Fragment 6

BEYOND TECHNOLOGY

Strangely enough, we have already left the era of technology. Another emergent event has already occurred. The lightening has been seen by the thunder which has not yet been heard. In a single stroke we are moved into a new gestalt patterning for all of Western civilization. This new era might be called the ‘meta-technical.’ It takes us beyond technology into a new realm where technology becomes intensified, self-embedded, and qualitatively different. And already there are glimmers of the next emergent event repatterning this new gestalt yet again before the dust has settled. We are living in a time of completion in which deeply buried seeds are coming to fruition. We need to try and understand these fundamental changes in which the teleonomic filter of Western civilization is locking in on its final determination.

Meta-technology is fundamentally different from technology. It is an intensification of technology which is so intense as to be a total qualitative transformation. Like coal turned to diamond under extreme pressure. Technology intensifies and turns into meta-technology. When meta-technology emerges, the lifeworld itself is threatened with total submersion under a blanket of total coverage by technology. All the different socio-technical control structures merge into a chain-mail of interlacing fields. The common face of all the different socio-technical control systems appears. It is our face. Not a mad computer taking over the world. It is seen in the emergence of a viable general systems theory under George Klir. Klir speaks of two dimensional science. One dimension is the specialized disciplines, and the other dimension is the development of a general systems science linking similar systems in disparate fields. In technology the program of technology was implicit and applied only to specialties. With the advent of meta-technology the program of technology becomes explicit and generalized beyond the specific disciplines.

We know when a new era is inaugurated for then new kinds of things come into existence which exemplify that era. In the case of the meta-technical era the pre-eminent representative new entity is ‘software.’ Software, also called computer ‘code,’ is written in a programming language and executes on computer hardware. Software controls the hardware and is what makes computers general purpose machines. Software appears as text statements in a particular programming language. The text must be very precisely written to tell the computer what to do in each step of its operation. Software, through the operations embedded in the programming language, tells the computer how to manipulate its memory to perform complex transformations by a series of simple steps done like a recipe for a particular type of memory manipulation. This is called an algorithm. The memory of the computer holds a number of text values in specific locations. The algorithms manipulate these values in a step-by-step ordered procedure to transform those values into other values. Software embodies and implements these algorithms.

Where hardware machines are technological in nature, software is instead meta-technological. Software provides the equivalent of changeable circuitry for digital computers. Software is called ‘soft’ because it is changeable. When it is less easy to change, because it exists in read-only memory, it is called firmware. Software exists in the computer either on magnetic storage media called disks when it is not in use, or in ‘random access memory’ (RAM) where the computer can use the software to determine its actions. Normally the software guides the computer to do operations on other parts of the random access memory. Such a software system is called and information processing system because it transforms and manipulates information coded in computer memory. However, software can use certain memory locations to receive information from external hardware sensors and to control hardware actuators that can cause effects in the real world through the artifacts produced by the technological system. Software systems which respond to real world events are generally called real-time systems. These may be ‘reactive’ when their response time lag is fixed or ‘hard real-time’ when they must respond at a specific instant in time in the real world. Such systems are called ‘embedded’ when they exist in hardware and do their job with no direct human interface. They are embedded in the machinery controlling it and reacting to external events. Like the computers in many new automobiles, you probably will not know that they are there until they break down. Then the mechanic says he must replace the computer in your car, or dish washer, or microwave or any of a myriad of consumer products that today contain microprocessors embedded in them controlled by software.

We hear a lot about computers and see them proliferating in our environment. But the important new addition to our environment is not the computers -- it is the software that adapts them to a particular task. Software is important because it is a completely new kind of thing different from the kinds of things we are normally used to encountering. Computers are merely a new and different kind of machine, whereas software is something genuinely emergent; something we have a difficult time dealing with because it is so different. Something which is changing our whole world as we watch. Software production and consumption is becoming a significant part of our economy. Yet few people ever see it. There is a software crisis talked about by software engineering professionals and company managers. Yet it is rarely noticed by the general public. However, everyone is effected by electronic financial transactions, large-scale information storage and transfer, and other software controlled aspects of our environment which have direct impact on our lives. For instance, computer trading on the stock market was blamed on the October ‘87 stock market crash. A bug in some new software recently shut down the AT&T phone system for nine hours. Computer viruses have shut down whole electronic communications networks. These glitches bring the issue of software to the forefront of the news. But software itself is a strange new thing which is little noticed. We talk about the importance of computer literacy, but few realize that this literacy is the ability to not just use computers, but to read a new kind of written text called software code.

Software is generally written in a higher order programming language. Examples of these languages are called C, ADA, MODULA, PASCAL, etc. However, there exist today a myriad of programming languages, some general and some special purpose. These languages are formal systems which specify the syntax and semantics for formal descriptions of what the computer is to do. These formal descriptions basically tell the computer how to represent data in memory and how to transform it. Because software programs specify data transformations where all data is reduced ultimately to bit manipulations, the computer hardware is an excellent representation of a structural system controlled by software. In fact, it is worthwhile to consider the computer hardware as the archetypal example of a formal-structural system. At its most basic level all information in a computer system is represented as 'bits' (on/off distinctions) which are grouped in sets of 'bytes' in groups of eight. The central processing unit (CPU) takes a byte from memory and places them in a special register called an accumulator for manipulation. In the accumulator the bits of the byte are changed or transformed from one pattern to another. Then the byte is written back out to memory. All programming languages are directed to control this basic behavior of the computer CPU to different degrees of sophistication. The pattern of bits in a byte combinatorially gives 256 possibilities. One bit is left free so that 128 separate forms are defined as corresponding to these patterns. This is called the

‘ASCII’ code. For instance, one bit pattern ‘01000001’ represents a capital ‘A.’ The capital ‘A’ is the form, and the bit pattern is the information content of that form from a computer programming viewpoint. By changing the bit pattern form ‘01000001’ to ‘01111010’ the ‘A’ is changed into a small ‘z.’ These changes are effected by bit manipulations in the accumulator of the CPU. By bit manipulations any ASCII defined form may be changed into any other form. Bit patterns are perfectly malleable. Software defines the order and sequence of these transformations, but the ability to make the transformation is part of the hardware. Thus, the computer hardware is the perfect example of a formal-structural system. It defines the forms and their structural transformation through bit manipulation. Software defines which transformations will occur in what order. Thus, software is a step above the formal-structural system embedded in the computer hardware. Software is based on formalism and structuralism, but goes beyond these, using them dynamically to effect a particular end result. These end results are of myriad different kinds and are called application programs. Application programs allow the software user to do different tasks without writing software himself. Application programs adapt the general purpose formal-structural computer hardware system to a specific purpose. We notice the impact of software on our lives as we use software-based tools to do our work or in our leisure activities. What is difficult to realize is that software represents a new kind of writing totally different from previous forms of writing. And that this new kind of writing is not just changing our environment, but is a totally new kind of object in the world which is difficult for us to deal with.

