Beware of Agents when Flying Aircraft: Basic Principles Behind a Generic Methodology for the Evaluation and Certification of Advanced Aviation Systems

Denis Javaux, Michel Masson, & Véronique De Keyser

University of Liège

Introduction

There is currently a growing interest in the aeronautical community to assess the effects of the increasing levels of automation on pilots' performance and overall safety.

The first effect of automation is the change in the nature of the pilot's role on the flight deck. Pilots have become supervisors who monitor aircraft systems in usual situations and intervene only when unanticipated events occur. Instead of "hand flying" the airplane, pilots contribute to the control of aircraft by acting as mediators, instructions given to the automation.

By eliminating the need for manually controlling normal situations, such a role division has reduced the opportunities for the pilot to acquire experience and skills necessary to safely cope with abnormal events (Bainbridge, 1987).

Difficulties in assessing the state and behaviour of automation arise *mainly* from four factors:

- the complexity of current systems (e.g., Billings, 1991) and consequent mode-related problems (Sarter & Woods, 1993)
- the intrinsic autonomy of automation which is able to fire mode transitions without explicit commands from the pilots (e.g., Sarter & Woods, 1992)
- the bad quality of feed-back from the control systems displays and interfaces to the pilots (e.g., Norman, 1990 ; Sarter & Woods, 1992), and
- the fact that the automation currently has no explicit representation of the current pilots' intentions and strategy (Onken, 1992 a; 1992 b).

The conjunction of those factors induces a large set of crew-automation interaction problems that pose questions to the current research: difficulties in anticipating computer generated mode changes, difficulties assessing the implications of changes to previously given instructions,

Human Factors Certification of Advanced Aviation Technologies
Edited by J. A. Wise, V. D. Hopkin, and D. J. Garland

difficulties in reacting to unanticipated events and to command changes, difficulties in finding, integrating and interpreting relevant data for situation assessment and difficulties in building extended and refined mental models of how automation is working and how instructions have to be input (Sarter & Woods, 1992).

For pilots, the consequences of those difficulties are an increase in cognitive workload and the development of "unofficial" strategies to override or "hijack" the automation, in an attempt to satisfy "official" goals (Amalberti, 1992).

As a result, *certification is facing a range of new and complex problems* that challenge the aeronautical community to predict and account for all kinds of pilot-automation interaction patterns arising from the introduction of new and sophisticated technologies in cockpits.

> The rapid pace of automation is outstripping one's ability to comprehend all the implications for crew performance. It is unrealistic to call for a halt to cockpit automation until the manifestations are completely understood. We do, however, call for those designing, analysing, and installing automatic systems in the cockpit to do so carefully; to recognize the behavioural effects of automation; to avail themselves of present and future guidelines, and to be watchful for symptoms that might appear in training and operational settings. (Wiener & Curry, 1980) (Mentioned by Billings, 1991, p. 67)

In particular, this paper tries to characterize the added complexity and problems created by the introduction of autonomous agents as (intended as automated resources) in new generations of aircraft.

As an example of the potential for catastrophic consequences of these problems, we would like to refer to the China Airlines B747-SP accident, 300 miles Northwest of San Francisco, of February 19, 1985, using the accident report proposed by Billings:

> The airplane, flying at 41,000 ft. enroute to Los Angeles from Taipei, suffered an inflight upset after an uneventful flight. The airplane was on autopilot when the n. 4 engine lost power. During attempts to relight the engine, the airplane rolled to the right, nosed over and begun an uncontrollable descent. The Captain was unable to restore the airplane to stable flight until it had descended to 9500 ft.
>
> The autopilot was operating in the performance management system (PMS) mode for pitch guidance and altitude hold. Roll commands were provided by the INS, which uses only the ailerons and spoilers for lateral control; rudder and rudder trim are not used. In light turbulence, airspeed increased. As the airplane slowed, the PMS moved the throttles forward but without effect. The flight engineer moved the n. 4 throttle forward but without effect. The INS caused the autopilot to hold the left wing down since it could not correct with rudder. The airplane decelerated due to the lack of power. After attempting to correct the situation with autopilot, the Captain disengaged the autopilot at which time the airplane rolled to the right, yawed, then entered a steep descent in cloud, during which it exceeded maximum operating speed. It was extensively damaged during the descent and recovery (1991, p. 98).

As noted by the author, the NTSB concluded that:

> ... the probable cause was the captain's preoccupation with an inflight malfunction and his failure to monitor properly the airplane's flight instruments which resulted in his losing control of the airplane. Contributing to the accident was the captain's over reliance on the autopilot after a loss on n. 4 engine. The Board noted that *the autopilot effectively masked the approaching onset of loss of control of the airplane.* (ibid., p. 98.).

Without stating too much about the concepts that will be developed in the following sections, yet in contrast to the first elements of analysis retained by the NTSB, this paper claims that this accident's main contributing factors are *flaws in the design* of the information and control systems combined with the presence of *agents* that *operate independently* of any pilot's control action but *without adequate feedback.*

More precisely, as revealed by this incident, the breakdown of the pilot-automation system onboard this aircraft – which is typical of a design currently known as "technology centered automation" – is mainly due to a lack of *controllability* of the automatic systems involved, coupled with a lack of *visibility* and *predictability* of those systems' status, effects and interactions over the considered flight phase, and an engine failure.

Assuming certification has among its major goals to guarantee the passengers' and pilots' safety and the airplane integrity under normal and abnormal operational conditions, the authors suggest it would be particularly fruitful to come up with a *conceptual reference system* providing the certification authorities both with a theoretical framework and a list of principles usable for assessing the quality of the equipment and designs under examination.

This is precisely the scope of this paper. However, if the authors recognize that the conceptual system presented is still under development and would thus be best considered as a *source of reflection* for the design, evaluation and certification processes of advanced aviation technologies.

The Multiple Resources of Automation

> We consider automation to be a tool or resource – a device, system or method by which the human can accomplish some task that might be otherwise difficult or impossible, or which the human can direct to carry out more or less independently a task that would otherwise require increased human attention or effort. (Bilings, 1991, p. 7)

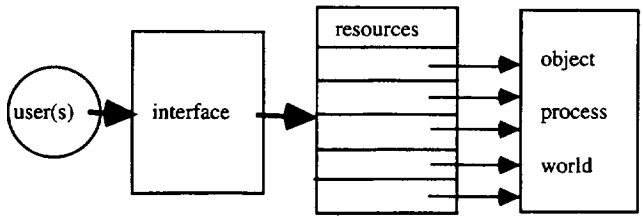


Figure 1. A simplified diagram of automated control of automation

Four components define the classical automated control situation (Figure 1) (e.g., Sheridan, 1988):

- A set of users, operators or pilots with some goals or tasks to achieve

- An object, a process, or a world, characterized by a set of stated variables, upon which the users want to act
- A set of automated resources which possess the capability to change the state of the world on the behalf of the user
- An interface which provides the user with the means to activate and control these resources.

It is clear from everyday life experiences (Norman, 1986) that resources can display very different behavioral characteristics, and that this influences the way we use them as well as the type and amount of knowledge we need to do this efficiently.

The following three essential categories of resources can be identified according to their different behavioral characteristics.

Functions constitute the simpler type of resources and affect the state of the world in a straightforward way. Their effect only depends on the state of the world prior to their activation. Moreover, this effect can be described by a simple state transition: the state of the world before and after the activation of the function. Functions are thus usually extremely predictable resources (e.g., manual seat-belt and non-smoking signs activation, manual throttle control, etc.).

