Abstract

It would be useful to define a generic protocol for information sharing on the air-ground link, so that the flight deck could access information services in a standardised, compatible and interoperable way around the world. Such a protocol, here called Total Information Sharing Protocol (TISP), is defined in this paper. The usefulness of TISP derives from the concept definition of Traffic Information Services in Contract (TIS-C) mode [1], which itself is justified [2] with the introduction of TIS-C in the operational concepts Airborne Separation Assurance System (ASAS) and Controller-Pilot Datalink Communications (CPDLC), and their need for Pilot Situational Awareness.

The paper will give an introduction to the concept of TIS-C and discuss the need for TISP, which is its underlying generic protocol. A definition of Total Information Sharing Protocol (TISP) will be given.

The Total Information Sharing Protocol is a generic software protocol in client-server software architectures. TISP customises the client-server protocol for mobile consumers and for safety-critical applications. It is conceived to operate in an environment of service providers and service users, competition between service providers, and free choice of services for the service consumer. Therefore special attention has been put on the discovery of service providers, the negotiation of contracts between service providers and service consumer, and a pre-negotiated seamless hand-over between service providers. In addition the notion of third parties has been introduced so that contracts can be negotiated on behalf of a party, e.g. an airline negotiating a company contract for all of its aircraft, or ATC imposing standard contracts for all aircraft and service providers. All these features will be presented in the following paragraphs.

TISP is composed of a set of protocol patterns that are the dynamic service discovery, service negotiation, service subscription, service delivery, and seamless service hand-over.

Introduction

It would be useful to define a generic protocol for information sharing between system components and their offered services, so that interaction of services can be standardised and operate on a global basis. This would be a prerequisite for the mobile user like the aircraft. The generic protocol would form an important part of the architecture or framework of large distributed systems. Air Traffic Management is such a large, distributed and heterogeneous system, when we consider all Air Traffic Service Providers, airline operation centres, airports, military operations and the aircraft. The future ATM system will link all these with an information-sharing network. A generic protocol for component interaction for information sharing would then allow for the combination of services in so called service-federations. In addition the generic protocol will handle mobility of service users in a seamless and transparent way (See Figure 1).

There is no equivalent protocol on the mass-market of Information Technology, but the TALIS (Total Information Sharing for Pilot Situational Awareness Enhanced by Intelligent Systems) project foresees that such a protocol will be needed soon. E.g. mobile devices like telephones, palmtops or laptops will increasingly have access to any kind of services that are available on the Internet and that have to be paid for.

The outcome of a generic protocol like TISP could result in standardised middleware components and platform programming constraints.
The generic protocol is defined as the Total Information Sharing Protocol (TISP). In the following sections we will define TISP.

Discussion

The discussion will contain both, the definition of the Total Information Sharing Protocol (TISP) and its rationale.

Overview of TISP

TISP is a generic software protocol for client-server software architectures. TISP customises the client-server protocol for mobile consumers and for safety-critical applications. It is conceived to operate in an environment of service providers and service users, competition between service providers, and free choice of services for the service consumer. Therefore special attention has been put on the discovery of service providers, the negotiation of contracts between service providers and service consumer, and a pre-negotiated seamless hand-over between service providers. In addition the notion of third parties has been introduced so that contracts can be negotiated on behalf of a party, e.g. an airline negotiating a company contract for all of its aircraft, or ATC imposing standard contracts for all aircraft and service providers. All these features will be presented in the following paragraphs.

TISP is composed of a set of protocol patterns that are the dynamic service discovery, service negotiation, service subscription, service delivery, and seamless service hand-over. A simplified TISP is shown in Figure 2.

Figure 1. Illustrates That Air-Ground Applications Can Be Categorised Into Information-Sharing and Decision-Making Applications. Information sharing applications should be based on TISP.

Figure 2. Shows the Simplified TISP, a generic protocol for information sharing of mobile users, with dynamic service discovery, service negotiation, service subscription, service delivery, and seamless service hand-over.

The following paragraphs define the elements of the TISP protocol one by one. We will call them protocol patterns or behavioural patterns, or in short patterns.

TISP Pattern: Dynamic Service Discovery

Dynamic Service Discovery allows the building up of a system during runtime, e.g. services can be discovered during a flight and then used in the aircraft. The service is used, and when it is no more provided – or needed – or available- it is released and another service is looked for.

This is somehow close to the Context Management application of Aeronautical Telecommunications Network (ATN), but has substantial differences! ATN has the same requirement for dynamic use of services for mobile users, but it builds upon a fragile chain where only special service providers may forward other available services to the users, and the user may not make an active search for services. That is very limiting, and possibly even dangerous. In contrast,
Dynamic Service Discovery could lead to highly flexible and dynamic system behaviour that would intrinsically lead to self-healing systems [3], and herewith increase system availability, and system safety (See Figure 3).

