LAWS AND TECHNIQUES OF INFORMATION SYSTEMS

J. Mende
Department of Accounting,
University of the Witwatersrand,
Johannesburg, 2000

Research has a utilitarian role of providing the knowledge needed by the designers of mankind's artefacts. In this context four distinct categories of knowledge are identifiable.

1. *Laws of Nature* describe inherent properties of natural entities found in artefacts.
2. *Laws of the Artificial* describe contrived properties of artificial entities.
3. *Natural Techniques* help combine natural entities to form artificial entities.
4. *Artificial Techniques* help designers combine artificial entities to form artefacts.

This classification can be used to identify deficiencies and special features of research on information systems.

1. META-RESEARCH

"Information Systems" is one of the youngest subjects of academic enquiry. Researchers have not had much time to consider what kinds of knowledge are necessary, let alone to find that knowledge. Accordingly, our subject matter has not reached the level of precision and consistency of more mature fields such as Physics and Electrical Engineering.

So in parallel with research on actual information systems, there is a need for "meta research" on the research process itself, to identify desirable directions of exploration.

A previous article [8] in this journal has suggested some potential inputs and outputs of research projects. The present article focuses on the output question: What kinds of knowledge should research produce? In an attempt to answer that question, it analyses the development of human artefacts in the context of increasing economic productivity.

2. INCREASING PRODUCTIVITY

Economic History [3] shows that centuries ago man toiled strenuously to secure the basic necessities of life - consumables such as food and drink, clothing and shelter. But over the ages people have created machines such as tractors, railways, computers etc, and they have established procedural systems such as mass production and information systems. These "artefacts" [10] have increased human productivity, allowing more consumables to be produced with less labour. As a result, Man today enjoys a much higher standard of living than his ancestors.

But Man is not satisfied with his present standard of living. He continually strives to replace existing machines and systems with new artefacts that produce even more consumables with still less labour.

However, the designs of these artefacts are constrained by Man's knowledge. For instance, the earliest civilisations had comparatively little knowledge of Science or Engineering. So designers could only develop relatively labour intensive artefacts such as bellows-driven smelters and oared galleys.

Today, Man's knowledge is both more extensive and more precise. Electrical Science and Engineering provide a basis for the invention of arc furnaces- which eliminate bellows from smeltering. Mechanical Science and Engineering have made it possible to invent steamships - which transport goods with a fraction of a galley's labour input. Chemistry and Chemical Engineering have allowed us to devise oil refineries and leach plants of unprecedented efficiency. Therefore better knowledge now enables us to design artefacts which produce consumables with less labour.

In order to achieve even more productive designs in the future, we need to improve the knowledge base for design. Firstly, we must extend it by gaining new insights on phenomena

which are still shrouded in mystery. Secondly, as our understanding is rarely perfect, we also require more accurate knowledge of familiar phenomena [7]. Accordingly Man's urge to develop more productive artefacts calls for ongoing discovery of new knowledge.

That means the design process depends on the research process - which suggests that research output requirements can be identified by *needs analysis* of design activities.

3. THE DESIGN PROCESS

Watching an electrical designer, a chemical designer or a program designer at work, one observes that he performs a series of physical actions : he writes, he draws, he calculates. But these actions do not happen at random. Each step must be activated by a prior decision. For example, before an electrical engineer draws a block on a circuit diagram, he determines whether it should be a resistance, a capacitor, a diode, etc. Similarly when a programmer draws a flowchart, he has to select the appropriate symbol before drawing it. So the physical steps in the design are accompanied by a series of corresponding design decisions.

Designing the thousands of artefacts required to enhance human productivity involves millions of such decisions. However, experience shows that many of them are similar. That means the same kinds of design decisions are being made over and over again - which presents an opportunity for standardisation. Suppose formal "techniques" of making every recurrent design decision were published in professional journals. Then any designer can simply apply the published technique whenever he faces such a decision. Therefore standard techniques diminish the effort of design.

For example, "Kirchoff's Rules" [6] constitute a general technique for determining unknown currents (I), resistances (R) and electromotive forces (E) in an electrical circuit. The technique reduces circuit calculations to three easy steps:

1. For every function, set $\Sigma I = 0$
2. For every loop, set $\Sigma IR = E$
3. Solve the resulting equations.

Similarly Jackson's technique of structure charting [4] reduces program design to the following simple steps:

1. Hierarchically decompose the input and output data into sequences, selections and iterations.
2. Draw a procedure structure chart with sequences, selections and iterations that correspond to the superposition of the data hierarchies.

In the interests of efficiency, the knowledge base of design should therefore include *techniques*.