Functional patterns constitute the second type of resource. The behaviour of functional patterns is also only dependant on the state of the world prior to their activation. Nevertheless, contrarily to functions, their effects are not described as simple state transition but as sequences of successive states. Predictability of these patterns is still high, but requires more information than with simple functions (e.g., landing gear extraction and retraction, flaps retraction).

Agents finally are described by sequences of successive states. Nevertheless, with agents sequences are not only influenced by initial conditions but also by conditions varying during execution of the sequences themselves (e.g., agents range from low-level automatisms [attitude stabilizers...] to high-level pilot aiding devices [the Flight Management Systems or the Performance Management Systems]):

> In more automated systems, the level of animacy of *machine agents* has dramatically increased. Once activated, systems are capable of carrying out long sequences of tasks *autonomously*. For example, advanced Flight Management Systems can be programmed to automatically control the aircraft from takeoff through landing. (Sarter & Woods, 1993, p. 6)

As suggested by the previous examples, automated functions, functional patterns and agents are present in most technological contexts. Processes and agents are especially useful in task-intensive situations and have come to progressively replace functions in modern airplanes. Several reasons account for that evolution.

The first is that, as any human operators, pilots are limited in their perceptual abilities (e.g. they cannot fine control the airplane's attitude without assistance or manually react to events in milliseconds) and in their capacities to process information (cf. the classical concepts of

bounded rationality (Simon, 1957)* and short term (Miller, 1956) or working memory (Baddeley & Hitch, 1974; Baddeley, 1986; Reason, 1987) limitations in cognitive psychology).

Some external resources can be introduced to cope with these limitations, but it should be clear that purely functional resources cannot suffice in highly dynamic situations such as piloting an airplane. Because of humans' limited bandwidth I/O channels and because of their limited and rather slow processing capabilities, it is not possible to ensure correct coordination and activation of multiple functional resources. Agents, on the other hand, because they can be considered as functions with autonomy, display that ability to coordinate, at least locally, several specialized functions (Maes, 1989). Agents integrate the logic behind functional integration and activation (acting on the process through functions – see the notion of competence – or recursively through simpler agents).

Producers (airplane designers) and consumers (commercial airlines) have extended the scope of the tasks expected from the global system crew/airplane/ATC. The necessity to enhance safety and performance while flying in highly dense and crowded airspace are among the main motivations for the introduction of agents in airplanes. As a result, the complexity of the flying task has grown to such levels that it has become necessary to extend the perceptive, motor and processing capabilities of the crew. The task itself has been broken down into simpler primitive subtasks that have been allocated to specialized agents.

Thus, there has been a continuous trend in aeronautics to introduce more and more automation into cockpits. However, some problems related to this approach have been described by several human factors experts, like Sarter and Woods:

> New automation is developed because of some payback (precision, more data, reduced staffing, etc.) for some beneficiary (the individual practitioner, the organization, the industry, society). But often overlooked is the fact that new automated devices also create new demands for the individual and groups of practitioners responsible for operating and managing these systems. The new demands can include new or changed tasks (setup, operating sequences, etc.), and new cognitive demands are created as well. There are new knowledge requirements (e.g., how the automation functions), new communication tasks (e.g., instructing the automation in a particular case), new data management tasks (e.g., finding the relevant page within the CDU page architecture), new attentional demands (tracking the state of automation), and new forms of error or failure (e.g., mode error). (1992, p. 17)

This kind of role sharing and interaction pattern has the long term effect of removing the pilot from the loop and decreasing system awareness especially as feedback on automation status and behaviour is poor and difficult to obtain.

While our goals are not to ignore these very important problems, we would like to draw the attention to the problems specifically related to the interfacing of functions and agents in modern aircrafts. We especially believe that some of the problems encountered in modern glass-cockpits

---

* " The capacity of the human mind for formulating and solving complex problems is very small compared with the size of the problems whose solutions is required for objectively rational behaviour in the real world - or even for a reasonable approximation of such objective rationality. "
(Simon, 1957, quoted by Reason, 1987, p. 76).

appear because the "agent" character of some resources has not been sufficiently recognized, and that in some case agents have been interfaced as if they were mere functions.

As will be shown later, the main question behind usable interface design is:

> *How do we provide the user with the necessary knowledge and the means to interact with the resources in order to act and react in the world according to goals or tasks?*

We will first show how the approach adopted by the classical HCI community regarding the interfacing of functions on a static setting has succeeded in its attempts to answer to this question, how it has provided designers with principles to support evaluation, certification and designs methodologies and, in the end, end-users with highly usable forms of interfaces. We will then show how such a strategy could be applied to interface agents in dynamic worlds.

In the end, we will have provided the reader with two sets of principles, respectively for functions and agents interfacing, that could influence the way evaluation and certification of interfaces incorporating these two types of resources are performed.

Interfacing Functions in Static Problem-Spaces: Classical HCI

The now classical domain of Human-Computer Interaction has proven its ability to solve interfacing problems with powerful computerized tools.

Such successes must be related to three factors:

a) cognitive theories of human-computer interaction have been produced

b) some general principles that interfaces have to verify have been defined, either as a subproduct of the cognitive theories of the interaction, or of empirical data (controlled experiments, analysis of errors, etc.)

c) some generic forms of interfaces conforming to these principles have been designed and have received a wide acceptance.

Cognitive Theories of Interaction

Cognitive theories of interaction between users and computers have existed for several years now. Strongly influenced by the early attempts of Artificial Intelligence to produce models of problem-solving and planning (such as GPS, Newell & Simon, 1972), nearly all rely on the same approach and assume that the user achieves goals by solving sub-goals in a divide-and-conquer fashion (Dix, Finlay, Abowd, & Beale, 1993): GOMS (Card, Moran,.& Newell, 1983), CCT (Kieras & Polson, 1985), TAG (Payne & Green, 1986).

The GOMS model, which has served as the basis for major research in cognitive modelling applied to HCI (Dix et al., 1993), considers for example that an interaction situation can be described in terms of Goals, Operators, Methods and Selection.

Goals are the user goals; they are "what has to be achieved".
Operators are the basic operations the user can perform to affect the system state (represented as state transitions).
Methods describe how alternative sub-goals decomposition can help the user to reach the same goal.
Selection rules attempt to predict which methods the user will use to reach goals depending on the user itself and the state of the system.

In such models, the computer is clearly considered as a static setting; that is, one whose state only changes as an effect of the actions of the user considered as the application of operators.

To illustrate how the distinction between a static problem-space and its related operators or functions encounter personal experience, we will analyse how the file management problem is treated on most personal computers.

Interface designers confronted with the file management problem have to define ways to represent files as they appear on some physical support (a hard disk for example) and provide users with the means to manipulate them. Files are usually organised on this support according to a hierarchical structure (a tree). This structure is static; it remains as it is unless the user attempts a modification. Files and directories can be moved, copied or deleted. Individual files can be transformed thanks to applications (word processors, spreadsheets,...) that change their internal structure. All these operations are under control of the user.

The desktop metaphor (Booth, 1989) elegantly solves this problem:

a) *The static problem-space:* the desktop metaphor is a possible alternative to the problem of representing static problem-space. Files and directories are represented by icons or names in lists. Files which are in use are represented by open windows.

b) *Functions or operators:* most of the functions are accessible directly on the desktop (Direct Manipulation Interface; Hutchins, Hollan, & Norman, 1986). File displacement and deletion are operated through moves of the mouse or function activations through menus. Activation of an application on a specific file is possible through double-clicking commands or menus.

