Le Travail humain, tome 60, n° 4/1997, 363-385

THEORIES AND METHODOLOGIES
THEORIES ET METHODOLOGIES

COGNITIVE ARCHITECTURE FOR MODELLING HUMAN ERROR IN COMPLEX DYNAMIC TASKS

by S. GRANT[1]

RÉSUMÉ

ARCHITECTURES COGNITIVES POUR LA MODÉLISATION DES ERREURS HUMAINES DANS LE CAS DE TÂCHES COMPLEXES EN SITUATION DYNAMIQUE.

La littérature traitant des interactions homme-machine et des activités dans les systèmes complexes et dynamiques à haut niveau de technologie, relevant de l'ingénierie cognitive, témoigne d'une substantielle discussion sur la notion d'erreur et de modélisation cognitive. Une raison à cela est que de bons modèles cognitifs pourraient être utiles à la conception de systèmes reconnaissant les erreurs, ou réduisant leur probabilité d'apparition. Ces modèles pourraient être utiles lors de la réalisation d'interfaces, de la conception de systèmes de sécurité, de l'estimation de risques ou de fiabilité, de cours sur les facteurs humains ou lors de l'analyse des causes d'accidents. Une moindre attention a été portée à l'architecture à la base de tels modèles, même si l'on cherche maintenant à faire que des architectures telles que ACT-R et SOAR puissent servir de support à la modélisation des tâches complexes. Chaque modèle cognitif se base sur une architecture, qui peut être cognitive. La nature cognitive de l'architecture limite la facilité de construction des modèles cognitifs, et, en particulier, de modélisation de l'erreur humaine.

Une approche pour évaluer l'utilisation probable et la valeur d'une architecture pour modéliser les erreurs est de comparer les types d'erreurs observées lors de l'exécution de tâches complexes réelles avec les mécanismes potentiellement fournis par l'architecture pour modéliser les erreurs. On présente ici, une première étape de la comparaison entre trois différentes approches: celle de J. R. Anderson dans ACT-R, de J. Reason dans la Machine Faillible (à la base du modèle COSIMO), et de M. J. Young dans l'Architecture Cognitive Holon (HCA). Des architectures telles que ACT-R ne sont pas spécifiquement conçues comme un cadre de modélisation du type d'erreur considéré dans cet article, mais pourraient être étendues afin de créer une architecture convenable. Des architectures telles que la Machine Faillible de Reason, semblable à l'architecture à la base de COSIMO, ont été conçues pour modéliser des erreurs mais la Machine Faillible nécessite de meilleures spécifications, allant au-delà de celles fournies par COSIMO. Des architectures telles HCA semblent pouvoir supporter des modèles riches bien que parcimonieux, cependant ceci n'a pas été validé. Dans tous les cas, on constate le besoin de modéliser des exemples réels d'erreurs lors de tâches dynamiques et complexes en utilisant les différentes architectures et de les tester en faisant des prédictions, par exemple, sur les conséquences d'une modification de l'interface, sur le comportement humain et ses erreurs.

Mots-clés: *Modèle cognitif; Architecture cognitive; Cognition; Tâches complexes; Erreur.*

1 (Then at: Connect Centre, Department of Computer Science, University of Liverpool, L69 3BX) Work carried out at JRC, ISIS, Ispra (VA), Italy. Current e-mail: asimong@gmail.com

I. INTRODUCTION: COMPLEX TASKS AND HUMAN ERROR

I.1. COMPLEX DYNAMIC TASKS

This paper is rooted in an established research approach to a particular widespread kind of complex task. From the 1960s onwards, researchers such as Jens Rasmussen (1980, 1983, 1986), Erik Hollnagel (1993), David D. Woods (1988), Woods, Johannesen, Cook and Sarter (1994), James Reason (1990), and many contributors to a number of collections (e.g., Goodstein, Andersen, & Olsen, 1988, Weir & Alty, 1991, Hoc, Cacciabue & Hollnagel, 1995) have been studying and reflecting on complex, dynamic, real-time supervision or control tasks involving complex joint human-machine systems, including automatic subsystems. These tasks are here referred to as 'complex dynamic tasks'. An appropriate name for the particular subject matter was coined by Hollnagel and Woods (1983): 'cognitive systems engineering'. The field is usually seen as linked closely with HCI, while having its own distinct character.

Examples of relevant complex systems include industrial and (particularly nuclear) power plant; modern aircraft and ships; traffic control and management systems; and high-technology systems found in medical practice. In each case, the information which supports the operator or practitioner in his or her task is handled at least partly electronically, by means of computers. Accidents involving such systems often involve the loss of human life, as well as extensive and costly damage to the environment or to the technology itself. Reason (1990) outlines six famous cases. This is sufficient to give great importance to the study of complex tasks and systems, for preventing such accidents, and for designing more robust, error-tolerant systems, often through attention to the human-machine interface.

Complexity of a task or of a system is difficult to define unambiguously. For example, there is no single measure of complexity which will completely predict differences in the ability of people to manage this kind of task. Woods (1988) lists a number of factors that contribute to the complexity and difficulty of tasks. Some factors are about the world: dynamism, many highly interconnected parts, uncertainty and risk; while other factors are about the agent (usually the human) and the representation of the world used by the agent. Whatever criteria one uses to judge the complexity of a system or task, in practice what one sees is that complex tasks have multiple goals, no clearly-defined optimal behaviour, and a variety of strategies used by different individuals (Grant, 1990).

The literature under consideration here is not to be confused with the distinct literature discussing theories of expertise. Holyoak (1991) mentions examples from chess, medical diagnosis, solving algebra word problems, mathematical expertise, learning from verbal instructions, and analogical thinking, referred to as 'complex'.

A related distinction is elaborated by Cacciabue and Hollnagel (1995) who use the term 'macro-cognition' to refer to the complete cognitive skills that are relevant to real complex dynamic tasks, as opposed to 'micro-cognition', referring to the cognitive skill components studied within the laboratory-based paradigm. Student modelling could be seen as between the two, but currently seems to share less with complex dynamic tasks, and is therefore not considered in this paper.

I.2. HUMAN ERROR

Accidents involving complex dynamic systems have increasingly been attributed to human error. Hollnagel (1993) reviews a number of past studies, that show that the proportion of system accidents attributed to 'human error' has increased from as low as 20% in some earlier studies in the 1960s and 70s, to nearer 80% more recently. Possible reasons for the increase include the improvement in reliability of engineered components, and increased recognition of the possibilities of human error.

Today, civil aviation accidents, for example (one is Monnier *et al*. (1993), others discussed by Woods *et al*. (1994)), are frequently seen as involving human error. How much the human is really to blame is another question. For example, in some cases it may have been that the human-machine interface of the flight deck was less than ideal, not giving the best information for supporting a correct sequence of actions by the pilot, or perhaps tying up attention or cognitive resources that could have been used to prevent the accident. The human error in this case could be seen as more in the design or management of the system than in its operation.