Writing has a long history, from cuneiform inscriptions down to the printed text. It is the fundamental basis through which all sustained cultural transmission occurs. However, until the advent of software, all writing has been static and has not controlled any machinery except the wetware of the human mind. Now with the advent of software, writing has suddenly become dynamic and controls a hardware embodiment of a formal-structural system. On the other hand, the new writing has somehow lost its meaning, taking the language of machines instead of humans. Each written software statement has significance within the total complex of the software/hardware computer system. But the statement is seen from a human viewpoint as almost devoid of meaning. Only the whole software program within a human-computer interaction context may be said to perhaps have some degree of meaning. But even that is extremely limited; thus, in software production human beings are reduced to producing reams of meaningless written text which controls computers. There has obviously been some trade-off where meaning has been replaced by the controlled dynamism of the software/ hardware configuration.

The philosopher J. Derrida has already pointed out the difference between speech and writing and how our culture has a prejudice towards speech over writing in spite of the fact that culture itself is carried forward more faithfully by writing rather than speech. Speech acts occur and then vanish. Only written material endures as a repository of cultural information. Derrida's analysis of speech and writing is interesting when one considers software. Software is an anomalous middle case which combines aspects of both speech and writing. Speech is dynamic, and writing is static. Speech carries its meaning within it implicitly, whereas writing depends on interpretation of the reader removed from the writing act. We do not grasp the meaning of written text directly as we do the speech of another. The meaning of the written text always occurs in a different context from that in which it was constructed. Whereas speaking and hearing share the same context. Of course, these clear distinctions fade when one considers video and tape recordings of speech acts which are then played back at a later time. With these recording devices speech is written and then read like a text. On the other hand, writing enters into speech with the telephone and telegraph or

FAX machine. As one speaks, the written image appears and changes what the narrative is about. What is interesting is that many of these technological anomalous situations which mix the attributes of speech and writing are based on software-controlled computer systems. Software itself is anomalous in relation to speech and writing in a different way. Software is a dynamic writing. It is written as a static representation of what the computer should do, but once the computer begins to read the software, then the software can be read and changed. So software, though initially a static representation of dynamic action, can be used to change itself. This recursive dynamism is unique to software as a written form. As software begins to take control of the computer hardware formal-structural system, it produces a sequence of data transformations in the accumulator. Since hardware gets faster and faster, these transformations occur so fast as to appear as a continuous stream where one form blends into another. This is, of course, an exact replica of the working of ideation. Speech produces a similar effect on the listener. In speech one becomes immersed in the illusory continuity of the speaker’s discourse. Writing produces a similar effect when read, but more readily collapses back into being just a text. However, listening to a discourse the illusion of continuity is rarely broken. Speech is, in fact, the basic human phenomena that all ideation takes as its basis. In computer software the idea is to manipulate the illusory continuity to give continuous transformation of forms for the user. In this way the user becomes enveloped in the operation of the software application. The ultimate in this envelopment is virtual reality. With other attempts to create illusory continuities that replicate ideation such as television or motion pictures, the content of the illusory continuity is fixed. We see the same film over and over, and it does not react or change based on our actions. Software, on the other hand, allows the content to the illusory continuity to change in reaction to our actions. Thus, a much deeper possibility of crafting an illusion is realized. Anyone who gets caught up in the action of a movie should know that if the movie reacted to your thoughts, speech, and actions, how much more compelling it would be. Airplane pilots are really the first group to have all encompassing simulated environments created to train them. In these simulated environments the movie seen outside the window of the cockpit changes based on their own actions. This is done so that they can try out maneuvers necessary for dangerous situations without risking property or lives. An all encompassing illusory environment which reacts to the actions of those watching is very expensive to build. However, this is the new kind of relation between man and his mechanized embodiments of ideational processes made possible by software.

Software is a written text that controls a dynamic formal-structural hardware system. Once the hardware starts reading this text it can manipulate the text, and change it in ways which in turn changes the behavior of the hardware formal-structural system. Software controls the ongoing transformations of the computer which produces illusory continuity within which forms change. A crucial difference with all prior embodiments of ideational processes is that software can react to the viewer enveloped in the virtual reality of the simulation so that the ‘program’ changes in response to the participant’s speech or actions. Thus, with software the ideational embodiments come of age, presenting a complete simulated virtual environment which the participant can manipulate. We are no longer spectators tied to our chairs in Plato’s cave. Instead, we can participate fully in the illusion which reacts to us as we in turn react to it. Suddenly we enter into a new realm in which a real dialectic can occur between ourselves and our embodiments of ideational processes. Software makes this possible. Software raises the interaction with ideational mechanisms to a new level of intensity by mixing the attributes of speech and writing in a completely new way.

With software we have entered into the meta-technical realm. It is necessary for us to understand how this meta-technical realm differs in its attributes from the realm of technology. Fandozi's analysis of the attributes of technology have already been mentioned. Let us return to these attributes and attempt to differentiate these attributes of technology from the attributes of meta-technology which are their intensification to the point where a qualitative change occurs.