General Principles

Thanks to the coherent framework provided by the analogy with problem-solving or planning on static problem-space, it is possible to produce a structured and theoretically sound (contrarily to most of the guidelines) set of principles about properties of usable interfaces.

These principles rely on four underlying ideas.

a) In order to act efficiently on a static problem-space, the user must have access to some knowledge about the problem-space itself and the functions that can be applied.

- The user must be able to assess the *current state* of the problem-space;
- He/she must know which *operators* or *functions* can be applied to this state; and
- What *transition* will occur if an operator or function is applied;

Without this information, the goals cannot be reached (one would say, in terms of problem-solving or planning theory, that the problem cannot be solved).

b) Part of this knowledge is related to the static problem-space and the other part concerns the functions themselves.

c) The knowledge required to interact with static problem-spaces can be distributed within the interface/user system. Well designed interfaces provide the user with a lot of knowledge about the current state of the problem-space (visibility), the functions that can be applied (availability) and the related transitions (predictability). To paraphrase Norman (Norman, 1988) in such interfaces, "information is in the world." In badly designed interfaces, the current state of the problem-space is not visible and a lot has to be remembered (in short-term memory). It is hard to tell which functions can be applied or what will be their effects. In such interfaces, "information is in the head."

d) Principles (necessary principles) that warrant the presence and availability of the necessary knowledge can be stated. Secondary principles, considered less important, can be proposed to indicate how to make the interface more usable or how to support the user in its tasks.

Principles for Static Problem-Spaces

Visibility. Can I see what I need to see? The goal of this principle is to ensure that the user might have a full and accurate representation of the current state of the problem-space.

Interpretability – Do I see what I'm supposed to see? It is not sufficient for the user to have access to a representation of the problem-space. A representation conveys some meaning about some real situation which is abstracted into symbols, and have to be interpreted by the user. This principle ensures that the user correctly interprets the representation. Some simpler but nevertheless essential principles usually support interpretability: consistency or coherence of the symbols and of their interpretation, familiarity and generalizability of the symbols.

Flexibility – May I change the way I see? The possibility of tuning the representations, in particular to modulate the informational flow according to the bandwidth of the human cognitive processing limitations and the particular needs of the current situation, is a specially desirable property of usable interfaces.

Reliability – Is this thing the real picture? This one deals with a critical feature of any interface. It must be reliable, and the user must be confident with the information it provides or the resource it helps to use. When applied to problem-space representation, the reliability principle wonders whether the representation presented to the user constitutes an accurate and reliable representation of the problem-space and how this can be assessed by the user itself.

Learnability – Can I avoid headaches? This principle is important because of the way users accept new products or interfaces is influenced by their learnability. In the case of a static problem-space representation, how easily can the user learn the rules that help to interpret

correctly the representation. Once again, simpler principles such as consistency, familiarity and generalizibility strongly contribute to facilitate learnability.

Principles for Functions

Availability – What can I do? In order to apply functions on the static problem-space as if they were operators, the user must be in a position to decide which functions can be applied on the problem-space. General availability refers to the complete list of functions provided by the interface. Local availability concerns the limited list of functions applicable to specific states of the problem-space. Knowledge concerning both types of availability should be accessible to the user.

Accessibility – How can I do it? Once the user has gained some knowledge about which functions can be applied on the problem-space and has chosen one or a sequence of them to apply, he/she has to specify it for the interface. Knowledge about how to access functions and how to activate them on the correct objects should available to the user. Consistency, familiarity and generazibility are once again among the simpler principles that help the user to access functions.

Predictability – What will happen? Predictability is without any doubts *the* essential principle to conform to. In problem-solving and planning models, the ability to predict how the state of the world will change when an operator iterface with a machine is crucial for resolution or task satisfaction. The user must possess the necessary knowledge to be able to generate plans or sequences of actions on the interface that in the end will meet its goals. Modes, if any, have to be made visible to the user because they influence, by definition, the way functions behave and thus constitute a threat to predictability.

Feedback – How is it working and what are the effects? Feedback is essential because it permits the user to assess that the intended state has been reached, and hence that the activated function has been applied correctly. Feedback is thus associated to error detection, but also to the ability to learn (see learnability principle) the necessary knowledge to predict functional effects (see predictability principle). Two forms of feedback are usually encountered. The first type concerns the visibility of the function status (progression bars) and help to confirm that the access to the function has been successful (see accessibility principle). The second type of feedback ensures that the effects of the activated functions are visible. In the strictest sense, this second form of feedback is more concerned with visibility of the representations, and is thus not a pure functional principle.

Controllability – How the hell do I stop this thing from erasing my hard disk? As dramatically stated by the previous sentence, controllability is a particularly desirable feature. Nevertheless, in general, interfaces provide a very limited set of interactions between a running function and the user (otherwise, it would be an agent). Control is usually limited to interruption (either temporary or definitive) of the function execution.

Flexibility – Can I do it the other way? Users are not machines, and they are faced with very variable tasks. Moreover, users all differ. They have different backgrounds, different cognitive

styles, and usually different goals. For such reasons, while not resorting to the major, necessary principles, flexibility is generally appreciated by users.

Automatibility – Can I automate this sequence of operations? There are two facets to automatibility whose advantages are obvious. Machine-supported automatibility refers to the possibility for the user to define "macros," automated sequences of functional activations, with or without parameters. Cognitive automatibility concerns the ability of the user to automate the motor and cognitive processes that support its access to functions. This form of automatibility is strongly conditioned by good visibility of the problem-space and easy and consistent access to functions.

Task Conformance – Does it really cover all my needs? This principle concerns the scope of the available functions regarding the nature of the task they are to perform. It can be considered from a general point of view (the global availability) or more locally according to specific situations (the local availability compared to the local task): i.e., is the function available when needed?

Error Management – What if I err? Users are fallible (Reason, 1990). Good interfaces have to take this into account and exhibit error resistance (prevent users to make errors, e.g. Masson & De Keyser, in press) and error tolerance (help users to correct effects of errors through reversibility, escapability, recoverability of function applications).

Reliability – Is this stuff really working as it is supposed to? While being extremely reliable systems, modern computers are nevertheless mere material human artifacts and consequently suffer from design errors as well as from the usually hidden effects of the second-law of thermodynamics. At the software level, bugs are present in any complex application. At the hardware level, problems and troubles sometimes occur due to heat, dust, fatigue or even failure of a component. Interfaces should furnish the user with means to ensure that the functional resources effectively affect the state of the problem-space as reflected by its representation and the different feedback mechanisms.

Learnability – Can I avoid headaches? Learnability of functions is essential. As already stated for problem-space related principles, it is generally a necessary condition for the acceptance of an interface. Several aspects can be learned and thus lead the user to eliminate exploratory or information seeking behaviors. Every piece of the necessary knowledge related to the primary principles (availability, accessibility, predictability) can be learned. Some rules that help the user to deduce such essential pieces of information from the representation of the problem-space can also be abstracted and then greatly contribute to simplify the activation of the functional resources; hence the pervasive character of the consistency principle.

Generic Forms of Interfaces

Classical HCI has also succeeded in its attempts to apply these principles to interface design. Graphical user interfaces, and especially WIMP (windows, icons, menus and pointers) interfaces (Dix et al., 1993), which constitute the standard interface for interactive computer systems (Macintosh, Windows-based IBMs and compatibles, desktop workstations) have proven their usability to millions of end-users.

Such kinds of interfaces indeed provide users with an excellent visibility over the current state of the problem-space (e.g., the desktop of the Macintosh) as well with consistent and familiar rules to interpret the representation (the desktop metaphor). Users habitually have the opportunity to tune these representations (e.g., different ways to display files in a directory) and this contributes to the interface flexibility. Moreover, such interfaces are highly learnable, especially because of their coherent and metaphoric nature.

