The Role of Opportunistic Behaviour in Specification Comprehension

FRANCISCO SIMPLICIO FILHO[1]

Embrapa, Brasília D.F.
Imperial College of Science, Technology and Medicine
Department of Computing
180 Queen's Gate, London SW7 2BZ UK

fcs@doc.ic.ac.uk

Resumo. O comportamento oportunistico (ad hoc), caracterizado por desvios de atenção entre diversas soluções parciais, tem sido reconhecido como um comportamento legítmo e esperado durante a atividade de compreensão de especificações de software. Em contraste com o comportamento orientado por objetivos (e.g. o comportamento top down), o comportamento opportunistico procede sem o suporte de um método ou um plano para solução de problemas. Não obstante, cerca de metade da atividade de compreensão tem sido experimentalmente caracterizada como opportunistica. Portanto, não se pode desprezar este comportamento quando do projeto de métodos e ferramentas para suporte as atividades de construção e compreensão de especificações. Este artigo apresenta um modelo engenharil do comportamento cognitivo durante a compreensão de especificações. O modelo enfoca os mecanismos de controle que orientam os comportamentos oportunistico e orientado por objetivos. O modelo é baseado em uma interpretação de modelos estabelecidos em psicologia, e descrito em termos do formalismo visual stategraph. Este formalismo é orientado a especificação de sistemas distribuidos e reativos. Em conclusão, o modelo sugere que o comportamento oportunistico não exclui totalmente o comportamento orientado por objetivos. Um certo numero de atividades cognitivas podem operar em paralelo, e isto torna o sistema cognitivo capaz de manter o curso de ação de acordo com planos ao mesmo tempo em que executa outras atividades. O modelo especifica os mecanismos que limitam e operam esta concorrencia

Abstract. Opportunistic behaviour, which is characterized by shifts between partial solutions and deviations from planned actions, has been identified as legitimate and expected behaviour during software specification understanding. In contrast to goal-oriented behaviour, opportunistic behaviour proceeds without support of a method or general problem solving plan. Nevertheless, about half of the understanding behaviour has been characterized as opportunistic. So, it cannot be ignored in designing methods and tools for supporting specification building and understanding. This paper presents an engineering model of cognitive behaviour in specification understanding. It focuses on the control mechanisms that drive opportunistic and goal-oriented behaviours. The proposed model is based on an interpretation of an established psychological model and described in terms of a set theoretical visual formalism, the stategraph, able to represent concurrent and distributed components. As a conclusion, in contrast to the view in which opportunistic behaviour is described as plan violation, the model suggests that opportunistic behaviour does not totally rule out goal-oriented behaviour. A number of understanding activities can operate in parallel and to some extent, this enables the cognitive system to keep the course of actions according to plan while undertaking other cognitive activities.The model specifies the mechanisms that underlie this concurrence

[1] Supported by Embrapa and the Brazilian National Research Council 'CNPq'

1 Introduction

Although it is recognized in software engineering that prescriptive methods, such as the top down method, are idealizations that will hardly be followed strictly, little has been done to understand and support properly ad hoc activities. These activities are regarded difficult to manage, costly, and bound to produce incorrect solutions. However, as pointed out by Guindon (1990), Dijkstra (1976) called structured program a discipline, suggesting that it may be not a natural way of programming.

The idealistic view of software development suggests that it is a rational process. By rational it is meant that each step taken can be shown to be the best way towards a well defined goal. Considerable amount of research in software design, programming methods and related topics concentrate on the derivation of programs from a specification in the same way that theorems are derived from axioms in a published proof. For example, Lehman (1984), Tursky & Maibaum (1987), to name a few. Prescriptive methods, such as the classical top-down approach or structured programming have been proposed with the goals of managing complexity and producing artifacts that are easy to understand, test, verify, and modify. This idealistic approach might suggest a naive view that software engineers actually follow the prescribed plan. However, when software development based on people is concerned, the established view (Parnas & Clements, 1986) is that we will never find a process that allows us to design software in a perfectly rational way. Nevertheless, we can fake it; "we can present our systems to others as if we had been rational designers". A brief review of Parnas & Clements arguments for this two assertions will be considered in turn.

Among the arguments for why a software design process will always be an idealization are the difficulties in establishing all the relevant facts before start implementation and the inherent evolution of these facts before the end of the project. Even if these difficulties can be overcome, there will remain other difficulties related to human limitations. Human beings are unable to fully comprehend the huge amount and variety of details that must be taken into account in order to design and build a correct system. Large part of the effort in software development is a process in which we attempt to separate information into manageable pieces. Even after this separation, errors are bound to happen, and can only be avoided if one can avoid the use of humans.