Since we are dealing with the particular area of Software Engineering Technology, it is proper to illustrate the foregoing abstract analysis of the relation between the formal-structural system and its ontological grounds with an example form this specific technological area. In order to facilitate this, let us analyze the relationship between embedded software and the characteristics of technology described by Fandozi in his study Nihilism & Technology. By establishing the relation between software and technology, we automatically create a link between software and nihilism because nihilism is the essence of technology. Nihilism is the face of erratic change within the social system that produces technological artifacts. Nihilism as a form of differing/ deferring, masks the presence of pure immanence. Nihilism exists so that non-nihilistic distinctions may be made. This is the equivalent of the old Zoroastrian moral dichotomy, good (Spenta Mainu) and evil (Angra Mainu) within the technological sphere. Just like evil and good, nihilism and the non-nihilistic distinction cannot exist and be seen without each other. They are complementary interdependent opposites.

Fandozi enumerates the following characteristics of technology:

> Technology is pervasive.
>
> Technology tends toward autonomy.
>
> Technology is repressive.
>
> Technology tends to conceal its own nature.
>
> Technology is anonymous.
>
> Technology emasculates ideology.
>
> Technology attempts to make everything available.
>
> Technology is a process of formalizing and functionalizing the world.

No one would deny that embedded software is a kind of technology. But how can we say that embedded software exhibits these characteristics of technology enumerated above. Embedded software represents a symbiosis of the old mechanical technology with the new meta-technical structures. In the development of this symbiosis there is an intensification of the technical project. This transformation of the technical sphere in many ways clarifies its nature, bringing to the forefront these characteristics described by Fandozi.

Technology is pervasive. It effects all aspects of society. It effects what we do and how we do it in all spheres of life. It profoundly changes the lifeworld by altering structures of everyday behavior. Our structures of behavior are conditioned by the artifacts that we produce and that operate within our environment. These artifacts, such as telephones, radios, computers, automobiles, planes, etc., have a synergistic compounding effect. They work together to create a complex technical environment which we work with every day. This interlocking of the aspects of the technical environment is what gives it a deep pervasiveness. Think of the launch of the Hubble telescope satellite. Here all the artifacts mentioned above, and many more, are used together, each fulfilling a particular function to bring about a successful launch. And what is it that allows this synergistic effect of combined technologies to occur besides human coordination? For the most part it is software embedded in various artifacts which allows the various technical artifacts to work together. The technical environment is knit together with embedded software so that different machines can work together efficiently. The deep pervasiveness of the interlocking technical environment is made possible by embedded software. That environment is symbiotic with the technical human community. It is a socio-technical system. It is software that even further deepens that symbiosis. The technical environment forms a continuous whole within which the human being functions. The cockpit of a jet is a good example of this. It is software that makes the technical environment whole and also adapts it to the human whose behavior is adapted to the technical system. Without software, the pervasiveness of the technical environment remains superficial. The picture is of an environment filled with many different machines that do not work together except by human coordinated use. These machines are not adapted to the human, but instead it is the human that must adapt to each individual machine. The deep pervasiveness allowed by embedded software occurs because the various machines can cooperate without human intervention and because the machines can dynamically adapt themselves to the user.

It is thus clear that technology is pervasive, not just because wherever we turn we find ourselves falling over some machine. It is pervasive because of coordinated use of machines so that the whole work structure is determined by the necessities of coordinating machines. The human coordination of machines, as in a production line, is limited by natural human capacities. Software has the potential to supplement and take over some of the work of coordinating machines. This kind of software is called the Command and Control system[1]. Embedded software in each machine makes it able to be controlled and for its current state to be sensed. Embedded software and command and control software working together allows a whole new level of adaptive and interlocking coordination far beyond what is humanly possible for complex technical environments. Thus, we see that software allows an intensification of the pervasiveness of the technical system by allowing adaptation and interconnection in ways that are not possible without software. This intensification of the pervasiveness of technology through increased levels of coordination may be deemed meta-technical.

Technology tends toward autonomy. This is to say that technology has its own agenda which supersedes that of user and maker of the technology. The agenda of technology is increased movement toward total control of everything including the human that created or uses the technology. <u>The Parable Of The Tribes</u>[2] expresses the dialectic out of which the autonomy of technology arises. If any tribe achieves a technological advantage and uses it against its neighbors, then all the tribes are forced to

1. See Beam on COMMAND AND CONTROLL SYSTEMS
2. Reference

adopt that technological way of doing things or be dominated. Domination means the loss of autonomy. Colonization is just this type of domination through technological advantage. Under domination the adoption of the dominant technology becomes necessary for different reasons. Only those tribes who adapt to technological dominance by adopting the technology which gives advantage can remain autonomous. By this dialectical relation between competing tribes, technology begins to evolve. This mutation of techniques takes on the appearance of a Darwinian evolution. In fact, the Darwinian model fits technology far better than it fits biological evolution[3]. In Biology there is no mutation of one species to produce a wholly new species to be found in nature, even though this is demanded by the theory. There is only variance within species. The Darwinian model does not explain the discontinuous nature of biological evolution well. On the other hand, studies of the evolution of technology show that the Darwinian theory fits very well what happens when technologies evolve into newer technologies. The parable of the tribes is the story of the survival of -- not the fittest -- but those with the technological advantage. Over the long progress of human history, since the Kurgen invasion of the whole of the known world in about 4000 BC, this technological advantage (in the case of the Kurgen people it was the use

3. Reference

of the horse in warfare) has become recognized as the key element in economic and political warfare. Thus, as time goes, on we recognize technology as an almost autonomous driver of human history. It gives unfair advantage to those who are at the technological cutting edge of their times. Everyone contributes to the growth of technology whether they like it or not. It is an out of control positive feed back spiral.