However techniques alone are not enough. Every artefact consists of constituents with characteristic properties of their own. Unless these properties are recognised in the design process, the artefact will not perform as intended. Therefore design decisions should be based on "Laws" - statements which describe the properties of design components.

That implies techniques should be derived from laws. For example, Kirchoff's technique mentioned earlier depends on two laws of electricity.

Step 1 ($\Sigma I = 0$) is a direct consequence of the law of Charge Conservation:
"The total charge in an isolated circuit never changes".
Step 2 ($\Sigma IR = E$) follows from Ohm's law:
"The current in a conductor is proportional to the voltage".

Similarly, Jackson's technique of structure charting derives from Böhm and Jacopini's law of flowcharting [1] plus three (unstated) laws of the computer. These computer laws can be expressed in terms of arbitrary procedures P and Q which process arbitrary data sets D and E:

1. The sequence P-Q processes the sequence D-E
2. The selection P or Q processes the selection of D or E
3. The iteration of P processes the iteration of D.

Law-based techniques such as the above implicitly ensure that recurrent decisions are consistent with underlying laws. But for non-recurrent decisions the designer also needs explicit statements of the properties of design components. For example, many electric circuits will malfunction unless the designer recognises the internal resistance of the energy source - Thevenin's Law. And some computer programs will fail unless the designer knows that division by zero in impossible.

Therefore, in addition to techniques, the knowledge base for design must include *laws*.

These two types of knowledge are different in character. Laws specify the properties of the design components; techniques help the designer combine the components into an artefact. Laws describe the *operands* of design; techniques prescribe the *operations* that can be performed on the operands.

Next, different types of laws and techniques will be identified by analysing the components of artefacts.

4. DESIGN COMPONENTS

In the beginning, Man's artefacts were uncomplicated. Their components were obtained form natural sources: for example, primitive smelters produced iron from surface ores and charcoal. Their designs were based on observations of a few basic patterns of Nature, such as carbon's ability to reduce oxides.

However, today's artefacts are much more complex. Natural components are first combined into functional components, and those are then assembled into useful artefacts. For example, at the first level of segmentation, an electric power drill consists of manufactured components: a motor, gears, capacitors etc. At the next level, those functional components consist of manufactured components, such as wire, steel plate etc. Finally at the next level those components in turn consist of natural materials such as iron and copper.

The natural components at the lower levels of the "systems hierarchy" [9] are characterised by *"Laws of Nature"*. Higher-order manufactured components have properties not found in nature, so they are described by a different type of law. For example, Ohm's law merely refers to the low-level copper wires in an electrical motor. The device as a whole is described by the formulæ of alternating current motors - *"Laws of the Artificial"*.

Therefore the knowledge base for design should contain two kinds of laws: natural and artificial.

These two types of law describe entities of quite different origin. Motors, capacitors and gears would not exist if Man had not specifically constructed them. Thus artificial entities originate with Man. On the other hand, natural entities such as copper and iron have existed well before Man. So their origin is non-human.

The dichotomy of origin affects the properties of natural and artificial entities. Natural entities are imbued with inherent properties which stem from their non-human origin. But whenever Man creates an artificial entity from natural entities, he gives it additional attributes which suit his own utilitarian purposes. So within broad limits imposed by the Laws of Nature, Man *contrives* the properties of artificial entities. Thus Laws of Nature describe inherent properties, but Laws of the Artificial specify contrived properties of an artefact's components.

A similar distinction also applies to techniques. When a design involves natural components, one employs "natural" techniques based on Laws of Nature. For example, Kirchoff's Rules are used to combine natural conductors into circuits. And these are derived from the Laws of Nature which describe the inherent properties of electric conductors.

On the other hand, when a design involves artificial components, one needs "artificial" techniques based on Laws of the Artificial. For example, Jackson's technique is used in combining artificial hardware functions to form information systems. And the technique depends on contrived properties of flowcharts and computers which are described by artificial laws.

Therefore the knowledge base for design should also include two types of techniques -

natural and artificial.

Again, the dichotomy of origin gives rise to differences between the two types. Natural techniques are consequent upon inherent properties of natural design components. Artificial techniques are consequent upon contrived properties. Man cannot influence inherent properties, whereas he can and does determine contrived properties. Consequently, artificial techniques depend on Man's utilitarian purposes, but natural techniques do not.

5. KNOWLEDGE NETWORK

The foregoing shows that research in general should produce four distinct types of knowledge:

- Laws of Nature (such as Ohm's Law), which describe inherent properties of natural entities;
- Natural Techniques (such as Kirchoff's Rules), which prescribe how to combine natural entities;
- Laws of the Artificial (such as the Böhm-Jacopini Law) which describe contrived properties of artificial entities;
- Artificial Techniques (such as Jackson's structure charting), which prescribe how to combine artificial entities.