Among the arguments for why pretending to follow an idealized software development process is useful nonetheless, are arguments related to good management of progress and project's achievements. In addition, there are arguments that we can relate to practical human requirements for good understanding and communication. Experience shows that designers need guidance. When a large project is undertaken one can easily be overwhelmed by the enormity of the task. It is believed that a good understanding of the ideal process will help the designer in deciding how to proceed. In addition, the designer will come closer to a rational design if he tries to follow the process rather than proceed in a pure ad hoc basis. In summary, it is believed that ideal processes lead to better communication. It makes design easier to review by outsiders and easier to transfer ideas and software between projects, project phases and project members. Specification understanding is an underlying component of the communication process in software development. However little is known about it .

The comprehension of software specifications plays a much more important role in software engineering than has commonly been thought. Specifications are related to the notion of abstractions, and abstractions are important means for coping with complexity. Using a given specifications to achieve understanding is a common part of many activities dispersed throughout the life cycle, including learning, design, programming, inspection, maintenance and reuse. The key to the specification's role is precision and understanding (Tursky & Maibaum, 1987). Specification is a process of achieving common understanding at a suitable level of precision. In contrast with formal methods research, which focuses on precision, our work focuses on the complementary issue of understanding. Many issues regarding how people understand specifications seem intimately related to human cognitive or problem solving limitations. Suitable tools and techniques may help to overcome some of them (Guindon, 1990). However, the development of such tools based on a sound engineering foundation requires a detailed comprehension of the cognitive requirements in this task.

2 Modelling Cognitive Behaviour

Among the many attempts to provide this comprehension, two broad groups are noticeable: 1) the development of specialized representations, and 2) the development of models of the understander's behaviour. In the first approach, research on the development of specialized representations addresses the knowledge held by the understander, its structure and its relationships. Representations such as "plans" (Detienne, & Soloway, 1990) and "design schemas" (Guindon, 1990 and Soloway & Ehrlich, 1984) to name a few, offer different, but related constructs to describe different organizations of the understander's knowledge (Brooks, 1990). This approach is based on the assumption that our ability to automate parts of the understanding process may be increased if more information can be captured in mechanically processable forms. A major problem with models of understanding based on knowledge representation is their inherent incompleteness. The background knowledge cannot be represented as a set of explicit propositions (Winograd & Flores, 1986). The overwhelming nature of knowledge may make it difficult, impossible perhaps, to construct a complete explicit representation. In the second approach, knowledge organization is not everything; the mechanisms that use knowledge are equally relevant. A scientific account of how knowledge is acquired, retained, and used in interpreting the world does not necessarily call for a complete specification of all knowledge (Johnson-Laird, 1989). In order to understand how knowledge is used, research focuses on the cognitive behaviour, in particular on the control mechanisms that produce the behaviour and guide the understander in deciding what to pursue and which parts of the problem to work on next. The emphasis is on assisting the activity of understanding rather than automating it. Models of these dynamic processes are in their infancy.

Following this second approach, the development of a model able to describe cognitive behaviour faces some representational difficulties. The characterization of complex behaviour in a way that aids understanding and at the same time is amenable to more precise analysis has been an issue in software and system engineering. One of the difficulties of this characterization relates to the description of the model's reactive character, through which the system continuously reacts to external and internal stimuli. The reactive behaviour can not be adequately described in terms of simple relationships between inputs and outputs, as commonly used to describe other less complex systems; the allowed combination of inputs and outputs may vary in time and depend on internal states of the system. Such behaviour is better described in terms of the set of allowed sequences of events, actions and conditions restricting them. Another difficulty of this characterization relates to describing concurrence and distribution. In concurrent and distributed systems several processes, which operate concurrently and without hierarchical relationship, should communicate and be controlled. This characteristic can not be adequately described in terms of relationships between states of the system because it is very difficult to get a coherent picture of the system global states. Systems with concurrent components may involve a large number of possible global states which result from the Cartesian product of all sequences of states and events of the components.

Blackboard architecture has been used to characterize cognitive behaviour, for example (Visser, 1990). The use of this architecture to represent cognitive models may have been induced by the fact that cognitive modelling and blackboard architecture share a similar basic approach: they try to locate the source of cognitive activity in the characteristics of the underlying machinery. However, in contrast to blackboard architecture, cognitive modelling is not directly concerned with how the architecture is to be implemented, neither is it limited by the computational tractability of its models. Since its origin, the potential of blackboard architecture for concurrence and distribution has remained an attractive possibility. In fact, most recent blackboard implementations are conceptually parallel and distributed models. However, the promise of multiprocessing and distributed blackboard architecture remains largely untapped. Constructing effective sequential blackboard based applications has been hard enough (Corkill, 1989). The system's approach to control determines its potential behaviour, its coherence and comprehensibility. Control of reasoning in blackboard architecture is still an active research area (Hayes-Roth, 1989). Nevertheless, analogies can be drawn between cognitive models and blackboard architecture, so sometimes it may be useful to describe the one in terms of the other from which more knowledge is available. However, we found that the implementation oriented approach of blackboard architecture imposes unnecessary constraints on discussing cognitive behaviour.