However, this autonomy of technology which arises from the struggle for power between peoples and nations, where everyone contributes to the growth of technology so that the agenda of technology is raised above everyone's agenda -- this is only the superficial autonomy of technology. The deeper type of autonomy is that given to technology by the presence of computing machinery and the software that controls it and its peripherals. The autonomy of independent artificial processors is a truer autonomy. Machines have always acted by themselves via a transformation of energy. Feedback circuits have allowed machines to be self regulating. However, through the addition of computer control, machines achieve a degree of self regulation which senses the environment and reacts in ways similar to an organism. This autonomy is set up by the human being because self-generating machines have not been produced yet. But once the machine has been set up, it can be programmed to sense its environment and react. Satellites are a good example of this sort of robot. Programming is a fundamental part of the 'set up' for autonomy. It allows flexible adaptation and response of the robot. It is like a wind-up toy which is wound and then let go to walk about independently. Thus, programming manifests the slavery of the automated technical system to its programmers. But once programmed, the technological complex acts independently of the programmer; it appears in the environment as an autonomous independent agent. This is true even if the programmed system offers a control interface to the user. The user is controlling the autonomy of the programmed technical automaton. The automaton is still acting independently from the user. The point here is that the program user -- a technologically adapted human -- is part of the technological system. He is the wizard of OZ behind the curtain. He is the sophist or magician who hides behind his sophistry or sleight of hand. The autonomy of the technological system as independent actor with its own agenda has appropriated the human slave. The master/ slave dialectic of Hegel comes to play here so that it is impossible to tell which is which. Technology has had its own agenda which has through economic and political warfare been raised above the agendas of the players of the power game. Through the evolution of computing machines as a general autonomous processor, the technological system has itself achieved autonomous independence through which it can pursue it own agenda in a way hither to not possible. The human being has been co-opted to serve the agenda of technology and provide it with the autonomy of movement and action as well. The autonomy of action given to technological products intensifies the application of the technological agenda to the world. Software is the mediator between the programmer: slave/master -- and the automation: master/slave. It is embedded software that makes the master/slave dialectic possible. Without software as a means of communication between automation and technical humans, the actions of machines would remain disjointed and non-adaptive. Through software embedded in computing machines controlling other devices, technology begins to exhibit the kinds of independence which we attribute normally only to organisms. The autonomy of technology before was that of a preeminent cultural artifact. Now that artifact is automated and acting independently in the environment. The first killing of a human by a robot occurred in Japan not many years ago. Who is the murderer? An environment filled with autonomous technological artifacts is certainly different from one which is just filled with machines of various types closely watched by human manipulators. The autonomy of technology has deepened considerably with sensors and servomechanisms controlled by embedded software. The autonomy of the technological complex has deepened. This deepening, dependent on the presence of software and independent computing devices, may be described as meta-technical. The autonomy of the technological agenda has been supplemented with the true autonomy of the technological artifact.

Technology is repressive. This is a corollary of the surfacing of the technological agenda. Because the agenda of technology surfaces and comes to dominate the agendas of the players of the power game, all other agendas ultimately become submerged. The human diversity becomes subservient to the technological system, and all features that are of no technical advantage are devalued. As such they slowly become identified as hindrances to efficiency. This is happening to language barriers in the European Economic Community. These barriers cause joint technical projects to cost five times what the same project would cost by speakers of a single language. These human differences are inefficient and thus, despite claims of cultural worth, English is becoming the technical language of choice everywhere in the world. The dominance of English is repressive because it causes other languages not to be used, and other kinds of human diversity connected to language becomes threatened. The success of American media that fosters English detracts from local cultural media products. Technology as a cultural system only accepts a very narrow band of human behavior. All other human behaviors are irrelevant to the technological society, and as such, they face extinction.

This repression of technology which rejects inefficient or noncontributory human diversity is a surface phenomenon. In many places around the world this repression takes on a more ominous character as the police state using technical weaponry against its own citizens. Political prisoners all over the world are held enthralled by the sinister face of technological gadgetry for eavesdropping, information collection, processing and control, as well as torture. Most of this activity is aimed at controlling diverse human populations so they will be docile markets, and work forces which do not threaten the stability of the dominant economic and political systems that foster the technological society. In many countries the police state becomes identical with the technological system, and the sensing and information processing becomes essential control mechanisms. Without software, this machinery of repression could not blanket entire societies. Software is essential to the expanded control of the police state. Yet there were police states much longer than there were the elaborate technical apparatuses that they use today. In fact, the police state first appeared as the Catholic church used the inquisition to stamp out the Cathar heresy.

On a deeper level, repression of individual differences becomes the channeling of the human being into a technological mold. This is done by the education system. Technical societies require a highly educated docile population. This is why Japan is so successful as a technical society. American individualism was good to elevate the technical agenda through the great strides of early innovation. But a technical society must repress the individual differences so that individualism becomes inefficient in the long run. Education is a kind of programming of youth that makes them docile and amicable to serving the technological system. It teaches them the doctrine of relativism which disarms all independent thought and action. Relativism is the positive face of nihilism. It is nihilism whose negative aspects have not yet become apparent. Relativism is the equivalent of religious dogma for the technocrats of the technological society. Education is an early repression of individual differences in order to mold good managers and workers.

Slowly computers are entering education curriculums. Computer-based educational materials are starting to be developed. Educational software is an expanding field.

This kind of software makes learning without teachers or books possible. The educational program lets self-paced learning occur. It allows training to take place in a simulated environment. This type of software for teaching shows a completely different aspect of the usefulness of software. It is universally regarded as a positive and fruitful aspect of software development. No one sees educational software as repressive. It is the means of turning talents into skills.

‘Repressiveness’ is a word with negative connotations. A better word might be ‘channeling.” Technology causes channeling. It channels the diverse human populations into a narrow range of alternatives. Those who resist this channeling are either controlled by the police state or join the ranks of the unemployed. Channeling, when directed at the young, is called education. Software takes educational channeling to a new level of sophistication. Education becomes a game. Education becomes adapted to individual needs and differences, allowing one-to-one teaching again by automated means. Computerized education is a channeling environment which prepares the student for control by making him docile in a computer-controlled environment. Thus, educational software allows us to see the deeper nature of technological repression. This repression of human differences is inherently meta-technical.

