INTELLIGENT INFORMATION AND INTERACTIVE SYSTEMS FOR PILOT SITUATIONAL AWARENESS ENABLED BY A FEDERATION ARCHITECTURE

Rudi Ehrmanntraut, Joseph Bauer, Aubin M.A.C. Hally

EUROCONTROL Experimental Centre EEC, CNS, 91222 Brétigny sur Orge, France

Keywords: System Wide Information Management, Air-Ground Integration, Total Information Sharing

Introduction

The domain of digital air-ground integration is relatively young for civil aviation, and the move from procedures between pilots, controllers, airport and airline operators using voice towards digital integration of aircraft with ground systems will take its time. The digital datalink that is used or is currently under conception will make the aircraft an integral part of an information system. However, CNS (Communication, Navigation, Surveillance) technologies for datalink have followed a bottom-up development approach, without taking into account a system architecture view. The lack of this global view resulted in customized applications per technology.

A federation architecture is needed in applying a top-down approach on system architectural level to integrate the different applications and technologies. The applications that are federated in this architecture must span the widest possible scope of co-operative and collaborative air-ground integration, and include Controller-Pilot Datalink, Airborne Separation Assurance, Collaborative Decision Making, Conflict Detection, Prevention & Resolution, Automatic Dependent Surveillance, Enhanced Surveillance, Surface Movement Guidance, Flight Information Services for ATC, and many other airline applications like pre-flight and in-flight management. The technologies that are federated must be complete and cover ACARS, SATCOM, VDL Modes, Future satellite and terrestrial telecoms, Internet technologies, telephone technologies, broadcast technologies like UAT, MODE S Extended Squitter, VDL4, and future satellite broadcast. The mechanisms for the federation is by abstraction for components, data, connection, management, security and platforms. We will show in this paper that such an approach is feasible, and develop the building blocks of this federation architecture.

Intelligent information for pilot situational awareness is the principle of showing only minimal needed and necessary, relevant information to the flight crew to enable correct human decision making in co-operative and collaborative processes. This information to be 'intelligent' requires a process of data gathering and preparation. The data gathering is based on the principles of ubiquitous services – with reference to the Internet principles of everywhere, every time, everybody – and we will develop a concept of information services for the aeronautical sector. The data selection mechanism is based on the notion of 'context awareness', and we will show how the context of the aircraft can be analyzed, abstracted, and converted into architectural elements that enable the treatment of information in an intelligent way. Furthermore we will elaborate the concept of dynamic federation of information services, to combine and associate information.

Interactive systems for pilot situational awareness are the ability for the pilot to get more than the automatically presented information if needed. We will elaborate a system in analogy to the World Wide Web browser that enables the pilot to browse for more information on a graphical presentation system. The information sources will again be information services, whether in the own ship, the ground or other aircraft. We will develop the architectural mechanisms needed for this type of flight crew interaction.

This paper has been created by the TALIS project (Total Information Sharing for Pilot Situational Awareness Enhanced by Intelligent Systems, http://talis.eurocontrol.fr) which is funded by the European Commission, DG IST, and the project partners LIDO, NLR, Skysoft, THALES Avionics and EEC. The next section treats the federation architecture, followed by sections on intelligent information and interactive systems, to conclude with a summary.

Federation Architecture

Federation Architecture Principles

The tendency to create Business-to-Business (B2B) applications coupled with the needs for a short Time-To-Market (TTM) has introduced the requirement for machine-driven search and integration of distributed services, which is known as dynamic service discovery.

Integration of data and application has for many decades been the focus of the industry. This has led to the development of many distributed, component-based, object-oriented system technologies. Nowadays distributed systems are described as a collection of components and the interaction among those components. Components however, are subject to failure whether partial or total. These partial failures that plagued distributed computing for many years now, have led the industry to shift the focus to service-based architecture. The need for a distributed architecture that can accommodate elements joining and leaving the network in an ad-hoc manner, the need for integration and interoperability with legacy systems based on the service concept, led in its turn to federated systems.

A Federation is an interoperation relationship that exists between independent software components or subsystems. Each software component or subsystem can act as a client to each other. The components or subsystems that form the federation are called federates. A federated architecture provides infrastructure services that allow federates to communicate and inter-operate without the need of being concerned about system boundaries.

A federation is event-based and service oriented. A federation can be implemented as 'Publish-subscribe' (event-based) system, where components are loosely coupled and communicate through multicast. Components in a federation can be added or removed without the direct knowledge or cooperation of other components.