A set-theoretical visual formalism called hygraph has been proposed for describing complex systems (Harel, 1987 and 1988). A hygraph-based extension of the standard state-transition diagrams, the statechart, has been proposed for describing behaviour of reactive systems (Harel & Pnueli, 1985). It provides means of representing complex relationships between components, like concurrence and depth, in a way to produce a concise diagram. It also allows the representation of a broadcast mechanism for communication between concurrent components. A hypothesis put forward in this paper is that this formalism may be used to describe and discuss cognitive behaviour.

In this paper we advance an engineering model of cognitive behaviour. The model makes use of the hygraph representation in order to convey some established hypotheses on the cognitive system functioning. It is based on existing models, in particular on the *model human processor* (Card, Moran & Newell, 1983, Lewis, 1990 and Newell & Card, 1985) so, it can be described as an attempt to provide a statechart based interpretation of a subset of that model. An advantage of this interpretation is that it provides basis for discussing these hypotheses in more objective terms. It also concurs to the elaboration of more complex combinations of behaviour. Here, three layers of behaviour are discussed. Section 3 introduces the basic representation of the cognitive system components and first and the second layers of behaviour. These layers allow for the discussion of the basic mechanisms of understanding. At the first and lower layer, a pattern of behaviour termed *dispersion* is counteracted by another pattern termed *focusing*. They are used to explain *spontaneous* and *rational understanding*, at the second abstract layer. Section 4 discusses the third layer in which *opportunistic* and *goal oriented behaviour* are explained. As a conclusion we suggest that goal-oriented behaviour and opportunistic behaviour can co-occur. A number of understanding activities can operate in parallel and, to some extent, this enables the cognitive system to keep the course of actions according to plan while undertaking other cognitive activities.

3 A Distributed Model of Understanding

This model is based on the description of the underlying cognitive components and their possible interaction. This description is presented in figure 1. A detailed account of the development of this interpretation may be found in Simplicio, (1992). In the diagram, rounded rectangles represent states; arrows represent transitions between states or sets of states; dashed lines represent orthogonal, parallel, partition of states. Small doted arrows represent initial states. When a state occurs it may cause further transitions in other components. This is represented by the condition *[in (state)]* related to a transition. It reads: if *state* occurs then the related transition also takes place. The recurring rectangles in the working memory illustrate that a number of activations may be stored simultaneously.