GUIs and WIMPs equally perform at their best regarding functions. Availability is usually very well documented by the interface. This is a least true of the most used functions. Less common functions are not well known to users, especially in case of very powerful tools such as word-processors that provide users with hundreds of functions. Accessibility is extremely good, thanks to the mouse and its clicking commands and to menus (that also contribute to availability). Predictability is good (at least for simple operations on the desktop) because of the coherence of the access rules and the already quoted metaphoric nature of the interface. Feedback is immediate, but restricted to objects visible on the desktop. Controllability is limited, but it is enhanced for functions that have destructive effects on the desktop. Flexibility is usually good, thanks to the several different ways to perform operations (directly on the desktop or through menus). Macros are provided as default features or can be added thanks to dedicated applications. Task conformance is the principle where these graphical interfaces are at their worst: the possible scope of what can be done is somehow limited, especially if compared to very powerful command-languages (e.g. Unix) dedicated to files management. Errors are handled differently by the manufacturers of common GUIs. Operations that imply a displacement of files between two places (a move operation) can generally be undone, but file deletion is sometimes an operation that cannot be reversed without specifically dedicated tools. Learnability, finally, is usually extremely good (perhaps it is in the end the main reason for the success of these interfaces within a computer-illiterate population) especially because of the so-praised consistency of the interface (even between applications) and the metaphor of the desktop.

Interfacing Agents in Dynamic Problem-Spaces: HCI Goes to the Real World

We have carefully analysed the approach followed by classical HCI to solve the problems related to the interfacing of functions in static problem-spaces. Now we would like to see how such a strategy can be applied to interface agents in dynamic problem-spaces. According to other authors (Kay, 1990; Laurel, 1990), the interfacing of agents is *the* challenge of tomorrow's HCI. We already have shown how agents constitute invaluable resources for users, operators or pilots in their respective tasks. That is why manufacturers and designers have introduced them at several different levels of automation used in process control.

Cognitive Theories of Interaction

There has been for a few years an emergent interest about ideas related to the integration of a distributed work or processing force into a coherent and goal-oriented whole. Computer Science, for example, has already produced numerous formal studies about parallel processing

and synchronization problems. Distributed Artificial Intelligence (DAI) aims designing systems or societies of agents (multi-agent systems) that collectively exhibit the ability to solve complex problems with more robustness than classical approaches (Maes, 1990). On the linguistic side, Winograd and Flores (Flores, Graves, Hartfield, & Winograd, 1988) have developed linguistic-based theoretical perspectives for analysing group actions. Coordination Theory (Malone & Crowston, 1990) as a general and abstract theory tries to establish the connections between several different disciplines that are concerned with similar coordination phenomena. On the applied side, Computer Support to Cooperative Work (CSCW) is aiming at providing organizations or groups of users with better ways and tools to work together (Dix et al., 1993). As demonstrated by the next excerpts, concerns about agents and modelling human/agent interaction have even been expressed in aeronautics by human factors authors.

> Pilots were surprised when the aircraft did not respond as expected; they did not realize or understand why their instructions to the automation had not resulted in the desired change. In some sense, this is a good example to show how pilots try to communicate with the system in a way analogous to communication with another human agent. They assume that entering the desired target value is sufficient for the system (as it would be for a human crew member) to understand that it is supposed to achieve this new target and how it is supposed to do so in detail. (Sarter & Woods, 1993, p. 12)

> (This) direction is to consider supervisory control of automated resources as a kind of cooperative or distributed multi-agent architecture. (Sarter & Woods, 1993, p. 12)

Despite these efforts and remarks, there is nothing today like a single and coherent body of theory about coordination between agents (Malone & Crowston, 1990), and it hard to think of any integrated cognitive theory of interaction between humans considered as agents, or between humans and automated agents. Nevertheless, there is more and more awareness of the similarities between the problems encountered by researchers involved in these approaches to cooperative systems as is witnessed by the increasing number of workshops or conferences on the topic. On the cognitive side, expectations about future progress will rely on domains such as social or developmental cognitive psychology as well as psycholinguistics to produce a coherent and integrated theory of human interaction with agents.

General Principles

Designers faced with the problem of interfacing agents are still left without the sort of powerful framework they used to rely on when designing functional interfaces. Nevertheless, some important principles that interfaces with agents should verify can already be stated, thanks to extensions of the basic principles for functional interfaces, reflections about the necessary knowledge required for usable interaction, and to recommendations formulated by analysts when incidents with such interfaces were reported. We will thus try to rely on these excellent studies of problems and incidents encountered with automation in modern glass-cockpits as major sources for defining general principles.

On the epistemic side, it is at least clear from a formal point of view that more knowledge (distributed between the user and the interface) is needed to control a dynamic problem-space than a static one. Anticipatory behaviors, of which some researches have shown the heuristic value (Van Daele, 1992), are only possible if the user, operator or pilot has some knowledge or

ability to predict how the controlled system will naturally evolve if no action is taken. Interfaces to dynamic worlds or problem-spaces should provide the user with such knowledge or resource (see the predictability principle for dynamic problem-spaces).

More knowledge is also needed to interact with agents than with functions. Agents can be of numerous different types and differ in terms of complexity (ranging from reactive to cognitive agents; Erceau & Ferber, 1991). Whatever the importance of such factors, the main difficulty with agents certainly comes from their flexibility (complex agents can exhibit different behaviours in similar situations) and from their autonomy (agents incorporate their own logic behind functional activation and act autonomously on the world). As a consequence, agents must be considered as generally less predictable resources than functions.

Respective Scopes or Competences and Cooperative Modes. A supplementary and rather essential distinction must be introduced before devoting some attention to the principles. It concerns the distribution of competence between the user, the operator or pilot and the agent. To use a multi-agent terminology, one would say that only two cooperation modes are possible: either the job is done by the function or it is done by the user. The situation is quite different with agents. Because such resources display possibilities for an extended amount of controllability these resources provide the capability for more complex cooperation modes.

Three classes of cooperation modes have to be considered:

a) The job is done by the user. The agent is not active or works on another task.

This corresponds to the concept of "direct manual control."

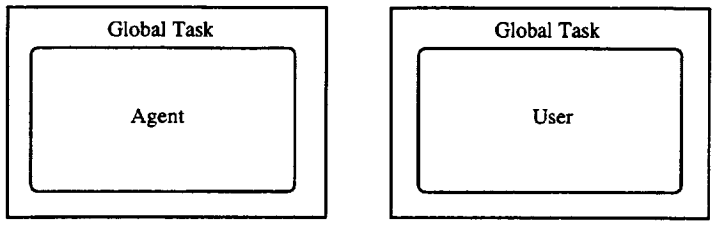
According to Billings (1991, p. 27) *direct manual control* is characterized by the pilot's direct authority over systems, manual control using raw data, unaided decision making and manual communications.

However, as pointed out by the author, no modern aircraft can be operated entirely on that mode. "Indeed, an aircraft operated even by direct manual control may incorporate many kinds of control automation, such as yaw dampers, a pitch trim compensator, automated configuration warning devices, etc." (Billings, 1991, p. 26). For example, landing gear retraction and extension are still manually operated in all transport aircrafts.

b) The job is done by the agent. The user is not active or works on another task.

This is precisely the meaning of the "autonomous operation" concept. As summarized from Billings, autonomous operation is characterized by the fact that the pilot has no role to play in operation, that the pilot has normally no reason to intervene and that the monitoring is limited to fault detection (Billings, 1991, p. 26).