Technology is anonymous and tends to conceal its own nature. We do not think of ourselves as living in a culture controlled by technology, even though the artificial environment is everywhere around us. We see the artificial environment as a means to furthering human ends. Human actors that own and control technological means are what we see around us. The technological equipment are stage props to the actions of other humans. This is merely another way of saying that technology is not present-at-hand. It is not the focus of our projects unless we are building and maintaining it. Because it is usually not present-at-hand, it fades from view no matter how it clutters the environment. But this concealing of itself is but a prelude to its concealment of its own nature. The nature of things are their essential properties. Technology not only hides itself, but it hides what it is really like from us. This is why technology has been part of human life for thousands of years, but it has not been focused upon by human society except only exceptionally. Our era is one of those exceptions. Technology blends into human action. We see the actions and not the technical means. Humans rarely realized that the technical means had any nature different from the enabled actions. It is only in eras of great technological change that the distinction between the way we do things and the technical means can be made. Thus, the attributes of technology discussed here appear. They are normally hidden. It is only with great difficulty that they have been discerned by philosophers over the centuries. This concealment of 'technical essence' or 'technical nature' is itself an attribute of the technical. We might call this a meta-attribute of technology. Through the meta-attribute of essence, concealment appears the next level of Being. That will be described in later sections of this essay as Hyper Being. It is the absolute concealment of pure immanence. Software takes its nature from this meta-level of Being. Self concealment of attributes is a meta-technological manifestation. With preceding attributes, the surface phenomena of technology has been differentiated from the deep meta-technical phenomena. 'Self concealment' as opposed to the 'concealment of nature' are two such levels. Fandozi recognizes both of these levels. He speaks of the anonymity of technology. This is self concealment of what is not present-at-hand. The concealment of concealment is the associated meta-technical attribute. Both of these apply directly to software. Software is almost never seen directly. Normally only its effects are seen. Embedded software even lacks a user interface. It is hidden in the bowels of the machinery. The user might not even know it is there. This software, unlike other kinds of writing, is nearly invisible to everyone except the programmer. When it is seen, it is very difficult to understand. Software is by nature hidden. The elegance of software designs is rarely appreciated. Besides hiding itself, software hides its nature. Software appears as ‘just a text.’ Yet it hides through the operation of multiple perspectives the singularity of pure immanence beyond the event horizon of Derrida’s differing/deferring of differance. Probably no one would agree that software has this inner nature. The nature is hidden. We only see it in our nightmares as yet another project fails, as a werewolf for which we seek an elusive silver bullet[4]. So technological anonymity is the surface phenomenon that glosses over the meta-technical manifestation of technology, hiding its own nature.

Technology emasculates ideology. Human differences are not the only kind that are repressed. Technology also represses ideological differences. Ideologies are the motive forces behind political systems. They occur when a particular set of ideas like freedom, liberty and fraternity become the central rationales of behavior. So it is interesting to note that although the agenda of technology is raised above all others in order to give power to the tribe or state, the state is ultimately a road block which must be removed by ‘progress.’ Multinational companies express the agenda of technology more perfectly. The corporation is an imaginary person created as a legal fiction. Imaginary

4. See F. Brooks “No Silver Bullet”

persons have no need of ideology. Their behavior is not motivated. Only human beings need motivation. The corporation is the equivalent of the self for the technological system. It is a vortex of human activity aided by technical means which forms a system producing surplus value. The corporation is the point at which the economic and technical systems merge. By becoming a legal fictional entity, it is recognized by the state. But the state only serves as a host for the corporation. Its ideology that motivates the political system is not necessary for the technological system. In fact, as the agenda of technology is raised -- and that agenda is the intensification of its attributes -- these human motivators become inefficient like other human differences. The Soviets are slowly recognizing how their ideology is counter productive. They are losing technical advantage because they insist on ideological control. Free flow of information, and open markets with free movement of people, is more conducive to the flourishing of the technological system. The police state is ultimately counter productive. However, the emasculation of ideology is a surface phenomenon. The deep phenomenon is the ideational phenomenon that produces ideologies. Ideologies are political theories. At a deeper level technology harnesses ideation but devalues the product of ideation which are ideas. Technology is practical. It functions by using the ideational process to produce theories. These theories are embedded in machinery for practical use. The pure ideas are discarded. Technology dwells in the realm of pragmatic intelligence defined by Kant as practical reason. The discarding of the pure ideas is what separates science from technology. Science is the opposite of technology in this respect. Science reveres the pure ideas and is only interested in technology as a means of getting at those pure ideas. There is a natural partnership between science and technology because the end products of ideation are useless to technology. Technology uses the ideational process itself. It creates theories embedded in machines rather than free floating -- nonembedded -- theories. An example of an embedded theory is a software design. As shown by Peter Naur software is a nonrepresentable theory which serves as a software design. Software design depends on ideation -- which produces illusory continuity -- to give wholeness to the design as it evolves. Without ideation as a capability of the mind to produce a differentiated theory and do mental simulations, the theory would not be a whole. The computing machine is an artifact that externalizes the ideational process. The software artifact, when represented as code, executes at such a speed that an illusory continuity for the processors actions is created. This illusory continuity is crucial to the manifestation of the software design. So at a deeper level, technology rejects ideas (the products of ideation) for the process of ideation which in software is externalized in the form of the running computer animated by software.

Technology makes everything available. The anonymity of software shows it is not present at hand. But this attribute attaches it definitely to the ready to hand. ‘Made available’ is equivalent to Being ready-to-hand. Total availability means ready-to-use to get some project we are focused on done. Unavailable is a hindrance to the technological project. Our conditioning to this attribute is shown by anger at the slightest inconvenience. Availability means the free flow of information, people and goods for immediate consumption. Any barriers to this flow is unacceptable. This free flow shows the likeness of technology to fire which consumes its fuel. The fire consumes resources and lives to produce artifacts which are consumed by the market. The fire is the vortex of activity within the corporation which feeds the frenzy of consumption within the free economy. Free economy means open to total availability. Technology strives to make everything available which fits into the technological system, including itself. Technology propagates its own availability even as it hides itself and its nature. This contradictory set of attributes gives us a hint that technology has its own mode of disclosing. It hides in relation to the present-at-hand, but becomes conspicuous in the ready-to-hand. As another mode of disclosing, we see that technology functions in a different kind of Being. Being is manifestation. Different kinds of manifestation will allow hiding in one mode with exposure in another. There is no meta-technical level associated with this attribute. Technology takes its Being from Heidegger's strange mixture of Being and Time called 'Process Being.'