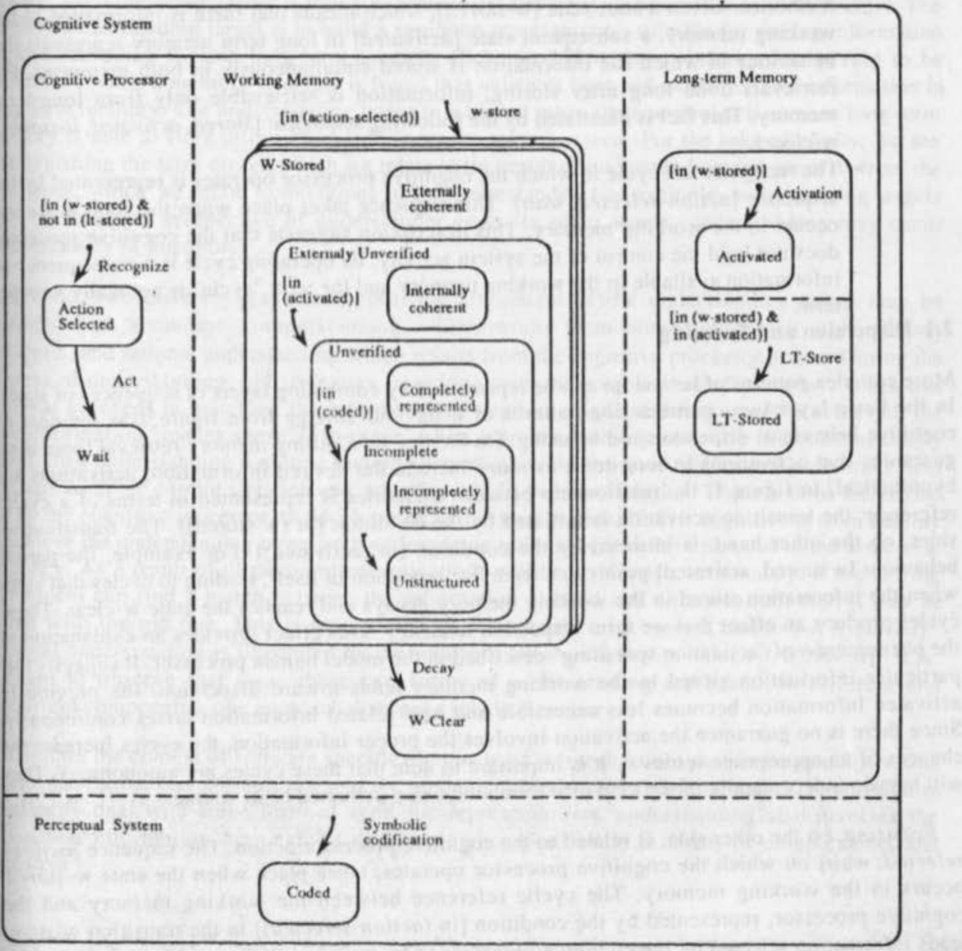

Figure 1. A statechart interpretation of the cognitive system components and their interaction. Rounded rectangles represent states; arrows represent transition between states or set of states; dashed lines represent orthogonal, parallel, partition of states. Small doted arrows represent initial states.

Based on this description, sequences of states can be inferred. Sequences of states represent behaviour. Some patterns of behaviour described in the model human processor can be discussed in terms of particular sequences of states. For example:

- Many memory retrieval failures are explained in terms of a spontaneous decay over time. This spontaneous decay is explicitly represented as a transition into the working memory, leading to the state *w-clear*, which represents the fact that the information is no longer available in the working memory.
- The memory retrieval behaviour depends on the time elapsed between the storing and retrieval of the information. This period will determine which memory, if any, holds the information. For retrievals done shortly after storing, information may be held in both memories. Given a start state [*w-stored*], which means that there is information in the working memory, a subsequent state [*activated*] in long term memory represents the behaviour in which the information is stored simultaneously in both memories. For retrievals done long after storing, information is retrievable only from long-term memory. This fact is illustrated by the following sequence: [*stored, activated, lt-stored, w-clear*].
- The recognize-act cycle in which the cognitive processor operates is represented by the sequence [*action-selected, wait*]. This sequence takes place when the state *w-stored* occurs in the working memory. This description suggests that the cognitive processor does not hold the control of the system activity. Its operating cycle is a consequence of information available in the working memory and the term "cycle" is not really a proper term.

2.1 Dispersion and Focusing

More complex patterns of behaviour can be represented by combining layers of sequences of states. In the basic layer two counteracting patterns of behaviour emerge from figure 1 as the core of cognitive behaviour: dispersion and focusing. On the one side, during memory retrieval there is no guarantee that activations in long-term memory include the desired information; activations are hypothetical. In figure 1, the relationship between memories is represented in terms of a cyclic reference: the transition activation is initiated by the condition [in (w-stored)]. The transition w-store, on the other hand, is initiated by the condition [in (activated)]. For example, the partial behaviour [w-stored, activated] points to a recursive repetition of itself, leading to cycles that stops when the information stored in the working memory decays and reaches the state w-clear. These cycles produce an effect that we term dispersion tendency. This effect provides an explanation of the phenomenon of "activation spreading" described in the model human processor. It suggests that particular information stored in the working memory tends toward dispersion: the previously activated information becomes less accessible and new related information arises continuously. Since there is no guarantee the activation involves the proper information, the cycles increase the chances of an appropriate retrieval. It is important to note that these cycles are autonomous. They will happen independently of the kind of information being activated or its source.

Focusing, on the other side, is related to the cognitive processor action. The sequence [*action-selected, wait*] on which the cognitive processor operates, takes place when the state *w-stored* occurs in the working memory. The cyclic reference between the working memory and the cognitive processor, represented by the condition [in (*action-selected*)] in the transition w-store, leads information selected by the cognitive processor to be re-stored in the working memory. The model human processor broadly refers to this as a "refresh". A primary effect of this is to focus the system on particular relevant information. A secondary effect is to restrict the dispersion tendency.

This restriction happens because dispersion is based on storing new related information in the working memory; refreshing old information reduces the chances for dispersion.