Until the introduction of the A320 and MD11, very complex systems were operated in a full autonomous fashion. In those new aircraft, however, major systems operate this way. For example, in the MD11, failure detection and subsystem reconfiguration are performed autonomously. Longitudinal stability and control wheel steering are also autonomous operations.



(a) agent fully in charge (b) user fully in charge

Figure 2: Agent and User interaction.

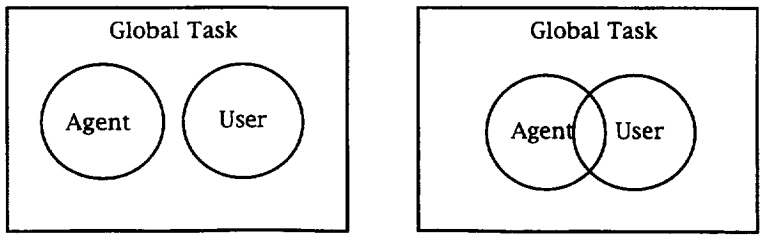
c) The job is done by both the user and the agent. Each of them has its own part of the task. Two situations have to be distinguished:

1) The two tasks are exclusive. This occurs for example when the agent and the user work on two different sub-systems. If the two tasks are interdependent (the two sub-systems interact) then the agent and the user have to synchronize their actions.

An example of such a sharing pattern is given by Billings: "the pilot may elect to have the autopilot perform only the most basic functions: pitch, roll and yaw control...he or she may direct the automation to maintain or alter heading, altitude or speed, or may direct the autopilot to capture and follow navigation paths, either horizontal or vertical...In all cases however, the aircraft is carrying out a set of tactical directions supplied by the pilot. It will not deviate from these directions unless it is capable of executing them." (1991)

2) The two tasks share a common part. This could occur when the agent and the user do work on the same sub-systems. In such cases, conflicts are likely to arise, and resolution techniques have to be provided.

For example, in the 320, the flight control system incorporates an envelope limitation system that operates at all times and interacts with pilot's commands, in order to guarantee that safety barriers are not overcome. For example, bank angle, pitch and angle of attack cannot be exceeded by the pilot unless the flight control computer is turned off.



(c.1) exclusive scopes (c.2) common scopes

Figure 3: Agent and User scopes.

Moreover; cooperation modes with agents cannot solely be considered in a static perspective (they are fixed and cannot be changed over a task session or a flight). Dynamic mode changes are also observed in modern cockpits (modes change over the course of a task session, either through agent or user instruction).

> An important characteristic of *automatic flight-path control* is the high degree of dynamism. Transitions between modes of control occur in response to pilot input and changes in flight status. Automatic mode changes can occur when a target value is reached (e.g., when levelling off at a target altitude), or they can occur based on protections limits (i.e., to prevent or correct pilot input that puts the aircraft in an unsafe condition). (Sarter & Woods, 1992, p. 306)

For such reasons, as stated by Billings, feedback (see the feedback principle) should be given to the user or pilot whenever an important mode change occurs.

> Automation should never permit a situation in which "no one is in charge"; pilots must always "aviate" even if they have delegated control to the autopilot. It is for this reason that autopilot disconnects are usually *announced* by both visual and aural alerting signals (Billings, 1991, p. 85).

To confirm the importance of issues related to cooperation modes, Sarter and Woods also describe how the pilot's inability to dynamically change modes can lead to some drastic measures.

> During the final descent, the pilots were unable to deselect the APPR mode after localizer and glideslope capture when ATC suddenly requested that the aircraft maintain the current altitude and initiate a 90° left turn for spacing. They tried to select the ALT HOLD and HDG SEL modes on the MCP to disengage the APPR mode and comply with the clearance, but neither mode would engage and replace the APPR mode. They finally turned off all autoglide systems (Sarter & Woods, 1992, p. 311).

This leads us to some critical remarks about the way evaluation or certification of agent-based interface relying on principles should be performed.

- The analysis should begin with a very careful study of the possible cooperation modes between the user and the agent.

- It should detail *who is in control* of
  - The cooperation mode changes
  - The relative scopes of the user and the agent within a given cooperation mode (task migrability).

- For each possible cooperation mode, consider how the duality user/agent is positioned according to the principles. However, due to the very different ways the task is conducted in different cooperation modes, principles have to be applied with some nuances in mind and be related to the current specificities of the current mode.

In a short-response time agent (a regulator), whose capabilities are far beyond those of the pilot, the cooperation mode is such that the task is exclusively under the agent control. The main principles in this situation are a) reliability, and b) the capability for the pilot to assess that the agent is working in its competence domain. Principles such as predictability, that used to be essential for functional resources, are hereby not necessary (e.g. the gyroscopic stabilizer of Maxim, 1891, the stability augmentation system of Wright, 1907, and their successors in modern autopilots).

Principles for Dynamic Problem-Spaces

Visibility – Can I see what I need to see? Visibility of the problem-space acquires hereby a special status due to its dynamicity. In static problem-spaces where the world does not change spontaneously, the user or pilot can rely on short-term memory to maintain awareness and orientation. In dynamic problem-space, updating is necessary and this is only possible through predictions or observations of the future states. In a particularly complex dynamic context with heavy task constraints, the concept must even be extended to meet the notion of "situation awareness" (Sarter & Woods, 1991). In such situations, it is not enough to provide the user with the means to perceive the state of the problem-state, but also to ensure that it will be *attended.*

> Situation awareness has recently gained considerable attention as a performance-related psychological concept. This is especially true in the aviation domain where it is considered an essential prerequisite for the safe operation of the complex dynamic system "aircraft." There are concerns, however, that inappropriately designed automatic systems introduced to advanced flight desks may reduce situation awareness and thereby put aviation safety at risk. (Sarter & Woods, 1991, p. 45)

The problem of the amount of information that must be visible is also addressed by Billings:

> "How much information is enough? How much is too much?" Though pilots always want more information, they are not always able to assimilate it. (Billings, 1991, p. 46)

As pointed out by the author, such a question should be answered (as suggested above) according to a clear consideration of the cooperative mode between the pilot and the agent and their respective involvement in the control task.

> Pilots need much less information when subsystems are working properly than when they are malfunctioning (Billings, 1991, p. 46)

Interpretability – Do I see what I'm supposed to see? No significant difference with its functional counterpart.

Flexibility – May I change the way I see? Flexibility applied to the representation of the dynamic problem-space means that the user or pilot is capable of adapting this representation to its current or future needs. Such a possibility is present in several subsystems displays (zooming features) of glass-cockpits.

Predictability – What will happen if I stop acting? The new principle is based as already stated on the heuristic value of predictory or anticipatory behaviours in most dynamic control situations (Van Daele, 1992). Good interfaces for dynamic problem-spaces should provide the users with means to anticipate future states. Several examples of such an approach already exist in aeronautic contexts. On ATC radar control screens, airplanes can be represented with small tails indicating their speed and direction. This helps operators to anticipate their future trajectory. On TCAS screens, vertical speeds of surrounding airplanes are represented by small arrows. In general any graphical representation of a trend indicator can contribute to the predictability of the dynamic problem-space.

Reliability – Is this thing the real picture? The problem of reliability related to dynamic problem-space is perfectly stated by Billings:

> It must be kept in mind that sensors, processing equipment or display generators can fail, and that when incorrect information is presented, or correct information is not presented, there is the potential for confusion in the minds of pilots. (1991, p.40)

An interface on a dynamic problem-space should help the user to ensure that it functions correctly, both in its ability to display correct and accurate information about the real state of the monitored systems and in its ability to inform the user about future states (support to predictability principle). Redundancy of display equipments or availability of displays related to interdependent subsystems can help the user or pilot to ensure that the informational interfaces are functioning correctly thanks to comparison or inter-display coherence checking.

