A GENERIC ARCHITECTURE FOR IN-FLIGHT CREW ASSISTANT SYSTEMS BASED ON ADVANCED INFORMATION PROCESSING TECHNOLOGY

René G. Zuidgeest
Pierre J.M. Urlings
National Aerospace Laboratory
Anthony Fokkerweg 2, 1059 CM AMSTERDAM
+31-20-5113210 (fax), rgzuidg@nlr.nl (email)

Introduction

A crew assistant is an on-board automated system that supports an aircraft crew in performing its tasks. Aircraft crews are currently confronted with numerous displays and complex controls in their cockpit. An overwhelming amount of multi-source data is offered while simultaneously control over the aircraft and its systems has to be maintained. This may lead to situations of high workload, in which non-optimal decisions are made.

Crew assistant systems are planned to reduce this problem and hence to improve efficiency and flight safety. They are expected to rely heavily on Advanced Information Processing (AIP) technologies to organize data and control flow in such way that the crew is provided with concise and relevant information. At the same time the crew's control efforts will be considerably reduced. This will enable the crew to concentrate on essentials and to make decisions more effective.

Many developments are going on in this area. Pioneer programmes are the US "Pilot's Associate"[17], the British "Mission Management Aid"[11], the French "Copilote Electronique"[5] and the German "CASSY"[19]. These programmes go by different names but they principally have in common the automation of routine tasks and the provision of effective aids to the crew in solving problems and managing their tasks successfully. The efficiency and effectiveness of handling the aircraft should then increase, because (a) the crew is freed from various tasks and can concentrate on essential decisions, and (b) the crew is given concise information relevant to the mission without saturating them with data.

The architectures developed in these programmes have many elements in common and suggest a more generic architecture as presented in this paper. With respect to the used technologies, the programmes have in common that they consider advanced information processing (AIP) as key technology for a successful implementation of a crew assistant. AIP provides technologies able to handle the complexity of interaction between crew, the crew assistant and aircraft systems and sensors which should result in sophisticated man-machine interaction.

This paper focuses in particular on these two aspects: the generic crew assistant architecture and the application of AIP technology. First, the crew assistant is introduced as an automated system that supports the crew during a mission or flight. After this introduction, the generic architecture of the crew assistant is presented, independent of any type of aircraft or operation. Third, multi-agent systems are assessed as a driving AIP technology for architectural design and implementation of the crew assistant based on the generic architecture. The paper finishes with conclusions and further work.

Throughout the paper, the discussion on the crew assistant is mainly illustrated by an application of a single-pilot military aircraft*, but is equally relevant to multi-crew civil aircraft and hence some references to civil applications are made in the text.

Introduction of the crew assistant

This chapter introduces the crew assistant in its operational environment. The place of the crew assistant and example tasks are discussed.

Operational environment

The main task of any aircraft crew is operate its aircraft to attain its (military) mission or (civil) flight objectives. In the traditional situation, each aircraft system and sensor will interface directly with the crew through dedicated controls and displays in the cockpit. The crew has to interpret multiple displays and has to operate multiple controls simultaneously in order to perform the functions that are related with its main task. In the non-assisted, traditional situation, the interpretation of all sensor information and the control of all systems remain with the crew as opposed to a crew-assisted situation.**

In the traditional situation (Figure 1a), the upward arrows illustrate the information flow from sensors to the displays, downward arrows illustrate the control flow to the systems. For reasons of consistency, the cockpit

* The paper is based on results of EUCLID CEPA-6 RTP 6.5, a cooperation between The Netherlands (NLR), Germany (DASA), Italy (Alenia) and Turkey (Bogaziçi Universitesi). This project has the objective to realise a concept demonstration to show that a "crew assistant" for military aircraft (single fighter pilot being the most complex) meets the needs of future operational missions and improves mission capability in a cost-effective manner.

elements are functionally divided into displays (inputs from sensors to the crew only) and controls (output from the crew to systems only). The aircraft elements are divided into sensors (output to displays only) and systems (input from controls only). In reality, most cockpit and aircraft systems will integrate these functional elements.

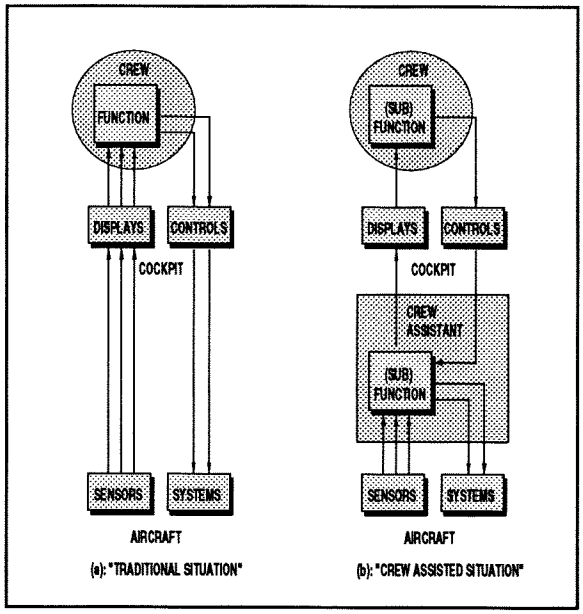

Figure 1. Operational environment of the crew assistant.

Figure 1b illustrates the situation when (a part of) a crew function is assigned to the crew assistant. The original function is then split into a (sub)function delegated to the crew assistant and a (sub)function that remains with the crew. Depending on how much of the original function is delegated to the crew assistant this results in a change in the amount of information offered to the crew and in a change in the amount of control required from the crew.