Integration Principles

In the past fifty years as technology evolved new forms of application architecture have emerged that present a need for meta-information services. This technological evolution has first led to networked systems, which simply connected computers and allowed remote access to and from any of the connected computers. The interoperation in a networked system was restricted to the exchange of files. These non-integrated platforms necessitated the development of what is now known as distributed systems. "Distributed-systems are a collection of independent computers that appear to the user of the system as a single computer".

In a federation emphasis is made on interoperability. Interoperability is a very important property that enables information and functionality to be shared among systems operating in the federation. The interaction model of modules in a traditional distributed system has a level of coupling that is related to the type of dependency that exists between these modules.

Processing dependency is when processing in a module is dependant on some work to be carried out by other modules (remote or local) in order to complete. Informational dependency is when a component needs to send or receive information to and from other local or remote components.

Integration is the second important principle of the Federation Architecture. The given requirement is to integrate all current, next-generation or future datalink technologies with a single, simple underlying paradigm: the connector. The divers technologies are e.g. ACARS, ATN, VDL modes 2/3/4, X.25, SATCOMs, IPv4, IPv6, mobile telephone modes like GSM, GPRS, UMTS and other 3G, but also broadcast technologies like VDL4, UAT, MODE S Extended Squitter, and future satellite wide-band broadcast. The architecture proposes to abstract these some basic categories of connectors: synchronous and asynchronous, point-to-point and broadcast.

Service Principles

Analogous to the World-Wide-Web (WWW), functionality is to the system via services. Services could be anything ranging from for example www.meteo.com, www.atm.com, www.atm.org, www.airport.com, www.flightplan.org, www.air-traffic.com, www.airspace.com, www.airline.com, www.your-travel-agent.com, to www.air-police.gov. These services although very useful by themselves, could serve a more powerful purpose if connected together through a federation. Within a federation, these services will be collaborating, reliable and able to tolerate and survive network failures.

Commonly provide services by a federation include:

- Starting and stopping of a federation,
- services location and discovery,
- services registry,
- federates management and
- data management services.

What is a service? *A service is a contractually defined behavior that can be implemented and provided by any component for use by any component, based solely on the contract [6].* This definition is based on the new paradigm called Service Oriented Programming (SOP), where the focus is on modeling problems in terms of the services that a component can provide or use. For a client component a service denotes a functionality that can be performed by its system environment. SOP identifies the following architectural elements and aspects of services:

- Contracts – An interface that contractually defines the syntax and semantics of a single behavior [6]. Contracts establish declared interfaces to the federation required for the cooperating components. Contract specification typically include data and its format, communication semantics, security and other protocols such as failure and recovery and data exchange.
- Components - An individually deployable, binary implementation that provides contractually specified services. Components are subject to third party composition and deployment.
- Mobility – The ability to move code around by means of a proxy. A proxy is a local object that replaces the remote object on the local machine. The proxy deals with any network-related functions, transmitting any parameters to the remote services and receiving any return values from that service.
- Availability – SOP has the goal to handle partial failures (local or remote), that reduces the availability of distributed systems. High availability in SOP, is provided through redundant network resources.

The high availability of services for its use in air traffic, with its severe safety requirements will drive requirements for dynamic services management. This means that services could be added and removed dynamically, i.e. at run-time. One of the key features enabled with dynamic service discovery is self-forming and self-healing systems. Self-forming because the system will not be statically pre-configured, but instead be built on system components that dynamically join and leave the distributed runtime environment, and user services that discover underlying services at runtime. Self-healing because the distributed runtime environment can be conceived in a way that all services are redundant; if one service fails or stops or leaves the federation, then it can be replaced at runtime with another, redundant service. That feature is of highest importance for the safety-critical air transport system.

As will be explained beneath, the dynamic service discovery will be used to introduce the notion of context-dependent services, i.e. agglomerations of services that federate to give a common service, dependent on the current context of the user.

Federation Architecture Building Blocks

A federation is a component-based architecture that is composed of modular and reusable components. It is a collection of collaborating software services. Components in a federation architecture are hardware and software components. Federation has the following properties:

- The Federation is network centric.
- The Federation is service oriented, i.e. federated systems are founded on the concept of service.

- The Federation must help achieve modularity and reuse of components that make up that architecture.
- The Federation must provide information and resources only when they are needed.
- The Federation must only provide visibility to needed information.
- The Federation must contain failures and information defects and must not propagate these through the entire system.
- The Federation must provide interoperability among its components. Interoperability is a fundamental feature of a federation.

In addition to the above-mentioned properties, we briefly discuss some properties found in certain implemented federated systems including TALIS.

