4 Symbols

Consider the Renaissance algebrist Franciscus Vieta, who demonstrated the power of symbols as well as anyone. Vieta's contribution merits a brief explanation for the nonmathematician.[1] The story begins with the fact that Medieval mathematicians contented themselves with the use of natural numbers. Recall that the natural numbers are positive whole quantities phenomena found in naturefor example, seventeen sheep. This usage may seem normal enough, until we note that even the most basic operations of arithmetic are not closed under natural numbers: that is, the outcome of the simplest operations may fall outside the set of natural numbers. For example, although six minus four yields a natural number, four minus six does not; similarly eight divided by two yields a natural number, but seven divided by two does not. The traditional solution to this problem was straightforward: Medieval mathematicians declared expressions of the latter sort to be "impossible." One simply *could not* subtract six from four.

Vieta proposed a better solution. In his *Logistica Speciosa* (1577) Vieta introduced a letter notation by which given magnitudes, to be represented by consonants, were distinguished from those that were unknown or sought, to be represented by vowels. This might be recognizable as the precursor to the Cartesian notation, such as $ax + by = c$, where letters to represent givens are taken from the beginning of the alphabet, and those for unknowns from the end. (This adaptation of Vieta's system has survived to this day, so let us use it here.) This might seem trivial to us, but in the sixteenth century it was a turning point in the history of algebra, much as the Arabic invention of zero had been the turning point for simple arithmetic. Here is the logic. Under this system, six minus four, and four minus six, both become simply $x = a\ b$. Similarly both eight divided by two, and seven divided by two, become simply $x = a / b$. Using letter equations for simple operations creates a *symbolic* solution. Using this notation it becomes difficult to hold that the expression $a\ b$ has meaning only *if* a is greater than b, or that a / b is meaningless when a is not a multiple of b. Moreover it also becomes impossible to tell from the face of the symbols whether an arbitrary expression cd brought before us is legitimate or not.2

Sooner or later, these conditions might suggest that there is no contradiction involved in operating on these symbolic beings as if they were legitimate numbers. Possibility and impossibility are not intrinsic to the operation, but rather are simply restrictions that arbitrary tradition had placed on the field of the operands. Remove these artificial barriers, and it becomes a simple step to recognize those symbolic beings a and b as numbers *in extenso*. This generalized set of numbers providing closure on simple arithmetic then needs a theory.3 Thus the use of symbols introduced by Vieta led to a powerful abstraction known as the rational numbers, and so to modern Cartesian algebra, and the rest is history.

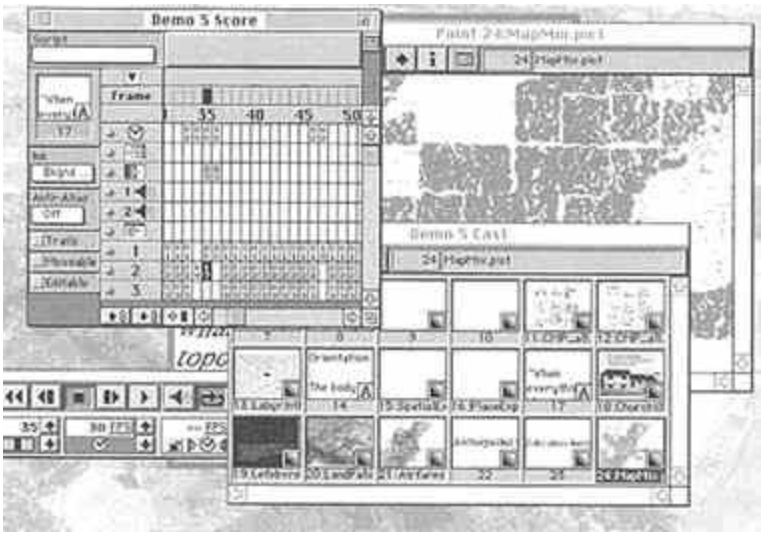
The point should be clear: in symbols we have the ultimate tools of abstraction. Not everyone may care to explore what this means, but it is important at least to recognize. Clearly the source of the symbol's power is that the symbol is not the signified. Symbols differ from that for which they

stand, and they are presumably easier to use: easier to reproduce, easier to transmit, easier to transform. Else, like the Sages of Lagado in *Gulliver's Travels,* we should carry about the subjects of our conversations and hold them up before one another to save speech. Furthermore, besides standing elegantly for more cumbersome things, symbols may represent concepts, conditions, or qualities lacking in embodiment or measure things we could not carry about at all.

In processed symbols we have a basis for formal reasoning. This power comes from the fact that, when replaced by a symbol, an object becomes a mere operand. As an operand, it may be manipulated in an abstract manner, such that the syntax of forming consistent expressions, rather than any representational significance of the symbols, governs the formation of new expressions. And as shown in the example, an operand used in such a formal system may be transformed to suggest a new meaning for which there is no objectno previous external significance. The rigor of the formal syntax suggests constructing an indicated meaning: a new representational significance emerges from syntactical operations alone. This is how symbolic processing yields abstract results.

In symbolic structures we have notations for formal reasoning in a variety of disciplines. Symbolic structures apply the principles of formal reasoning to many more kinds of operands: not just magnitudes or quantities, but coordinates and fields of numbers; not just algebraic numbers, but musical sounds, chemical compounds, logical assertions.