This doubt about blame gives one possible reason for hesitating to define human error exactly. Nevertheless, Reason offers the following as a definition.

> "Error will be taken as a generic term to encompass all those occasions in which a planned sequence of mental or physical activities fails to achieve its intended outcome, and when these failures cannot be attributed to the intervention of some chance agency." (Reason 1990, p.9)

This leaves open the questions of what was planned, by who, who intended the unachieved outcome, and who is attempting to attribute the error to what kind of chance agency on the basis of what kind of reasoning.

It is easy to be less than fully satisfied by these definitions, and Woods *et al.* (1994), in contrast, prefer not to offer a definition, but rather to discuss in great detail different aspects and factors of 'human error', which they put in quotation marks. The present paper follows Woods in refraining from attempting a clear, simple definition of human error. Instead, the approach to understanding error is through the attempt to model it.

I.3. WHY MODEL ERROR?

The fact of accidents, together with the proportion attributed to human error, give a strong incentive for the better understanding of human error. Perhaps a statistical approach can suggest ways of trying to avoid situations in which errors have been most frequently observed, but it needs a deeper model to be able to recognise errors at the time they occur, to support means of managing them, and to design new systems that reduce the chances of error occurring.

It would be a mistake to think of human error as a separate phenomenon from the rest of human performance. This indivisibility of error from skill or expertise is a common understanding of the research community. Woods *et al.* (1994) tell of Jens Rasmussen often quoting the philosopher Ernst Mach: "Knowledge and error flow from the same mental source; only success can tell one from the other" (Woods *et al*., 1994, p.20). Rasmussen (1983) says that useful human performance models must reflect the limits of human capabilities, so that 'errors' are modelled properly. He continues: "Successful performance does not validate a model, only tests of its limits and error properties can do this" (Rasmussen, 1983, p.264). So when we talk about models of error, we are really considering human performance models that are able to account for errors, rather than models of errors alone.

There are many possible practical uses for human performance process models able to explain the processes of error and their generic causes. Used as conceptual models, by other humans, they informally support interface design; safety systems design; risk and reliability assessment; human factors training; and analysis of accident causes. Used as user models, implemented as simulations run on computers embedded in the engineered system, they could be used in particular for operator support; diagnosis of errors; dynamic task allocation; and the operational management of cognitive resources more generally. Such simulations could also potentially help with design, training, assessment and analysis.

In contrast, models of cognition that neither explain nor predict human error are of limited usefulness for practical complex system design and related matters, because safety and reliability are critical to such systems in a way that they are not critical to, shall we say, the text editors studied by Card, Moran, and Newell (1983).

Having accepted the motives for modelling error, the next section briefly outlines the current position of cognitive models and architecture in the field of cognitive systems engineering. The main part of the paper then gives one approach to the evaluation of cognitive architectures for modelling error. This is intended to serve a number of purposes. Firstly, the cognitive architecture examples illustrate the possibility and potential importance of the evaluation method suggested. Secondly, exploring the method points out various challenges which will need to be addressed if this method is to be developed. These challenges are both to the architectures and to the evaluation method itself. Thirdly, even the limited analysis done in this paper reveals some issues concerning the suitability of the architectures for modelling error.

II. COGNITIVE MODELS AND ARCHITECTURES

M. J. Young (1993b) uses the term ‘human performance process’ (HPP) model to refer to engineering-style models “which emulate human behavior by simulating specific human information processing attributes and processes” (Young, 1993b, p.3). This highlights the difference between, on the one hand, the AI tradition, where human information processing models can be made just to test possibilities for the internal workings of cognition, and on the other hand, the cognitive systems engineering tradition, where ‘cognitive models’ could be made for the kind of practical purposes suggested above. Models intended for practical purposes need to give useful output which must be able to be tested in realistic scenarios. Thus, a cognitive systems engineering model must be a model of some particular skill or ability, and cannot be simply about abstracted cognitive processes.

There are two kinds of examples of the use of cognitive modelling in complex dynamic tasks. The first kind involves the general discussion of the characteristics of human cognition in these tasks, which may serve as objectives or requirements for more specific cognitive simulations, or may inform conceptual models used in design and related areas. Much of Rasmussen’s and Hollnagel’s work falls into this category. Such general work, while being of great potential importance, is difficult to test. One significant use in this way is in the analysis of incidents. Cacciabue, Pedrali, and Hollnagel (1993) refer to a very simple model of cognition (Hollnagel, 1993; Hollnagel & Cacciabue, 1991) comprising four stages of perception / observation, interpretation, planning / choice, and action / execution, along with a memory, illustrated by use in the analysis of an aircraft accident.

The second way is that of cognitive simulation modelling of specific tasks, reviewed by Cacciabue (1994). These simulations allow at least some kind of comparison with the performance of a human on a similar task, and this potentially allows the simulations to benefit from the feedback and develop greater accuracy and coverage. Amalberti and Deblon (1992) made a simulation of military aircraft pilot behaviour, while the COSIMO model (Cacciabue *et al*., 1992) has been used for modelling tasks and operator behaviour in nuclear power and civil aviation.

Any cognitive simulation or model could be regarded as having architecture, possibly at several levels. The top level architecture of a cognitive model is what remains when all the domain-specific knowledge is taken out, rather like an expert system shell. Thus a cognitive model built directly as a production system model would have the production system as its architecture. Lower-level architecture may involve the underlying computer architecture on which the top-level architecture is built. Referring to architecture as 'cognitive' makes sense if it specifies some general, rather than domain-specific, features or mechanisms which model some aspect of human cognitive processes.

For example, Amalberti and Deblon's (1992) simulation model has a cognitive architecture constructed specifically for the purpose. This is built on top of an 'actor' or agent-based, object-oriented computational architecture. The cognitive architecture underlying COSIMO includes part of Reason's (1990) fallible machine, which is discussed in detail below. It was implemented using a blackboard computational architecture. The architecture of cognitive models which are not simulations can be quite simple. For example, the simple model of cognition used by Cacciabue, Pedrali, and Hollnagel (1993) is basically an ordering of the four presumed stages of cognition.

There are clear advantages in principle to using existing cognitive architectures. Since they define the mechanisms which use the knowledge, and the structure of that knowledge, building a cognitive simulation becomes a matter of fitting the knowledge gained by task analysis or knowledge engineering into the predefined formats given by the architecture, and then letting the architecture do the relating together, running the model, and producing the simulation output to compare with human performance. Without a cognitive architecture, the available effort must be divided between implementing general cognitive mechanisms, and modelling using those mechanisms.

What of the recognised cognitive architectures such as Newell's (1990) Soar and Anderson's (1993) ACT-R (which is discussed in more detail shortly)? In the past, these architectures have tended to be used in the modelling of the 'micro-cognition' cognitive skill component tasks, rather than any full-scale complex dynamic skill. At the time of writing, Soar and ACT-R (e.g., Lee, Anderson, & Matessa, 1995) are intended to be used to model the Kanfer-Ackerman (1989) air traffic control task, which, though still a simulation game, is dynamic, and is a step towards a realistic complex dynamic task.