A federation implies a loosely coupled system distributed across a network. The key concept in a federation is that federates or participants, can join or leave the federation in an ad-hoc manner.

Components are the basic units of software that can be composed together to form applications. Components usually have their instances created and managed within a framework. This framework is called a container. Apart from managing components life cycle, a container also provides the basic services that components use to operate and to communicate with other components. Within TALIS we have a meteo component, a Traffic Information System (TIS) component, a Flight Management System (FMS) component and others. These components make use of the TALIS Federated Architecture (FA) services. The FA is a software environment that allows components to be dynamically instantiated, to provide their services and to use services provided by other components. Just how this propagation of services is achieved, is explained later.

The FA is designed to optimise the use of components that are distributed across a wide-area network and to operate different distributed computing and e-commerce technologies such as Jini, Openwings and J2EE. To accomplish this, we defined components that abstract some of the core services that are needed for discovery, startup, shutdown, maintenance, recovery, load balancing and monitoring of TALIS components. Some of these component services are discussed below and a conceptual architectural overview of TALIS is provided.

Connector services. Connectors represent the communication glue that captures the nature of an interaction beween components. Connector services provide TALIS with the following facilities:

- Transport protocol abstraction facilities.
- Connectors provide facilities to connect component that execute on the same or different networked machines.

The Architecture Description Language (ADL), which is the modeling language for service-oriented systems, defines two primary interface types for components: interfaces that components provide to other components and interfaces that use functionality from others. These interfaces are called ports and roles. Each role of the connector defines a participant of the interaction represented by the connector. There are two roles: the sender role and the receiver role. These roles translate into sender and receiver proxies in the connector services.

Event services. The event paradigm is often used to federate new or existing systems and components to interoperate. Event services allow components to communicate with each other via implicit invocation. In this architectural pattern, components publish the details of their services and clients can register their interests, so that they can be notified when a particular event of their interest occurs. An event is an asynchronous message containing details of an activity, which has occurred within a component or has been detected by the component. Events can be distributed on the network (LAN, Inter/Intranet). Examples of events in TALIS are weather updates, aircraft positions updates, severe weather notifications and others. The TALIS Federated Architecture has abstracted this event-based mechanism to allow for reusability and de-coupling from any particular event-based framework.

Registry services. The registry services are used by components to advertise their existence in the federation and to search for other advertised component services.

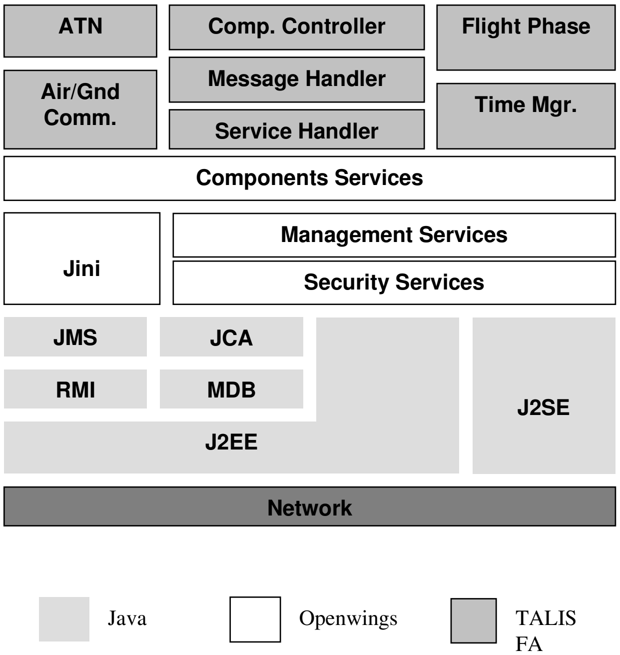

Figure 1 TALIS architectural overview

As depicted in the figure above, TALIS FA is composed of three layers: The FA layer, the Openwings layer and the Java layer.

The Java layer provides a set of services to its deployed components. These services are hidden from the application components through a simplified application programming model. These services include:

- *Load balancing* – By replicating its container services across the network, a share of processing load can be achieved and therefore a better overall performance of a deployed system.
- *Asynchronous Messaging* - J2EE provides this through Java Messaging Service (JMS).
- *Management Services* – J2EE provides component monitoring facilities through its Managed Beans technology. Managed Beans or MBeans are plug-ins that can be added to a component at run-time to monitor its operations.
- *Message Driven Beans (MDB)* – To take full advantage of the asynchronous messaging mechanism provided. MDBs participate in the messaging services provided by JMS.