![Dynamic Service Discovery Diagram](image)

**Figure 3. Illustrates Use Cases for Service Publishing and Service Discovery.** The user may search in the white pages (naming) or the yellow pages (trader), and will retrieve either an address or a proxy.

**Definitions.** Dynamic Service Discovery is a protocol pattern that allows a user to discover dynamically the name of a service (or component or object etc.) in a distributed environment through the matching of selection constraints against advertised capabilities. A Search Engine (or trader, or lookup) is a service that facilitates the offering and discovery of services of particular types. A Proxy Forward (proxy offer) is a service from the Search Engine to store a proxy from the service provider and forward it directly to a service consumer upon match of the Service Discovery.

1. In the first step the service consumer must formulate a search with attribute-value pairs, e.g.
   - Service: Traffic Information Service – Contract
   - Service Version: X.XX
   - Geo-Availability: Central Europe OR Germany
   - Service Functions: Position AND Flight Plan AND Trajectory AND Medium Term Conflict Alert

2. In the second step the service consumer must address this request to a known Search Engine. Different technologies handle this issue differently, e.g. JINI [4], a Java technology, discovers also the Search Engine via a special broadcast protocol. Others assume that the service consumer knows the Search Engine, in analogy with the World Wide Web (WWW)⁴.

3. In the third step the Search Engine executes the comparison of the request with available service providers and their published services.

4. In the forth step the Search Engine informs the service consumer about matches or mismatches in the request with a list of services.

As mentioned before, today there are three families of technologies that can provide Dynamic Service Discovery service, but which offer different features: CORBA, JINI and XML [5][6][7]. None have high market impact at the moment because of the young age of the technologies. The richest capability is provided by JINI (see Figure 4), which is however limited to the Java world. CORBA is as usual a very good technical candidate especially for large distributed systems with its integration constraints. XML, because of its similarities with the current WWW will probably be the most commonly used, regardless of its drawbacks like the rather complex SOAP protocol.

![JINI Lookup Service Diagram](image)

**Figure 4. Illustrates That the JINI Lookup Service allows for event registering, and forwards the service providers’ proxy to the already registered users.**

JINI has an interesting feature that allows a user to register with a request and to be alerted

---

³ Definitions are highly influenced by [13, 14].

⁴ WWW users know Google, Alta Vista and others.
whenever a service provider publishes a service that matches the requested criteria. This feature allows a very dynamic and highly asynchronous system behaviour, and helps system designers avoid static setups and system locks at start-up. This should lead to cheaper systems. To register with a Search Engine is a very useful feature that has an impact on the human users, and also has operational impact on Air Traffic Management. However, the other additional feature of JINI to use a broadcast protocol for the discovery of Search Engines is not useful in the context of air-ground integration, because these do not allow for this specific protocol.

Proxy-offer and proxy-forward have some problems in the aeronautical context, in that forwarding a proxy is equivalent to sending and loading executable code into the clients’ runtime environment. The size and capabilities of a proxy are flexible and depend on the service provider, and a proxy may be very thin and dumb and just providing an interface to the service, - or fat and intelligent and performing local computation in the client’s environment. That means that the aircraft, being the principle client, must allow for mobile code execution, which is a difficult safety issue. One of the difficulties is that the user environment which hosts the proxy must be able to execute it, i.e. its runtime environment must suit the proxy, i.e. runtime environment and proxy must be compatible. That is achievable if it is based on one single technology like Java. However it is not probable that all aircraft provide a single technology environment. CORBA and XML technologies are better for heterogeneous environments, and both allow in principle a kind of code mobility, however, this is not used very often. The TALIS project studies these issues in a dedicated work package [8].

The service discovery enables the user to find a service provider with a set of search-constraints. As a result the user gets an address, or a proxy, and potentially a number of attributes of the service. Once the user knows the addresses of the service providers, it can contact them and go to the next stage of the protocol, the service negotiation. This next step is only necessary for services that require negotiation; for other services the users may directly subscribe with the name of the service or application.

**TISP Pattern: Service Negotiation**

The TALIS system is conceived to operate in an environment of service providers and service users, competition between service providers, and free choice of services for the service users. Therefore special attention has been put on the negotiation of contracts between service providers and service users. In addition the notion of third parties has been introduced so that contracts can be negotiated on behalf of a party, e.g. an airline negotiating a company contract for all of its aircraft, or ATC imposing standard contracts for all aircraft and service providers.

![Figure 5. Use-Case Service Negotiation](image)

Service negotiation is based on contracts. Each service for each application defines its own contract specifics, i.e. the TIS-C application specifies which kind of contract is to be used to get TIS-C services [1], the Weather application specifies contract properties for Weather services and so forth.