Example tasks for the crew assistant

Typical (military) tasks to be supported by the crew assistant were identified during EUCLID RTP 6.5 (which have in most cases their equivalent in flying a civil aircraft). Interviews were conducted with 33 pilots from air forces of the participating nations, flying the fixed wing combat aircraft F-16, MRCA "Tornado" and AM/X. Key factors for identification were operational relevance, reduction of pilot workload, increase of mission effectiveness and expected applicability of AIP technologies[8]. These tasks include:

- ***System management:*** addresses monitoring of normal system performance (and in particular engine performance), trend analysis, and reporting of information on system status;

- ***Malfunction handling:*** relates to analysis of anomalies, to presentation of appropriate warnings, to (checklist) assistance in countering malfunctions, and to (when authorized) automatic execution of corrective actions;

- ***Mission/flight planning:*** includes the capability to monitor mission/flight progress, to evaluate the impact of environmental entities (e.g. adverse weather and enemy threats) on this plan and, if needed, to assist in or to perform an automatic (re)planning;

- ***Situational awareness:*** relates to the capability to combine and interpret all available environmental data in order to derive an easy to assess situational picture of this environment; situational awareness may be limited to navigational information but for military applications includes all relevant strategic and tactical information;

- ***Self defence:*** addresses management of self protection systems, assessment of sensor information, selection of available countermeasure options, and (automatic) execution of the selected tactics.

Generic architecture

This section presents the generic crew assistant architecture as a description of functions and interfaces to be embedded in and interfaced with its operational environment (see Figure 1b). These crew assistant functions provide assistance to specific (sub)tasks performed by the crew (see "Example tasks for the crew assistant").

Decomposition

This generic functional architecture is based on a modular 'horizontal' and 'vertical' decomposition. The crew assistant can be seen as a processing unit that
(1) processes in several stages low-level aircraft data from systems and sensors up to easy-to-assess information for presentation to the crew, and
(2) accepts control from the crew to direct and manage aircraft systems and sensors.

The crew assistant, however, can also be seen as a collection of relatively independent modules ("crew

** Example:
(a) *Non-assisted traditional situation.* An "oil pressure" warning on the system malfunction panel may indicate an oil pressure malfunction. The crew has to confirm this hypothesis by considering oil pressures at a variety of power settings indicated in his checklist. Once this hypothesis is confirmed, he has to adjust his engine power to delay further system breakdown, search for the cause of the malfunction and meanwhile replan his routing to a recovery base to land as soon as practical.
(b) *Crew-assisted situation.* Automated support in (a) should confirm that the oil pressure warning is indeed caused by an oil pressure malfunction and, depending on authorization, should execute corrective actions. In addition it could present routing to the nearest recovery base.

assistant functions") that assist the crew in different tasks and hence require different capabilities. These decompositions of 'horizontal' processing layers and 'vertical' crew assistant functions, respectively, are discussed in the next sections together with their interface aspects.

First, a generic decomposition into processing levels is discussed assumed to be typical for a *single* automated crew assistant function and discusses the interfaces between these levels. Second, interfaces of an automated function with cockpit and aircraft systems and other crew assistant functions are discussed and the problem of coordination between functions, crew and aircraft systems is addressed. The last section sumerizes the crew assistant functional architecture.

Internal interfaces

For each crew assistant function the basic flow of data is from the aircraft sensors to the cockpit displays. It is the goal of the crew assistant to direct this flow by processing aircraft sensor data into information for display. Main objective is "to provide the crew with concise and relevant information" (see "Introduction").

Processing levels. In this process, a number of steps can be distinguished. Each step represents a processing level at which data are combined with information, knowledge and procedures and interpreted into information for a next step. In Figure 2, four processing levels are distinguished: collection, assessment, decision and presentation.

- At the ***collection level***, data are collected and prepared for further assessment. This includes for example the collection and conversion of data from sensors and other input devices on-board the aircraft (including data link with other aircraft, sensor data fusion, and filtering or prioritzing of data.

- At the ***assessment level***, the collected data are assessed on normal or abnormal properties. This includes mutual or threshold comparison of data from the collection level, analysis of system trends in time and prediction in order to anticipate future problems, and assessment of the aircraft environment.

- At the ***decision level***, it is decided *what* has to be presented to the crew on basis of inputs from the assessment level and possibly provide autonomous control. This includes for example the filtering of data from the assessment level in order to prevent saturation of the crew's cognitive resources, the generation of advice on handling abnormal situations, and, if authorized, the execution of autonomous action, i.e. control the aircraft systems.

- At the ***presentation level***, it is decided *how* the information from the decision level is presented to the crew. This includes for example an assessment of the available cockpit display resources and crew preferences, and the presentation of information in such way that the crew is directly cued and able to process the information efficiently and effectively.

Interaction: data and control. Each processing level has a characteristic combination of type of data, information, knowledge and operations. Within a single function, these levels interact with each other hierarchically (see Figure 2):

- inputs from a higher level are intended for control or request for information; the lower level is obliged to act according to this input;
- reversely, inputs from lower levels are intended to be information only; a higher level is free to process this input.