It follows that a technology for processing symbolic structures should be a very powerful means of abstraction. Using computers we perform not only calculations, but also communications, visualizations, and abstract structural transformations; and we perform these not only in the fields of traditional notations, but also on attributes of people and places, all variety of documents, millions of colors, shapes, three-dimensional forms, industrial processes, and global communications networks. Today computers dominate information storage and formal reasoning. Tomorrow, as we are often assured, they will become the province of many new forms of thought processing.4



4.1 Many nontextual processes now have notations, such as this multimedia orchestration in *Director*

Symbolic Context

Symbolic notation has taken many forms. Probably the first writing was less like word processing and more like a spreadsheet: tables and formulas for accounting purposes, presumably for taxation. (Note the parallel with early applications of computing.) Only later did there arise a need or desire to inscribe the spoken word. Tallies required only a few symbols, and tools for pressing on clay tablets favored pictographs. Pictographic writing existed for over three thousand years before the invention of a phonetic alphabet. But once developed, linear sequences of phonetic characters slowly and steadily became the dominant means of learning and recorded communication, at least in the West, and have so remained for six thousand years. Only in our century did information tasks finally become too much for

writing. New technologies arose in the form of pictorial telecommunication and computers, which now are uniting.

In a sense, any markings made for explicit purposes of representation, record keeping, or transmittal could be understood as notation. Pictures, numbers, and text have been the usual formats (and today these are being united by their underlying representation in bits). Of these, pictures have been around the longest, and still reach the widest audience. Images feed the imagination especially well, and as discussed earlier, images speak to the greatest range of people. They are the stock-in-trade of contemporary culture. However, as also noted, while images are very easy for electronic media to record and transmit, they are very difficult for computers to interpret. If we wish to understand notation as an unambiguous symbolic code, images are the least manageable format. Neither images nor their readings are based on distinct elements or syntactic conventions. Furthermore, any given image may have many implicit meanings. For example, classical painting relies on conventional identities rather that overt expressions. This is a matter of affinity: the symbol suggests a meaning but does not express it literally. For example, a lion may be taken as a symbol of strength, but a lion is not strength itselfit is a lion.5

More often, context and structure shape the use of symbols. For example, three sides of a triangle do correspond with the three persons of the Trinity, but a triune God has many attributes not possessed by a triangle, and the same triangle might equally well symbolize something else, such as the Nile delta, and so fertilitydepending on the context. On a highway, however, a triangle means to yield the right of way, an act that bears no affinity to the triangle. So relieved of any requirement for affinity, the symbol becomes the sign, and the context becomes arbitrary, constructed, and relative. With signs, resemblance is unnecessary, and indeed any picture may stand for anything. These propositions underlie countless disciplines: aesthetics, structural linguistics, cognitive science, psychoanalysis, software engineering, signal processing, art.

Take text. Arguably the richest symbolic contexts occur in writing. According to the philologists, the meaning of even a single word depends

on time, place, and usage. A complete text, then, is not only a construction of time and place and outlook of the author, but also those of other authors, and also (on each occasion) of a reader. No wonder that growing webs of intertextuality may merit a genuinely infinite number of studiesstudies of the investigation of methodology, investigations of the methodology of study, methodologies of the study of investigation, and so on. And whatever is understood from such hermeneutics can then be taken as a metaphor to be applied to other symbolic contextsthe built environment for instance. Contexts are everywhere. So it would not be entirely facetious to speculate how among dogs there exists a metaphorical "text" composed of smells.

Perhaps the Enlightenment philosophers were anticipating just such difficulties when they sought to establish a universal language of discourse. The modern literary critic Hugh Kenner draws our attention to one Thomas Sprat, Bishop of Rochester, of the late seventeenth century, known today on the merits of just two sentences, his classic plea for scientific writingan indictment of "Eloquence, Luxury, and Rhetorick," albeit amusingly phrased in those very manners.

> They have therefore been more rigorous in putting in Execution the only Remedy that can be found for this *Extravagance;* and that has been a constant Resolution to reject all Amplifications, Digressions and Swellings of Style; to return back to the primitive Purity and Shortness, when Men deliver'd so many *Things,* almost in an equal Number of *Words*. They have exacted from all their Members, a closed, naked, natural way of Speaking; positive Expressions, clear Senses; a native Easiness; bringing all Things as near the mathematicall Plainness as they can; and preferring the language of Artizans, Countrymen, and Merchants, before that of Wits, or Scholars.6

It was in a similar spirit that Gottfried Leibniz proposed the *Lingua Philosophica* (1666). The great mathematician was convinced that a univer-

sal logic would help solve political misunderstandings resulting from the inadequacies and multiplicities of language. His logic would be applied to basic elements of reason, and it would be operable by formal manipulations of charactersa *calculus ratiocinator*. Such reasoning processes could be executed automatically, and indeed Leibniz was the first to propose a reasoning machine.7

Of course it was impossible to foresee the difficulties of achieving a universal formal language. From what we now know about linguistics, it was utterly naive. But it was also impossible to foresee the power of some particular formal systems. It turns out that the route to a more generally useful *Lingua Philosophica* was to get more explicit about semantics and notation. When in the twentieth century Alan Turing specified a symbol-processing scheme that could be implemented in an electronic machine, there finally existed the means to do just this. Some time would have to pass before these schemes would develop sufficient vocabulary and structure to pass for languages, but Turing understood his formalism to be for much more than just numerical calculation. Today we might say that programming language differs from natural language mainly in its basis for semantics, that is, in the unambiguously declared meanings of words. Even when contemporary computer languages extend this idea to high-level abstractions such as class libraries, code is pure convention.

Convention is a form of abstraction, or a means to abstraction. A mathematician might tell you that choosing the right symbols is half the battle in proving a theorem. Although the symbols are merely conventions, and the theorem itself is the main focus of abstraction, the choice of notation does affect the relative ease or difficulty or achieving the results.