This brings up some serious and interesting questions. Several years ago, Rasmussen (1983) expressed doubts about whether production system models would be able to represent the error properties observed in humans in complex dynamic tasks. Can ACT-R, for example, provide adequate architecture for complex dynamic simulations that include error? Or will the architecture have to be extended to include complex error phenomena? On the other hand, are the architectures from existing complex dynamic task simulations useful for simulating other tasks? To approach answers to these questions we need to consider how to evaluate cognitive architectures for a particular purpose.

III. COGNITIVE ARCHITECTURES FOR MODELLING ERROR

When we consider the capabilities of an architecture for modelling human error, we are not talking about formal power, since formally most architectures have the same power. We are talking about how well matched the architectural mechanisms are to the modelling of error, comparing errors observed in real complex tasks with the potential mechanisms available in the architecture with which to model error-prone behaviour. This can be done by examining either particular examples of error, or some error categories, and examining how straightforward or complex the model of such errors would have to be in terms of the architecture. The other way round, one can look at the architecture, and see what observable errors are likely to result from the operation of any component or aspect of the architectural mechanisms or knowledge structures. These could then be compared with the errors observed in operation with humans.

Ideally, an architecture intended for modelling particular kinds of error would be tested by being used in the modelling of just those kinds of error. Choosing a representative selection of errors for this would itself be a major task. An easier option, satisfactory for an initial exploration, is to take classes of error already described by other authors. This is less than ideal, because any predefined classes of error are likely to have already imposed some kind of theoretical structure on the data. Reason's (1990) analysis assumes, as fundamental, differences between what he calls 'failure modes' in skill-based, rule-based and knowledge-based performance, following Rasmussen's (1983) distinction, which may or may not fit in with the causal error categories arising from explanation in terms of a cognitive architecture. Nevertheless, any reasonably broad classification of errors at least gives a range of errors which may be explained in terms of a cognitive architecture. For this reason, and since Reason is a very well-known author in the field of human error, his GEMS (generic error-modelling system) classification is here taken to stand for the different kinds of observable errors, with which are compared the errors that can be generated by the cognitive model.

This is not the place for explaining Reason's terminology, but to get a general idea of his GEMS classification, Table 1 reproduces the content of Reason's Table 3.3 (1990, p.69). Although it is not a theory-free classification, it will be used here as a rough indication of errors observed in practice, including those arising in complex dynamic tasks.

The comparison of these observed errors against errors theoretically predicted by a cognitive architecture can be done at varying levels and degrees of precision. A complete comparison of any one cognitive architecture would be a major study, requiring thorough, and practical, knowledge of the architecture. Here, a preliminary analysis is undertaken, in which some of Reason's error types are used selectively to illustrate particular points of interest in the architectures.

The different approaches compared in this way are: Anderson's ACT-R; Reason's fallible machine (underlying the COSIMO model); and Michael. J. Young's Holon Cognitive Architecture. ACT-R is chosen as arguably the most cognitive of the architectures originating from the AI tradition. The fallible machine is chosen because it is perhaps the only architecture designed expressly to model error. The Holon Cognitive Architecture is less well-known and developed, but is chosen because it is one of the few examples offering an approach different from the first two.

TABLE 1

Summarising the main headings for the failure modes at each of the three performance levels (Reason, 1990 p.69)

Résumé des principales catégories de modes de défaillance à chacun des trois nivaux d'activité (Reason, 1990 p.69)

Skill-based performance

Inattention	*Overattention*
Double-capture slips	Omissions
Omissions following interruptions	Repetitions
Reduced intentionality	Reversals
Perceptual confusions	
Interference errors	

Rule-based performance

Misapplication of good rules	*Application of bad rules*
First exceptions	Encoding deficiencies
Countersigns and nonsigns	Action deficiencies
Information overload	Wrong rules
Rule strength	Inelegant rules
General rules	Inadvisable rules
Redundancy	
Rigidity	

Knowledge-based performance

Selectivity
Workspace limitations
Out of sight out of mind
Confirmation bias
Overconfidence
Biased reviewing
Illusory correlation
Halo effects
Problems with causality
Problems with complexity
- Problems with delayed feed-back
- Insufficient consideration of processes in time
- Difficulties with exponential developments
- Thinking in causal series not causal nets
- Thematic vagabonding
- Encysting

III.1. ANDERSON'S ACT-R

ACT-R (Anderson, 1993), like Soar (Newell, 1990), is a production system architecture in which various cognitive skills can be modelled. At its simplest, a production system comprises a set of if-then or condition-action rules, along with a working memory. In execution, the system checks for matches between data (in working memory) and conditions, and when a match is found, executes the action part of the rule whose condition has been matched. The action parts of rules typically alter the contents of working memory. It is possible to write simple models of cognitive skills as production systems in which the rules at least in part correspond to rules which humans might plausibly be explicitly aware of when they attempt to perform the skills. Arithmetic is a favourite example, and Anderson gives possible production rules for addition (on paper) on page 5 of the book.

But ACT-R, like its predecessors, adds a great deal more structure and mechanism to the basic production system architecture. It deserves to be called a cognitive architecture particularly because of the attention which has been given to the plausibility of the mechanisms and of the model behaviour, in the light of knowledge from cognitive psychology.

III.1.A. Architecture and error in ACT-R

The most straightforward kind of error that can be modelled with a production system architecture is on involving wrong production rules. This could correspond to Reason's broad category 'application of bad rules'. So there is one simple possibility for explaining an observed error type in terms of an architectural structure. Whether it is a good explanation depends on the more detailed matching of possible models with observed phenomena. Anderson (1993, p.32-35) gives persuasive evidence that the production-rule unit is good for analysing performance in learning LISP.

The existence of an erroneous rule begs the question of how it came to be there, and how it is maintained. For Anderson, researching the learning of arithmetic for example, it may be reasonable to accept that these bad rules arise from some unspecified process. In contrast, in a real complex dynamic task the operators or practitioners tend to have learned the rules very well over a long period of time, and the retention of the bad rules as well as their source would have to be explained.

Can other observed errors be modelled as faulty production rules? This is a key question for illustrating the nature of this exercise in comparison of observed errors with mechanisms. Production system models may have productions that, rather than encoding domain knowledge content, instead represent some more subtle and general feature of the cognitive processes. Such productions may represent very different kinds of knowledge to the productions that straightforwardly code the domain rules (e.g., of addition). The mechanisms may differ for acquiring, maintaining and checking these different kinds of production. If this is the case, a useful analysis of the different observed errors would have to go beyond the idea that they were both faulty production rules, to include the architectural differences in the mechanisms affecting the errors in question. If one architectural mechanism is to explain more than one manifest type of error, there needs to be a explanation why this is so, and how one mechanism of producing error is manifest in two or more ways. The more clearly different types of error are shown as arising in distinct ways, the more useful the analysis will be.

