge of the state vector. These features of non-local quantum interactions have no equivalent in current human-computer interaction. All interface techniques change the state of the computing system in a deterministic manner. No information about the code or the data is of necessity destroyed or even damaged. All "classical" interfaces act locally on one particular variable of the computer.

The developing process in human-computer interaction has not overcome the "classical" limit like quantum mechanics did for classical mechanics. Since quantum interactions, including perturbations and fluctuations, are well established in our physical models of the (microscopic) world, one should allow the question, whether a restriction to "classical" local human-computer interaction does limit the abilities of computing resources from the beginning on. The physical models of nature are - at least - one step ahead.

In this section we have not touched the wide research area of quantum information [15]. The fast growing community for quantum computing, quantum teleportation or quantum cryptography indicates a huge interest in new computing abilities beyond the classical machines. However, present quantum computers get use of the parallel computing resources within quantum systems and speed up "classical" computations, e.g. the factorization of large numbers. A clear understanding of the challenging non-local interface techniques (human quantum computer interaction (HQCI)) in quantum computing is still lacking.

## References

[1] J. Walker: "Through the Looking Glass", in: "The Art of Human-Computer Interface Design", edited by Laurel, B., Addison-Wesley Publishing Company, 1990

[2] B.A. Myers: "A Brief History of Human Computer Interaction Technology.", ACM interactions, Vol. 5, no. 2, 1994, pp. 44-54

[3] T. Winograd, F. Flores: "Understanding Computers and Cognition: A New Foundation for Design", Addison-Wesley Publishing Company, 1997

[4] D. A. Norman: "Things That Make Us Smart", Perseus Books, 1999

[5] E. v. Goldammer, R. Kaehr: "Polycontexturality: Theory of Living Systems - Intelligent Control". In: E. Kotzmann (Ed.): "Gotthard Günther - Technik, Logik, Technologie", pp. 205 - 250, Profil Verlag, Munich, 1994

[6] M. Serres: "The Parasite", Johns Hopkins University Press, Baltimore, 1982

[7] M. Heidegger: "Being and Time. A translation of Sein und Zeit", translated by Joan Stambaugh, State University of New York Press, Albany, 1996

[8] J.J. Sakurai: "Modern Quantum Mechanics", Addison-Wesley Publishing Company, 1985.

[9] C. Itzykson, J.B. Zuber: "Quantum Field Theory", McGraw-Hill Publishing Company, 1985.

[10] R.P. Feynman: "The strange Theory of Light and Matter", Princeton University Press, New Jersey, 1985.

[11] O. Nachtmann: "Particle Physics. Concepts and Phenomena", Springer Verlag, Berlin, 1998.

[12] M. Creutz: "Quarks, Gluons and Lattices", Cambridge University Press, Cambridge, 1983; H.J. Rothe: "Lattice Gauge Theory: An Introduction", World Scientific Publishing, Singapore, 1992; G.S. Bali, K. Schilling, C. Schlichter: "Observing long color flux tubes in SU(2) lattice gauge theory.", PhysRev.D51, pp. 5165-5198, 1995.

[13] S. Güsken: "Flavor Singlet Phenomena in Lattice QCD", WUB 99-02 (hep-lat/9906034), Wuppertal, 1999.

[14] J. Viehoff: "Die Bestimmung von disconnected Diagrammen in flavor-singlet Matrixelementen der vollen QCD", WUB-DIS 99-17 (PhD thesis), Wuppertal, 1999; DESY-THESIS-1999-036.

[15] J. Gruska: "Quantum Computing", McGraw-Hill Publishing Company, 1999; A. Steane: "Quantum Computing", Rep.Prog.Phys. 61, pp. 117-173, 1998; C.P. Williams, S.H. Clearwater: "Explorations in Quantum Computing", Springer Verlag, New York, 1998; S. Braunstein (Ed.): "Quantum Computing, Where do we want to go tomorrow?", Wiley-VCH, Weinheim, 1999.