Formal Notation

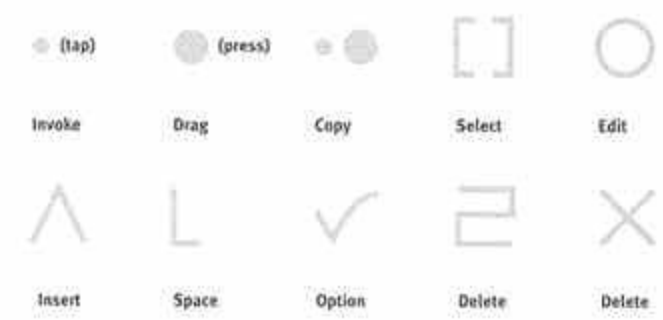
Mathematical symbols are an explicitly formal notation, and formal notation is a special case of symbol usage. Although any representation might loosely be understood as a notation, certain formal systems deserve

particular attention. This is not an easy subject for conversation, nor one that summarizes well, but it is one worth getting a sense about as a foundation for studying the computer as a medium.

For a rigorous theory of formal notation and its relation to creative work, we may turn to Nelson Goodman's groundbreaking book, *Languages of Art* (1976). Notation, as formulated by Goodman, is essentially defined as a "symbol system" consisting of a symbol scheme unambiguously correlated with a field of reference. For example, letters unambiguously represent quantities in algebra. A symbol scheme is generally a set of distinct characters plus a syntax for combining them. The essential features of such a scheme are that the characters are finitely differentiable (ultimately you can tell which is which) and that the syntax is character-indifferent (or substitutableone instance of a character is as good as another). These properties are common to many familiar schemes, including alphabetical, numerical, binary, telegraphic, and basic musical characters. They are not universal, however: they are absent from informal representations such as sketches. A sketch with its dense field of overlapping, ambiguous, uniquely executed marks is not a symbol scheme.

According to Goodman, a symbol system is the correspondence by which meanings comply with characters in a symbol scheme. This has similar properties of articulation. For example, all inscriptions of a given character must have the same semantic reference, or compliant. No one compliant can be referred to by multiple characters. And for the system to be a useful notation, these semantic distinctions themselves must be finitely differentiable. For example, in a musical notation, a C sharp quarter note represents a particular pitch and relative duration; all instances of this note in a given register have the same meaning; those are intelligible from other sounds; it is clear when you have an instance of the one; and no other note means the same sound.[8]

The most elegant instances of formal notation occur in music. Standard musical notation has endured for centuries, is common to many different cultures and languages, and has successfully incorporated many



4.2 A very small system of differentiable symbols:
gestures for a pen-based computer

modern practices. It has also extended well into digital media such as time-coded sequencers. A work of music exists in the abstract, and it may be performed, transcribed, arranged, etc., according to its notation. Two different performances are clearly instances of the same work. Note the preeminence of the score. Goodman observed, "The composer's work is done when he has written the score, even though the performances are the end-products, while the painter has to finish the picture."[9]

Painting is different. Here, authorship and execution are united. Even the most accurately duplicated copy would not be the same as an original. Painting defies notation, which Goodman claimed is because we lack any real definition of what constitutes a work of painting. We can develop a nominal notation that is essentially a new coinage for each piece of work; we can develop a procedural notation based on history of production (e.g., an advanced color-by-number for producing copies); but not both.[10]

Architecture is a mixed case. It does use intermediate representationsdrawingsyet we do not think of these as the work. Construction plans, in which relative position, script dimensions, and established tolerances govern the reading, and in which appearance to scale is just for convenience, are true notations, whereas sketches are not. Yet plans are not like

scores. "We are not as comfortable about identifying an architectural work with a design rather than a building as we are about identifying a musical work with a composition rather than a performance."[11]

Goodman introduced terminology to distinguish fine arts on the basis of notation. Arts such as painting for which the artifact is the work and there exists only one original he calls "autographic." Arts such as music where the notation carries the work and multiple instances are possible he calls "allographic."[12] We may note that the latter is more abstract. One route to abstraction is to incorporate formal notation.

At first consideration, traditional crafts are purely autographic. They lack notation, are made by hand, and produce no two pieces exactly alike. Yet if we discount the exception of the masterworks, and concentrate on routine artisanry, we find a different situation. Here among the countless indifferent pieces made according to any one trade, one piece is as good as the next.[13] Although the pieces are not identical, or modular, like machine-made wares, nevertheless they are effectively interchangeable. No one stands out as the original, and certainly there are no forgeries. Here one might argue that the intellectual property is not so much the artifact as the tradition by which it is madesuch was the stuff of apprenticeships. As we have noted, this tradition is tacit. Even in the unusual case where there exists a written specification or a drawing, that notation is not considered the work. The work of craft is neither the design nor the individual artifact: it is the tradition of the very production. It is the presence of many objects identical in their conception, and interchangeable in their use, but unique in their execution.

Notation confers several advantages besides the obvious capacity to obtain multiple instances of the end product. For one thing, it is reproducible also in the sense that the score itself may be copied. Ownership is not of an object but a copyright. But more important, notation allows composition in the abstract. This means that modifications proceed quickly without the encumbrance of executing or performing every stage of the evolving composition. Multiple instances of the notation might represent different stages or versions of that work. The intrinsic syntax of the nota-

tion might suggest compositional approaches. At the simplest level, these might include elementary shifts and substitutions. One could think in terms of variables and transformations. There would be nothing to prevent the use of logical variables. Thus at an advanced level, the nature of notation might invite us to employ iterations and conditionals, which together with variables and their relations would turn scripts and scores into true programs. In sum, symbolic processing may assist compositional explorations. We can make rules, and we can identify structure.