In the case of ACT-R, there are indeed many more mechanisms, and more structure. Three essential theoretical commitments form the basis of the ACT-R architecture (Anderson, 1993, p.17).

1. There are two long-term repositories of knowledge: procedural memory and declarative memory. This differs from Soar, where there is only one production memory.
2. The 'chunk' is the basic unit of knowledge in declarative memory. Chunks can be organised hierarchically, such that one chunk can appear as part of another chunk.
3. The production is the basic unit of knowledge in procedural memory. Productions are essentially independent of each other, and learned separately.

This allows a possible distinction between wrong productions, already mentioned, and wrong units of declarative memory. Wrong declarative knowledge about the world is an easy problem to imagine. In Reason's scheme this seems to correspond to some knowledge-based errors, but it is not clear exactly which ones could be explained

simply by wrong declarative knowledge. In practice, a human performing a complex, dynamic task would be unlikely to possess long-term declarative knowledge that is usually wrong. For working memory, on the other hand, erroneous knowledge will presumably arise from unreliable procedures, which again leads back to questions about the architecture of those procedures.

An example of erroneous belief, in aircraft, involves the surprisingly common accident type of controlled flight into terrain, where the aircraft does not have a technical fault, but for whatever reason the pilots allow it to fly into the ground. In these cases there are undoubtedly false assumptions about the situation being a certain way when in fact it is not that way at all. The pilots may assume they are on the correct glide-path when in fact they are not. The next question is, how did that error arise? Answers could be attempted *ad hoc*, by listing the immediate precursors of the error; but that will not help in the quest for solutions to that particular type of accident. What would be most valuable would be an architectural analysis of the error type, leading to prediction of design strategies for combating the problem. It is an open question whether or not ACT-R affords the required architectural analysis in this case.

Much detail of the operation of ACT-R is focused on what is a central issue for all production systems: how to select a particular production rule from a range of ones whose conditions match to a similar extent. In ACT-R, this 'conflict resolution', as it is generally known, is determined by a list of factors: the goal that is currently active; the past history of use of various declarative chunks; the elements in the current context; the complexity of the rule; the past frequency of use of the production rule; the past history of success of the production rule, the amount of effort put into solving the problem so far; the similarity between the goal state and the state resulting from applying the production rule; and what other options for behaviour are available.

This mechanism is clearly of the right nature to produce errors where normally suitable rules are applied inappropriately. But it is difficult to know which part of the conflict resolution mechanism to put in correspondence with which error or type of error. In order to be able to identify certain mechanisms with certain types of error, one would have to perform simulations, systematically varying the different aspects determining production rule selection, and see whether any patterns emerge which correspond with observable error phenomena.

At places in ACT-R, what is called 'rational analysis' is invoked. This means that there are assumed to be mechanisms that work optimally in some sense, without specifying how each mechanism works. This may be justified, in that there may be levels of cognitive mechanism that work well, but are very difficult to model in a particular framework. However, to the extent that rational analysis is used, it eliminates the possibility of attributing any errors to that part of the architectural mechanisms, unless being rational contributes to an error.

While ACT-R can be used for modelling existing skill without any learning, skill acquisition is a significant focus for the theory, and this has implications for the relationship of the architecture with models of error. The theory is stated as follows.

1. The knowledge underlying a skill begins in an initial declarative form (an elaborated example), which must be interpreted (problem solving by analogy) to produce performance.
2. As a function of its interpretative execution, this skill becomes compiled into a production-rule form.
3. With practice, individual production rules acquire strength and become more attuned to the circumstances in which they apply.

4. Learning complex skills can be decomposed into the learning functions associated with individual production rules. (Anderson, 1993, p.143)

Including a learning mechanism provides a deeper level of potential explanation. Instead of being limited to accounting for errors in terms of the present state of knowledge and the way that interacts with the environment, a learning model enables the cause of error to be traced back to an explanation of how the knowledge involved in the error came to be there. A model including learning is stronger, predicting more, and equally is more open to refutation.

Interestingly, Anderson (1993) sees this theory of skill acquisition as central to his book. Recognising this helps to explain the strengths of the ACT-R architecture. Since one of the key questions guiding the development of ACT-R has been 'how is cognitive skill acquired', it makes sense that the phenomena best modelled are the studied examples of skill acquisition. It is to be expected that ACT-R is well-adapted to modelling in those domains.

This contrasts with cognitive systems engineering, where modelling learning is not a prime concern. One may speculate what difference it would make if an architecture were developed being less concerned with learning and more under the influence of the question, 'how do errors in complex dynamic tasks happen?' This suggests reversing the analysis, and looking at some of Reason's error types in terms of the ACT-R architecture.

III.1.B. Error types and ACT-R architecture

Which of Reason's error types pose difficulties for ACT-R? A few examples are now quickly outlined. Attention and perception do not feature explicitly in ACT-R, as the kinds of tasks modelled are not ones where attention and perception are critical. As a consequence, errors due to attention problems, given in Reason's GEMS under 'skill-based performance' are not explicitly modelled in ACT-R. From the rule-based category, 'informational overload', seems not to be explicitly modelled. At the knowledge-based level it appears that there are many relevant human failure modes that have no directly corresponding mechanism in ACT-R. Since, for example, there is no explicit limitation on working memory usage in the architecture, error types such as workspace limitations, confirmation bias, overconfidence, and problems with complexity would have to be modelled in terms of other aspects of the architecture, if at all.

These points are made briefly, because without the kind of knowledge that comes from deep involvement in the architecture, it is difficult to be sure what architectural mechanisms could perhaps be invoked to explain some of these error types. The explanation of phenomena that have not yet been explicitly accounted for is a creative exercise, and must be left open to those actively working with the architecture.

Table 2 summarises the major points of the discussion. It is not intended to discredit ACT-R as a potential architecture for modelling errors in complex dynamic tasks, but rather to draw attention to some of the areas where close correspondence remains to be achieved, or perhaps where ACT-R could be developed or supplemented to provide a architecture more closely adapted to error modelling.

TABLE 2

Some mechanisms in ACT-R compared with some error categories

Comparaison entre quelques mécanismes de ACT-R et des types d'erreurs

ACT-R mechanism or structure	*Error category*
Procedural production rules: missing or 'buggy'	Errors of omission, inability to complete task
Declarative chunks (erroneous)	Widespread errors
Conflict resolution	Sub-optimal rule selection
Skill acquisition mechanisms: analogy, production creation, production tuning	*Correspondence unclear*
Could be modelled by partial matching	Some errors of commission
Unclear	Several of Reason's skill-based errors
Unclear	Some of his knowledge-based errors

III.2. REASON'S FALLIBLE MACHINE AND COSIMO

Reason's Fallible Machine contrasts with GEMS, discussed above. GEMS is a classification of error types, based on a broad conceptual model of the logical stages of erroneous processes. Only a minimal internal cognitive structure and mechanism are used. In contrast, his Fallible Machine design, appearing in the same book (Reason, 1990) is in effect a cognitive architecture.