3.2 Autonomous and Rational Understanding

Unlike dispersion, focusing is not an autonomous behaviour. Therefore its action must be driven by mechanisms in a higher abstract level. A second layer of cognitive behaviour may be built based on the specification of: 1) what the understander is trying to achieve, 2) what cognitive actions may support the attainment of this goal and, 3) what constraints the task faced by the understander imposes on these actions.

First, what the understander is trying to achieve defines what we term *understanding target*. The primary understanding target is to build a symbolic representation suitable for storing information in, and retrieving it from long-term memory (Schank, 1986). Input information is said to be understood when a memory structure is found that indicates how and where to store information in long-term memory. The understanding target is achieved when the component *lt-store* in long-term memory is able to store information leading to the state *lt-stored*. For the sake of clarity, we are distinguishing the term *target*, which we relate to the result of understanding as a process, from the term *goal*, which we relate to intentions of the understander. For example, understanding targets may be achieved irrespective of understander goals. In other words, understanding may occur independently of intention.

Second, two different ways to support the attainment of the understanding target may be distinguished: autonomous understanding, which results from immediate match with memory retrievals, and rational understanding which results from the cognitive processor action. During the process of understanding, old structures from long-term memory are called into play. When this retrieval succeeds in matching the information at hand, the understanding target is achieved; the new information "makes sense". However, when a proper memory structure is not immediately found, then understanding requires: finding an applicable old structure, determining to what extent it differs from the current situation and, adapting it to fit the new situation. These actions are carried out by the cognitive processor, to which we relate the term "rational". The cognitive system's ability to achieve the understanding target without involving rational actions is in part due to the dispersion tendency. As a result of dispersion new activations will be produced continuously. If the storing component can find a match between the information at hand and an activation, then it may be stored with the old one. This is represented by the sequence [*w-stored, activated, lt-stored*]. Rational understanding is illustrated by the sequence [*w-stored, action-selected, lt-stored*]. It is relevant to observe that since these two forms of understanding are based on different and concurrent components, one approach does not exclude the other.

Third, in the context of software specification, at least two restrictions set out the conditions that ordain what processes are necessary to the attainment of the understanding target. In addition to necessarily deal with some form of symbolic representations, understanding also involves the notion that information may exhibit different elaboration levels in terms of completeness and consistency.

3.3 Elaboration Levels and Promotions

Information being manipulated by the cognitive system may exhibit different levels of completeness and consistency. In ideal terms, at the highest level all relevant information is collected and expressed in a selected representation scheme and that is coherent with all other structures available. At the lowest level, information symbolically represented provides simple associations between external stimulus and knowledge eventually activated in memory. In the continuum between these extremes, we distinguish five discrete points that we term elaboration levels. Information may be: unstructured, incompletely-represented, completely-represented, internally-coherent and externally-coherent. High elaboration levels are desirable properties of the information used by the cognitive system. Nevertheless, this is not always required or attainable. In order to provide a proper response to the environment, it may be neçessary to elaborate the available information to higher levels. This elaboration we term promotion. Four promotions are conceived. Promotion 1 seeks to lead information from the state *unstructured* to the state *incomplete*. Promotion 2 aims to lead information from *incompletely-represented* state towards *completely-represented*. Upgrading from *completely-represented* state towards *internally-coherent* state is carried out by promotion 3. Finally, from the *internally-coherent* state promotion 4 seeks to lead to the *externally-coherent* state. A detailed description of the promotions is not necessary here. It is relevant to observe that understanding may be achieved at any elaboration level and that promotions may involve different patterns of behaviour.

3.4 The Control Mechanism of Understanding

The cognitive system components are not equally able to deal with all elaboration levels; some are restricted to handle only lower levels. This is represented in figure 1 by the conditions [in (*state*)] associated with transitions in working memory. The perceptual system activity is restricted to *incomplete* states. This is represented by the condition [in (*coded*)]. The long-term memory retrieval is restricted to *unverified* states by the condition [in (*activated*)]. The cognitive processor activity is related to all states by the condition [in (*action-selected*)]. This indicates that only the cognitive processor is able to deal with all elaboration levels. All components may deal with promotion 1, so several patterns of behaviour may lead to this promotion.

Since some promotions may be accomplished by different means, the cognitive system faces a possible choice between them. For example, the promotion 1, from *unstructured* to *incompletely-represented* states, may be carried out by both, long-term memory activity and cognitive processor. This is represented by the sequences [*coded, unstructured, activated, incompletely-represented, lt-stored*] and [*coded, unstructured, action-selected, incompletely represented, lt-stored*]. These sequences are possible because both long-term memory and cognitive processor can deal with those levels of elaboration. However, because the first sequence results from the autonomous behaviour characterized by autonomous understanding it will take place first.