Structure

Formal notation invites the study of structure, which in natural language consists of grammar. It is a cornerstone of twentieth-century intellectual history that comparative grammars reveal common underlying structures. Words themselves may vary between languages, and indeed may be arbitrary, but certain correspondences exist between languages at the level of phonetic systems. Ferdinand de Saussure's *Course in General Linguistics* (1916) established these principles and in so doing opened the floodgates for the development of disciplines such as semiology, structural linguisitics, developmental psychology, and structural anthropology. These many endeavors have adapted the notion of structural equivalence from descriptive grammars of language to interpretation of culture at large. In the process, there emerged much thought that linguistic structure somehow reflected the structure of the mind itself. Structure was inherent, and could be latent. The psychologist Jean Piaget wrote: "The discovery of structure may, either immediately or at a much later stage, give rise to formalization. Such formalization is however always the creature of the theoretician, whereas structure itself exists apart from him."[14]

By midcentury, there reemerged an interest in *building* symbolic expressions, rather than merely analyzing them, and this redefined grammars once again. Noam Chomsky's *Syntactic Structures* (1957) introduced the idea of generative grammars, as Chomsky emphasized the creative use of language for constantly developing new expressions. Unlike his predecessors,

Chomsky felt that a more general grasp had to underlie the process of inventing or understanding legitimate sentences that had not been uttered before. He thus renewed the interest in universal language with his distinction between the syntax of any particular language, which he called "surface" structure, and this more general cognitive process, or "deep" structure. Furthermore, he focused new attention on formalism by developing a notational system of rules for generating syntactic structures.15

Abstraction in Data Structure

Not surprisingly, many computational theorists share an interest in generative notations and grammars. After all, computing is essentially the use of symbolic notation on a huge range of scales, from microscopic arrangements of charges that yield logical interpretations (e.g., memory) right on up to planetary webs of computer networks.

Computing is structure manipulation. It is an abstract medium based on generative symbolic notation, particularly data structure, which can represent increasingly high-level concepts. To begin to see how, it is worth reciting two fundamental tenets of computer science: data structure, and data abstraction.

Data structure characterizes software: what you can do, and how a program looks and feels, depends on its underlying abstractions and assumptions. In essence a data structure organizes symbols for specific purposes. It formulates vocabulary and operators useful to some specific end, whether its record of symbols is to be treated as a document or an artifact, as real or as virtual. For example, one kind of structure might define a repertoire of shapes and area fills for use in an illustration program. This structure is itself a notation in a programming language, and it also establishes a format for creating and recording specific notations in the language of data elements it defines. What to the computer is a record of specific instances of elements within a defined repertoire is to you a drawinga record of a session or body of work in a particular symbolic context.

Much as in Vieta's simple algebra, declaring a variable is the most fundamental act of building a data structure. A particular selection of

variables opens up a particular set of possibilities, or casts the solution to a given problem in a very particular way. Of course, individual variables are grouped into compound elementsspecifically structured typessuch as shapes in an illustration program. Thus despite being based on the simplest, atomistic, data types of numbers, letters, and true/false values, a limitless variety of formal elements can be described: from words to sounds to forms to motions. Once established, these data objects may be given names, associated with positions in menus or maps, identified with pictographs, manipulated with sliders and knobs, and so onthere is no limit to the operations. As use of abstract data structures, computing is not so much calculation as generalized symbol manipulation. What makes data structure such an essential characteristic of a digital medium, then, is *which* kinds of elements and operations it establishes in any given context. A robust illustration program such as *Freehand* has a very rich data structure that shapes a useful symbolic context for fine linework. You can draw beautiful splines with control over second derivative of curvature, for example, but you cannot sweep those curves into surfaces, or animate them in time. To do so would require some other data structure.

This should already suggest something about this second and related fundamental idea, namely data abstraction. The history of programming may be understood as largely a matter of increasing abstraction. The earliest, most primitive software stored data solely in terms of indexed addresses in physical memory registers, and to use it required physical knowledge of the computer memory. Since that time, there have been three major advances in the independence of data structure from data storage technology. First came the capacity for named variables, which allowed memory allocation to be accomplished in software, and for this the new languages were heralded as "high-level languages." Second, languages introduced pointers, such as the identifier keys used in relational databases, which allowed for much more efficient programs by processing simple addresses rather than manipulating the more cumbersome data itself that is stored at those locations. (This is analogous to using weightless words rather than holding up bulky objects like the sages of Lagado). Third, and more recently, came the means to represent the identity of an instance

independently of how it is found, where it is stored, or what it contains. This is known as object orientation.

The basic idea behind object orientation is to provide a single identity for any arbitrary set of properties and capacities. By uniting the two main acts of earlier programming methods, namely declaring structure and defining procedure, it enables a level higher of abstraction that insulates the outward behavior of a system of elements from its underlying implementation. This fundamental building block is known as an abstract data type, or class, instances of which are objects.

Programmers may find this trivial and nonprogrammers may find it opaque, but object orientation has several consequences for the growing perception of computing as a medium. Much of this simply reflects increasingly flexible software engineering. Three particular conceptual constructions obtain: first, one class may inherit properties (structure) from another; second, one same operation may be "overloaded" to be applicable to many different object classes; third, different operations now have a way of finding out whether they are referring to the same object. All this allows higher-level software design with greater independence of specific technological implementation. It makes for a more robust and modular structure capable of underpinning a much greater variety of software artifacts and operations. It is more than engineering performance, then: better abstraction lets programmers manipulate concepts at a higher level. Abstract data types have been a big advance, because like Vieta's alphabetic symbols, they allow semantic ideas to be manipulated independently of their actual compliants. From a software designer's standpoint, then, a class library is a powerful generative structure.[16]

The non-technician should note: here in what is fittingly known as object orientation is the very root of one of the biggest advantages of digital media, namely the ability to operate on abstractions as if they were things.