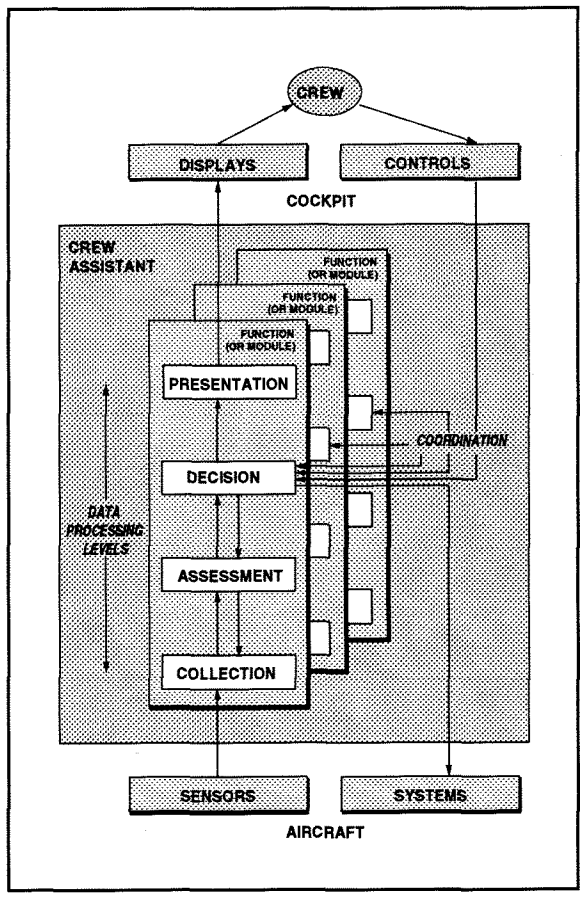

Figure 2. Processing levels and internal interfaces of the crew assistant.

The decision level is modeled to be the only level that receives external control from the crew and it is the only level that provides control of aircraft systems. This crew control includes preferences for display presentation and authorization to the crew assistant to control aircraft systems. Control of aircraft systems includes "sensor

systems". When data collection requires activation of a "sensor system", this control is subject to crew authorization and does not differ from control of other systems.

Co-ordination. When several crew functions are delegated to the crew assistant, interactions will take place which require co-ordination between the corresponding functions within the crew assistant. In Figure 2, co-ordination is performed at decision level because this is the only level that receives and provides control. Typical co-ordination tasks are:

- translation and decomposition of the request for assistance by the crew into the activation of all needed functions within the crew assistant;
- prioritization between crew assistant functions when simultaneous execution of crew assistant functions results in conflicts that are related to the crew (limited cognitive capabilities*), to the aircraft (limited available cockpit displays or supporting sensors) or to other resources (computer memory, processing power or throughput capability);
- cooperation between crew assistant functions when a specific function needs specific results from another; this control (or request for data) is performed at the decision level, although the actual exchange of data may remain at the assessment level.

External interfaces

The crew assistant externally interfaces with displays and controls in the cockpit and with sensors and systems on-board the aircraft. The crew assistant architecture adds capabilities to organize the corresponding data, information and control flows. These capabilities are organized in the functional architecture in Figure 3 in three functional modules: presentation management, coordination management and control management. The next two sections discuss these interface aspects and related management capabilities both for the single function situation and for the multiple function situation, respectively.

Single function. This section discusses external interface aspects from a single function's point of view, i.e. if the crew assistant consists of one function only.

Crew assistant authority.
By delegating a function to the crew assistant, the crew inevitably has to specify the nature of its interaction and the authorization for presentation and control. The specification of this delegated authority can be attained by defining crew assistant operating modes to be selected by the crew, e.g. "off", "standby", "manual" (CA function running in background and information-only), "semi-automatic" (dialogue-driven, advise options for control/action) and "automatic" (automatic action if authorized).

The crew remains in the loop and may regain control at any time. The common "off" and "standby" modes allow the crew overall control over the crew assistant. The modes "manual", "semi-automatic" and "automatic" are at least required to control the individual functions that are delegated to the crew assistant. In the functional architecture of Figure 3 this capability is allocated to co-ordination management.

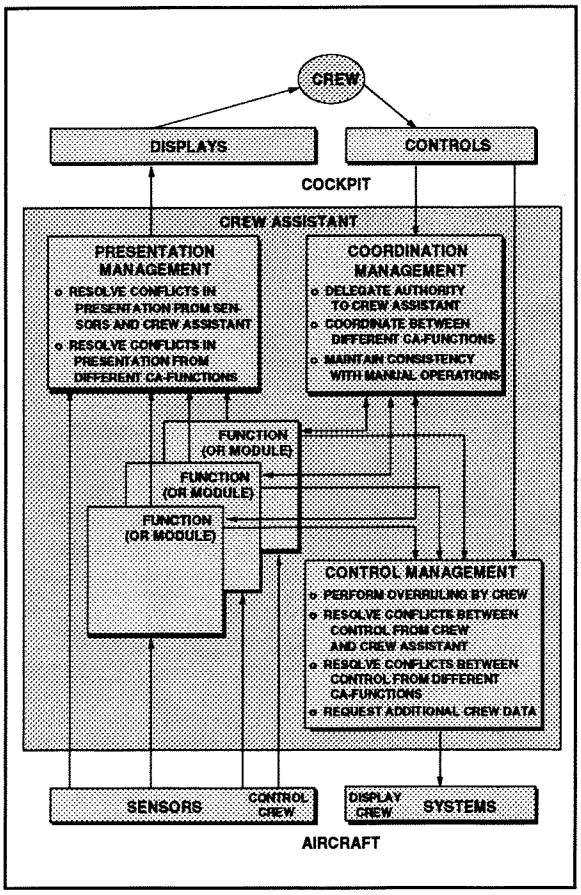

Figure 3. External interfaces and required capabilities of the crew assistant.

Since full automation is outside the scope of the crew assistant (the crew may authorize crew assistant autonomy but shall at all times be allowed in the loop instantaneously), the "automatic" mode should at least inform the crew on the status of its activities and should accept a reset by the crew instantaneously at any time. The crew should have a correct and complete

* Example of a limited coginitive resource. The crew has four communication or interface channels with the aircraft[10]: vision, audio, speech, and tactile. In principle, all four channels could be used independently, but in practice, human motor coordination and data processing limits will restrict simultaneous use of more than two channels. This should be taken into account as a constraint on the use of resources.

understanding of the functioning of the crew assistant in all modes in order to allow a graceful transition between different modes and to maintain consistency with manual (non-CA) operations.