Autonomous understanding comes before rational understanding. However, because they share the same target, there will be no need to pursue the second unless the first fails to accomplish the understanding target. A detailed mechanism to explain how long-term memory informs the cognitive processor that it has failed is not necessary here. The cognitive processor will be aroused to action if long-term memory fails to match information and an activation. This is represented by the condition [not in (lt-stored)] in the cognitive processor in figure 1. Once aroused to action it should recognize what actions are appropriate to deal with the present elaboration level. Understanding may be accomplished by promoting information to higher levels of elaboration. If the new elaboration level can be still handled by memory activity, then again, autonomous understanding will be attempted first. If autonomous understanding finds its limit of action, in the *unverified* elaboration level, then promotions can be carried out only by rational understanding.

4 COGNITIVE BEHAVIOUR IN UNDERSTANDING

To sum up so far, we have examined patterns of behaviour in terms of sequences of states built onto a description of the cognitive components. Two layers of behaviour have been discussed. At the basic layer, two counteracting patterns have emerged as relevant: dispersion and focusing. At the second layer, two other patterns, autonomous and rational understanding support the attainment of the understanding target of building a representation able to be stored in and retrieved from long-term memory. The control mechanism of understanding was described by combining these ways of achieving understanding and differences in the nature of the information at hand which may exhibit different elaboration levels for completeness and consistency. This mechanism underlies a third layer in which the cognitive behaviour is discussed.

4.1 Goal Processing as Understanding Based Activity

Reasoning about goals and ways to achieve them calls for the construction of plans. This notion highlights a functional distinction between control knowledge, characterized in terms of plans, and other uses of knowledge, which can be called problem knowledge. In the context of software specification understanding, the distinction between plans and problem knowledge corresponds to distinguishing between 1) reasoning about what to understand, and 2) the actual understanding. At first glance, the importance of consistent plans as a way to succeed in dealing with the world may suggest that the cognitive capability of building and revising plans should be carried out by a special component of the cognitive system, or at least, by a specially adapted mechanism. However, there is no evidence of a component or processor specifically allocated to this function. Goal processing shares the same cognitive resources used for processing other forms and use of knowledge. The distinction between them is simply functional. Therefore, goal processing is based on the same cognitive mechanisms used for understanding. Plans will have to be understood in the same way that problem knowledge is understood. This view of goal processing amounts to saying that goal-oriented behaviour is based on plan understanding.

Since goal processing and other information processing share the same cognitive resources, they may eventually be in competition. If goal oriented behaviour is based on plan understanding then it is supported by autonomous understanding, or when that fails, rational understanding is called into play, so two different situations may be envisaged. On the one hand, plans that are frequently used acquire stable representation and effective memory links, so proper activations can increase the chances of autonomous understanding. In addition, usually there is no need to have plans represented in high elaboration levels, so autonomous understanding is enough. Because the process of activation is concurrent with the process of storing in long-term memory, and the working memory may hold a number of activations simultaneously, it follows that goal-oriented behaviour based on autonomous understanding may operate in parallel with other processing. On the other hand, when autonomous understanding fails then goal processing will be based on rational understanding. In this case, the need for goal processing may compete for the use of the cognitive processor with other needs of information processing. This competition is possible because the working memory is able to store several activations simultaneously, but the cognitive processor is able to focus on only one at a time.

4.2 A Competing Situation

We devise the following example to illustrate this competing situation. This example was inspired by the empirical study conducted by Visser, (1990). Let us suppose an understander verifying the external coherence of an arbitrary specification segment. To be in this state implies that this specification segment was not autonomously understood; a representation scheme was activated from memory to accommodate it; this representation was promoted to the internally coherent elaboration level, and now, it is being promoted to the externally coherent elaboration level. This promotion may involve testing consequences of the description against scenarios of the problem domain. During the promotion the cognitive processor is focusing on a particular activation, so, autonomous dispersion is restricted to very closely related activations. For example, dispersion may help to activate information about scenarios of the domain. Nevertheless, the understander may also activate the representation of another specification segment which is analogous to that segment at hand, which is also not fully understood, but whose domain the understander is more familiar with. This activation could provide the basis for autonomous understanding of that first segment. However, because the new specification segment is not yet understood and because memory is not able to deal with information at this high elaboration level, the promotion will probably continue. But, which segment will be focused? The understander may continue with the first segment or deviate to the second segment with which he is more familiar. The decision of focusing on one activation leads to abandoning the other. The abandoned segment will fade away and will be lost from the working memory. There is no guarantee that the lost activation can be retrieved later on.