Generative Structure

Much as mathematics is largely a matter of choosing the right symbols, software design is a matter of defining an appropriate structure for serving

a task or problem. Likewise software usage is a matter of grasping the uses and limitations of such structure. Often the best way to do this is through manipulation. Generative structure is the beginnings of a medium largely because it invites manipulation. Piaget wrote:

> As a first approximation, we may say that a structure is a system of transformations. Inasmuch as it is a system and not a collection of elements and their properties, these transformations involve laws: their structure is preserved or enriched by the interplay of transformation laws, which never yield results external to the system nor employ elements that are external to it. In short, the notion of structure is comprised of three key ideas: the idea of wholeness, the idea of transformation, and the idea of self-regulation.17

A structure has a feel based on internal laws and self-regulation, which we experience by working on itthrough transformation. This is a fundamental idea behind any understanding of the computer as a medium. The structures we manipulate are more than collections of elements: they are dense notational *contexts for action*. The experience of structure is difficult to describe in words, but obvious to anyone using software. Available operations, accumulated work, and the organization of sessions all contribute to a strong sense of the rules of the game.

Creative computing gives special emphasis to generative structure. Developing new artifacts and expressions depends on sound aesthetic theory informed by syntax and grammar. Computing is not a radical departure but a natural extension of intellectual development that can be expressed in terms of our knowledge of notations.

As a point of departure for a digital aesthetic, Steven Holtzman has summarized the history, function and significance of generative structures in his recent book, *Digital Mantras* (1994). Holtzman places Chomsky's deep structures and generative grammars squarely in the path of an intellectual history dating back to ancient India and Panini's formulation of the *Astadhyayi* grammar of classical Sanskrit (which literally meant "perfected"). According to this interpretation, the *Astadhyayi* was the first

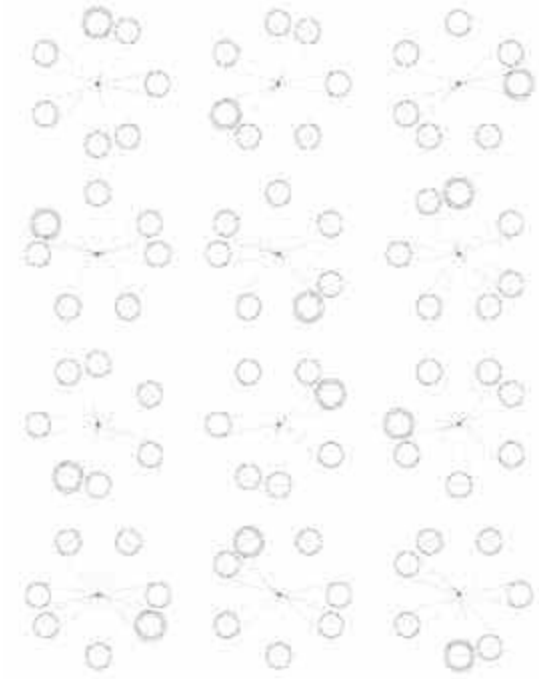
generative grammar, and its preeminence at such an early point in intellectual history may suggest some fundamental human dispositions. Thus it comes as no surprise that there are counterparts today: using examples from modern music and painting, Holtzman explores the use of structured symbolic processing as the basis for creative expression.

For example, he studies examples of tonality in Western music. In particular, the interval of seven semitones, called the fifth, which since the ancient Pythagoreans has been thought to be especially harmonious, allows us to cycle though the twelve semitones of an octave in such a manner that each of the notes will be heard once before any is reached a second time: (C, G, D, A, E, B, F#, C#, G#, D#, A#, E).[18] Holtzman describes classical Western music essentially as "an exploration of the circle of fifths, an exploration of the expressive possibilities of a system of music based on the special characteristics of the interval of the fifth, [and] also an exploration of the musical structures that can be built exploiting these expressive possibilities."[19]

This approach generalizes effectively to other notations. Formal structure provides a basis for understandingand generatingexpression within a variety of media. Computation introduces formal notation and structure where formerly there were none. We might say that it makes some autographic media allographic.

For example, sculpture now has a notation. Surface and solid modeling systems construct formal representations of complex forms. Constructive solid geometry hierarchies based on the union, intersection, and subtraction of elemental forms such as boxes and cylinders introduce a sense of cutting or filling volume, like with clay. Spline surface lofting systems based on sweeping arbitrary curves along arbitrary paths introduce a sense of bending and morphing surfaces, like with sheets of plastic or metal. Forms so described may then be fabricated in physical material even carved in stoneor else they can be animated like Spielberg dinosaurs. Faster technology continually improves the capacity to work these notations not only by the measures and constructions of design, but by the free-form gestures of sculpture. You may believe that sculpture requires

Starting points precess through set of
twelve without omission or repetition



Each version differs from the original by one more
(or fewer) element than its predecessor
4.3 A graphical rendition of the circle of fifths

palpable massand certainly we will have to examine the perception of medium in immaterial formbut many artists are happily exploring the possibilities of these highly structured three-dimensional notations.