There are three classes of contracts: custom, standard and company. Custom contracts are negotiated on-the-fly, whereas standard and company contracts are negotiated long before service use. Standard contracts are valid for an entire set of users, e.g. all aircraft that want to use TIS-C in Europe. Company contracts are valid for a number of users, possibly for all aircraft from one airline e.g. to access to one specific service provider at special rates. Custom contracts are special contracts for special requests by the users.
One user can make one or more contracts with one or more service providers. Several contracts with the same service provider are useful to have a concatenation of data.

Figure 6. Classifies Contracts In Custom, Standard, And Company Contracts.

Contract negotiation can become quite complex, because it tries to imitate a contract negotiation between persons, which culminates in a legally binding agreement. The topic has arisen in the context of Business-to-Business on the WWW [9][10][11][12], but there are no standards available yet. The World-Wide-Web Consortium (W3C) has started (slowly) to act on standardisation of contract management. TALIS will propose a protocol for contract negotiation in the absence of a standard in information technology. However, if an aeronautical standardisation body like RTCA, EUROCAE or ICAO in the future would like to standardise TISP, then it should investigate whether there are available standards for contract negotiation in the field at that moment in time.

Figure 7. Each User Can Negotiate Several Contracts With One Provider. Each user can have several contracts with many providers. Each provider can serve many users with many applications. One application allows for more than one contract per user, e.g. TIS-C up to 4. The contracts for one application can be with different service providers.

The following sequence diagrams indicate the primary use-cases for contract negotiation. The outcome of a successful negotiation is a service ID that is provided to the user, and with which the user can consequently subscribe to a service.

Figure 8. Shows The Nominal Sequence For A Custom Contract Negotiation.

Figure 9. Shows The Nominal Sequence for a Standard Contract “Negotiation”. The same applies to company contract negotiation.

A further simple scenario is when the proposed contract is rejected. In that case the service provider that rejects the contract should give a reason, like service unavailable etc. In case that there are problems with an ATC related standard service, the service provider should alert the third party of ATC.
This concluded the service negotiation patterns. The user should now have a service ID to subscribe for a service.

**TISP Pattern: Service Subscription**

Users can subscribe to services with the service subscription pattern. There are services that require the negotiation of a contract, and others that do not. In the first case the user subscribes by issuing the service ID that s/he retrieved from the negotiation, in the latter case the user issues the name of the service.

---

**Figure 1. Shows That Any Contract Can Be Rejected.** The third party is informed only in the case of company and standard contracts.

These simple cases barely reflect a real-life negotiation. There is also the possibility to go into a real negotiation with a ‘ping-pong’ effect between the user or its third party and the service provider.

---

**Figure 2. Shows a Failed Contract Negotiation For a Company Contract Between a Third Party and a Service Provider**

As can be seen from the figure above, there is an exchange of counterproposals. This is allowed to last as long as the negotiation parties wish, and in any detail of the contract. Nevertheless, each negotiation party should take care that the negotiation makes sense, and can stop it by rejecting. If a custom negotiation is done over the air-ground datalink, then the number of counter proposals should be limited to 1 to avoid waste of bandwidth. For in-depth negotiations each party should have a contract manager that sets up the negotiation strategy and fixes limits for negotiated values.

---

**Figure 3. Shows the Two Cases of Subscription, One With a Contract ID and the Other One With a Name**

A service may be subscribed to on behalf of the user, either in case that this was contractually fixed, or simply if the service allows for it. In this case the third party must give an identity of the user that will benefit from the service. One could consider higher-level applications where the third party subscribes on behalf of sets of users, e.g. an airline wishing that all of their flying fleet gets specific services, or an ATCO wishing that all aircraft in the sector get specific services. This application, however, is not (yet) defined.
Service Provider subscribe to service (service ID, target user ID)
subscribe to service (name, attributes, target user ID)
subscribe to service (service ID, sets of user IDs, grouping criteria)

Figure 4. Shows Service Subscription by a Third Party; the last case where the third party may indicate sets of users or other grouping criteria has not been specified yet.

**TISP Pattern: Service Delivery**

The service delivery pattern is straightforward. Services are always delivered to the user, never to the third party.

In the case of problems in the service delivery, the user is informed. The same applies in the case that the underlying system encounters problems like transmission problems on the air-ground datalink.

**TISP Pattern: Seamless Service Hand-Over**

The seamless hand-over of services is very important for safety related mobile services if there are many adjacencies between services, or a lot of competition. To simplify service hand-over we suggest that none should be the case.

The behaviour depends on whether the service is contractual or consensual. If the service is of contractual nature, then the clauses for service-hand-over are fixed in the contract that is negotiated between the user and the first service provider. That means that the first service provider negotiates with its peers for the hand-over conditions. The negotiation depth is also contractual, i.e. whether only the new or adjacent provider is concerned or whether there are chains. The specifications of pre-negotiated hand-over will not be covered here, because of its current irrelevance.