To prevent losing one of these activations, a plan has to be built to include the goal of dealing with the abandoned activation in the future. Two issues arise in this case: 1) when should the postponed activation be dealt with properly and, 2) where to store the plan in memory in order to retrieve it in due time. This plan may be simple enough to be processed in terms of autonomous understanding. In this case, the plan is restricted to low levels of elaboration and there is a relative high risk that it will not be activated in due time. For example, if at the right time the situation is handled in terms of autonomous understanding then the system will probably provide no focusing to activate this plan. Also, the future situation may not happen in the way it was conceived; the plan representation may not be suitable for representing the future event. So, an effective plan to deal with this conflict will require rational understanding. The problem in this situation is that by focusing on the plan, the two segments of the specification are at risk of being lost from memory, even for very simple plans. This configures a competing situation. For example, the understander may decide to follow dealing with the first segment with an interruption just long enough to take notes and guarantee that the second segment may be dealt with later. But, because taking notes very probably requires translating the information into a different representation scheme, carrying out this plan will call for rational understanding. As a consequence the first segment is bound to fade in memory.

4.3 Opportunism as a Promising Behaviour

The competing situation illustrated above may be summarized as: 1) the understander is focusing on a specification segment ***a***; 2) a related segment ***b*** is activated in memory as opportune; 3) there is a goal of dealing with both segments; 4) building a plan able to support the achievement of this goal may result in losing one or even both specification segments from memory. The underlying issue is which segment has better chances of being recovered later. An answer to this issue lies in the plan that was actually being followed before the activation of the segment ***b***, that is, the plan that was driving the behaviour to focus the segment ***a***. If this plan can be recovered, then segment ***a*** may also be recovered. Three situations may hold: 1) If the plan is general, such as following the text of the specification, it probably can be recovered easily because the text will provide the auxiliary memory to help the plan resumption. 2) If the plan is specific and at a high elaboration level then it probably can also be recovered by reasoning about the plan itself; 3) If there was no plan, then focusing ***a*** or ***b*** will make little difference, so other distinctions, such as the relevance of each segment may be considered. However, in the same way that to build a plan to deal with segments ***a*** or ***b*** may lead to the lost of one or both segments, considering which one is more relevant may also put them at risk of fading in memory. As a conclusion, segment ***a*** offers, in its worst case, the same chance of being retrieved as segment ***b***.

If instead of building a plan to deal properly with segment ***a*** or ***b***, the understander just deviates to segment ***b***, then there will be less risk of losing relevant information in the future. In addition there will be less effort spent on rational processing. In general, segment ***a*** is more likely to be retrieved later than segment ***b***; building a plan to deal with the situation may not improve the chances of success; and segment ***b*** will be lost if it is not dealt with at once. Thus, opportunistic behaviour constitutes the more promising way to succeed in dealing with unpredictable events. Since many world events are of this sort, opportunistic behaviour is the typical way of reacting. It is a way of taking advantage of the knowledge available at the time. To behave opportunistically corresponds to following the general rule: if something has to be done, it is better to do it at once.

One might suggest that, in the long run, to follow opportunistic behaviour in the way described above will lead the cognitive system to chaos. However, it should be observed that this does not exclude goal-oriented behaviour totally. Since goal oriented behaviour may also be carried out by autonomous understanding it may function effectively in parallel. This means that the cognitive system is able to keep track of general and simple plans while undertaking opportunistic behaviour. A restriction to this sort of plan is that it is limited to low elaboration levels. This is a characteristic of autonomous understanding. For example, this plan is not capable of registering multiple layers of deviations; it is local. Yet, there is no need for plans to be at high elaboration levels to succeed in many frequent activities.

However, autonomous understanding may not be enough to provide understanding. This may occur in both cases: in understanding the plan or in understanding the problem. In the example above, autonomous understanding may fail while understanding the specification segment and also while keeping the course of actions according to a plan. Exceptions are bound to happen, which requires rational understanding. When the process of understanding the problem is what fails, then this does not necessarily require changing the general plan. A local plan may be activated in order to deal with this situation in an opportunistic way. As we have pointed out, opportunistic behaviour will provide better chances for achieving understanding and the retrieval of the abandoned activations later on. The general plan may be resumed afterward. However, if in addition to the failure in understanding the problem, the process of understanding the plan also fails, then we have a breakdown: what one was trying to do has failed, and there is no new plan to indicate what to do next. In this case goal processing will require rational understanding, and this may be found competing for the use of the cognitive resources with the need for understanding the problem.