Overruling by the crew.
For each task that is delegated to the crew assistant, the crew shall be able to overrule the crew assistant. Overruling may cause sensors and systems to receive control inputs from both the crew and the crew assistant which may be conflicting. This conflict is prevented within the design of the crew assistant by routing all control inputs through the crew assistant. Note that the overruling of system control is basically different from resetting or deselecting the crew assistant*. In Figure 3, the capability of overruling is allocated to control management.

Conflicting system control.
A conflict in system control by the crew assistant and the crew exists when the same system is employed simultaneously both by the crew and the crew assistant while each performs a different task**. This occurs when e.g. the crew assistant performs a mission planning function and directs a radar in its ground mapping mode while simultaneously the crew independently selects that radar to operate in an air-to-air mode. The solution of such a conflict should be provided within the design of the crew assistant, in Figure 3 this capability is allocated to control management. Control management prioritizes and solves such conflicts and, when required, informs the crew and requests additional guidance.

Limited display resources.
The crew assistant may be in conflict with a sensor outside the crew assistant if both apply for the same display in the cockpit. This is likely to occur with Multi-Function Displays (MFDs). Since such a conflict emerges by the addition of the crew assistant, it should also be solved by the crew assistant. In the functional architecture of Figure 3 this capability is allocated to presentation management. The conflict could also be solved by displays that are dedicated to the crew assistant, but such additional displays are undesirable because these would increase the information load to the crew.

Crew requested input.
Occasionally, the crew assistant may not be able to collect all data required to perform a function, e.g. because a sensor is malfunctioning or because there is no sensor available. Such data can be obtained by requesting the crew to provide these. This capability is included in the functional architecture of Figure 3 by supposing the crew to be a sensor as well. Consequently, the aircraft sensors and systems are considered to include a "display-crew part" to request additional information and a "control-crew part" for input of requested information, respectively. Loading mission data via a crew inserted data cartridge is part of this capability.

Multiple functions. This section considers the aspect of interfacing if multiple crew assistant functions have to co-ordinate (see also "Co-ordination").

Consistency with manual operation.
A pilot may authorize CA-autonomy but shall at all times be allowed in the loop instantaneously. Hence the interactions and hierarchy between crew assistant functions have to match with the way the crew would organize these functions in a non-crew assistant situation.

When multiple functions are delegated to the crew assistant, these will interact with each other and their interaction (information and control flow) has to be organized. This requires the translation of the operating modes (selected by the crew) into a matching hierarchy and co-ordination of all related functions inside the crew assistant. This capability is allocated to co-ordination management in Figure 3. This hierarchy determines how the crew perceives the crew assistant and should be (transparently) available at the external interfaces (i.e. at presentation management and control management).

Conflicting system control.
If authorized, the crew assistant will control systems. When multiple functions are assigned to the crew assistant, these may conflict in controlling the same systems. This may e.g. occur when (short term) self defence functions and (long term) mission planning functions simultaneously request the same threat sensor-system to provide information. In the architecture of Figure 3, solving this conflict is functionally allocated to control management. Solving these conflicts has to match with the way the crew would solve these.

Limited display resources.
The crew assistant will be operational in a cockpit environment that is expected to rely heavily upon MFD technology. This implies that conflicting requirements in the presentation of information are likely to emerge when multiple crew assistant functions simultaneously require the same display. In the functional architecture, the capability to solve these conflicts is allocated to presentation management, see Figure 3. Solving these conflicts as to match with the way the crew would solve

* Example: present autopilots follow this concept. The crew selects the operating mode, the autopilot combines this input with inputs from the aircraft's sensors on the actual state of the aircraft and uses this to control the aircraft's rudders. The crew can overrule the autopilot by simply moving the control stick. Once stick inputs are discontinued, the autopilot continues in the originally selected operating mode.

** Hence it differs from "overruling" when crew and Crew Assistant perform the same task.

these. Remaining conflicts should be prioritized and, when required, additional guidance should be requested from the crew.

Crew assistant functional architecture

Previous sections discussed various aspects that should be part of a crew assistant architecture. Figure 4 shows the functional architecture integrating these aspects. The architecture is modular in two dimensions:
- the various separated crew assistant functions (or modules), and
- the different levels of data processing within each function (or module).

Modularity gained by separated crew assistant functions. The functions relate to crew (sub)tasks that are to be supported by the crew assistant. Single crew assistant functions may correspond directly with single crew tasks, but it is also possible that the crew assistant includes modules of multiple functions supporting strongly-related (sub)task, which will yield a maximum mutual coherence and a minimum interaction with other modules.

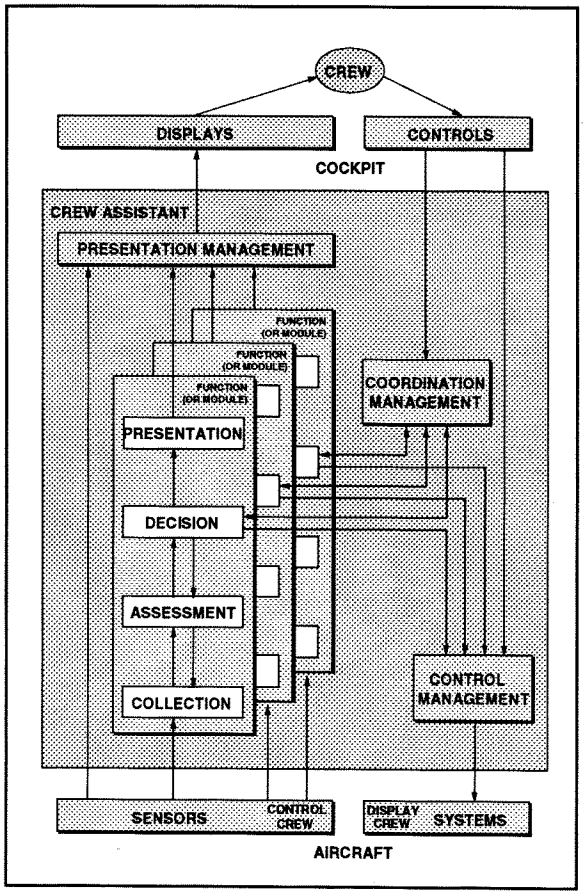

Figure 4. Functional architecture of the crew assistant.