---

5 Note that TIS-C [1] does not allow the service provider to end a service, yet the service provider may abort it, but should have good reasons to do so.

6 It is anticipated that transaction protocols will perform well here.
Figure 7. Is a Simplistic Illustration That the Complexity of Pre-Negotiated Service Hand-Over Depends on the Granularity of the Service Providers. Simplest would be if there were only one provider for ATM services in Europe with one backup.

However, the service hand-over itself is important, because it must be sure that the service in the aircraft is not interrupted between the transaction of services.

![Diagram](image)

Figure 8. Indicates the Sequence of a Service-Hand-Over. It is ground-initiated, ground forwarded, user-confirmed, ground ended. This will give high reliability and performance because the air-ground link is only used for confirmation.

**TISP Pattern: Service End**

The end of service depends whether the service is consensual or contractual. If the service is consensual, then any party may request the end of service. If the end of service is contractual, then only those parties that are empowered may end the service. E.g. TIS-C defines that only the service user and the third party may end service, but not the service provider. If the service provider has to end the service, it must use abort and give a good reason for it.

The user or third party may give a reason to the requested end of service.

If the service is handed over, then the confirmation of the reception that is forwarded between service providers replaces the service end request.

If the request for end of service is omitted and the service provider thinks that the service is not really used anymore, it may send a request to end the service to the user or the initiating third party, and should wait about 2 minutes for a reply. If there is no reply, then the service provider may abort with reason. This may happen e.g. if the aircraft is stopped without driving down all services properly.

![Diagram](image)

Figure 9. If the Service Provider Thinks That the Service Is Not Used Anymore, then it may request the end of service. The user may re-confirm with the subscription primitive; otherwise a time-out occurs and the service is aborted.

**The TALIS Project**

The TALIS (Total Information Sharing for Pilot Situational Awareness Enhanced by Intelligent
Systems, [http://talis.eurocontrol.fr](http://talis.eurocontrol.fr) project develops specifications and prototypes for a distributed information-sharing system providing TIS-C and Weather services. The approach of the project is to focus on the overall integration of existing system components into a system of systems with the help of a Federation Architecture. This Federation Architecture will handle collaborative, co-ordinated, distributed, and consistent information-sharing and decision-making.

Figure 10. Illustrates the Vision of the TALIS Project: A collaborative, distributed, interoperable, consistent, available and integrated information sharing system. [13]

Detailed specifications of TISP will be available on the WWW soon. A demonstration is planned for early 2004.

Conclusions

The Total Information Sharing Protocol (TISP) is a generic software protocol for client-server software architectures. TISP customises the client-server protocol for mobile consumers and for safety-critical applications. It is conceived to operate in an environment of service providers and service users, competition between service providers, and free choice of services for the service consumer. Therefore special attention has been given to the discovery of service providers, the negotiation of contracts between service providers and service consumer, and a pre-negotiated seamless hand-over between service providers. In addition the notion of third parties has been specified so that contracts can be negotiated on behalf of a party, e.g. an airline negotiating a company contract for all of its aircraft, or ATC imposing standard contracts for all aircraft and service providers.

TISP is composed of a set of protocol patterns that are the dynamic service discovery, service negotiation, service subscription, service delivery, and seamless service hand-over.

The TISP protocol could underly many client-server applications between the air and the ground, and the first candidates are the TIS-C and Meteo applications that are developed in the TALIS project; others will follow.

A standardised TISP protocol is a cornerstone for an interoperable global system. We recommend that these findings be forwarded to the international standardisation bodies for Air Traffic Management.

Acknowledgements

This paper has been created for the TALIS project funded by the European Commission, DG IST, and the project partners LIDO, NLR, Skysoft, THALES Avionics and EEC.

We thank the European Commission officers, as well as stakeholders in EUROCONTROL for their support.

References


4.B.1-9


### Acronyms

- **ADS-B** - Automatic Dependent Surveillance - Broadcast
- **ATN** - Aeronautical Telecommunications Network
- **ATS** - Air Traffic Services
- **ATSU** - Air Traffic Service Unit, sometimes used as synonym for Air Navigation Service Provider (ANSP)
- **CDM** - Collaborative Decision Making
- **CPDLC** - Controller-Pilot Data Link Communications
- **EEC** - EUROCONTROL Experimental Centre
- **FA** - TALIS Federated Architecture
- **Mode-S ES** - Secondary Surveillance Radar Mode-S Extended Squitter
- **SATCOMs** - Satellite Communications
- **TALIS** - Total Information Sharing for Pilot Situational Awareness Enhanced with Intelligent Systems
- **TIS** - Traffic Information Service
- **TIS-B** - TIS Broadcast
- **TIS-C** - TIS Contract
- **WWW** - World-Wide-Web
- **W3C** - WWW Consortium