These are interrelated (fig. 1). Natural techniques are based upon Laws of Nature. Artificial techniques are based upon Laws of the Artificial. And Laws of the Artificial are constrained by Laws of Nature.

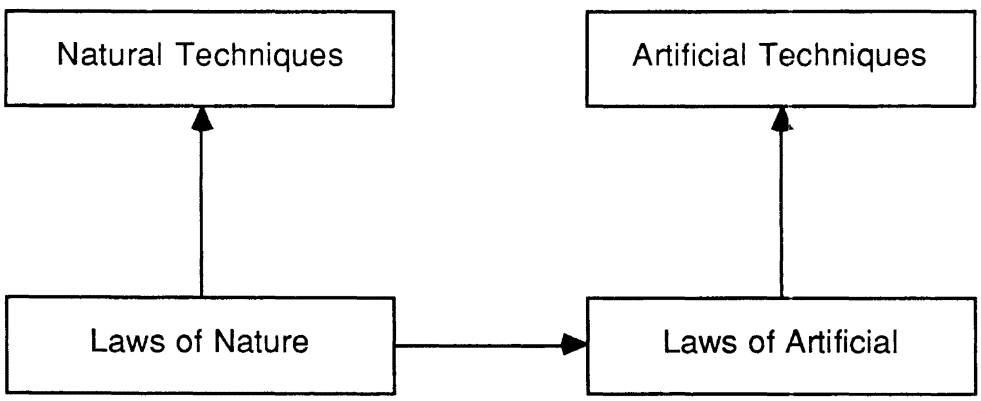

figure 1

Knowledge for Design

All four categories are represented in the subject Information Systems. The systems we study involve natural entities such as people, as well as artificial entities such as computers and information. The natural entities are subject to natural laws, and design decisions affecting them require natural techniques. The artificial entities are described by artificial laws, and are incorporated into systems with the aid of artificial techniques.

6. IMPLICATIONS

Researchers in the field of Information Systems can derive many implications from this hierarchy. For instance, the *classification scheme* reveals significant contrasts between our work and that of colleagues in other fields.

At present, much of our research is concerned with artificial systems components - computers and information. On the other hand, scientists are primarily concerned with natural

entities. Therefore we mainly produce *artificial* laws and techniques, while they establish *natural* laws and techniques.

These differences have two implications. Firstly, our artificial laws and techniques depend on Man's utilitarian purposes. But those purposes can and do change. So our research results are subject to rapid *obsolescence*, whereas our scientific colleagues' results are more permanent.

Secondly, in order to ensure a longer effective life for our artificial laws and techniques, we need to consider Man's *future* purposes. This suggests that some research should perhaps be based on forecasts, not simply on current technology and information needs.

Thirdly, research on the human aspects of information systems is likely to be discomforting. Humans are natural rather than artificial components. Therefore a researcher accustomed to the mechanistic artificial components of information systems may have to re-orientate himself drastically in humanistic research.

Further, the *logical relationships* between the four categories point out some significant deficiencies in current publications on information systems.

Firstly, techniques should be based on laws. This shows up serious weaknesses in many of our published techniques. Some textbooks omit explicit statements of the underlying artificial laws, e.g. Jackson's Principles of Program Design [4]. Others fail to show the logical derivation, e.g. Gane and Sarson's Structured Systems Analysis [2]. Both omissions cast an unfortunate shadow of doubt on otherwise magnificent work.

Secondly, the logical relationships affect our research methods. Popular textbooks of Research Methodology (e.g. 5) urge the researcher to adopt "empirical" methods. That is sound advice for natural laws, but not a sole requirement for artificial laws and techniques. Why verify an artificial law or a technique by exhaustive empirical testing when it can be derived logically from underlying laws? So empirical methods should be the exceptions rather than the norm in a subject which is largely concerned with the artificial.

7. IN CONCLUSION

Researchers of different backgrounds might find the above taxonomy illuminating. It may renew a scientist's hopes of finding laws in a field presently dominated by techniques. The engineer may look forward to establishing techniques that are solidly based on laws. And the business specialist may be influenced to join the search for laws and law-based techniques.

It is also recommended for consideration by authors and lecturers. In writing a textbook it may be advantageous to emphasise underlying laws before presenting the consequent techniques. Similarly in lecturing on a technique, one may find it helpful to discuss laws and establish the logical necessity of the technique before proceeding to its use.