Modularity gained by separated data processing levels. The different data processing levels and their interfaces within a crew assistant function separate collection, assessment, decision, and presentation from each other. When e.g. a cockpit display is replaced, only the presentation part that addresses that display has to be adapted, the others remain unaffected.

The relation or hierarchy between the elements in the functional architecture of the crew assistant determine the contents of their interfaces: it ranges from information-only to control-only. In the crew assistant, internal interfaces exist between the processing levels within a function and between different functions. External interfaces exist between the functions that constitute the crew assistant and the cockpit and aircraft elements. Specific capabilities from potential conflicts and required behaviour are identified and allocate these to co-ordination management, presentation management, and control management.

Advanced information processing technology

The presented functional architecture shows a modular approach in which various elements (potential crew assistant functions as well as co-ordination functions) can be marked as knowledge intensive. It also shows that interactions are complex but these interactions should at any time be transparent to the crew. Advanced information processing provides technologies able to handle this complexity and to build sophisticated man-machine interaction, in order to minimize the cognitive gap between man and machine. These technologies should allow for automation of (sub)functions (e.g. routine tasks) currently performed by the crew (see Figure 1), and for improvement of the level of pilot/aircraft interaction.

Candidate technologies for the crew assistant include:
- knowledge-based systems, e.g. to provide recommendations or alternatives to the crew based on pre-programmed crew expert knowledge and knowledge about the operational environment,
- natural language/speech understanding as an extra information and control resource for the crew,
- perception, including advanced sensor data processing and fusion,
- planning, e.g. for in-flight mission planning,
- learning to improve crew assistant capabilities, and
- distributed problem solving.

This section will focus on how to develop the functional architecture into a system architecture. It is argued that distributed problem solving (DPS) is the proper AIP technology to serve as basis for the system architecture. The other technologies are applicable to specific elements 'functions') of that system architecture. The first section lists criteria that an AIP technology has to comply with in order to be suitable for application in the crew assistant. The second section discusses how DPS can be

applied in the crew assistant. Finally, the last section shows through discussion of DPS with respect to the criteria that application of the technology is a merit to the crew assistant and is a good basis for the system architecture.

Criteria for application of AIP technology

The crew assistant is a complex and critical system as the previous sections have shown. Technology is required that supports the following features in order to be able to form a firm basis for the crew assistant system architecture:

- ***Modularity.*** The crew assistant shall be based on technology that allows for a logical decomposition of the system into simpler components (modules) with well-defined interfaces for "easy" development and reduction of life-cycle costs (increased maintainability).

- ***Real-time performance.*** The crew assistant shall have guaranteed response times in a highly dynamic environment. It may be better to provide an (optimal) answer in time than to provide an answer that is the best, but too late*. Therefore, real-time performance is a critical factor in user acceptance.

- ***Reliability.*** Reliability can be defined as the probability that the system performs its assigned functions under specified environmental conditions for a given period of time. The crew assistant shall have built-in software that reduces the probability of complete system failure, but allows at least for graceful performance degradation.

- ***Integrateability.*** The crew assistant includes many diverse functions needing different implementation methods and techniques. Technology should support integration with conventional as well as advanced technologies preserving modularity.

- ***Engineering methodology.*** The crew assistant shall be developed and maintained by a well-defined system engineering methodology. The technology should support such a methodology to reduce development and life-cycle costs.

- ***Maturity.*** The crew assistant shall be based on mature technology (possibly in near future) for a solid implementation. This is expressed by the availability of tools and prototype or operational applications.

* This finding can be extended with respect to available cognitive resources by a response being not only "too late", but also by being "too complex". Although the response was presented within the pre-set time limit, the message might be difficult to understand within the limits of crew's currently available cognitive resources (and hence interpretation will take more or too much time).

Distributed problem solving in the crew assistant

Introduction. An emerging candidate technology for realization of the crew assistant system architecture is *distributed problem solving* (DPS)[9]. This technology provides a natural transition from the crew assistant functional architecture to a system architecture where the inherent distribution and modularity of functions of the former is preserved in the latter by functionally-distributed problem solving modules.

DPS technology considers two main approaches: distributed knowledge sources (often referred to as blackboard system) and the multi-agent system. Both types of framework consist of multiple agents, but they differ in structure at both global architecture level and agent level. A multi-agent system normally consists of heterogeneous agents that have a range of expertise or functionality (e.g. a complete knowledge-based system performing a specific function such as mission planning or malfunction handling). These agents have the potential to function stand-alone but are also able to support cooperation with other agents[6]. In a blackboard system, the agents are knowledge sources interacting through a shared memory: the blackboard[18]. Here, only knowledge is distributed in knowledge sources, but data, information and control are central as opposed to multi-agent systems.

A common (shared) data structure for a complex crew assistant system with heterogeneous knowledge, data and functions is not likely to be obtainable, and additionally, central blackboard system control will be a bottleneck for real-time performance. The application of a blackboard system should be limited to single crew assistant functions only. In fact, the blackboard system concept provides a natural way to design and implement the layered processing structure of a crew assistant function (see "Internal interfaces"). Blackboard system technology can be suitable for specific crew assistant functions such as system status diagnosis[20], threat assessment[13], data fusion and object identification[22], and overall crew assistant system management[2].