Technology formalizes and functionalizes the world. Technology is structural, and is best represented by the formal-structural system such as that described by Klir. Technology can formalize and functionalize precisely because it is structural. 'Structural' means it has the ability to structure. To structure is to impose form. Functionality is an attribute of form. Structure determines the functionality of a form within a set of forms that can be transformed into each other. Functionality is the relation of a form from a structured set of forms to the whole set. Structuralism handles the discontinuities between the forms of this set. Structuralism builds bridges between forms across these discontinuities. The forms evolve through time, changing into each other. If this process is fast enough, the change of forms appears continuous, and then this is a model of ideation. Form and function is the means by which we understand what is present-at-hand. We concentrate on forms and apply our functions to transform them. This transformation changes the functions of the form. If that change is radical enough, they are replaced by another structurally-related form. Technology can only formalize because its nature is structural. This means that technology operates on the present-at-hand forms with their functions from the hidden space of the ready-to-hand. If it were not once removed in another mode of Being, it could not operate on forms. Thus, this attribute of technology connects it yet again closely to Process Being.

Software is a meta-technology. Where technology makes everything available and formalizes/functionalizes, the world-software does that and more. Meta technology intensifies technology. Meta-technology "technologizes" technology itself. This becomes apparent in the way computers make math skills more necessary while at the same time taking away the tedium of the computation. It is no longer necessary to be able to do the computation. Yet is it necessary to know what the end product of the computation means. Without this knowledge "garbage in-garbage out" becomes more than a truism. It describes precisely the situation where there are reams of overly precise data which when processed is meaningless. When everything is available, then relevance, significance, and priority becomes crucial. Thus, education becomes more important. One can no longer do production jobs without basic computer and thus, mathematical skills. Thus, while technology makes everything more available, the question becomes “available for what?” What do we do with what has been made available to us? What are the significant facts in our data base? How do we find out? Meta-availability is relevance. Meta-availability is filtering. Software provides the means of automatic filtering of information. It can automatically signal the arrival of an important fact while filtering out irrelevant information. So while software makes more information available as a technology, it also allows filtering of what is available which is a meta-technical aspect that allows the true nature of software to be glimpsed. All we need to do is look at the evolution of abstracting journals to see this exemplified. Now the references of articles are cross-referenced to discern related articles. Thus, has been taken another step where clusters of heavily referenced articles are identified and named as cutting edge subdisciplines. The identification of cutting edge subdisciplines allows a reader to immediately identify the crucial papers published each year by seeing how heavily referenced they are and whether they belong to a cluster of heavily referenced papers. This is a kind of information filtering which should intensify research and the dissemination of important findings.

Software plays a part in the formalization and functionalization of the world. Yet this has already been started by other technologies that foster the creation of formal-structural systems in the world. Software as a meta-technology must do more than merely formalize and functionalize. It must intensify the formalization and functionalization in some way to qualify as a meta-technology. The intensification of formalization occurs through the operationalization of formal languages. Formal languages are used to write software modules called functions that take inputs and transform them into outputs. Thus, is some way formalization, and functionalization is epitomized by software. Software must have its input in a certain form to make use of it. Each piece of software fulfills a specific function. It demands a certain set of formalized behaviors on the part of the user in order to work properly. Installation must be done exactly right. The program accepts a certain limited range of user behaviors and responds consistently to the instilled behavior patterns associated with its user interface. Software simulates the world, and by running its simulation within an environment, it sets the pace and conditions the environment. So software allows the formalization and functionalization of the world to be automated. The automation is a simulation of some aspect of the world which ends up driving the world and causes behaviors of people to change to accommodate the simulation. By epitomizing formalization and functionalization and confronting individuals in everyday life with formal and functional responses, it causes those individuals to change their behavior and conform. The result of conformity is that the individual enters into the simulated or artificial environment of the complex dynamic software artifact. Our vision of Tron[5] being sucked into the computer is becoming fundamental mythology that haunts everyone who fails to get a good credit rating. The software has a replica of us which in some sense controls our basic economic behavior. The stock trader who does computer trading has an active replica -- called strangely enough a daemon -- that watches for the price of a stock to change beyond a certain threshold, and it automatically buys or sells based on that movement. Software allows us to enter the simulated artificial world, and allows that artificial world to get mixed up with the tangible world of everyday life. The end result is a confusion about what is real. Fortunes are made and lost not on paper, but within the electronic media of the computer circuits. Software makes the world a simulation. Through this analysis of Fandozi's attributes of technology we have defined a set of meta-technological attributes related specifically to the role of software in the world today.

5. The Disney movie.

The meta-technical (like software) is not just pervasive, but interlocking and adaptive; not just tending toward autonomy, but truly autonomous; not just repressive, but channeling; not just concealing (anonymity) its nature, but hiding a singularity of pure immanence; not just emasculating ideology, but harnessing ideation; not just making all available, but filtering for relevance; not just formalization and functionalization, but simulating in an artificial environment.

Recognizing the meta-technical in the midst of the technological milieu is very difficult. Yet it is necessary in order to respond appropriately. The meta-technical, as exemplified by software, is an interlocking and adaptive environment composed of truly autonomous ‘daemons’ which channels behavior by harnessing ideation and filtering relevant information to produce a simulated artificial reality that hides a singularity of pure immanence. On the other hand, the technological exemplified by the formal-structural system is pervasive yet anonymous. It is repressive and emasculates ideology. It tends toward autonomy while making everything available. It conceals its nature while pursuing the agenda of formalizing and functionalizing the world.

Technology is intensified and guided by the meta-technical. Yet at the heart of the meta-technical is a phenomenon of cancellation. The meta-technology is the limit of technology. Upon reaching that limit, technology turns into its opposite which is the proto-technology of Wild Being. This is exemplified by the rise of interest in acupuncture and homeopathy and other types of ‘alternative traditional’ medicines. The meta-technical is like a veil where one encounters the limits of technology and sees those limits clearly. From this encounter comes alternative or appropriate technologies. As one passes through those limits, there then appears proto-technologies which harken back to archaic technologies. In our time each of these approaches to technology are flourishing side by side. Our culture as a whole favors technological means until the side effects become unbearable. Then small is suddenly beautiful and we opt for appropriate alternative technologies. Thus, we all use leaded fuel until it starts to appear in cow’s milk which causes it to go into our bodies from which it does not emerge. Then suddenly we are environmentalists attempting to manage the exploitation and pollution of the environment enough so that it does not effect us. When even management fails, we then become proto-technologists attempting not just to find alternatives, but to go back to archaic technologies that were hither to rejected alternatives. No one tries acupuncture until all other forms of medicine fail. It is a last resort. In ancient

China it used to be the first resort. When drugs fail, we try stress reduction and other health maintenance strategies. When these fail we attempt to learn what the ancients know that we have forgotten. This harkening back to Shamanism, which is now called ‘New Age Thought,’ is a reconstruction and not the original. The original has been lost, and only the archeological remains are left. The proto-technical is a kind of nostalgia for what has been lost.