Reason's intention in designing his Fallible Machine is very appropriate to the present study. His question is "*What kind of information handling device could operate correctly for most of the time, but also produce the occasional wrong responses characteristic of human behaviour*?" (Reason, 1990, p.125). In his book, he outlines a conceptual design for this Fallible Machine, for which the structural components are reproduced in Figure 1, reproduced from his Figure 5.1.

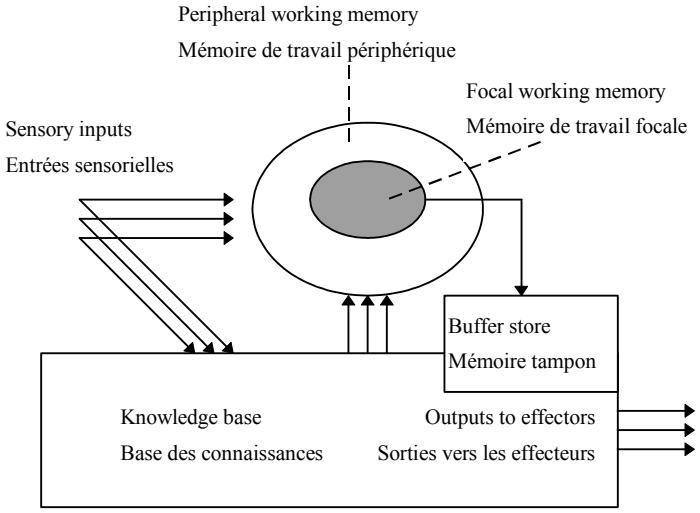

Fig 1: The principal structural components of the fallible machine.
Les composantes structurales essentielles de la machine faillible.Reason (1990) Figure 5.1, p.126

Peripheral working memory (PWM) receives inputs from senses and from the knowledge base, and selection takes place to determine what is to be processed by the powerful and restricted operation of focal working memory (FWM). The knowledge base contains procedural and declarative knowledge units, which are neither clearly the same as, nor clearly different from, those of ACT-R. Each knowledge unit has a modifiable level of activation. When this level exceeds a threshold, the knowledge unit outputs some product either to the effectors, or to PWM.

There are three mechanisms for bringing a stored knowledge unit into FWM. There is a kind of search mechanism modelling direct human inference, and then there are two mechanisms which are central to the error-prone nature of the architecture. These are similarity matching and frequency gambling. These come from Reason's observation that "*When cognitive operations are underspecified, they tend to default to contextually appropriate, high-frequency responses*." (Reason, 1990, p.97).

Similarity matching is similar to the basic production rule condition matching process in production systems. Partial matching is explicitly allowed, as in the example given by Reason: when a general knowledge question is put to someone, the cues in the question activate related portions of long-term memory. If an exact answer is not known, the answer guessed is likely to come from the set of items activated by the cues — that is, those partially matched. This gives the potential to model errors similar to those noted in the discussion of ACT-R.

In the Fallible Machine, frequency gambling may follow similarity matching. Frequency gambling is the selection between partially matched knowledge units on the basis of frequency of encounter. Compared with ACT-R's manifold factors influencing conflict resolution, this is relatively simple. It may be more clear, but if Anderson has good reason to include all the different factors, maybe the Fallible Machine loses in fidelity compared with ACT-R. Without a more precise specification of the Fallible Machine architecture, it is difficult to do more than guess differences in the modelling of rule-based errors between the two architectures.

Skill-based attentional errors could in principle be modelled in terms of the selection of PWM contents for transfer to FWM. In this case, the lack of detailed specification given in the exposition makes it difficult to judge whether the architecture can give a psychologically plausible and computationally implementable explanation of these skill-based error types. In that way, the Fallible Machine is in much the same position as ACT-R.

Similar problems occur in considering knowledge-based error types. It is relatively easy to propose on paper mechanisms which plausibly could serve as the basis for modelling these error types, but much more difficult to specify in detail the computational mechanisms responsible. As Reason says, "Inevitably, this design for a fallible machine dodges many crucial issues and skates over others. ... What it tries to do, however, is to convey a picture of an information-handling 'machine' that ... is in essence driven by a small number of simple computational principles." (Reason, 1990, p.137).

Computational implementation of Reason's theory has been done in two ways. One implementation is by Marsden (1993), whose work is briefly described by Reason (1990). The main application of this which is described by Reason is in general knowledge question answering. The implemented model gives error results that match well with experimental ones on this task. But the potential of Marsden's implementation in simulating complex dynamic task errors remains currently untested.

The other implementation based on Reason's work is the COSIMO (for Cognitive Simulation Model) system (Cacciabue *et al*., 1992), which implemented much of his architecture including similarity matching and frequency gambling mechanisms, within the pragmatic framework of a general blackboard computational architecture. The knowledge base contains rule-based frames and knowledge-based frames. A rule-based frame relates a particular type of accident with a frequency of encounter, a set of attributes (or symptoms) typical of the accident, and a set of appropriate actions. Each symptom has a physical salience, according to how easily the indication of it is perceived, and cognitive salience, according to its significance. The knowledge-based frames, in contrast, contain rules of thumb, together with general principles from engineering and (in the nuclear power plant case) physics.

COSIMO has a number of cognitive functions which manage the knowledge effectively while also being the mechanisms through which error is manifest. The filtering function selects data from the environment based on physical and cognitive salience; this is governed by a filter threshold, and the relative weight of physical and cognitive salience can be adjusted. There is then an interpretation threshold governing conversion to meaningful form. The diagnosis function is achieved by similarity matching and frequency gambling, producing a support score for each hypothesis. A hypothesis evaluation function then filters these hypotheses according to an evaluation threshold that can be dynamically modified to reflect the current cognitive state of the operator. The execution function then puts the rule-based frame associated with the selected hypothesis into working memory, and the actions associated with the frame are executed. Finer details of the implementation are given in the cited paper, but will not be discussed here.

For the purposes of the present paper, the real strength of the COSIMO analysis lies in the treatment of simulated errors. Three kinds of error are discussed, which Cacciabue *et al*. (1992) call 'cognitive collapse', 'unadapted change', and 'cognitive lock-up'. These error types are abstracted from observations of operators performing real tasks.

Their category of cognitive collapse is when the operator loses the capacity to reason clearly, perhaps due to acute stress. COSIMO models this by expanding the filter threshold, decreasing cognitive salience and increasing physical salience, increasing the interpretation threshold, increasing the importance of frequency gambling compared to similarity matching, and increasing the evaluation threshold. This seems to correspond fairly with Reason's 'informational overload'.

What is called 'unadapted change' is where the operator responds to a new event without taking past effects into account fully. This is simulated by reducing the evaluation threshold, and again frequency gambling is increased in weight relative to similarity matching. This could be compared to Reason's category of 'thematic vagabonding', though Reason gives it as a knowledge-based failure mode, rather than the rule-based problem which is suggested by COSIMO.