Learnability – Can I avoid headaches? Here, again, there is no significant difference with functional counterpart.

Critical State Assessment – Is this the right place to stand still? This new principle concerns the peculiar problems associated with dynamic problem-spaces. In such spaces, states are rarely equivalent. Some of them require special care or attention, either because the monitored system ventures within space regions where irreversible damages could be observed, or because its intrinsic dynamic could lead the user or pilot to lose control. Interfaces provide critical state assessment support users and help them to enhance their performance.

Principles for Agents

Availability – What can I do? Availability as such refers to the capability the user has to decide whether a resource exists and is available. Users or pilots should be informed of the different agents they might use as resources (global availability) as well as when these can effectively be used (local availability), and in which cooperative modes.

Accessibility – How can I do it? Users or pilots willing to use an agent as a resource should have access to some knowledge about how to activate and configure it in the cooperative mode of their choice (if they are in control of this variable).

Predictability – What will happen? Predictability is, without a doubt, one of the principles that must be considered with extended caution when trying to interface agents. As previously stated, agents are autonomous systems. They consequently present less predictable behaviours than

functions. Numerous examples of incidents related to a lack of predictability of some agents in glass-cockpits have already been reported. Sarter and Woods (1992) have realized a study through a questionnaire asking pilots to describe instances in which FMS behaviour was surprising, and to report modes and features of FMS operations that were poorly or not understood. 135 B-737-300 line pilots from an airline company participated to that survey (Sarter & Woods, 1992).

> Pilots indicated that the algorithms underlying the calculation of a VNAV are not transparent to them. They cannot vizualize the intended path; therefore, they are sometimes unable to *anticipate* or *understand* VNAV activities initiated to maintain target parameters...Several pilots reported that they have been *surprised* by VNAV when it failed to start the descent on reaching the top-of-descent (TOD) point...

The problem the user or pilot is faced with is one of agent modelling. Designers must ensure that they provide the user with a correct model of the agent. Two principal classes of models govern the theories about agents: mechanistic models and intentional models. In mechanistic models, the user relies on a kind of finite-state automaton approximation of the agent whose behaviour can be predicted thanks to the awareness of relations between some internal parameters or variables of the agent, the input it is actually processing and the resulting behaviour. In intentional models of agents, the user predicts future behaviors of the agent on the base of its goals or intentions. Consequently, and whatever the type of model hold by the user (depending on the type and complexity of the agent), it seems essential that any important autonomous change that might modify the way the agent will behave in the near future (a change of mode in mechanistic models or a change of goals or intentions in intentional models) is reported to the user.

Scope or Competence Awareness – What can this thing do and when? This new and essential principle concerns the competence of the agent: what it (can) do and in which circumstances. With purely functional resources, competence awareness is close to predictability. Functions induce simple state-transitions (what is does) from the states upon which they apply (in which circumstances). Due to the extended flexibility and autonomy of agents, this similarity does not appear and a new principle has to be introduced. Scope awareness is extremely important, at least when the user or pilot is in control of the cooperation modes and of task migrability: the pilot must be able to assess that the agent is performing reliably (reliability principle) and correctly (adapted to the task) in its domain (scope awareness) of competence. Designers must consequently provide the user or pilot with this knowledge through the interface, documentation, and/or training courses.

Feedback – How is it working and what are the effects? This is another essential principle for agents. Due to the new problems introduced with predictability of the agent and the correlated needs to model its behavior, the visibility of agent status increases in importance. As already reported, mode awareness (in mechanical models) is a condition for real cooperative work between the user or pilot and the agent.

> Pilots reported that they are surprised by uncommanded mode transitions that occur on reaching a target state or for protection purposes. Most often, the reports referred to the automatic reversion from vertical speed mode to LVL CHG mode, which occurs if the airspeed deviates from the target range due to an excessive rate of climb or descent.

> Pilots' reports seem to indicate that such uncommanded changes are difficult to track given current cockpit displays and indications. (Sarter & Woods, 1992, p. 311)

Visibility of the agent's effects are of equal importance. Because agents display autonomy, any change introduced on the dynamic problem-space by the agent should be reported or be at least visible to the user, especially in cooperative modes where both the agent and the user are in charge of the same task. It is also due to this visibility that the adequacy of the decision to activate the agent as well as its reliability can be assessed. See also Billings (1991, p. 85) and the concept of "fail-passive" control automation situations that describe hazardous conditions where visibility of the agent effects is lowered.

Controllability – How the hell do I stop this thing grounding my airplane? Because of the autonomy of agents, and their ability to induce disastrous effects on the controlled problem-space, controllability should remain high in every circumstances. Woods describes how "clumsy automation" can contribute to lower controllability in circumstance where it is especially needed.

> Clumsy automation is a form of poor coordination between the human and machine in the control of dynamic processes where the benefits of the new technology (i.e. additional tasks, forcing the user to adopt new cognitive strategies, new communication burdens, new attentional demands) occur during periods of peak workload, high criticality or high tempo operations. (Cook et al., 1990 ; Sarter & Woods, in press)
>
> Significantly, deficits like this can create opportunities for new kinds of human error and new paths to system breakdown that did not exist in simpler systems. (Woods, Cook, & Sarter, 1993) (Woods, 1993, p. 2)

It is a clear evidence that users or pilots should have the capability to disengage automation (agents) or at least change the current cooperative mode to some mode where they are more involved whenever they think it is needed. Billings states this very precisely:

Premise

The pilot bears the ultimate responsibility for the safety of any flight operation

Axiom

The human operator must be in command

Principles of Human-Centered Automation (extract). (Billings, 1991, p. 12)

The same author expresses serious concerns about recent examples of violation of such principles. The flight control system of the A320 and its envelope limitation operate at all times: they cannot be disengaged by the pilot. In the MD-11, major aircraft systems operate autonomously to a large extent:

> ...(civil aircraft) do on occasion have to take violent evasive action, and they may on extremely rare occasions need control or power authority up to (or even beyond) structural and engine limits to cope with very serious failures. The issue is whether the pilot, who is ultimately responsible for safe mission completion, should be permitted to operate to or even beyond airplane limits... (Billings 1991, p. 29)

Error Management – What if I err? As pointed by Billings, system operation errors are responsible for roughly two-thirds of air carrier accidents (1991, p. 24). It thus mandatory, as for functions, to design error-resistant and error-tolerant agent interfaces that attempt to minimize the effects of human error. Monitoring capabilities into the automation, system envelope limitations and procedural control are among the currently investigated techniques to enhance safety.

Task Conformance – Does it really cover all my needs? Here again, there is no significant difference with functional counterpart.

Flexibility – Can I do it the other way? The multiplicity of ways a given resource can be used is usually a rather desirable feature, especially because it provides the user or pilot with a choice within several different strategies to achieve the same goal.

> For example, an automated cockpit system such as the Flight Management System (FMS) is flexible in the sense that it provides pilots with a large number of functions and options for carrying out a given flight task under different circumstances. There are at least five different methods that the pilot could invoke to change altitude (Sarter & Woods, 1993, p. 2).