The Openwings layer is a service-oriented architectural framework, based on Service Oriented Programming (SOP). It is intended for building self-healing, self-repairing and network-centric systems. Openwings builds on Java and Jini by providing components that can be installed over the network.

Some of the services provided by Openwings are of particular interest to TALIS, like its Component and Connector Services. Openwings component services provide abstraction of the service discovery and lookup. The following are some goals, among others, as defined by the Openwings community to be provided by the Openwings component services:

- Service APIs that abstracts the communication between components.
- The ability to provide, locate and use services independent of any location/lookup mechanism.
- Allow services to be provided, located and used by components over the network.
- Support location of services based on a unique service identifier or a set of attributes describing the service.

The TALIS FA layer introduces a layer of abstraction that:

1. Componetises the services – By providing components that are self-contained and well specified for easy integration.
2. Greatly reduces dependency to legacy systems.
3. Introduces reusability through the use of patterns.
4. Promotes extensibility.

The Message Handler component implements the ‘Publish/Subscribe’ pattern, where events are published and subscribed to. An event could be a message or a service. The event handler hides the identity of its agents. Agents are message producers and consumers.

The event handler promotes:

- *Loose coupling of components* - Components do not depend on information about other component’s interface.

- *Extensibility* – Components can be added easily and participate in the messaging process.
- *Maintainability* is greatly improved partly through extensibility and partly through low coupling.

The service handler is another core component within the TALIS FA. In short it provides facilities for:

- *Service Registry* – Provides components with the facility to register and unregister their services.
- *Service Lookup and Discovery* - Provides components with the facilities to discover offered services within the system.
- *Service Propagation* – Facilities to propagate registered services through the entire network.

Dynamic discovery of services is becoming increasingly important. Services are being selected automatically taking into account its location, its context and other semantics information. Currently businesses are moving towards a model where services can be provided ‘anytime and anywhere’.

Air-Ground Communication component. This component abstracts the communication details between air and ground components by implementing a façade to the real air-ground communication API.

Intelligent Information

Intelligent Information Requirements

The execution of a flight requires a number of different tasks from the pilots - from navigating the aircraft, responding to air traffic control, exchanging information with the airline to some peripheral tasks like passenger entertainment. New concepts of air traffic management add new tasks to the pilot, in that the flight deck will be increasingly involved in decision processes, and a true co-operation takes place between the pilot and controller. This concept is called ‘Co-operative Air Traffic Services’ [2]. In addition, higher automation of airline and airport processes include the pilot in the information chain, and increase pilot involvement. This concept is called ‘Collaborative Decision Making’ in Air Traffic Management. That evolution of the pilots’ work results in an increased need for information treatment in the cockpit.

The difficulties of the increased information treatment in the cockpit and the pilot being part in the information chain come from their environment. Cockpit space is a limited resource that is already overloaded with technical devices, and the pilot must be enabled to conduct the current work, flying the aircraft, at the same or increased level of safety. New functionality needed for the enhanced concepts as explained above must be introduced in a non-evasive way, i.e. the pilot should not perceive them as additional workload and they should not interfere with the pilots’ basic tasks. The pilot must be helped in continuing making coherent decisions and not be distracted by the additional functionality.

The functionality that is needed for the new co-operative and collaborative concepts is of different nature:

- A basic functionality is the increased pilot situational awareness needed to evaluate the need for action in co-operative processes, and monitor the implementation of maneuvers. Here the flight deck is the drain for all kind of information from environment and ATC tactical data, e.g. the surrounding aircraft identification, position and velocity, clearances for the own ship and potentially also for other surrounding aircraft, airspace status information like dynamic route information, congestion or special use airspace, meteorological information for all flight phases, airport approach slotting, dynamically contributed SID and STAR, runway identification, runway visual range, taxi routing information, maps, gate management and much more.
- Decision support and decision making is functionality that traditionally involves the human, typically for controller-pilot exchanges and pilot-airline operator exchanges. In the future these decision processes will increasingly rely on digital information exchanges and protocols like CPDLC, and evolve towards at least triangular processes that include the pilot, controllers, airline operators, airport- and military control. Other types of decision support tools for conflict prediction, prevention and resolution will be supported by tools, provided those reach approximately the capacity of humans.

- The concepts foresee new logistics functionality beyond flow management, an evolution in the cockpit in the sense that the pilot will be involved in air traffic flow processes, from pre-flight planning at the last minute, to in-flight tactical flow management. For the airline operations side the pilots' role will evolve to higher anticipation in airline fleet management, and passenger flow support.