Finally, the taxonomy can affect attitudes. Intellectuals tend to regard Laws of Nature with considerable more respect than Laws of Artificial [10]. Indeed the latter are not usually even admitted to the status of "Laws": for instance Ohm's *Law* on the one hand but motor *formulæ* on the other. However the realm of the artificial is becoming increasingly important. People live in artificial dwellings, wear artificial clothes, and work with artificial devices in artificial organisations. So in terms of human progress, artificial laws and techniques may actually become more significant than their natural counterparts. Then the subject Information Systems could become one of the most prestigious in academe.

REFERENCES

1. Böhm, C. and Jacopini, G. (1966). Flow Diagrams, Turing Machines and Languages with Only Two Formation Rules, *Comm. ACM*, **9**, 366 - 371.
2. Gane, C. and Sarson, T. (1979). *Structured Systems Analysis*, Prentice-Hall, Englewood Cliffs, New Jersey.
3. Hohenburg, P. (1968). *A Primer on the Economic History of Europe*, Random House, New York.
4. Jackson, M.A. (1975). *Principles of Program Design*, Academic Press, New York, 15 - 91
5. Kerlinger, F.N. (1973). *Foundations of Behavioral Research*, Holt, Rinehart and Winston, New York.
6. Kip, A. (1962). *Fundamentals of Electricity and Magnetism*, McGraw-Hill,

New York, 153 - 157.
7. Kuhn, T. (1970). *The Structure of Scientific Revolutions*, University of Chicago Press, Chicago, p7.
8. Mende, J. (1986). Research Directions in Information Systems, *Quæstiones Informaticæ*, 4, 1 - 4.
9. Short, R. (1985). Levels of Abstraction and Systems Hierarchies, *in Proceedings of the 15th annual UCSLA Conference.*
10. Simon, H.A. (1969). *The Sciences of the Artificial*, Massachusetts Institute of Technology Press, Cambridge, Mass.

BOOK REVIEW

Macintosh Graphics in Modula-2, Russell L. Schnapp, Prentice-Hall, Englewood Cliffs, New Jersey, 1986. ISBN 0–13–542309–0. 190 pages.

Reviewed by: Philip Machanick, *University of the Witwatersrand, Johannesburg 2001*

At first sight, this seems to be a book addressing a very narrow niche market—a specific class of application in a specific language on a specific computer. Despite this apparent limitation, it is valuable in several respects. Of particular interest is the way it—without any pretensions at doing so—suggests the viability of Modula-2 as a first teaching language, as well as the utility of features supposedly supporting programming-in-the-large in relatively small programs.

The former point is illustrated by the author's approach to defining useful modules, then following them with examples using them. An interesting approach to teaching beginners to program would be to present them with such a predefined module as a starting point for exploratory programming in the style of Logo. The way Modula-2's abstraction and information hiding features are developed through examples (the full text of most is presented—a disk with the Modula-2 source is available at a modest price) and exercises suggests students could be gently broken in to such an approach.

An illustration of the way the programming-in-the-large features of Modula-2 are put to good use is the way events (how asynchronous actions like mouse movements and keystrokes are passed to Macintosh programs) are hidden inside a module in one of the examples; unfortunately, this approach is not carried through to the following example.

An unusual (but positive) feature is that the author doesn't hesitate to point out problems with the language as they are encountered.

Aside from a few minor errors, the book is useful as extra documentation for the Macintosh's Toolbox and QuickDraw routines. The author's occasional references to *Inside Macintosh*[1] and the MacModula-2 manual are a bit of a joke—his book would not be filling such an obvious need if that documentation were comprehensible. From this point of view, it is a pity he did not explain some of the details a bit more thoroughly—such as his use of CODE in procedures accessing built-in routines. This is a relatively minor issue, since this technique only applies to the 128K versions of the modules (at least with the current compiler). Of more significance is the fact that later releases have major enhancements which would have been worth covering—such as access to Resources (crucial for good use of the user interface). Nonetheless, his mainly well-written and documented examples are a valuable extension to existing documentation. Minor quibbles include excessive use of INTEGER where a subrange would have been enough, and the occasional over-use of globals (though this a problem in most module-based languages—variables private to the module cannot be passed as parameters to publically-known procedures).

The book would have been of more value if it had explained design as well as presenting complete examples. But then it would have lost one of its chief virtues: brevity. Perhaps a second volume about design—drawing on the examples of this book—is called for.

REFERENCE

[1] *Inside Macintosh* (three volumes), Addison-Wesley, Reading, Massachusetts, 1985. ISBN 0–201–17737–4.