Perceived complexity versus internal complexity
Did we take into account expertise, reliability and cognitive stability?

Guy Boy[1] and Jeffrey Bradshaw[2]

[1] European Institute of Cognitive Sciences and Engineering (EURISCO International)
4 avenue Edouard Belin, 31400 Toulouse, France
Email: guy.boy@eurisco.org

[2] Florida Institute for Human & Machine Cognition
40 South Alcaniz Street Pensacola, FL 32502, USA
Email: jbradshaw@ihmc.us

Abstract. Human reliability issues in safety-critical systems, in aviation for example, motivated and still motivate the design and use of protections that can be tool-based or organizational. Software and hardware have been developed to overcome human reliability to enable both tolerance and resistance to human errors. Consequently, systems have become more complex and the distance between people and actual production machines never stopped to increase. Most of the time, the perceived complexity tremendously decreased when the automated product matured, sometimes after a difficult start where it was high to very high. This paper presents a synthesis on complexity and cognitive stability in human-machine systems, and more specifically in highly automated systems. It emphasize several issues such as technological complexity, complexity and expertise, reliability of machines and people, and complexity and resilience. The paper emphasizes interaction between people and highly automated safety-critical systems. What do people expect from their cooperation with their "friendly" automata? Do they need to know about their internal complexity to interact with them? How do they perceive their external complexity? What is the right level of abstraction required to interact safely, efficiently and comfortably?

1 INTRODUCTION

Human reliability issues in aviation motivated, and still motivates, the design and use of protections that can be tool-based or organizational. Software and hardware have been developed to overcome human reliability to enable both tolerance and resistance to human errors. Consequently, systems have become more complex and the distance between people and actual production machines never stopped to increase. Most of the time, the perceived complexity tremendously decreased when the automated product matured, sometimes after a difficult start where it was high to very high. Humans and automata are now working together for the best and, we hope, not for the worst. An automaton, that could be a software agent in the modern sense, is defined in a specific context and can be used in very different contexts. We are often working on the definition limit, and sometimes over that limit. This is mainly due to the fact that things don't go wrong and we believe that the tool can be used in a way that is very different from

the way it was designed for. A machine or a tool, whether it involves physical force or information processing, has a role that must be clearly defined. In return, its use induces a cognitive function (or several cognitive functions) that has also a role. In the aeronautics industry as in all safety-critical industries, it is always important and necessary to define roles. In addition to the role, we cannot avoid defining the context of use, even if we know that context is typically difficult to grasp. What do we mean by context? Do we mean context of design or context of use? In addition, we need to rationalize the resources that will be used to fulfill the role. We have defined a cognitive function as a cognitive process that has a role defined in a specific context and fulfilled using a set of resources. Such resources can be cognitive functions themselves. Therefore, the cognitive function concept is recursive (Boy, 1998). It is then possible to start describing a cognitive function of an agent and by developing it we may end up in a different agent that could be a human or a machine. We will defend the claim that the internal complexity of a system must be available to its operator when it is not entirely mastered, robust and reliable. People need to understand what is going on inside, and, of course, capable to articulate the internal complexity in order to eventually recover from failures. Conversely, it would be counter-productive to show the strings of the magician trick when the magician very well masters the complexity of the trick! In other words, when a system is reliable and robust enough for a routine use, internal complexity is no longer useful and must not be visible.

The problem comes with perceived complexity. Whether a system is internally complex or not, it could be perceived as complex anyway by its operator. How can we simplify its use? There are cognitive functions that are devoted to the task supported by the system, and others that are devoted to the interaction with the system (Boy, 1995). We need to be very careful in distinguishing these two types of cognitive functions in the design of a system. Designers are usually focused on automating the task, forgetting interaction issues. They think that they simplify the work by removing a large part of the burden involved in task performance, but if interaction with the freshly-automated system is difficult, boring and/or inefficient, then the human operator will not like it! Perceived complexity is a matter of designing for appropriate emerging cognitive functions supporting an efficient, nice and easy interaction. In addition, we need to consider normal and abnormal situations. Systems are usually designed for normal situations, taking into account a set of identified abnormal situations. When things go wrong in unpredicted abnormal situations, human operators have to cope and find solutions. At this point, there are systems that are designed in such a way that they “naturally” bring us back to a normal situation. We will say that they are cognitively stable. Other systems may diverge from the normal way of doing things when something is going wrong. They are cognitively instable. We will develop this physical metaphor of stability where balance errors will be replaced by human errors, or system failures also. One of us already proposed the concepts of cognitive stability and cognitive support that will be further developed in this paper (Boy, 2005). In particular, there are three processes that frame our investigation: anticipation, interaction and recovery. Well-trained and knowledgeable people tend to anticipate reducing workload, stress and other humans factors likely to provoke human errors. There are systems that facilitate such anticipation, other

that don't. In the same way, there are systems that facilitate cognitive stability during interaction and recovery after errors or failures.