Cognitive lock-up is where the operator continues to act in terms of a hypothesis that should have been replaced. The disconfirming evidence is not effective. In this case, again the strength of frequency gambling is raised, but this time the evaluation threshold is also raised. This can be compared with Reason's 'confirmation bias' and has some points in common with 'encysting'.

COSIMO's explanation of errors shows well how the cognitive architecture can be used to explain these errors using mechanisms that are also responsible for effective performance. There is no attempt, however, to model a complete range of errors, nor to

map all COSIMO's potentially error-prone mechanisms onto observed error types. The authors also recognise limitations: they state that COSIMO does not model limitations in inferential, temporal and analogical reasoning.

The comparison with Reason's failure modes in these three cases highlights the difficulty of using predefined categories of error in doing this comparison. One area for development would be to assemble a set of real human errors (whether taken from a real complex dynamic system or a simulation) described in detail without any theoretical preconceptions.

The above discussion shows that COSIMO can model some errors in terms of alterations of the parameters built into its cognitive architecture. There is much potential for development of COSIMO, or a similar architecture. One improvement could be a clearer distinction between the theory and the implementation detail, such as is recommended by Cooper, Farringdon, Fox, and Shallice (1995). A principled account could be given of what would cause the model parameters to vary to produce errors. Hollnagel (1993) approaches a similar issue with his discussion of 'control modes'. This has not been implemented, though some work towards this end has been undertaken.

III.3. YOUNG'S HOLON COGNITIVE ARCHITECTURE

ACT-R and the Fallible Machine have several things in common, and thus it is worth also exploring cognitive architectures for modelling error that have a different basis. An example of this is M. J. Young's (1993a, 1995) work, which springs from a similar motivation to that of the cognitive systems engineering community. The aim is to support operability analysis of designs for complex dynamic systems, investigating optimum crew arrangements and task allocation between humans and machines. As it was still under development, the brief discussion here is intended simply to outline the concepts, and to suggest possibilities for future investigation.

Young's Holon Cognitive Architecture (HCA) is not based on production rules. The concept of a holon is taken from Koestler. It is an autonomous unit comprising rules, execution strategies, and its own internal representation of the relevant parts of the environment. Holons communicate by message passing, and are arranged in hierarchies, called holarchies. One motivating concept is to devise an architecture which is not dependent on a sophisticated central executive mechanism, but instead where control is distributed, consistent with the possibility of parallel execution and increased plausibility in terms of human neural architecture.

One of the more interesting ideas in the HCA is that knowledge is distributed through the holons, rather than being in the form of complete units of declarative knowledge. Young calls the memory construct a 'thread'. A thread is a pattern of connection between different holons. Young (1995) gives the example of a simple 'dog' thread. The attributes that together define a dog are located in separate holons: for colour; form; habitat; sound made; autobiographical information; and so on. Recognition of a dog would involve activation of a number of these holons, as an input holarchy, while an operator performing some action would involve the activation of an output holarchy.

Young (1995) briefly explains how Reason's three basic error types can be modelled in terms of the HCA. Skill-based slips occur when spreading association triggers a routine that is inappropriate, but when the evaluation of whether the action has been carried out correctly is not triggered.

Rules are implemented with their conditions as threads. Rule-based mistakes can occur when a thread is activated which is similar to the actual condition, but has a lower

activation threshold. Alternatively, a correct condition thread triggers a sensorimotor routine with a lower threshold than the appropriate one.

Knowledge-based mistakes occur when the operator formulates an inappropriate plan. This can either be due to perceptual bias, and the processing of the wrong perceptual cues, or due to cognitive bias invoking a well-learned plan instead of a more novel approach.

Without examples of implementation, as have been given in the cases of ACT-R and COSIMO, it is difficult to assess the real strengths of the HCA in the modelling of errors. Young recognises that the architecture is underspecified, and collaborators meanwhile have produced a modelling framework intended to allow the construction of HCA-like cognitive models (Deutsch, Cramer, & Feehrer, 1995). If substantial cognitive models are implemented in terms of the HCA, it will become clearer the extent to which it really offers significant differences in either capability or simplicity.

IV. DISCUSSION OF THESE DIFFERENT ARCHITECTURES

The review above has been made from the position of considering what kind of cognitive architecture to use to model tasks including errors. The goal of answering precisely which architecture is currently suitable for which models is out of reach, but for those interested in the aim of using existing cognitive architecture for modelling, some points are made here concerning features of cognitive architectures that pose problems or opportunities.

It is clear that architectures that have not been implemented tend to be too underspecified even to be sure of what kinds of error they would be suitable to model. They are not much help to the cognitive engineer interested in using simulation models of error, as the engineer would have to do much work filling in the specification of the architecture, without even being sure that it would be effective when the work had been done. Implementation and testing are therefore important, both practically, and also because implementation reveals problems which can lead to improvement of the architecture.

The analysis of COSIMO showed that even when an architecture is implemented explicitly to include errors in cognitive models, it can still be a major effort to relate particular categories of error to the architectural mechanisms. This is especially the case in architectures which have much internal complexity, and so a good objective is to devise architecture that covers the same or a greater range of error phenomena in a more parsimonious framework, with, particularly, a clearer distinction between the cognitive architecture and the computational implementation. However, simple architectures are of no practical use if they do not clearly allow modelling of the types of error of practical importance: they must have no less that the necessary complexity (similarly to requisite variety, as in Conant and Ashby (1970), and others).

Error is not just one thing, and the kind of error that needs to be modelled creates different demands on the architecture. ACT-R, for example, does not set out to support models of errors in complex dynamic tasks, and it may be expected that substantial work will have to be done to adapt ACT-R for supporting such error models. Cognitive models cover different phenomena, depending on their intended use, and it is no more than common sense that an architecture needs to be designed with the use of the supported models in mind. If a cognitive model is to include learning, in particular, the underlying architecture must, like ACT-R, include learning mechanisms. It currently seems unlikely that predictive simulations of learning will be practically used for

complex dynamic tasks in the near future, though such an architecture would be very impressive and potentially highly useful, since it would enable models of operator training and learning from experience, and could thus be used in initial and in-service training design.

Evaluation of the likely potential of an architecture, as approached here, is not the same as testing it in practice. Since an architecture is not directly testable, testing must involve creating models on the basis of the architecture, and comparing the performance of the model with the performance of humans on similar tasks. The most stringent testing involves using such a model to make predictions about the effects of, for example, changing the interface to a complex task, and using this capability to design or redesign a complex dynamic system in a way that fewer errors occur. This is a major future goal for the practical use of cognitive architecture in cognitive systems engineering.