However, with complex cooperative agents, flexibility can strongly contribute to the "clumsiness" of automation and lead to very serious problems, as is witnessed by the same authors:

> This flexibility is usually portrayed as a benefit that allows the pilot to select the mode best suited to a particular flight situation. But this *flexibility has a price*: the pilots must know about the functions of the different modes, how to coordinate which mode to use when, how to "blumplessly" switch from one mode to another, how each mode is set up to fly the aircraft, and he has to keep track of which mode is active. These new cognitive demands can easily congruate at high tempo and high criticality periods of device use thereby adding new workload at precisely those time periods where practitioners are most in need of effective support systems.
>
> *Clumsy use of technological possibilities,* such as the proliferation of modes, creates the potential for new forms of human-machine system failure and new paths towards critical incidents, e.g. the air crashes at Bangalore (e.g., Lenorovitz, 1990) and Strasbourg (Monnier, 1992) (Sarter & Woods, 1993, p. 2).

Reliability – Is this stuff really working as it pretends? The reliability principle is extremely important with agents, especially because of their capability for autonomy and of the corresponding tendency of users to rely blindly on them.

As an example of overconfidence in automation, we would like to mention the accident of Scandinavian Airlines DC-10-30 occurred at Kennedy Airport on February 2, 1984. In this

accident, the airplane touched down 4700 ft beyond the limit of an 8400 ft runway, was then steered to the right and landed in water 600 ft beyond the runway. The accident was due mainly due to a failure of the throttles to respond to the autothrottle speed control system commands and to the excessive confidence of the Captain in the reliability of that autothrottle system, in spite of a one month history of malfunctions. As noted by the NTSB among other causes, the "performance was either aberrant or represents a tendency for the crew to be complacent and over-rely on automated systems" (quoted by Billings, 1991, p. 99).

As pointed out by Billings, the ability to assess reliability is related to visibility of the problem-space, and to predictability of the agent behaviour (either based on mechanical or intentional models).

> It is thus necessary that the pilot be aware both of the function (or dysfunction) of the automated system, and of the results of its labors, on an ongoing basis, if the pilot is to understand why complex automated systems are doing what they are doing (Billings, 1991, p. 83).

However such a strategy might fail simply because of the stable condition the controlled process is in. Automatic monitoring of the agent reliability and visibility on its status is badly needed in such situations.

> "Fail-passive" control automation represents a particular potential hazard, in that its failure may not change aircraft performance at the time if the airplane is in stable condition. Such failures must be announced unambiguously to insure that the pilots immediately resume active control of the machine (Billings, 1991, p. 85).

Learnability – Can I avoid headaches? As with functional interfaces, comments must be made about the strong relation between the learnability of agent interfaces and their success measured in term of acceptance by users as means to access the full capabilities of the resources, in a safe and error-free fashion, and without the side-effects (clumsy automation, shift or loss of expertise, etc.) usually observed. Given the amount of knowledge that must be learned to interact cooperatively with an agent (this point will be developed later), learnability of agent interface *must* be (very) high. A few possible solutions will be described in the section about generic interfaces design.

Generic Forms of Interfaces

To begin with interfaces, some special points must be made about the amazing amount of knowledge required to interact fruitfully with agents. Users or pilots must be educated about the availability of agents (when they can be used), their accessibility (how they can be used), their scope or competence (what they can and can not do), their predictability (how they will behave or act on the problem-space under control), and the related mental models of their functioning, and finally about their controllability (how can they be controlled). Moreover, they must develop skills or mental processes dealing with how to communicate with them, how to evaluate their reliability through predictability and visibility of the problem-space, how to use or require feedbacks to enhance predictability itself, how to manage errors when they occur, etc.

To gain this knowledge or develop the means to access it is an extremely important task user/pilot face (hence, the "clumsy automation" problems and shift of workload toward more cognitive tasks reported by many authors). Moreover, to add to the task, the knowledge is required for each agent the user or pilot is interfaced with!

Our claim is that many problems described in modern glass-cockpits could be avoided if these simple – but overwhelming – considerations were taken into account.

A possible and promising solution, as already demonstrated with function interfacing through Direct Manipulation and metaphoric interfaces, is *to provide the user with a lot of the necessary knowledge* embedded *in the interface itself and with the means to* extract *it whenever needed*. Whether such interfaces should rely on graphical DMI-type interfaces or even more futuristic solutions (e.g., virtual realities) remains an open question.

A second and complementary approach is to reduce the *amount of knowledge* required to interact with agents. This is especially true at the level of the cockpit considered as a global work environment (or *macro-interface* with functions, functional patterns or agents provided as resources to interact with the airplane, the airspace and the ATC). Introducing intra- and inter-agent coherence into cockpits seriously contributes to limit the necessary knowledge to use them: agents can be classed according to the kind of cooperative modes they entertain with the crew and coherent communication protocols, feedback techniques and support to mental modelling can be established. The current situation with cockpits might be similar to the situation of interactive computers prior to the introduction of coherent GUIs, when every application had its own way to interact with the user.

Another important issue already considered by designers as decreasing the amount of knowledge that is not intuitive is familiarity. Thanks to the introduction into cockpits of more "natural" cooperative and communication modes (e.g., multi-modal and multi-media), the everyday life experience of interaction situations could be made more usable.

Conclusion

The influences of the introduction of new and sophisticated automation technologies in the last generations of commercial aircraft regarding the pilots-systems interactions has been extensively described by numerous experts in aeronautics and human factors engineering.

Technology allows a proliferation of interaction possibilities with increasing level of automation autonomy and poor feedback capabilities. These changes create new cognitive demands for the pilots, demands that turn to be he highest precisely during the most critical flight phases, where one would have expected the automation to be of the highest utility (Sarter & Woods, 1993. See also Moll van Charente et al., 1992, for similar results in the medical domain).

In summary, the complexity and lack of transparency of current automation challenge the pilot's ability to *cooperate* with the sophisticated systems he is provided with. At least three sets of measures can be explored to tackle the difficulties showed between current technologies and designs. The first set of measures would aim at improving the crew-automation interface as suggested above. A second approach to improve the quality of the cooperation is to decrease the cognitive demand on the pilot. More "natural" cooperative and communication modes are considered by cognitive psychologists as rather "effortless" processes, thanks to the many years spent to learning and automate them to interact with other humans. Improving mutual

models of each other (it reduces the need to communicate), increasing reliability and the means to assess it, giving agents the possibility awareness of their own scope or competence, or providing dynamic feedback for important modes or intentional changes (e.g., Billings, 1991; Onken, 1992; Sarter & Woods, 1993) are among the several paths designers follow. The third set of measures is to conceive of the interactions between the pilots, the various automated resources, and even the ATC and other airplanes as a distributed cooperative multi-agent architecture in which each partner is engaged, in collaboration with all other agents, in the pursuit of a common system goal.

To sketch the current problems encountered with the "technology-centered automation," Wiener (1989) reports that the most common questions asked by pilots in glass cockpits are: "what is it doing?", "why did it do that?" and "what will it do next?"; to which Sarter and Woods (1993) add: "how in the world did I ever get into that mode ?"

We believe that all those interrogations could be reinterpreted in the light of the concepts and methodology developed in this paper.

According to the analysis made on the effects of current automation in cockpits, we suggest to extend that list by adding: *"how can I coax agents into performing what I want them to?"*

But as we have tried to highlight, this might not the right way to envisage operator-automation interactions. We here suggest that a shift in view could be fruitful, which would envisage both human and artificial agents as collaborative partners. And new technologies should facilitate that shift.

The question to be asked should rather be: "how can *we together* perform the missions I am in charge of ?"

Facing that new complexity, we suggest that the certification of future equipment and designs could benefit from a systematic methodology aimed at identifying the most critical problems in pilot-automation interactions. This paper constitutes one attempt to come up with such a methodology.