From this impressive list of newly introduced functionality into the cockpit it seems obvious that the pilot must get the best possible support: 1) not to be overloaded with information that is not relevant and that does not contribute to the situational awareness, 2) to have coherent information to support decision and logistics processes and the right tools for the decision protocols. To this end the information must be consequently treated so that it is convenient for the human user. That is meant with the keyword 'intelligent information'.

However, any type of intelligence is difficult to implement in information systems as has learnt us the domain of Artificial Intelligence. The characteristics of this 'intelligence' can be given to reduce its scope and put it on feasible basis. First, the information is tailored to the driving concepts to reduce the possible information. Second, system behavior is in many cases predictable, which helps in the filtering of information. Third, the status of the user or the context can be considered to apply further filtering or preparation of information. The context of the aircraft can be analyzed, abstracted, and converted into architectural elements that enable the treatment of information in an intelligent way.

The following paragraphs detail the notion of context and context dependent information.

Context Dependent Information

A "Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves "[4].

In an air traffic scenario the aircraft is an entity. Contexts of an aircraft are for example "status", "activity" and "identity". The context "status" contains criteria such as: fuel consumption, remaining fuel, navigational and kinematics data. The context "activity" contains criteria, like the intended flight path (trajectory).

Additional entities are the pilot and weather phenomena (e.g. turbulence, storm). They have related contexts (e.g. location, status) with their related criteria. Usually this information (turbulence with its location and status) is not provided in the format of entity and context, but it can be seen as it. This is similar to the relation between information and meta-information.

The context can be used by an information system to provide specific information to the user. Based on the contexts, the information system can select information that is relevant to the users current situation.

"A system is context-aware if it uses context to provide relevant information to the user, where relevancy depends on the users task" [4].

This implies that all defined context information is available to the information system. Information scenarios arise by correlating the contexts of different entities, e.g. comparing intended flight trajectories from different aircraft may result in detecting possible conflicts. All useful context combinations should be identified and described in the information scenarios.

Information logistic scenarios can be developed based on the context information. "Turbulence" is an example for such an information scenario. The activation can be computed by correlating the context "intended flight path" of the entity aircraft and the context "location" of the entity turbulence. That means, the information scenario "turbulence" is activated, if the turbulence is located on (or in the near of) the flight path. The time of the occurrence of the turbulence has to be taken into account too. The activation of the information scenarios is not a mandatory part of the information system. This can also be done by external components. In fact these kind of systems exist (a medium term conflict can be detected by the medium term conflict detection system – MTCD).

After the activation of the information scenario it has to be clarified: "Which information should be available for whom, where, when and how?"

The information analysis of the scenarios defines: actors, involved components, work flow, information need, information flow and presentation of information.

1. The actors and involved components define who needs information and who will provide it.
2. The work flow defines the necessary actions. This leads to the related information need.
3. The information need defines what kind of information are relevant and desired in the specific situation.
4. The information flow defines the distribution of information.
5. The information presentation defines how the information will be presented to the user.

For the definition of these information scenarios a generic description has to be found. This makes it possible to add future scenarios to the information system. Furthermore, existing scenarios can be extended without changing the implementation.

The information scenario "turbulence" can be defined as follows: The actor is the pilot. If the information scenario is activated, the pilot wants to receive information about the turbulence (e.g. strength, location, duration). Based on this information he can follow his predefined flight path or bypass the turbulence. The second solution defines an information need: surround traffic. The pilot needs information about the surround traffic to find an alternative flight path. This information is currently available via different systems, e.g. TIS-C, ADS-B, TIS-B. The information gives the pilot the possibility to find a solution, which he can communicate with the sector controller. The pilot has to implement this solution in co-operation with the sector controller.

More information scenarios can be identified and analyzed that are based on the context. These scenarios are based on the main principle of the information logistic – only relevant information are provided to the user.

Interactive Systems

Changing Role of Pilot

The cockpit is subject to constant changes through the past and the future, with increasing automation of functions that have been or are conducted by humans. Therefore the trend to reduce the number of pilots is still ongoing, and there are serious plans for one-man cockpits and Unmanned Aerial Vehicles (UAV). However, both the one-man cockpit and the UAV are concepts that require still a lot of technical and operational conception and validations, and it is difficult to anticipate the time for the implementation of these concepts in civil aviation. The future concepts that influence the cockpit must take these evolutions into account, as is the case here.