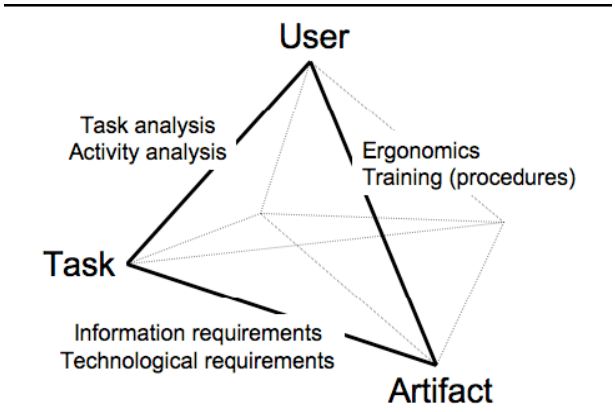
2 MODELING COMPLEXITY IN HUMAN-MACHINE SYSTEMS

Javaux et De Keyser (1998) defined cognitive complexity of a human-machine situation (in which specific tasks are performed) as the quantity of cognitive resources that a human operator must involve to make sure that tasks are executed with an acceptable level of performance. Several authors worked on complexity in human-machine interaction (HMI). Amalberti analyzes complexity by making a distinction between nominal and non-nominal situations (Amalberti, 1996). He related HMI complexity to the dynamics of underlying processes, time pressure, acts irreversibility, unpredictability of the processes, the number of systems to be managed at the same time, risk, factors coming from the insertion of safety-critical systems in cooperative macro-systems and factors related to the human-machine interface. Van Daele made another distinction between situation complexity and the complexity of task and operational goals (Van Daele, 1993). Van Daele relates complexity to HMI, i.e., constraints blocking the execution of a task, remote character of goals, multiplicity of goals to be satisfied at the same time, interdependence of goals and environment dynamic attribute, multi-determination, uncertainty and risks. Pedersen provided several kinds of complexity definitions (Pedersen, 1990). He distinguished objective and subjective complexity, system complexity, representational complexity and agent-related complexity. Theoretical computer science provided its own definition of complexity, and Pedersen in particular dissociates computational complexity into algorithmic and informational complexity. Card introduced KLM (*Keystroke-Level Model*; Card et al., 1980) and GOMS (*Goals, Operators, Methods and Selection rules*; Card et al., 1983) to study text processing in office automation. KLM and GOMS enable the prediction of the required time to perform a specific task. They assume task linearity (i.e., tasks can be hierarchically decomposed into sequences), belong to the class of analytical models, and work well in very closed worlds. Kieras and Polson (1985) developed the *Cognitive Complexity Theory* (CCT) as an evolution of GOMS. They proposed several measures of HMI complexity such as the number of necessary production rules and the learning time, as well as the number of items momentarily kept in the working memory in order to predict the probability of errors. Norman proposed a generic model that takes into account human actions, learning, usability and possibility of errors (Norman 1986). He proposed the following concepts: physical versus psychological variables; physical versus mental states; goal as a mental state; and intention as a decision to act to reach a goal. He expressed interaction complexity in terms of execution gulf and evaluation gulf. In particular, the distinction between physical and psychological variables enables showing complexity factors related to interaction induced by the use of the physical system and the task that the user is required to perform. Boy and Tessier developed the MESSAGE system in order to predict human-machine system performance in early glass cockpits (Boy, 1983; Tessier, 1984; Boy et Tessier, 1985). MESSAGE was a multi-agent simulation system taking into account pilots, aircraft automation and air traffic controllers. Human agents were modeled