In a generative system, symbolic processing of data structures establishes an underlying structure, which under the proper conditions can take on aspects of a medium. This structuralist argument is therefore an important contribution to the question of digital craft. As Holtzman puts it, "The essence of the structuralist view is that the key to studying language is the distinctions and relationships that emanate from the underlying systemand that this view is applicable to all systems of meaning." This condition is general: "All languages (in the broader sense) can be studied in terms of formal structure. Even the most expressive and subtlest languages can be studied in terms of formal structure, including the languages of poetry, painting, sculpture, and music."[20]

This Platonic and universalist position stands in obvious contrast to the relativistic, poststructuralist and (perhaps not coincidentally) anticomputational stance of much current academic theory. Yet its renewed structuralist emphasis may reflect a widespread interest at least among the digerati. And as a polemical approach toward a digital aesthetic (which Holtzman does call just a stake in the ground) its focus on generative structure seems appropriate and welcome.

So here is another stake in the ground. Although it is important to respect generative structures, we must also acknowledge that their power will be realized through the complementary role of personal sensibility, and that this may occur through the perception and practice of medium. *Theories of design computing demand attention to the skillful nature of accomplished symbolic manipulation*. Explorations of generative structure obtain power from hand, eye, and tools. They arise from personal knowledge, practice, and commitment of the sort found in traditional handicrafts, now applied to symbolic systems. Skillful operations of physical devices are given leverage through effective symbolic structures such as human-computer interfaces and cumulative software constructions. Through the abstraction of symbolic representation, practiced, playful talent finds new outlets and develops new kinds of appreciation.

The Limits of Structured Symbolic Thinking

One might argue that the ultimate symbolic systemsgreat softwarecould eventually just do all the work. But this is not the case, for there will always be intentions that we cannot or choose not to express in symbols. Specifically, there are limits to the domain of formalized code. In all likelihood, a human-computer partnership will continue to surpass either the unaided human or the autonomous algorithm for some (important) aspects of work. Neither decidable formal manipulations alone, nor traditional craft unleveraged by symbolic systems, should be able to keep pace with a partnership between inarticulable insight, or impetus, and rigorous symbolic reasoning. So long as this is the case, personal practice will prosper primarily in its coupling to digital notational systems, and digital notational systems will be useful just as much as they encourage human imagination.

According to a great many contemporary theorists, the mind has a structure of its own. It is generally agreed that particular knowledge does not reside in any explicit place, but that the mind is an epiphenomenon of brain structure, perhaps like the glow on burning coals. Nevertheless it is believed that language, vision, and perhaps even creativity may have corresponding physio-neurological structures. And though much of such thinking has centered on the formality of language, more general propositions have appeared on the formality of mind. Jung's archetypes, Saussure's linguistics, Lévi-Strauss's mythologies, Piaget's developmental stages, and a host of related theories all have emphasized structure.

It has been very easy, then, for some people to impute a symbolic processing model of mind. Because computers are the product of our minds, the mind must be somehow like a computer. The artificial intelligence research community of the 1980s was especially prone to take this view. Techniques such as inference chains, connectionist learning (neural nets), case-based and frame-based reasoning, etc., were at the vanguard of design research, and the rapidly emerging discipline of cognitive science informed and built a foundation for these investigations. The educational psychologist Howard Gardner became a most articulate advocate of symbolic-

processing models of mind, and he has taken a special interest in the role of symbol systems in creativity.

> To my mind, there is a crucial leap that a structuralist study of the mind must take. The leap involves a recognition that the basic unit of human thought is the symbol, and that the basic entities with which humans operate in a meaningful context are symbol systems. Attention to symbol systems is possible (indeed natural) within a structuralist framework, but such a focus opens up the possibility of the endless devising of meaningful worldsin the arts, in the sciences, indeed in every realm of human activity ... The key to an understanding of artistic creation lies, I believe, in a judicious wedding of structuralist approaches to philosophical and psychological investigation of human symbolic activity.21

There is something of a jump from believing that symbolic processing reflects the mind to believing that symbolic processing *is* the mind. This is an especially appealing prospect to academicians, who presumably wish to build more complete, possibly definitive theories of mind, as opposed to "endless devising of meaningful worlds."

As a result, many recent studies of creative computation have confined themselves to decidable methods in symbolic processing. Despite the loss of certainty having been demonstrated even in mathematics, orthodox academics in countless disciplines hold fast to deterministic knowledge. It is as if scientism is the only way to legitimate creativity, as though otherwise we are spiraling back, hermeneutically, to square onewhich you may recall as some courtly "Eloquence, Luxury, and Rhetorick."

Of course this very obsession with externalized knowledge has stigmatized this research in the eyes of traditionally creative people. Professional designers, for instance, have mostly spurned computation for anything other than task automation by direct manipulation. This has not just been out of necessity, but of choice. We have yet to escape the state where a sensible person can quickly dismiss computer usage for creative

work on very simple grounds: one, it's too arbitrary; two, it cannot record feelings; three, you cannot get a hold of it; four, it is difficult and time-consuming; and five, it's not much fun.

On the other hand, we have arrived at the time where there exist people who would challenge every one of those grounds. This is largely a generational issue: younger people who have grown up with computing do not seem to experience as much frustration. Furthermore, each next generation of technology (far more frequent than generations of human beings) becomes more usable on the basis of faster components, increasing practicality of more intuitive designs, and general accumulation of technological wisdom.

Maybe you can't touch it now, but there are signs that haptic interaction will arise. Maybe it can't record feelings, but no medium captures anything but a partial and implicit record of the state of an author. In any case, the growing perception of computer as medium demonstrates a slowly increasing capacity for implicit emotional content.