The role of the pilot is changing. His workload is shifting from tactical to strategic navigation, and towards increased implications in flight management tasks for the airline. The operational concepts of co-operative ATS and Collaborative Decision Making, as mentioned above, demonstrate this well. For pilot-controller co-operation e.g. it is obvious that the pilot's work will evolve in parallel to the changes of the controller's work; if the controller's workload is shifted from tactical to strategic, then the pilot workload will as well. This means that planning tasks and involvement in strategic decisions will increase, and execution of tactical clearances decrease. Another example is the participation in flow management e.g. for departure and arrival, and also en-route (keyword tactical flow).

The pilot needs the right tools that help to fulfil the new roles in new decision processes, and the new tools need the correct information. Today's systems in the cockpit are not adequate for that, because of their poor human-machine interfaces for information handling, and very poor link with the ground and hence lack of information. Interaction with the system is sometimes cumbersome, e.g. the keyboard of the Multi Cockpit Display Unit. The next generations of cockpits overcome many of the defaults and put the human in the center of an information system, with more displays, multi-function displays and pointer input devices.

The military cockpit shows a further way ahead with the use of enhanced vision (or augmented reality), i.e. an overlay of the natural vision and data about objects, using head-up displays. This type of technology is already available for CATIII landing systems. In the future the same visualization devices can be used to inform the pilot about much more events also coming from outside the aircraft, e.g. runway meteorological information as wind and visual range, runway clearances, taxi routing, gate allocations etc., provided there is a link to ground-based services that can deliver that information.

Interactive Cockpit

The interactive cockpit as conceived by the TALIS project (see next paragraph) illustrates how human machine interaction is seen in the next-generation cockpit, and how information services are used for increased pilot situational awareness.

Information is shown on graphical user-interfaces, and pilots may select objects and browse for further information on objects, very much as on a WWW page. The Multi Cockpit Display Unit treats only textual information, the other displays like the Navigation Display with an additional vertical window, and also the Primary Flight Display can treat visual and textual information. The information is presented in objects that may have visual instances on several displays at a time. Depending on the design of an application, the behavior of objects may depend on which screen the user interaction takes place.

The Traffic Information Service e.g. gives information about other aircraft in the vicinity of the own ship. That information could come from an ADS-B source or a ground-based data server. The pilot may select other aircraft and browse for more information about them, like for example the other aircraft flight plan or the other aircraft trajectory, or its arrival airport, -runway and –slot. This information will be provided by other services from possibly a multitude of information servers at airlines, airports and ATC. The following figure is an example of how the pilot can obtain information on objects for traffic information.

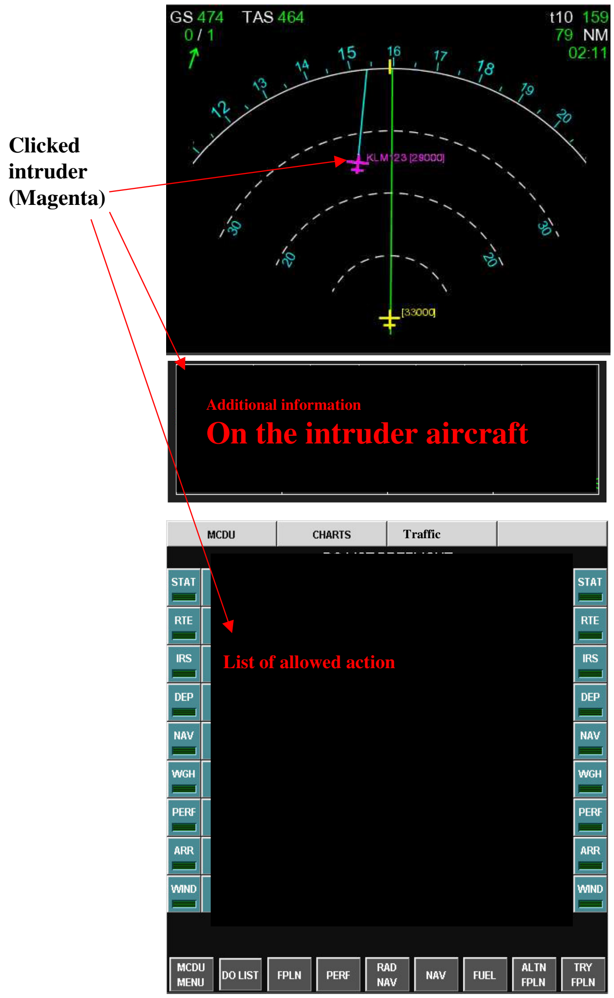

Figure 2 illustrates that a service is executed by clicking on an object in the Navigation Display. That service gives additional information related to the object in the vertical window, plus associated actions in the window of the MCDU. The information and further actions that are shown are a property of the service. The location of the service is transparent to the users, and herewith the functionality of the service might depend (if specified) from the actual service that is used. E.g. a TIS service provided by an airport may provide different functions from a TIS service for en-route. The services will then propose different user interfaces.