The remaining of this chapter focuses on application of multi-agent technology as a basis for the crew assistant system architecture.

Application of multi-agent technology. Blackboard system technology might be suitable at the crew assistant function level, but the multi-agent system technology can form the basis for the overall system architecture. This architecture will be based on multiple cooperative agents, where each agent implements a crew assistant function. Each agent has its own local data, information, knowledge, operations and control that are relevant for the problem or task domain of the function. This

encapsulation will increase modularity and reduce complexity.

The agent capabilities and their potency to cooperate in order to reach goals beyond the capabilities of a single agent determines the total system's functionality. Co-operation in particular is the key to a sound crew assistant system architecture that is compliant with basic requirements as modularity, reliability, real-time performance and comprehensible, predictable system behaviour (not in the last place for the crew), and directly touches the key problem in the crew assistant as shown in the functional architecture: how to manage interaction between crew, crew assistant functions and aircraft systems, all being agents on their own?

Research on multi-agent systems particularly focuses on this cooperation problem. Important methods for optimal coordination in the crew assistant are:

- a well-defined, pre-designed ***system organisation*** in order to oversee the complexity and enhance real-time performance. A relatively fixed community-like organization following a set of rules of behaviour to avoid system conflicts or harmful behaviour is preferred above a market-like organization (which has dynamic negotiation as key strategy and assumes well-defined task hierarchies that can be dynamically decomposed into nearly independent sub-tasks, which is likely not be the case with the crew assistant) or a centralized organization (where a single coordinator will be a bottleneck).

- ***localization*** of knowledge, responsibilities, control and capabilities in crew assistant agents through specialization, dependency reduction, and increased local capabilities so that coordination decisions are part of local decisions rather than a separate layer (coordinator) above local problem solving.

- ***planning*** to synchronize agent behaviour and resolve conflicts before or during actual execution. See e.g. the plan-goal graph in Pilot's Associate[2].

- Other methods to improve coordination in the crew assistant as a multi-agent system are to increase common ***contextual awareness*** of agents so that they can make better and less conflictuous decisions (e.g. a function Situation Assessment that manages a world model that is accessible by all other functions), to perform ***communication management*** to be aware what, how and when to communicate in which relevance, timeliness, and completeness are key items[7], ***resource management*** in order to avoid conflicts in use of resources, and ***data abstraction*** and ***meta-level information*** about the problem domain and inter-agent communication respectively, which provides a common representation of information and directs coordination.

For reasons of modularity, real-time performance and reliability it is argued not to leave co-ordination and management with a single agent, but to distribute it and make it an integral part of each agent's cooperation capabilities. This means that in a multi-agent crew assistant system architecture, there should not be a central coordinator as is present in the functional architecture. Hence, a (non-trivial) system solution has to be designed by considering the above-mentioned methods for achieving coherent behaviour with distributed coordination.

Evaluation

In this section DPS technology will be discussed and evaluated against criteria that are relevant to a successful implementation of the crew assistant.

Modularity. The ability to structure a complex problem or task into relatively self-contained processing modules (agents) leads to a modular system. The functional decomposition of the crew assistant by functions and levels of processing provides a basis for a modular system architecture of multiple cooperating agents. Specialized agents work on a problem, possibly through co-operation with other agents using message passing or shared memory, but actual processing and information structures are encapsulated in the agent itself. The concepts of encapsulation and cooperation through well-defined interfaces with simple communication primitives allow the crew assistant to be constructed in a parallel, incremental and evolutionary way, and allow for scalability, extensibility, maintainability, and adaptability. These are features that makes the system flexible towards the ever changing operational environment, aircraft systems and military demands.

Performance. The inherent parallelism of multi-agent systems and the possibility to run on multi-processor hardware through natural distribution of agents among the available processors allow for *fast* response. Increased fast response is also obtained in multi-agent systems where agents are co-located as in the crew assistant and make use of shared memory rather than message passing as communication primitive, see e.g. Copilote Electronique[12].

In order to be useful in the crew assistant, however, DPS technology must cope with *hard real-time* and *any-time* requirements and deal with the basic trade-off of quality and time under the assumption that resources (e.g. computation, communication) are limited. Promising methods and techniques are:
- *PAO**, a planning technique for meeting event deadline specifications[16].

- *dynamic notice boards* for real-time operation using blackboards[14].
- *partial global planning*, addressing the trade-off between predictability, quality and responsiveness and the effect on coordination and system coherence[7].
- *approximate reasoning* which uses multiple methods to solve a problem, where each method (or approximation) is a trade-off between the time required to generate the response and the quality of the response[15].
- *progressive reasoning* provides guaranteed response by quickly producing a coarse-grained solution that is refined incrementally if time is available[15].

Reliability. Reliability is one of the potential benefits gained from the application of multi-agent systems through achievement of the following characteristics:
- *Modularity*, decomposition of a system in relatively simple, cooperative agents makes a complex system comprehensible and will reduce number of system bugs.
- *Redundancy*, redundant allocation of system capabilities to agents provide flexibility in case of agent failure, showing graceful system performance degradation rather than complete system failure.
- *Integration of results* obtained from different viewpoints (e.g. different sensors) or agents by cross-checking and triangulation, increases reliability and reduces data uncertainty.
- *Multi-processor hardware* allows each agent to run on a private processor, increasing system reliability at both software and hardware level (by agent-processor reallocation).