In software engineering we encounter the limits of technology by dealing with complex representations of systems we cannot see, which are nonintuitive and may act in counter intuitive fashions which are hard to explain. We work in an environment where our tools are software artifacts that help us build software artifacts. These software artifacts are simulations of parts of the world which must be simulated themselves in order to know if they work correctly. This simulation is done with test software which simulates the environment of the simulation to see whether the built software will track what is happening in the environment correctly. The software simulates without actually knowing anything about the environment. Whatever knowledge is not coded into the software product is thrown away. The attempt to regain and use this discarded knowledge is called ‘knowledge engineering.’ Knowledge engineering is proto-technical in that it attempts to get at the knowledge in the human being that allows him to know what is significant. It is an attempt to code the knowledge into a software artifact that will not forget it. Software forgets knowledge and preserves behavior. Knowledgeware sacrifices behavioral coherence for knowledge preservation. Passing through the event horizon of differing/deferring, the meta-technical realm collapses. The singularity is not seen, but disappears. What is left after the collapse is the proto-technical. In software this is the knowledge of the builder of the software. Automatic code generation is a proto-technical activity. Software reuse is a proto-technical activity. All activities which attempt to transcend the “software problem” are inherently proto-technical and deal with knowledge acquisition, preservation anduse. Software is trapped at the behavioral level. It ultimately lacks the knowledge that would allow complete adaptation to its environment. It is the human expert who best exemplifies this total adaptation to the environment. In the proto-technical state software attempts to become its own designer by replacing the expert designer. Software has become too complex for the human being. Now software defines software. The human reenters the arena of ignorance about software. Everyone is a user. The software engineer merely uses the software construction tools rather than their end products. At that point we will have passed through the barrier of software which was too difficult for humans to handle directly. Software with expert design knowledge must handle software. Software is in this way destined to become a non-human artifact. Humans built software. Software extracted the knowledge of how it was built. It then started generating itself, and humans became excluded because it was too complex and costly for humans to handle the building process. This sounds like a science fiction scenario. Yet already, code generating systems have been shown effective on simple problems. It is the nature of the meta-technical to collapse and push those who experience the limits of technology beyond technology into the proto-technological realm. In this we do not solve the problem of software because we are the problem. When we are excluded and pushed into the proto-technical realm, we, as a problem, have been solved by the technical system anonymously seeking autonomy. We have met the software problem, and it was us -- our incapacities and our limitations as human beings.

What is strange is that as the meta-technical arose outwardly, we simultaneously discovered our own nature as programmed artifacts intrinsically based on software. Crick and Watson discovered DNA, and slowly we began to read the software of all living things. Now we dream of carrying out the human genome project to understand fully the semantics of the program for which DNA is the text. As we project software outwardly, we discover it everywhere including in our biological foundations. Thus, the meta-technical was at work within us before we ever discovered how to embed it in our artifacts. In fact, much of what we call organic in the biological realm is very similar to the meta-technical in the realm of artifacts and socio-technical systems. Organisms are pre-eminently symbiotic and adaptive. They display a genuine autonomy as sources of independent action. Organisms display channeling through their adaptation, causing interspecies differentiation. Our current ecological perspective on organisms fitting together into interlocking niches to make up a whole, owes much to the meta-technical understanding of how technical systems work together. It is clear that with each new vantage point achieved in our own thought, we see nature differently. In the formal phase we saw animals as mere machines. In the structural phase we saw them in terms of organic chemistry. Now in the meta-technical phase they are seen as interlocking systems within an overall meta-system. It is the meta-technical that projects this meta-system as a way of understanding how things work together in the world. But since the meta-system is only apparent in the dynamic of interlocking systems, it is always hidden from full view. It is a projection of unity and continuity which is not apparent in the static things.

It is in this way a virtual environment within which dynamic systems interact. In that interaction the teleonomic filters of the different formal-structural systems work together to produce a super-filtering by the meta-system. The different ideational embodiments react to each other and create an all encompassing simulation in which the forms within the illusory continuity react to each other. In the meta-technical realm a complete virtual reality is for the first time possible. The meta-system is another name for this virtual reality which is a whole of a different kind. Similar to the wholeness of organisms which exhibit behaviors qualitatively different from molecules and atoms. So to the virtual reality of the meta-technical exhibits attributes qualitatively different from the technical. Our appreciation of that difference allows us to appreciate organisms in a new light, especially since we find a form of software embedded in both the biological and the meta-technical realms.

The meta-technical has not yet taken hold completely of the technological environment. And already a new shift away from the meta-technical towards something else is in the wings. We might call the next phase after meta-technology: *proto-technology.* Meta-technology is the intensification of technology to the point of all encompassing domination. Ironically, the point of total domination is also the point of collapse of the technological sphere. What is left after that collapse is the proto-technical. Just as the meta-technical sits on top of the technological formal-structural system, so too, the proto-technical sits on top of the meta-technical. One might think of the proto-technical as the inhabitant of the virtual reality created by the cocoon of software. In this virtual reality the rules of technology no longer apply. Instead, totally new non-technological rules apply. Rules we have yet to fully discover. Several different points of departure into the virtual space of the proto-technological realm have been taken. To name a few, there are rule-based systems, neural nets, simulations (continuous and discrete), autopoietic systems and cellular automata. These are all examples of proto-technical artifacts. They exist completely within the cocoon of the software virtual reality. They are each attempts to explore that reality in different ways. One might call them ad hoc exploratory devices. They all have in common the fact that they exist as experiments in the workings of virtual reality. Where software must still deal with the outer technical sphere, these experiments can take place completely buffered from the demands of the technical sphere.