The meteo service developed by TALIS follows the same rules, the pilot is informed about meteorological information via the meteo service, and can browse for additional information through the selection of graphical objects.

The Services Concept

The TALIS Services Concept [1] is a system engineering approach for the implementation of the multitude of services defined in [2] and that can be derived from [3]. As mentioned before, the Internet and the WWW with its evolution towards Business-to-Business applications are the paradigm for the Services Concept, i.e. that all functionality of the system is delivered by services, and that services are ubiquitous. Services themselves make use of other services to build up more complex functions.

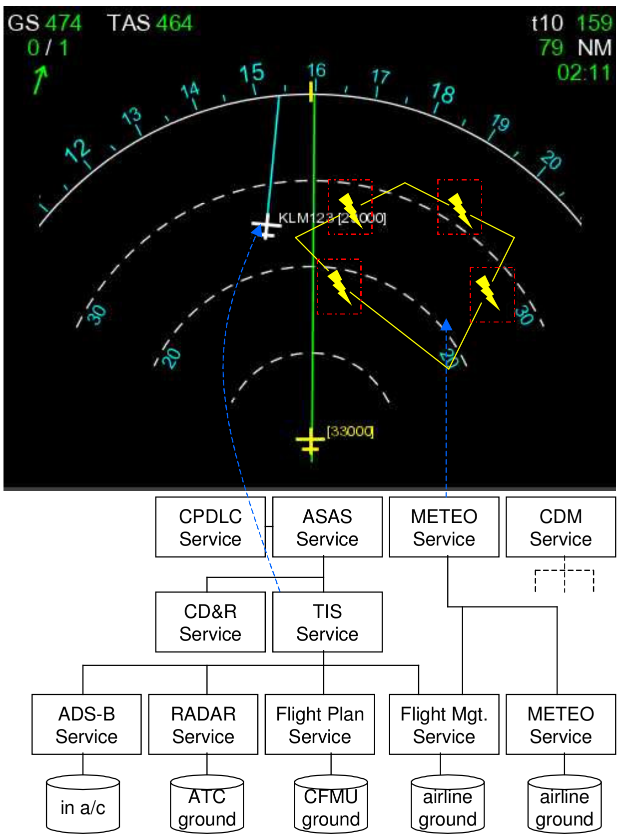

Figure 3 could be a snapshot for a scenario where the pilot is requested to fly station keeping on a target aircraft to avoid thunderstorms. The pilot has access to the application services like CPDLC, ASAS, METEO and CDM. These application services build upon underlying services, which may be transparent to the user. The location of the source information is transparent to the hirarchy of services, via the abstraction of the architecture.

The "agglomerations" of services are not static but build up dynamically instead, through dynamic service discovery and dynamic service binding, to produce the federations of services. Services join and leave to these federations as the aircraft moves, e.g. a flight from New York to Paris will use the New York airport CDM, SMGCS, meteo and traffic services; then Atlantic meteo, and later the Paris, meteo, traffic, SMGCS and CDM services. The almost impossibility to configure such a system statically leads to the requirements of dynamic service binding. It allows also for much more dynamic local implementation of services, without need for upgrading aircraft. That leads ultimately to very fast implementation times.

The TALIS Project

TALIS (Total Information Sharing for Pilot Situational Awareness Enhanced by Intelligent Systems) is a project carried out by a consortium in the context of the 5th Research Framework of the European Commission, DG Information Society. The partners are EUROCONTROL, LIDO, NLR, SKYSOFT and THALES Avionics. The objectives are to produce a verification prototype including a cockpit (A380), weather and flight data sources, that demonstrate the Federation Architecture. A special work package focuses on the possibility for avionics certification, considering that the Federation Architecture builds on commercial-off-the-shelf components. The duration of the first phase of TALIS is two years.

Conclusions

This paper has investigated on automatic and interactive features of the future cockpit as needed for future operational improvements. In this cockpit the services may be presented to the pilot with graphical objects, and related information and actions about that object are provided to the pilot with the other graphical or textual user interfaces. The pilot uses pointer devices and the keyboard to browse for more information or execute an function from a service. The services that are used in the cockpit applications are transparent to the user and may be distributed on the ground, and bind together to form federations of services. Therefore the entire system is conceived as services.