However, there is no denying the arbitrariness of symbol systems. Like any other medium or notation before them, digital media have no special claims on reflecting any creative processes more directly. Consider that notations are improvised ad hoc. For instance, you could keep time by hand. Start tapping your foot to a regular rhythm, and then count twelve beats. Count them on one hand, as the ancient Babylonians did, by moving the thumb from fingertip to fingertip, then from midsection to midsection, then across the bottom segments of each of four fingers on the same hand. This lets you count to twelve. Using your other hand, raise one finger for each round of twelve on the first. This way you could count to sixty. The Babylonians built a whole mathematics on this practicebase sixtybut it didn't last because it made arithmetic unusually difficult. But one aspect of it is still with us on the face of a clock. Since the Babylonians gave us timekeeping, to this day there are sixty seconds in a minute, and sixty minutes in an hour.22

We must remember that the most basic symbolic systems were created out of convenient objects, long before there existed any theories of

mind. Such ad hoc arrangements act in partnership with the human brain, but even in the most modern, sophisticated cases, such as computer programs, they don't necessarily work in the same way.23

Advocates of symbolic reasoning therefore need to recognize ways in which the mind is unlike a computer. For example, although rigorous symbolic contexts may help articulate qualities of imagination, they are no substitute for insight. We have explored this relationship as it pertains to visual thinking: human learning is by example, by context, and by intentbut not necessarily by rules. All of this makes it especially difficult for any notation, even with the power of computers, to extract useful information from pictures. It also suggests that we are mistaken to assume that even if the mind has intrinsic structure, it has *only* structure. It has structure *sometimes,* or it *changes* structures in the process of exerting multiple intelligences. We humans are very agile.

Besides, there seems to be a problem with symbolic systems that are completely context free. Consider the fact that being context free is both the main strength and the main weakness of numerical models. It is a strength in intelligibility: whereas extracting much content from pictures or texts remains beyond today's computing methods, extracting *all* the content from well-formed numerical expressions is the very basis of the technology. As demonstrated by Vieta, formal symbolic manipulations arrive at indicated solutions that suggest new concepts. But it is a weakness in contextual interpretation: numerical models have the unfortunate disadvantage of containing no information about when their contents are appropriate.

Inappropriateness might be an issue at a micro scale; for example, using a particular inventory and distribution model might be very profitable for one business and disastrous for another. Or, as the critics of computing point out, it could become global, and indeed become our biggest problem: abstractions such as bottom lines, admirably optimized in context-free numerical models, seem to have negative effects on social and environmental milieus. Pure abstraction and solely formal reasoning obviously need to be governed by human practice and context-based understanding. Perhaps this is why medievalists played it safe by sticking to the natural numbers. But where else, without rejecting modern mathematics,

logic, and computer programming as a whole, might we begin to draw the line? How can we represent skill and judgment?

Inspiration and Intent

To uphold reflectivity, imagination, and contextual appropriateness as practicable but unknowable does not require one to become a hippie mysticit is an increasingly legitimate intellectual position, whose heritage includes philosopher-biologists such as Henri Bergson, Pierre Teilhard de Chardin, and Gregory Bateson. Its current label is "biological naturalism."[24] It is an intellectual direction being taken by a lot of other fields besides visual studies.

In *The Rediscovery of the Mind* (1992), John Searle has provided a good recent summary of this position. Searle, a perennial opponent of the dualist crowd who regard the body as a mere "meat machine," argues that cognitive scientists have been working on false assumptions. Here are his opening assertions:

1. Consciousness does matter.

2. Not all of reality is objective; some of it is subjective.

3. Because it is a mistake to suppose that the ontology of the mental is objective, it is a mistake to suppose that the methodology of a science of the mind must concern itself with only objectively observable behavior.

4. It is a mistake to suppose that we know of the existence of mental phenomena in others only by observing their behavior.

5. Behavior or causal relations to behavior are not essential to the existence of mental phenomena.

6. It is inconsistent with what we in fact know about the universe and our place in it to suppose that everything is knowable by us.

7. The Cartesian conception of the physical, and the conception of physical reality as *res extensa*, is simply not adequate to describe the facts that correspond to statements about physical reality.[25]

Searle has effectively challenged the lack of affect in cognitive science, the overreliance on behavioralism in the study of software usage, and the wisdom of confining our discourse only to what shows up on our instruments. He has argued how consciousness is a natural feature of certain biological structures, much like being wet is a feature of the structure of water molecules. He asserts that intent is an undeniable component of behavior, but that you cannot detect intent in a detached experimental settingyou have to ask.