Multi-agent systems are potentially more reliable than conventional, centralized systems, but on the other hand, multi-agent systems might have negative effects on reliability to account for:
- *Non-determinism*, due to data uncertainty and a-synchronous agent communication. The line of reasoning will not be fully traceable and hence the system can not be fully verified or validated. This allows for forward error recovery only. It is remarked[16] that the system's reaction to unexpected events should be both predictable and reliable if it is to gain acceptance by a user community (i.e. the crew). Non-deterministic behaviour can be reduced by incorporation of prescribed strategies in the system's design with fixed coordination patterns and control flow.
- *Performance*. Reliability is partly realized by redundancy of agent capabilities and integration of results, but the resulting intensive agent interaction is at the cost of performance. Reliability is very important in the crew assistant, but ignoring response time constraints makes the system useless.
- *Testability*. A number of features make multi-agent systems difficult to test[1]:
  - many loci of control (simultaneous intervention is difficult),
  - communication delays (determining the system's state at a given time is difficult),
  - non-determinism (reproducibility is difficult),
  - system monitoring alters behaviour (stopping or slowing down one process alter behaviour of the entire system), and
  - large amounts of data (magnified in DPS systems that are often large).
- *Qualification and certification*. The discussion above provides an indication of the problems that could emerge in the process of *qualification* and *certification* and associated flight safety, with as main problem the inherent non-determinism.

Integrateability. Multi-agent systems allow for a heterogeneous integration of methods and techniques where each has encapsulated its own knowledge representation, reasoning capabilities, data bases, etc. Blackboard systems do not have complete encapsulation, but still allow for different knowledge and data representation techniques.

Engineering methodology. The application of multi-agent systems has a positive impact on system engineering (see "*modularity*"). To improve system engineering the following is recommended:
- apply *concurrent/parallel development*, if the interfaces have been agreed, agents can be developed relatively independently;
- apply *incremental/evolutive development*, both at the agent and architectural level. Architectural prototyping can be done separately from agent prototyping if the agents' characteristics are known to some extent. The SAHARA tool[3] is an example tool used for Copilote Electronique to prototype and evaluate alternative architectures on real-time performance and behaviour considering parameters such as agent organization, granularity, resource availability, etc;
- use *mature tools* that not only support a computational model to build DPS systems, but should also pursue a clear system engineering methodology that considers prototyping, production and maintenance.It is argued[23] to combine agent-oriented design with object-oriented design principles into criteria for a DPS system design methodology.
- because DPS is closely related to knowledge-based systems, the engineering methodology should also consider knowledge-based system engineering approaches. So, DPS system engineering should be a *migration* of conventional, object-oriented and knowledge-based system engineering methodologies with additional agent-specific features;
- consider *user interfacing* as a key aspect in system engineering[1]. The crew assistant is to be classified under functionally decomposed systems. It is advised to have one agent being responsible for user

interaction. This responsibility could be dynamically shift among agents providing a specific functional view (e.g. mission planning, system malfunction handling), possibly supported by information of other agents. This enables an integrated, agent-focused view of an agent's local reasoning and global system behaviour to the crew;
- comply with certification requirements from the start onwards.

Maturity and next generation. Maturity of the technology can be measured by the availability of tools and the realisation of operational applications. A rich set of tools are currently on the market, so that in this respect maturity and state-of-the-art of DPS is relatively high. The blackboard concept is older than the multi-agent concept. But because of the nicer properties of multi-agent systems, companies are focusing more and more on multi-agent technology which is reflected by the growing list of multi-agent development tools. In fact, most of these tools have integrated blackboard system technology in their architecture.

Example crew assistant systems that already apply DPS technology are:
- *Cockpit Information Management* prototype system CIM uses a blackboard architecture as basis[4].
- *Pilot's Associate* adopts a distributed blackboard architecture in order to structure the system as a heterogeneous, loosely coupled system in which individual agents are not restricted to a particular development environment or software approach[2]. Communication between modules was centrally coordinated by a blackboard-based agent called the Mission Manager, but this centralized approach has been abondoned in future system design for complexity and performance reasons and is being decentralized and distributed among the agents.
- A prototype application[10] of an *expert system that manages a set of cooperating expert systems*. It provides interaction management towards the multiple expert systems as well as interaction management towards the pilot, so that the complexity of the multi-expert system is hidden from the pilot.
- *Copilote Electronique* is based on a flexible heterogeneous implementation paradigm[5] which is evaluated on performance with the simulation tool SAHARA[3].

Note that although DPS technology is expected to reach maturity on short term, application of this technology in crew assistant systems is still in the research or prototype phase.

Conclusion

A crew assistant as an on-board automated system will assist the crew in performing their tasks to improve efficiency and flight safety in a demanding, complex operational environment. This is achieved by assigning (a part of) a crew function to a crew assistant. Depending on how much of the original function is delegated to the crew assistant, the amount of information offered to the crew and amount of control required from the crew will be significantly reduced, enabling the crew to concentrate on essentials and to make decisions more effective.

Based on the operational environment in which the crew assistant supports the crew in performing its tasks, a generic functional architecture of the crew assistant has been presented. The architecture identifies crew assistant function modules which implement (parts of) crew functions, management functions that address aspect of coordination, authorization and conflict resolution, and its internal and external interfaces.

The functional architecture shows a modular approach in which various elements (crew assistant functions as well as management functions) are knowledge intensive. Advanced information processing provides technologies able to handle the complexity of the operational environment and to build sophisticated man-machine interaction based on knowledge-based processes, in order to minimize the cognitive gap between man and machine.