An approach has been made to analyze the possibility of filtering 'useful' information with the notion of 'context', 'context-awareness' and 'information scenario', in the hope that these abstractions will permit to react intelligently to unknown extensions of the system, the system being built on dynamic services federations. An initial approach has been worked out that consists of the information analysis of the scenarios as defined by actors, involved components, work flow, information need, information flow and presentation of information.

The requirements of the TALIS Federated Architecture have been specified and its components described. It has been pointed out that a major innovation in comparison to existing component-based and distributed systems are the requirement for dynamic service discovery, and a complete abstraction of telecommunications technology for the air-ground link with the use of 'connector'. As a positive side-effect, the notion of dynamic service discovery leads to self-forming and self-healing systems. The Federated Architecture has then be broken down into three layers of distinct functionality, and the building blocks have been shown.

That summarizes in brief the first findings of the TALIS project. Further scientific work is needed to clarify 'contextual awareness' for intelligent information. The Federated Architecture and the cockpit will be further developed and result in a demonstration platform, with the potential of becoming a reference platform for air-ground integration. The documents from the TALIS project will be a first step towards system standardization.

The evolution of the cockpit and the pilots' roles is just about to start in a co-operative and collaborative environment. The full integration of the aircraft into an overall information system will give entirely new possibilities to the improvement of the future air transport system. At the end the traveling citizen will profit from better and safer air transport services.

Acronyms

3G	3rd Genereation Mobile Telephone
A/c	Aircraft
ACARS	Aircraft Communications, Addressing, and Reporting System
ADL	Architecture Description Language
ADS-B	Automatic Dependent Surveillance
ATN	Aeronautical Telecommunications Network
ATS	Air Traffic Services
B2B	Business To Business
CD&R	Conflict Detection, Prevention and Resolution
CDM	Collaborative Decision Making
COOPATS	Co-operative ATS
CORBA	Common Object Request Broker Architecture
CPDLC	Controller-Pilot Data Link Communications
EEC	EUROCONTROL Experimental Centre
FA	Federated Architecture
FA	TALIS Federated Architecture
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communication
IPv4	Internet Protocol Versions 4 and 6
J2EE	Java 2 Enterprise Edition
JCA	Java 2 Connector Architecture
JMS	Java Message Service
MCDU	Multi Cockpit Display Unit
MDB	Message Driven Beans
MODE S	Secondary Surveillance Radar MODE S
SATCOMs	Satellite Communications
SID	Standard Instrument Departure
SOP	Service Oriented Programming
STAR	Standard Instrument Arrival
TALIS	Total Information Sharing for Pilot Situational Awareness Enhanced with Intelligent Systems
TIS	Traffic Information Service
TTM	Time To Market
UAT	Universal Access Transceiver
UAV	Unmanned Aerial Vehicles

UML	Unified Modeling Language
UMTS	Universal Mobile Telecommunications Service
VDL	VHF Datalink
WWW	World-Wide-Web

References

1 TALIS "Concept of Dynamic Services for Total Information Sharing in Air Traffic Management", Rudi Ehrmanntraut, EUROCONTROL Experimental Centre, CNS, 19 June 2001

2 "Operational Requirements for Air/Ground Cooperative Air Traffic Services, EUROCONTROL, AGC-ORD-01, Edition 1.0, 02 Apr 2001"

and

"Towards Co-operative ATS - The COOPATS Concept", Version 0.5, Nov. 2000, EUROCONTROL - EATMP – AGC PROGRAMME

3 Concept Definition for Distributed Air-Ground Traffic Management (DAG-TM), Version 1.0, Sep. 1999, NASA – AATT Project

4 Anind K. Dey "Providing Architectural Support for Building Context Aware Applications" Georgia Institute of Technology; 2000

5 David Garland, Robert Monroe and David Wilde. ACME: An Architecture Description Language. 1997

6 Guy Bieber, Jeff Carpenter. Introduction to Service Oriented Programming.

7 Wade Wassenberg. Protocol Independent Programming Using Openwings Connector Services.

8 General Dynamics, Decision Systems, 2002. Service Basecx Architecture as Enabler for Next Generation Battlefield Systems.

9 Randall Bramley and Co, Indiana University. A Component Based Services Architecture for Building Distributed Applications.

10 Anna Liu, Len Bass and Mark Klein. Technical Note, CMU/SEI-2001-TN-025. Analyzing Enterprise Java Beans Using Quality Attribute Design Primitives

11 Aubin M.A.C. Hally. Eurocontrol White paper, TALIS-WP3-WHP-3104, June 2002. TALIS FA Coneptual Architecture.