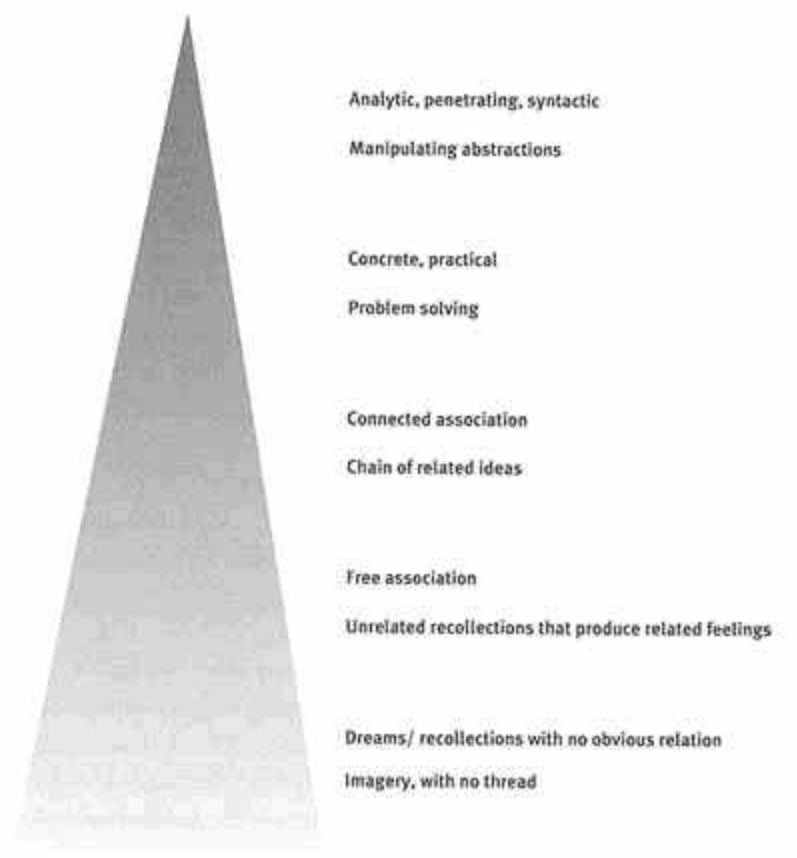
This argument is as pertinent to the humble digital craftsperson as to any lofty theorists of mind. If in essence consciousness is natural, and neither we nor our technology should ignore it, then more advanced conditions of design computing should differ from algorithmic, industrial machine operation precisely by reflectivity and intent. Yet generally, our routine computer usage has involved not the fullest and best forms of consciousness, and because computers are powerful means of production, which amplify and transmit our states, the results have often been unsatisfactory. This situation ought to be changed at the sourceboth the technology and its practitioners. Here again are some central themes. Digital craft is to will some sort of reflective consciousness in the practice of habitual computer-based work. It is to use the phenomena of tools, media, artifacts, and settings as means for focusing that consciousness. It is to cultivate contextual awareness.

Searle emphasizes how active intent occurs against a background context of capacities and presuppositions.[26] Here we might strike a crude analogy in the relation between a tool and a medium: a tool conducts intent, a medium forms a background of possibilities, and the two are inseparable. Note too how Searle's discussion of background ability bears similarity to Polanyi's concept of personal knowledge. An example used by Searle: a beginner may require a simple functional intent, for example to put his or her weight on the downhill ski; an intermediate skier can intend to "turn left"; and an expert can just "ski the slope."[27] Similarly in interpretive intent: one person might see colored patches in a painting, another might see a picture of a woman standing next to a window, and another might see Vermeer's respect for a newfound rigor in spatial illumination principles.

It is as if vision, thought, work all occur on a spectrum. David Gelertner has recently presented just this idea in his book, *The Muse in the Machine* (1994). Not only must there be a relation between focused reasoning and free association, he argues, but this relation is spectral, and not just not polarized as in the tired right-brain/left-brain metaphor. Gelertner's spectrum ranges from a "high" of detached formalism all the way down "low" to pure reverie. This spectrum essentially measures focus, as opposed to alertness. High focus suppresses associative reasoning. This focused, analytic state is best for manipulating abstractions, because it adheres best to syntactic and symbolic rigor. It suppresses associations like "Sheepare we counting sheepyou can't have negative two sheep!" By contrast, other states benefit from association. Practical, intermediate levels of focus such as concrete problem solving maintain a deliberative thread but allow for recollection out of the blue, as in, "Didn't we do it this way that time?" Lower, still partially focused states may lack any thread, but nevertheless maintain a sensibility for when two recollections engender the same feeling. This latter condition is the best for creative insight.[28] Gelertner emphasizes that moving up and down this spectrum is more effective than maintaining a uniformly analytic focus. Those who confine themselves to just one part of the spectrum, even if at a higher focus, are missing some opportunities. This is of course the big problem with computation. Although complete thought includes what Gelertner calls "affect linking," nevertheless cognitive scientists have been ignoring affect (feeling) for years. This Gelertner likens to sailors who, knowing how to measure latitude but not yet longitude, were just as good as lost.[29] Searle, incidentally, draws a similar comparison between exclusively functional-analytic mind theorists and the drunk who, having lost his keys in the dark bushes, looks for them out under the streetlight, because the light is better there.[30]

Achieving a Balance

In the balance, symbolic processing is too powerful to ignore, but not powerful enough to rely upon exclusively. We must balance the use of powerful notations with the use of tacit knowledge: visual thinking, kinetic skill,



4.4 Spectrum of focal attention

and symbolic reasoning can complement one another, whether in business planning, scientific analysis, or artistic improvisation. They not only can; they must.

How can we approach these theoretical propositions from the standpoint of practical artisanry? If rapidly improving dynamics of structured symbolic processing lead to wider use of computers in the creative process, does that mandate that we understand the role of talent, skill, and insight? Does it suggest that creative computation overcome the traditional separation between tool users and symbol users? Will the balance shift as physical images, forms, and fabrication processes become abstractly coded?

Or conversely, will the balance shift as abstract schemas are given sensory manifestation? Can the power of visual and kinetic understanding allow us to pursue new realms of abstraction, different from the ones opened up by nonvisual, nonmanual computing?

So we must askas increasingly it all comes down to bitscan there be a reunion of manual and mental work? May the coordinated hand, eye, and tool at last benefit from what has traditionally been the most powerful means of abstractionprocessed symbols? And conversely, may the work of symbolic processing be made more human by the ways of the eye and the hand?

In our pursuit of answers to these many questions, there is wisdom in coupling symbolic processing to all the human insight and skill we can muster. Here amid rapid advances in formal symbolic knowledge, this is no time to forget tacit, personal knowledge. After all, only human beings will make computing humane. Human beings like to make things; they like to use their hands at least as much as their brains; they like to play; they tend to do better with practice. If we are to tap the increasing visuality and dynamics of computing in order to open new realms of abstraction, we should depend very much on these humane traits to do so. So if there is a digital aesthetic to be had, getting there may well be a matter of abstracting craft.