From the viewpoint of the technical sphere represented by the formal-structural system, the inner sphere of complete virtual reality is irrational and enigmatic. It is a world in which free form conceptual structures may be created and set to interact to produce strange dynamics. These conceptual structures and their dynamics do not have to bear any resemblance to physical laws or be bound by physical constraints. Because those constraints do not apply, many times it allows the theoretician the freedom to explore avenues unavailable in physical experiments. The results of these thought experiments carried out in virtual reality may provide new insights into why the world is exactly the way it is. The irrationality of the proto-technical can be its very advantage. The pilot does not have to take a chance of killing the passengers and wrecking the plane to learn a delicate and complex maneuver. The rule-based system can come up with strange and enigmatic results which are non-intuitive, which then applied to the real world turn out to be true or offer a new perspective on some obvious facet of existence hither to not appreciated. An instance of this is when theorem provers “discover” a new theorem which no one thought of before by automatically exploring the conceptual landscape of the formal system in a way no one had thought of exploring before.

Because the proto-technical has not emerged as fully as the meta-technical, it is not possible to be as definitive in describing this new realm. All that is really known is that it is non-technological and appears highly irrational to those of us brought up to expect everything to fit the mold of the formal-structural system. The current centerpiece of proto-technology is the Mandelbrot set which displays chaotic patterning and is the most complex mathematical object yet discovered. It could not have been discovered without software to control the repetitive calculations that makes the Mandelbrot set visible. However, software is just a means into the virtual reality of the Mandelbrot set. Once there, the Mandelbrot set has its own reality and complexity and beauty which boggles our minds. Software provides the means of seeing the Mandelbrot set, but the set itself has its own reality which would have been there if we had never seen it and with which we must come to terms with independently of the software which lets us explore that reality. At this point we do not know how many monestrous things like the Mandelbrot set might inhabit the virtual reality cocooned within software. We are like explorers on the edge of an alien wilderness. There the norms of the formal structural system collapse, and we see strange objects which we must find new ways to understand. Mostly, we use that virtual reality to simulate our designated as real world[6]. But some explorers like those who found the Mandlebrot set will explore virtual reality for its own sake divorced from the constraints of mimicking our designated as real world. Slowly these phenomena from virtual reality will impact

6. See David Gelertner MIRROR WORLDS

our designated as real world. For instance, we are already starting to understand the strange behavior of computer networks in terms of chaos which we are in turn finding in natural systems. Chaos was first discovered by an attempt to construct a simulation of weather. When separate runs of the simplified atmospheric model diverged when started again and again from what seemed the same point, it was a strange phenomena for a deterministic system. Now we understand that many deterministic systems are in fact chaotic. And we have glimpsed the fact that the sensitivity to initial parameters is only the first level of Chaos. There are some kinds of Turing compatible machines that exhibit a deeper form of chaos, the ultra-complex, not dependent on initial starting conditions.

Thus, the exploration of the irrational nature of virtual reality inside the cocoon of software has begun. What we will find in that imaginary universe is unknown. But it is strangely accessible in a way that the physical universe is not. There is no light speed barrier here. All that is needed is complex software instruments to carry out the exploration. What emerges from the proto-technical virtual reality will have deep and profound effects on our existence. Already the emergence of ‘chaos’ has set limits on our ability to understand much of the physical world. Our ability to handle complexity is extremely limited, and one of our first discoveries within the proto-technical realm is that physical and biological reality is *very very very* complex. Also, experiments, except of a very gross variety, are all essentially non-repeatable as slight differences in initial conditions may have profoundly significant effects on the outcome. Slowly we see the characteristics of virtual reality manifest in the physical world, and we come to understand the designated as real “physical and biological” worlds in a completely different light. Where that will lead us is anybody’s guess. But just as we discovered software at our biological core, we can be sure that as we explore the inner more accessible virtual reality, we will discover even more amazing things about ourselves. Software and the meta-technical gives us an entryway into the inward realm which is culturally and intersubjectively defined. In it we will discover the true interpersonal archetypes as aspects of virtual reality. We will enter a new mythic space in which our dreams and fantasies take on a life of their own and become dynamic entities in a virtual environment with which we interact. Perhaps there we will discover the aliens we seek on other worlds, as the irrational aspects of the virtual proto-technical reality are embodied by our ‘daemons’. When that deeply irrational element from the virtual reality begins acting in the world and controlling machines through software interconnect, then very deep impacts on the lifeworld will be seen.

Technology has laid the groundwork, but it is the meta-technical that has brought us to the threshold of the proto-technical virtual reality. There we can simulate the interaction of particles in the first moments of the universe, and perhaps through that simulation get a glimpse of what lay beyond the beginning, going in our imaginations to places where we would have no access physically, and still learning something beyond mere speculation. This is the threshold on which we stand -- beyond technology and even reaching beyond meta-technology on the shores of an unknown, almost totally unexplored virtual reality which is our own inward dimension.

Publisher:

Apeiron Press

PO Box 4402,
Garden Grove, California
92842-4402

714-638-1210
palmer@exo.com
palmer@think.net
palmer@netcom.com
Thinknet BBS 714-638-0876

This book was typeset using Framemaker document publishing software by the author.

Publication Data:

Library of Congress
Cataloging in Publication Data

Palmer, Kent Duane
(aka Abd al-Alim al-Ashari)

THE FRAGMENTATION OF BEING AND
THE PATH BEYOND THE VOID:
Speculations in an Emergnet Onto-
mythology

Bibliography (tbd)
Includes Index (tbd)

1. Philosophy-- Ontology
2. Sociology -- Theory
3. Mythology -- Indo-european

I. Title

[XXX000.X00 199x]
9x-xxxxx
ISBN 0-xxx-xxxxx-x

Keywords:

Being, Ontology, Sociological Theory, Indo-european Mythology, Plato’s Laws, Emergence, Technology, Worldview, City Form

Electronic Edition:

Adobe Acrobat PDF

Available from http:\\server.snni.com:80/
~palmer/dialognet.html