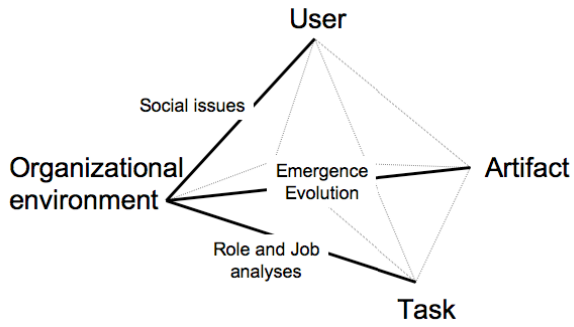
as information processing systems, in the Newell and Simon's sense (1972). The aircraft was modeled by a set of flight dynamics equations and system logics. The main objective was to measure workload and complexity as information-processing difficulty. Several difficulty indices were developed including visibility, observability, accessibility, operability and monitorability. Rasmussen proposed the SRK model to capture three types of behavior, i.e., skills, rules and knowledge (Rasmussen, 1986). He also developed an ecological approach based on five levels of abstraction hierarchy. This approach was used by Vicente to develop his work analysis approach (Vicente, 1999). Interaction blocks were developed to take into account interaction chains among expert agents in order to better understand the possible proceduralization of underlying operations (Boy, 1998). The description of interaction using interaction-blocks requires the elicitation and specification of the interaction context, and therefore structuring various relevant situations. Five generic interaction-block structures were proposed including sequence, parallel blocks, loop, deviation, hierarchy and blocks leading to either weak or strong abnormal conditions. These generic structures enable the analysis of interaction complexity.

3 COMPLEXITY, EXPERTISE AND RELIABILITY

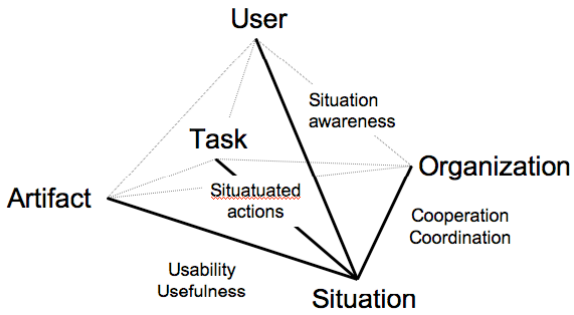
As announced in the title of the paper, we make a distinction between perceived complexity and internal complexity. Almost all models and approaches that are presented above have been designed and used to assess internal complexity. They are analytical. Complexity in the real world needs to be elicited from observation and interaction with appropriate agents. The main difficulty in complexity assessment is that complexity is related to expertise, and even if expertise is easy to assess it cannot be anticipated correctly. Users, tasks and artifacts should then be designed in an environment that is clearly specified. The AUTOS framework was proposed to handle this kind of human-centered design for safety-critical systems (Boy, 2005). The AUTO tetrahedron (Boy, 1998) was introduced to help relate four entities: Artifact (i.e. system), User, Task and Organizational environment. Artifacts are aircraft systems, devices and parts for example. Users may be novices, experienced or experts, coming from and evolving in various cultures. They may be tired, stressed, making errors, old or young, as well as in very good shape. Tasks vary from handling quality control, flight management, managing a passenger cabin, repairing, designing, supplying or managing a team or an organization. Each task corresponds to one or several cognitive functions that related users must learn and use.



The AUT triangle enables the explanation of three edges: task and activity analysis (U-T); information requirements and technological limitations (T-A); ergonomics and training (procedures) (T-U).



The organizational environment includes all the agents that interact with the user performing the task using the artifact. It introduces three edges: social issues (U-O); role and job analyses (T-O); emergence and evolution (A-O).



The AUTOS framework is an extension of the AUTO tetrahedron that introduces a new dimension, the "Situation", which was implicitly included in the "Organizational environment". The three new edges are: usability/usefulness (A-S); situation awareness (U-S); situated actions (T-S); cooperation/coordination (O-S).

In addition, complexity is also related to reliability. The complexity of things that are reliable, i.e., they don't fail or fail with an extremely low probability, is not perceived. In other words, when you can delegate with trust and the work performed is successful, the complexity of the delegate is not an issue. However, when it fails, you start to investigate why. You look into the "back-box"! For that matter, the "black-box" should be transparent, i.e., the complexity of any unreliable agent or tool should be made transparent to its user. This implies that the user should be knowledgeable about the way the agent or the tool works, expertise is then up to front. For a long time, and still now, aircraft pilots are experts in their job because the machine may fail from different angles. It takes a long time to make a pilot. However, reliability is not only technological. Amalberti talks about ultra-safe systems, extremely reliable, when he talks of airplanes (Amalberti, 2001). The problem is that the coupling of humans and these ultra-safe systems may not be safe in some specific situations. Even ultra-safe systems should be considered in context of use with real people operating them to observe the complexity of

the resulting system. Traditional engineering analyses of the reliability of systems intend to assess the probability and consequences of failure. Unfortunately, for the most serious and unacceptable types of failure, the probability cannot be even estimated because it is almost impossible to predict human errors, whether they are intentional or not, and terrorist attack for example. Even ultra-safe systems are vulnerable and this vulnerability is impossible to anticipate. However, we can develop categories of events that led to disasters or near misses in order to improve our understanding on the possible answers that could be brought in real-time either to anticipate, manage or recover from such events. We should keep in mind that any categorization of that kind will not be able to predict which particular trouble event will be most important. Human-machine systems resilience is therefore a matter of strong training and experience.