References

Amalberti, R. (1992). Safety in process-control: An operator-centred point of view. *Reliability Engineering and System Safety, 38*(313), 99-108.

Baddeley, A. D., & Hitch, G. (1974). Working memory. In G. H. Bower (Ed.) *The Psychology of Learning and Motivation, 8*. London : Academic Press.

Baddeley, A. D. (1986). *Working memory*. Oxford : Oxford University Press.

Bainbridge, L. (1987). Ironies of automation. In J. Rasmussen, K. Duncan, & J. Leplat (Eds.), *New technology and human error*. United Kingdom: John Wiley and Sons Ltd.

Billings, C. E. (1991). *Human-Centered Aircraft Automation*. NASA Tech. memo N° 103885. Moffett Field, CA : NASA-Ames Research Center.

Booth, P. (1989). *An Introduction to Human-Computer Interaction*. Lawrence Erlbaum Ltd.

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

Cook, R. I., Woods, D. D., & Howie, M. B. (1990). The natural history of introducing new information technology into a dynamic high-risk environment. In *Proceedings of the Human Factors Society, 34th Annual Meeting*.

Dix, A., Finlay, J., Abowd, G., & Beale, R. (1993). *Human-Computer Interaction.* Prentice-Hall International. Cambridge.

Erceau, J., & Ferber, J. (1991) *L' Intelligence Articielle Distribuée.* La Recherche. Volume 22. Juin.

Flores, F., Graves, M., Hartfield, B., & Winograd, T. (1988) *Computer Systems and the design of organizational interaction. ACM Transactions on Office Information Systems,* 6(2), 153-172.

Hutchins, E. L., Hollan, J. D., & Norman, D. A. (1986). Direct Manipulation Interfaces. In D. A. Norman and S. W. Draper (Eds.), *User Centered System Design,* pages 87-124. Lawrence Erlbaum Associates, New Jersey.

Kay, A. (1990). User Interface: A personal view. In B. Laurel (Ed.) *The Art of Human-Computer Interface Design.* Addison-Wesley Publishing Company.

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

Laurel, B. (1990). Interface Agents: Metaphors with Character. In B. Laurel (Ed.) *The Art of Human-Computer Interface Design.* Addison-Wesley Publishing Company.

Lenorovitz, J. M. (1990). Indian A320 crash probe data show crew improperly configured the aircraft. *Aviation Week & Space Technology, 132 (6/25/90),* 84-85.

Maes, P. (1989). *How to de the right thing.* A.I. Memo N° 1180. Massachussets Institute of Technology, Artificial Intelligence Laboratory. December 1989.

Malone, T. W., & Crowston, K. (1990). What is Coordination Theory and How Can It Help Design Cooperative Work Systems ? *Proceedings of CSCW 90 conference.*

Masson, M., & De Keyser, V. (in press). Preventing Human Error in Skilled Activities trough a Computerized Support System. *Proceedings of HCI International '93, 5th International Conference on Human Computer Interaction,* August 8-13, Orlando, FLO.

Miller, G. A. (1956).The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review, 63,* 81-93.

Moll van Charente, E., Cook, R. I., Woods, D. D., Yue, L., & Howie, M. B. (1992). Human-computer interaction in context: Physician interaction with automated intravenous controllers in the heart room. *Proceedings of the Fifth IFAC/IFIP/IFOEA/IEA Symposium on Analysis, Design and Evaluation of Man-Machines Systems.* De Hague, the Netherlands.

Monnier, A. (1992). R*apport préliminaire de la commission d'enquête administrative sur l'accident du Mont Sainte* Odile du 20 janvier 1992.

Newell, A. & Simon, H. A. (1972). *Human problem solving.* Englewood Cliffs, N.J. : Prentice Hall.

Norman, D. A. (1986). Cognitive engineering. In D.A. Norman and Draper S.W. (Eds) User Centred System Design: *New Perspectives on Human-Computer Interactions.* Lawrence Erlbaum Associates, Hillsdale, NJ.

Norman, D. A. (1988). *The Design of Everyday Things.* Double Day Currency, New York.

Norman, D. A. (1990). The "problem" with automation: Inappropriate feedback and interaction, not "over-automation." *Philosophical Transaction of the Royal Society of London,* B327.

Payne, S. J., & Green, T. R. G. (1986) Task-Action Grammars: a model representation of task languages. *Human-Computer Interaction, 2.* (2):93-133

Onken, R. (1992a). New Developments in Aerospace Guidance and Control: Knowledge-based pilot assistance. *IFAC Symposium on Automatic Control in Aerospace.* 8-11 September 1992, München.

Onken, R. (1992 b). Pilot intent and error recognition as part of a knowledge based cockpit assistant. *AGARD GCP / FMP Symposiom*, Edinburgh, 1992.

Reason, J. T. (1987). Generic Erro-Modelling System (GEMS): A Cognitive Framework for Locating Common Human Error Forms. In J. Rasmussen, K. Duncan and J. Leplat *New Technology and Human Error*. John Wiley and Sons Ltd, UK.

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

Sarter, N. B., & Woods, D. D. (1991). Situation awareness: A critical but ill-defined problem. *The International Journal of Aviation Psychology, 1*, 45-57.

Sarter, N. B., & Woods, D. D. (1992 a). Pilot interaction with cockpit automation : Operational Experiences with the Flight Management System. *The International Journal of Aviation Psychology, 2* (4), 303-321. Lawrence Erlbaum Associates, Inc.

Sarter, N. B., &Woods D. D. (1992b). Pilot interaction with cockpit automation: an experimental study of Pilots' Model and Awareness of the Flight Management System (FMS). CSEL Technical Report N° 92-J-11. Submitted for publication.

Sarter, N. B., & Woods, D. D. (1993). "How did I ever get into that mode?" Mode Error and Awareness in Supervisory Control. CSEL Technical Report . Submitted for publication.

Schneiderman, B., et al. (1991). User Interface Strategies '92. Video tape courses. Instructional Television System. University of Maryland.

Sheridan, T. B. (1988). Task allocation and supervisory control. In *Handbook of Human Computer Intaercation.* M. Helander (Ed.). North-Holland, Amsterdam, NL.

Simon, H. A. (1957). *Models of Man.* New-York : Wiley, USA.

Sullivan, J. W., & Tyler, S. W. (1991). *Intelligent User Interfaces.* Addison-Wesley Publishing Co (ACM Press), Reading, MA, 560 p.

Van Daele. (1992). *La réduction de la complexité par les opérateurs dans le contrôle des processus continus. Contribution à l'étude du contrôle par anticipation et de ses conditions de mise en oeuvre.* Thèse de doctorat. Université de Liège, Belgique.

Wiener, E. & Curry. (1980). *Flight Deck Automation: Promises and Problems.* Moffett Field, CA : NASA TM 81206, June.

Wiener, E. (1989). *Human factors of advanced technology. ("glass cockpit")transport aircraft* (NASA Contractor Rep. N° 177528). Moffett Field, CA : NASA-Ames Research Center.

Woods, D. D. (1993). The Price of Flexibility. *Plenary paper in Proceedings of International Workshop on Intelligent User Interfaces.* W. Hefley and D. Murray (Eds). ACM, January.

Woods, D. D., Cook, R. I., & Sarter, N. (1993). *Clumsy Automation, Practitioner Tailoring and System Failures. Cognitive Systems Engineering Laboratory Report*, The Ohio State University, Columbus OH, prepared for NASA Ames Research Center.