5 Conclusions

In conclusion, this paper focuses on modelling the cognitive control mechanisms that guide an understander in deciding what to pursue and which parts of the specification to work on next, during the activity of software specification understanding. Current research stresses the distinction between goal-oriented behaviour, in which a plan is followed, and opportunistic behaviour, in which no plan is observable. One view of opportunistic behaviour describes it as plan deviations (Visser, 1990). This view assumes that the cognitive system follows a comprehensive and well defined plan that is interrupted when new information or new events are perceived as opportune. In contrast, this paper suggests a rather different view in which opportunistic and goal oriented behaviours to a certain extent can proceed concurrently. A number of autonomous understanding activities can operate in parallel and this allows the cognitive system to keep the course of actions according to a plan while undertaking other cognitive activities.

The model introduced here analyses these mechanisms and provides the basis for further work to discuss how software specification understanding may be supported by providing an appropriate interaction between the understander and a set of software tools. Experimental evidences for the model are emerging, for example, Davies (1991) and Davies & Simplicio (1992).

Acknowledgements

I wish to thank Thomas Maibaum and Anthony Finkelstein for invaluable comments in earlier versions of this paper.

References

Brooks, R. (1990) 'Categories of programming knowledge and their application', International Journal of Man-Machine Studies,33,3, pp. 241-246.

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

Corkill, D. (1989) "Design alternatives for parallel and distributed blackboard systems" In Blackboard Architectures and Applications, V. Jagannathan, R. Dodhiawala and L. Baum (Eds), Academic Press, Inc.

Davies, S & Simplicio, F. (1992) Opportunistic and goal-oriented behaviour in software design: combining empirical and theoretical studies in cognitive modelling, in proceedings of Second International Conference on Artificial Intelligence in Design, AID'92 (to appear).

Detienne, F. & Soloway, E. (1990) An empirically-derived control structure for the process of program understanding", International Journal of Man Machine Studies, **33**,3 pp.323-342.

Guindon, R. & Curtis, B. (1988) 'Control of cognitive processes during software design: What tools are needed ?' in E. Soloway and S. Iyengar (eds.) *Empirical Studies of Programmers*, First workshop, Ablex. pp. 263-269.

Guindon, R. (1990) "Designing the design process: exploiting opportunistic thoughts", In Human-Computer Interaction,**5**,pp. 305-344.

Harel, D. (1988) "On visual formalisms", Communications of the ACM, 31,5,pp.514-530.

Harel, D. (1987) Statecharts: a visual formalism for complex systems. Science of Computer Programming, 8,3,pp.231-274

Harel, D & Pnueli, A (1985) On the development of reactive systems In Logics and Models of concurrent Systems. NATO ASI Series, 13, pp 477-498, K.R apt. (ed), Springer Verlag.

Hayes-Roth, B (1989) In Blackboard Architectures and Applications, V. Jagannathan, R Dodhiawala and L. Baum (Eds), Academic Press, Inc.

Johnson-Laird, P. (1989) 'Mental Models', in M. Posner, (eds.) Foundations of Cognitive Science, MIT.

Lewis, C. (1990) 'A research agenda for the nineties in Human-Computer Interaction', Human-Computer Interaction,5,pp. 125-143.

Newell, A. & Card, S. (1985) "The prospects for psychological science in human-computer interaction". Human-Computer Interaction,1,pp. 209-242.

Parnas, D. & Clements, P. (1986) A rational design process: how and why to fake it, IEEE Tras. Software Engineering, 12, 2 pp 251-257.

Schank, R. (1986) Explanation Patterns: Understanding mechanically and creatively. Lawrence Erlbaum Associates, Inc.

Simplicio, F. (1991) Modelling Cognitive Behaviour in Specification Understanding. In Nato Advanced Research Workshop, User-Centred Requirements for Software engineering Enviroments, Toulouse, France, Nato ISEP.

Simplicio, F. (1991a) Modelling Cognitive Behaviour in Specification Understanding. In Nato Advanced Research Workshop, User-Centred Requirements for Software engineering Enviroments, Toulouse, France, Nato ISEP.

Simplicio, F. (1992a) A distributed model of cognitive behaviour in Specification Understanding. In Proceedings of Nato Advanced research workshop, Cognitive Models and Intellegent Environments for learning programming, Genova, Nato ISEP.

Soloway, E. Ehrlich, K. (1984) 'Empirical studies of programming knowledge', *IEEE Transactions on Software Engineering*, 10, 5, pp. 595-609.

Turski, W. & Maibaum, T. (1987) The Specification of Computer Programs, Adisson Wesley Pub.

Visser, W. (1990) 'More or less following a plan during design: opportunistic deviations in specification', *International Journal of Man-Machine Studies*, 33, 3, pp. 247-278.

Winograd, T. & Flores, F. (1986) *Understanding computers and Cognition. A new foundation for Design.* Ablex Publ.