V. DEVELOPING ARCHITECTURE FOR MODELLING ERROR

There are two main development directions apparent from a wide consideration of cognitive architecture. One, as mentioned in the previous section, is to increase the amount of specification detail of the architecture, implement it, and make it usable to create cognitive models in whatever domain it is intended for: this could well be seen as including the provision of a modelling framework, offering the functionality of the architecture to cognitive modellers. From the examples discussed in the present paper, deeper specification may be appropriate for architectures like the Fallible Machine or the HCA. For the COSIMO implementation, the experience gained could be fed back and used to help create a better more generally usable one. There are also other unimplemented cognitive architectures that may be worth implementing. For complex dynamic tasks, a good example is the kind of architecture assumed in the work of Bainbridge (1974), and in particular her cognitive processing element concept (Bainbridge, 1993).

The other development direction is to extend the architecture to cover an increasing range of phenomena observed in human behaviour and error. This could be applied, in various ways, to all the architectures discussed. This does not only mean extending an architecture's capability: it may mean restricting its capabilities so that its limits match better the limits of human cognition. Working memory and attention limits are obvious targets. Other architectures to consider for development in this direction also include the Knowledge Blocks approach of Boy (1991) developed by Mathé (1990).

At this point it is helpful to summarise the options, in terms of architecture, for creating cognitive models in whatever field is of interest — here this is complex dynamic tasks. There appear to be three options for the development of cognitive models from architectures.

Firstly, one could use a relatively simple architecture, elaborating purpose-built structures and mechanisms for the particular cognitive model required. So, for example, one could build a model of expert decision-making as an expert system or a multi-criteria decision support system. Models of routinised skills could be built from a procedural programming basis, or using GOMS (Card *et al*., 1983). However, two problems seem likely in these cases: difficulty in modelling more complex phenomena such as correct performance and error in complex dynamic worlds; and lack of generality, and transfer to other tasks, of any phenomena modelled *ad hoc*.

Secondly, one may work to develop an existing cognitive architecture as the modelling need is seen to arise. A problem here is that the need is subjective. There is always the alternative of using existing mechanisms given by the architecture. The motivation for extension of an existing architecture will be that the architecture is not easy enough to use in some way for the modelling task in hand. This is perhaps the best feasible option at present, and will presumably be the path for Soar and ACT-R confronted with modelling complex dynamic tasks. Work at the European Commission's Joint Research Centre developing COSIMO in this way has been motivated by the needs of modelling aviation tasks, among others (Amat, 1995).

Thirdly, one may start with a more sophisticated architecture, which has been developed explicitly with the intention of supporting cognitive models of the kind required. Greater potential in a cognitive architecture could come through revising some of the basic assumptions of previous architectures. Both the HCA, discussed above, and the exploratory SimulACRUM framework (Grant, 1996), do without the intricate central executive mechanisms appearing in other architectures, preferring what could be called contextual modularity over functional modularity (Grant, 1994).

One important factor contributing to simplicity, clarity and ease of use of a cognitive architecture is the separation of its theoretical content from its implementation detail. Often, implementation of an architecture leads to over-specification, in the sense that more mechanism is implemented than is claimed to have theoretical validity (Cooper *et al*., 1995). Users of the architecture should be protected from this excessive detail, and allowed to focus on the cognitive aspects of the model. In COSIMO, for instance, there are simulation parameters that are not discussed under the section for theoretical foundations. A promising development approach for COSIMO or an architecture in a similar position would be to target the clarification of this separation, making explicit just what is the extent of the domain-independent theory. Any such careful examination is likely to lead to general improvement of an architecture, and resulting clarity and simplicity would also help towards creating a modelling framework.

To assist cognitive modelling effectively, all architectures need to be worked through on examples of cognitive models of the general type that is needed. However, focusing on one example, such as do COSIMO and Amalberti and Deblon (1992), may not give the required generalisability for the architecture to be easily usable for other models. More specifically, for modelling errors, it would be particularly useful to have a generally available set of relevant human errors described in detail, without theoretical presuppositions. These could guide both the development of the architectures themselves at a theoretical level, and the development of models from those architectures, to test out their usability in practical modelling, and to test the quality of derived models.

VI. CONCLUSIONS

Cognitive modelling and architecture may be used at a conceptual level for reasoning and speculating about the cognitive mechanisms responsible for human errors. But the main thrust of this paper is concerned with the use of architecture to support simulation modelling of errors in complex tasks, and for this, an architecture needs considerable sophistication. An approach to preliminary evaluation of cognitive architecture for this purpose has been introduced, and this approach can be followed to varying depths and levels of detail. Problems and opportunities have been identified which could lead towards methods of fuller evaluation of architectures' potential in this respect.

Architectures such as ACT-R do not directly attempt to model the kinds of error focused on in this paper, but perhaps could provide such models with development of the architecture. Architectures such as Reason's Fallible Machine, similar to the architecture underlying COSIMO, are designed for modelling errors, but the Fallible Machine's need of full specification is not ideally complemented by COSIMO. Novel, different architectures such as the Holon Cognitive Architecture may yet offer rich, parsimonious models, but this remains unproven.

Cognitive modellers interested in human error and safety need to maintain a critical awareness of the cognitive architecture that they propose to use. Those who wish to benefit from cognitive models can contribute to their development by providing for study real examples of error in complex dynamic tasks. Cognitive architectures should be devised taking real error examples fully into account, and tested with reference to that kind of example, ensuring that the mechanisms of the cognitive architecture adequately correspond to the kinds of error naturally observed.

Modellers will then be able to look forward to the easier construction of cognitive simulations in the domains of their choice, and this can be expected to contribute finally to better overall safety and reliability of complex dynamic systems which involve complex tasks performed by humans.

Acknowledgements – Helpful comments on earlier versions, and suggestions for improvement, have been received from Anne-Laure Amat, Gordon Baxter, John Dowell, and various referees.

REFERENCES

Amat, A.-L. (1995). Goals and decision making aspects in pilot behaviour simulation. In L. Norros (Ed.), *Proceedings of 5th European conference on cognitive science approaches to process control*. Espoo, Finland: VTT.

Amalberti, R., & Deblon, F. (1992). Cognitive modelling of fighter aircraft process control: a step toward an intelligent on-board assistance system. *International Journal of Man-Machine Studies, 36,* 639-671.

Anderson, J. R. (1993). *Rules of the Mind*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Bainbridge, L. (1974). Analysis of verbal protocols from a process control task. In E. Edwards & F. P. Lees (Eds.), *The Human Operator in Process Control* (pp. 146-158). London: Taylor & Francis.

Bainbridge, L. (1993). *Building up behavioural complexity from a cognitive processing element*. Unpublished manuscript, Department of Psychology, University College, London.

Boy, G. A. (1991). *Intelligent Assistant Systems*. London: Academic Press.

Cacciabue, P. C. (1994). *Academic and practical needs for operator modelling and simulation*. Paper presented at the *2nd Workshop on SuperSimulators for Nuclear Power Plants*. Tokyo, Nov.

Cacciabue, P. C., Decortis, F., Drozdowicz, B., Masson, M., & Nordvik, J.-P. (1992). COSIMO: a cognitive simulation model of human decision making and behavior in accident management of complex plants. *IEEE Transactions on Systems, Man, and Cybernetics, 22*, 1058-1074.