4 COMPLEXITY, COGNITIVE STABILITY AND RESILIENCE

We can see a resilient system as a shock absorber. For that matter, a distinction could be made between passive resilience and active resilience. The same as in car safety, we talk about passive and active safety. One of us introduced the concept of procedural interfaces that takes into account four mains high-level requirements, i.e., simplicity (as opposed to perceived complexity), observability/controllability, redundancy and cognitive stability (Boy, 2002). When a human being controls a system, there are two main questions that arise: (1) is the system observable, i.e., are the available outputs necessary and sufficient to figure out what the system does? (2) Is the system controllable, i.e., are the available inputs necessary and sufficient to appropriately influence the overall state of the system? A cognitive model is developed to control a system, associating observable states to controllable states (Norman, 1986, Rasmussen, 1986). There is a compromise between controlling a system through a large set of independent observable states and a small set of integrated observable states. "… The larger the number of degrees of freedom in a system, the more difficult it is to make the system behave as desired. Simply counting degrees of freedom, however, oversimplifies the issue. It is the manner in which degrees of freedom interact that determines the difficulty of controlling a system. For example, if the n degrees of freedom of a system are independent of one another, then the controlling system needs only to process an algorithm that is adequate for the control of a single degree of freedom; the algorithm can be replicated n times to control the overall system. Conversely, if the degrees of freedom are not independent (that is, if the effects of specifications of values for a particular degree of freedom depend on the values of other degrees of freedom), then a team of independent controllers is no longer adequate, and more complex control algorithms must be considered." (Jordan & Rosenbaum, 1989). The interface of a system is characterized by a set of n observable states or outputs $\{O_1, O_2, \ldots O_n\}$, and a set of m controllable states or inputs $\{I_1, I_2, \ldots I_m\}$. The interface is redundant if there are p outputs ($p<n$), and q inputs ($q<m$) that are necessary and sufficient to use the system. The remaining (n-q) outputs and (m-q) inputs are redundant interface states when they are associated with independent subsystems of the overall system. These redundant states need to be chosen in order to assist the user in normal, abnormal and emergency situations. In aircraft cockpits, for example, several instruments are duplicated, one for the captain and another for

the copilot. In addition, some observable states displayed on digital instruments are also available on redundant traditional instruments. Controlling a system state-by-state with the appropriate redundant information is quite different from delegating this control activity to an automaton. New kinds of redundancy emerge from the use of highly automated systems. Traditional system observability and controllability usually deal with the *What* system states. The supervision of highly automated systems requires redundant information on the "why", "how", "with what" and "when" in order to increase insight, confidence, and reliability: *Why* the system is doing what it does? *How* to obtain a system state with respect to an action using control devices? *With what* other display or device the current input/output should be associated? *Cognitive stability* is analyzed using the metaphor of stability in physics. Stability can be static or dynamic. Static stability is related to the degrees of freedom, e.g., an object in a three-dimensional world is usually defined by three degrees of freedom. A chair is stable when it has (at least) three legs. Human beings are stable with two legs, but this is a dynamic stability because they have learnt to compensate, often unconsciously, their instability. When an object is disturbed by an external event there are usually two cases: a case where the object returns to its original position, we say that the object is in a stable state; and a case where the object diverges from its original position, we say that the object is (or was) in an unstable state. Human errors have been extensively studied during the last two decades, and several categories have been derived (Norman, 1981; Reason, 1990; Hollnagel, 1993). When a user acts erroneously, there are two cases: a case where the user recovers from his or her erroneous action, we say that the user is in a stable state; and a case where the user does not recover from his or her erroneous action, we say that the user is (or was) in an unstable state. There are human erroneous actions that may be tolerated, and others that should be blocked. Error tolerance and error resistance systems are usually useful redundancy. Error tolerance is always associated to error recovery. There are errors that are good to make because they foster awareness and recovery. However, recovery is often difficult, and sometimes impossible, when appropriate resources are not available. The concept of action reversibility should be put forward and exploited whenever a user can backtrack from an erroneous action, and act correctly. The UNDO function available on most software applications today provides a redundancy to users who detect typos and decide to correct them. Thus, making typos is tolerated, and a recovery resource is available. Error resistance is, or should be, associated to risk. Error-resistance resources are useful in safety-critical systems when high risks are possible. They may not be appropriate in low-risk environments because they usually disturb task execution. For example, text processors that provide permanent automatic grammar checking may disturb the main task of generating ideas. Inappropriate learning and training, poor vigilance, fatigue and high workload are the main adverse influences on cognitive stability.