Distributed problem solving technology has been considered as a key technology for the crew assistant system architecture. The tendency is to let a multi-agent system form the backbone of the architecture that considers co-ordination aspects (in the context of agent priorities, deadlines, goals and resources) and to apply blackboard system technology to local task-dependent problem solving (perhaps serving as a backbone structure of an agent). With respect to real-time operation, the multi-agent system architecture will address the basic trade-off between agent communication and computation, and the agent's blackboard system is particularly suited to make problem-dependent trade-offs between quality and responsiveness.

DPS has a positive effect on modularity, real-time performance, reliability, integrability and system engineering. However, for the crew assistant some arrangements have to be made, including:
- a relatively fixed agent organization to let each crew assistant function map on a specific agent, to reduce control complexity and non-determinism (to guarantee consistent and predictable behaviour towards the crew) and to provide predictable load balancing of the limited resources such as communication bandwidth and computing power.

- consistent and predictable conflict resolution where agents opt for the same resources as on-board computer hardware, aircraft systems (e.g. chaff, flare, weapons), cockpit systems to provide the crew information, and the limited cognitive capabilities of the crew.

Although DPS contributes to crew assistant reliability (flight safety) if non-determinism is kept to an absolute minimum, flight safety is only *guaranteed* if the following conditions are satisfied:
- the crew assistant's task is to *support* the crew,
- the crew will always be in command as final *authority*,
- delegated autonomous operation may only be considered for simple, routinely tasks that ensures deterministic and predictable agent behaviour.

Further work is expected in:
- detailing the crew assistant architecture and demonstrate concept,
- development of crew assistant functions consistent with the architecture, and
- performing research on qualification, certification and flight safety aspects.

References

(1) Avouris N.M., *User Interface Design for DAI Applications: An Overview*, Distributed Artificial Intelligence, Theory and Praxis, Eds. Gasser, Avouris (Eds.), Boston Kluwer Academic, 1992.

(2) Banks, S.B., Lizza C.S., *Pilot's Associate: A cooperative Knowledge-Based System application*, IEEE Expert, June, 1991.

(3) Barat et al., *Design and Analysis of Multi-Agent Systems with Sahara Simulation Tool*, COGNITIVA 90, 1990.

(4) Baum L.S., *Recent Developments in Blackboard Frameworks (Foreword)*, Blackboard Architectures and Applications, p303-308, 1989.

(5) Champigneux G., "In-Flight Mission Planning in the Copilote Electronique", *AGARD Lecture Series on New Advances in Mission Planning and Rehearsal Systems*, October, 1993.

(6) Decker K.S., et al., *Evaluating Research in Cooperative Distributed Problem Solving*, in *Distributed Artificial Intelligence Vol II*, Gasser L., Huhns M.N. (eds.), London Pitman/Morgan Kaufmann, 1989.

(7) Durfee E.H., et al., *Trends in Cooperative Distributed Problem Solving*, IEEE Transactions on Knowledge and Data Engineering, Vol. 1, No. 1, March, 1989.

(8) Euclid RTP 6.5 "Crew Assistant", User Requirements Document, document D-URD-WP2.3-AL/4A), February, 1995.

(9) Gasser L., *An Overview of DAI*, Distributed Artificial Intelligence, Theory and Praxis, Eds. Gasser, Avouris (Eds.), Boston Kluwer Academic, 1992.

(10) Gerstenfield A., et al., *An Expert System for Managing Cooperating Expert Systems*, Artificial Intelligence in Engineering: Tools and Techniques, Cambridge, MA, Aug., 1987.

(11) Gibson C.P., Garret A.J., "Towards a Future Cockpit - The Prototyping and Integration of the Mission Management Aid (MMA)", *AGARD Symposium on Situational Awareness in Aerospace Operations*, 1989.

(12) A. Gilles, et al, *A Parallel Multi-Expert Architecture for the Copilote Electronique*, Dassault-Aviation, France, 1992.

(13) Holla K., Benninghofen B., *A Threat Management System*, Proceedings of the AGARD Avionics Panel Symposium on Machine Intelligence for Aerospace Electronic Systems, Lisbon, Portugal, (AGARD-CP-499), 1991.

(14) Holt J., Rodd M.G., *An Architecture for Real-Time Distributed Artificial Intelligent Systems*, Real-Time Systems, Vol. 6, p263-88, 1994.

(15) Kuiper H., Van de Poll E., *Real-time Aspects of Advanced Information Processing in Multi-Sensor Data Fusion*, NLR Contract Report 94608 L, 1994.

(16) Lane D.M., McFadzean A.G., *Distributed Problem Solving and Real-Time Mechanisms in Robot Architectures*, Engineering Applications Artificial Intelligence, Vol. 7, No. 2, p105-117, 1994.

(17) LaPuma A.A., Marlin C.A., "Pilot's Associate: A Synergistic System Reaches Maturity", AIAA-93-4665-CP, p1131-41, 1993.

(18) Nii H.P., *Blackboard Systems: the Blackboard Model of Problem Solving and the Evolution of Blackboard Architectures*, AI Magazine, Summer, 1986.

(19) Nordwall B.D., "Cockpit Safety Researchers Eye Verbal Advice System", *Aviation Week & Space Technology*, October 5, 1992.

(20) Pomeroy B., Irving R., *A Blackboard Approach for Diagnosis in Pilot's Associate*, IEEE Expert, p39-46, Aug 1990.

(21) Raulefs P., *Toward a Blackboard for Real-Time Interactions with Dynamic Systems*, Blackboard Architectures and Applications, p285-299, 1989.

(22) Sikka D.I., Varshney P.K., *A Distributed Artificial Intelligence Approach to Object Identification and Classification*, SPIE Vol. 1100 Sensor Fusion II, 1989.

(23) Hall L.E., Avouris N.M., *Methodological Issues of DAI Applications Interface Design: Transparancy Analysis*, Distributed Artificial Intelligence: Theory and Praxis, p163-78, 1992.