Cacciabue, P. C., & Hollnagel, E. (1995). Simulation of cognition: applications. In J.-M. Hoc, P. C. Cacciabue, & E. Hollnagel (Eds.), *Expertise and Technology: Cognition and Human-Computer Cooperation* (pp. 55-73). Hillsdale, NJ: Lawrence Erlbaum Associates.

Cacciabue, P. C., Pedrali, M., & Hollnagel, E. (1993). *Taxonomy and models for human factors analysis of interactive systems: an application to flight safety*. Paper presented at the *2nd ICAO Flight Safety and Human Factors Symposium*. Washington, DC, April.

Card, S. K., Moran, T. P., & Newell, A. (1983). *The Psychology of Human-Computer Interaction*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Conant, R. C., & Ashby, W. R. (1970). Every good regulator of a system must be a model of that system. *International Journal of System Science, 1,* 89-97.

Cooper, R., Farringdon, J., Fox, J., & Shallice, T. (1995). *A systematic methodology for cognitive modelling* (Report UCL-PSY-ADREM-TR14). London: University College, Department of Psychology.

Deutsch, S. E., Cramer, N. L., & Feehrer, C. E. (1995). *Research, Development, Training and Evaluation (RDT&E) Support: Operability Model Architecture* (Report AL/HR-TP-1995-0018). Wright-Patterson AFB, OH: Armstrong Laboratory.

Goodstein, L. P., Andersen, H. B., & Olsen, S. E. (Eds.). (1988). *Tasks, Errors and Mental Models*. London: Taylor & Francis.

Grant, A. S. (1990). *Modelling cognitive aspects of complex control tasks*. PhD thesis, University of Strathclyde, Scotland.

Grant, S. (1994). *Modeling complex cognition: contextual modularity and transitions*. Paper presented at the *4th International Conference on User Modeling*, Hyannis, MA, August.

Grant, S. (1996). *SimulACRUM: A cognitive architecture for modelling complex task performance and error*. Paper presented at the *8th European Conference on Cognitive Ergonomics*. Granada, Spain, Sept.

Hoc, J.-M., Cacciabue, P. C., & Hollnagel, E. (Eds.). (1995). *Expertise and technology: cognition & human-computer cooperation*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Hollnagel, E. (1993). *Human Reliability Analysis, Context and Control*. London: Academic Press.

Hollnagel, E., & Cacciabue, P. C. (1991). *Cognitive modelling in system simulation*. Paper presented at the *3rd European Conference on Cognitive Science Approaches to Process Control*. Cardiff, Wales, Sept.

Hollnagel, E., & Woods, D. D. (1983). Cognitive systems engineering: new wine in new bottles. *International Journal of Man-Machine Studies, 18,* 583-600.

Holyoak, K. J. (1991). Symbolic connectionism: Toward third-generation theories of expertise. In A. Ericsson & J. Smith (Eds.), *Toward a General Theory of Expertise: Prospects and Limits* (pp. 301-355). Cambridge, England: Cambridge University Press.

Kanfer, R., & Ackerman, P. L. (1989). Motivation and cognitive abilities: an integrative/aptitude-treatment interaction approach to skill acquisition. *Journal of Applied Psychology, 74,* 657-690.

Lee, F. J., Anderson, J. R., & Matessa, M. P. (1995). Components of dynamic skill acquisition. In J. D. Moore and J. F. Lehman (Eds.), *Proceedings of the Seventeenth Annual Conference of the Cognitive Science Society* (pp. 506-511). Hillsdale, NJ: Lawrence Erlbaum Associates.

Marsden, P. (1993). *Fallible Adaptive Machines: A Computational Model of Error Tendencies in Human Cognition*. PhD thesis, University of Manchester, England.

Mathé, N. (1990). *Assistence Intelligente au Contrôle de Processus: Application à la Télémanipulation Spatiale*. PhD thesis, ENSAE, Toulouse, France.

Monnier, A. et al. (1993). *Rapport relatif à l'accident survenu le 20 janvier 1992 près du Mont Sainte-Odile (Bas-Rhin) à l'AIRBUS A320 immatriculé F-GGED exploité par la Compagnie Air Inter*. Ministère de l'Equipement, des Transports et du Tourisme, France.

Newell, A. (1990). *Unified Theories of Cognition.* Cambridge, MA: Harvard University Press.

Rasmussen, J. (1980). The human as a systems component. In H. T. Smith & T. R. G. Green (Eds.), *Human Interaction with Computers* (pp. 67-96). London: Academic Press.

Rasmussen, J. (1983). Skills, rules and knowledge; signals signs and symbols, and other distinctions in human performance models. *IEEE Transactions on Systems, Man, and Cybernetics,13,* 257-266.

Rasmussen, J. (1986). *Information Processing and Human-Machine Interaction*. Amsterdam: North-Holland.

Reason, J. (1990). *Human Error*. Cambridge, England: Cambridge University Press.

Weir, G. R. S., & Alty, J. L. (Eds.). (1991). *Human-Computer Interaction and Complex Systems*. London: Academic Press.

Woods, D. D. (1988). Coping with complexity: the psychology of human behaviour in complex systems. In L. P. Goodstein, H. B. Andersen, & S. E. Olsen (Eds.), *Tasks, Errors and Mental Models* (pp. 128-148). London: Taylor & Francis.

Woods, D. D., Johannesen, L. J., Cook, R. I., & Sarter, N. B. (1994). *Behind Human Error: Cognitive Systems, Computers and Hindsight*. Wright-Patterson AFB, OH: CSERIAC.

Young, M. J. (1993a). *Human performance models as semi-autonomous agents*. Paper presented at the *4th Annual Conference on AI, Simulation and Planning in High Autonomy Systems*. Tucson, AZ, Sept.

Young, M. J. (1993b). *Successively approximating human performance* (Report AL/HR-TP-1993-0026). Wright-Patterson AFB, OH: Armstrong Laboratory.

Young, M. J. (1995). *Human error and the Holon Cognitive Architecture*. Paper presented at the *6th Annual Symposium on Analysis, Design, & Evaluation of Man-Machine Systems*. Cambridge, MA, June.

SUMMARY

Modelling human performance in complex tasks is progressing from lower-level architectures to now using cognitive architectures like ACT-R and Soar. The important differences between architectures include their capabilities for modelling human error. One can evaluate an architecture for modelling error by comparing observed types of error in real complex tasks with the potential mechanisms available in the architecture with which to model error. Anderson's ACT-R; Reason's fallible machine (underlying the COSIMO model); and Michael J. Young's Holon Cognitive Architecture (HCA) are here compared. Conclusions are in terms of the outstanding challenges for development of architecture towards being useful for models of human error in commercially important tasks, and towards greater ease for this evaluation.

Keywords: *Cognitive model; Cognitive architecture; Cognition; Complex tasks; Error*.

Paper received: July 1996
Paper accepted in modified form: December 1996