5 CONCLUSION AND PERSTECTIVES

Cognitive stability is enhanced by simplicity, redundancy as well as appropriate observability and controllability of the user interface. Users tend to use redundant sources of information whether they are personally constructed or deliberately provided in order

to maintain a reasonable cognitive stability. Such redundant sources of information will be called stabilizing cognitive functions. "What will happen if I do this?" Any redundant resource that contributes to answering a user question of that type is likely to support cognitive stability. Perceived complexity can be minimal when a safety-critical human-machine system is well "designed", i.e., when human operators of such a system is able to control and manage appropriate variables that enhance their cognitive stability, and therefore leads to an overall resilient human-machine system. This assumes that internal complexity is mastered, i.e., reliable and robust, otherwise internal complexity needs to be shown to human operators, which requires additional expertise and necessary training. The main difficulty is that cognitive functions that are necessary to control and manage these systems incrementally emerge from practice and therefore are difficult to anticipate during design and development when there is not enough time to bring the overall human-machine system to maturity.

REFERENCES

Amalberti, R. (1996). *La conduite de systèmes à risques.* Le Travail Humain. Presses Universitaires de France : Paris.

Amalberti, R. (2001). From little incidents to the big one. *EURISCO International Summer School on Design for Safety*, Saint-Lary, France.

Boy, G.A. (1983). Le système MESSAGE: un premier pas vers l'analyse assistée par ordinateur des interactions homme-machine. *Le Travail Humain*, 46 (2)

Boy, G.A. & Tessier, C. (1985). Cockpit analysis and assessment by the MESSAGE methodology. *Proceedings of the Second Conference on Analysis, Design and Evaluation of Man-Machine Systems*. (pp. 73-79). Oxford: Pergamon Press.

Boy, G.A. (1994). *La méthode GOMS pour l'analyse et la conception d'interface utilisateur*. Notes de cours SUPAERO, Toulouse.

Boy, G.A. (1995). Supportability-based design rationale. *Proceedings of the 6th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design and Evaluation of Man-Machine Systems*. Boston, MA, USA. June.

Boy, G.A. (1998). *Cognitive Function Analysis*. Ablex-Greenwood Publishing Group : Westport, CT, USA.

Boy, G.A. (2002). Procedural interfaces. *Proceedings of HIM'02* (the Francophone Conference on Human-Computer Interaction). ACM Press (ACM Digital Library), New York, USA.

Boy, G.A., (2005). Human-centered design: The AUTOS pyramid. *EURISCO International Newsletter #4*, Fall.

Boy, G.A. (2005). Human-Center Automation of Transportation Systems. *AAET 2005 Conference Proceedings*. Braunschweig, Germany. February

Card, S.K., Moran, T.P. & Newell, A. (1980). The keystroke-level model for user performance with interactive systems. *Communications of the ACM*, 23, 396-410.

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

Javaux, D. & De Keyser, V. (1998). *Complexité et conscience de la situation*. Rapport final SFACT/DGAC.

Jordan, M.I. & Rosenbaum, D.A. (1989). Action. In *Foundations of Cognitive Science*, Michael I. Posner (Ed.). The MIT Press, Cambridge, MA.

Kieras, D.E. & Polson, P.G. (1985). An approach to the formal analysis of user complexity. *International Journal of Man-Machine Studies*, 22:365-394.

Newell, A. & Simon, H. (1972). *Human Problem Solving*. Englewood Cliffs, NJ: Prentice-Hall.

Norman, D.A. (1986). Cognitive Engineering. In D. Norman & S. Draper (Eds.), *User-Centered System Design*. (pp. 31-61). Hillsdale, NJ: Lawrence Erlbaum Associate.

Pedersen S.A. (1990). Coping with Objective Complexity. In J. Rasmussen, B. Brehmer, M. de Montmollin & J. Leplat (Eds.), *Taxonomy for Analysis of Work Domains. Proceedings of the first MOHAWC workshop*. RISOE National Laboratory: Roskilde.

Rasmussen, J. (1986). *Information Processing and Human-Machine Interaction - An Approach to Cognitive Engineering*. North Holland Series in System Science & Engineering, A.P. Sage.

Van Daele, A. (1993). *La réduction de la complexité par les opérateurs dans le contrôle de processus continus*. Doctoral Thesis. Work Psychology Department, Université de Liège.

Vicente, K.J. (1999). *Cognitive work analysis: Towards safe, productive, and healthy computer–based work*. Mahwah, NJ: Lawrence Erlbaum Associates.