A LAYERED SUBSYSTEM ARCHITECTURE FOR A FLIGHT MANAGEMENT SYSTEM

BACKGROUND OF THE INVENTION

The present invention is generally related to flight management systems. More particularly, the present invention is related to a full-featured flight management system that extends a single architecture to diverse subsystems (e.g., communication, controllers, monitoring, and data link) that are normally independently managed and are not easily integratable.

Existing flight management systems (FMS) are notoriously difficult to modify and maintain. History has generally shown that FMS development programs that assume the strict reuse of code from previous FMS programs have unacceptably long development schedules and, therefore, excessively high costs. Furthermore, using prior art FMS development methodology, when there are several FMS development programs being executed concurrently, two or more development teams performing similar development tasks have a low probability of taking advantage of each other's work. These long development schedules, cost overruns, and lack of team synergy can be attributed to the previous FMS programs' lack of a software architecture that is well documented and easily extensible.

Because only a few software architectures that can be used for multi-system management within a FMS are well thought-out and documented, the primary artifact from previous FMS programs that can be used during development of new FMS programs is code. Lacking such architecture, modifications to existing code required to implement new FMS requirements

are typically made from an individual software developer's perspective. Often, software requirements that are used as the basis for implementation do not reflect the original problem being solved or the original capability being added. This leads to a software solution that can be virtually incomprehensible from a user's standpoint. Over time, as changes upon changes are made, the implementation becomes an unmanageable conglomeration of software modules that all have the potential to depend on each other. When this occurs, the result is typically "spaghetti-code." Further complicating the issue can be the fact that there are several FMS programs executing simultaneously. Because of this code-based development process, it can be virtually impossible for an ongoing FMS development program to take advantage of desirable changes that are being made on other concurrent FMS development programs. Therefore, it can be entirely possible that the personnel on two or more program teams are working on the same problem at the same time and are not generally able to collaborate.

There is a need for an improved flight management system (FMS) architecture that is compatible with the different pieces of flight-related equipment on various aircraft types. Such an improved FMS architecture could form the basis for a product line or family of related FMS systems, comprising functions common to all aircraft types and functions peculiar to specific aircraft types, such as a C-5 cargo aircraft.

SUMMARY OF THE PRESENT INVENTION

Our invention is an improved FMS architecture that can form the basis for a FMS product line. Each FMS within this product line comprises functions common to all aircraft types and functions peculiar to specific aircraft types, such as a C-5 cargo aircraft.

One aspect of the present invention is an FMS 'layered subsystem' architecture that enables diverse subsystems at associated hierarchical levels to interface with one another based on predefined dependencies within the overall FMS.

Another aspect of the present invention presents a method of controlling and coordinating communication between diverse subsystems within an FMS. A first FMS architectural rule is that each subsystem operation shall reflect a real-world problem being solved, such as a flight leg transition. A second FMS architectural rule is that subsystem dependencies, within the FMS, mirror corresponding real-world dependencies as defined by specific aircraft operations. These rules are verified during implementation and/or execution of software associated with each of the FMS subsystems.

Advantageously, our inventive FMS automates functions that a flight crew would routinely perform during flight operations and also provides additional high-level functions that would be impractical for a flight crew to perform manually under normal conditions. It is contemplated that the improved FMS of the present invention would comprise software constructed hierarchically as a layered subsystem architecture and residing on a computing platform. The FMS

subsystems, within this layered architecture, include the following.

(a) An operator interface subsystem provides a simple and easy to understand interface between the aircrew and aircraft and is responsible for collecting FMS inputs from the aircrew and presenting FMS outputs to the aircrew. This subsystem also provides operator services (application functions) allowing the aircrew to obtain information directly from aircraft sensors or to assume direct control of aircraft avionics equipment.

(b) A communications subsystem manages and interprets communications to and from the aircraft and controls onboard equipment used for communication.

(c) A flight management subsystem provides functions associated with complex flight operations, such as maintaining an electronic flight plan and providing aircraft guidance.

(d) A database management subsystem provides and controls access to various databases and associated information.

The novel features of the present invention will become apparent to those of skill in the art upon examination of the following detailed description of the invention or can be learned by practice of the present invention.

BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying figures, in which like reference numerals refer to identical or functionally-similar elements throughout the separate views and which

are incorporated in and form part of the specification, further illustrate the present invention and, together with the detailed description of the invention, serve to explain the principles of the present invention.

Figure 1 is a basic illustration of an FMS and its surrounding environment.

Figure 2 is a block diagram of subsystems associated with the FMS and related dependencies.

Figure 3 illustrates additional dependencies including subsystems from Figure 2.

Figure 4 is a layered representation of the subsystems depicted in Figure 3.

Figure 5 is a block diagram illustrating through arrows the conceptual dependencies among five avionics subsystems included in an overall flight management subsystem.

Figure 6 is a block diagram illustration of subsystem dependency rules for the flight management subsystem depicted in Figure 5.

Figure 7 is a layered representation of a decomposed FMS; and

Figure 8 is a substantially complete subsystem layered representation of an overall FMS architecture and associated subsystems and dependencies therein.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

Our invention is an improved flight management system (FMS) 11, which

comprises software architecturally configured as layered subsystems. One purpose of developing this FMS layered subsystem architecture is to document implementation and dependency rules that must be followed during software development or modification for individual subsystems contained with the overall FMS. This layered subsystem architecture can be used for a product line, or family of related FMS systems for various aircraft types. This new and inventive FMS product line will be rooted in a solid, domain-based software architecture, and each individual FMS software implementation will be checked against that architecture. In contrast, because there are generally no architectural guidelines for the implementers to follow, current systems are implemented in an ad-hoc fashion using whatever means are required to get the job done in the most expedient way. By mirroring real-world dependencies during construction of the FMS product line system, the spaghetti-code syndrome can be mostly averted.

The Flight Management System (FMS) described herein is a complex, integrated system that automates many flight crew functions that are routinely performed during flight. Because of its speed and ability, the FMS is also able to provide several high-level functions that were previously impractical for a flight crew to perform manually. Although the internal operation of such a system can be complex, from the crew's point of view it can be quite simple. To the flight crew, the FMS can easily be characterized as the single interface to most of the avionics systems in the aircraft and also as the system that houses an electronic version of the flight plan. Perhaps the biggest challenge for the FMS architects, especially with respect to human-factors, can be to continue implementation of

increased complexity behind the scenes, while preserving the simplicity that can be perceived by the crew.

Tasks being performed by flight management systems are numerous and generally complex. The improved FMS architecture presented herein implements complexity while presenting a user view of the system architecture that can be simple and easy to understand.

The FMS architecture can be generally described herein in terms of functionally oriented subsystems as viewed from the perspective of an end-user. While it can be ultimately important that the implementation of the system match the conceptual view of the architecture, that does not imply that the implementation must follow a functional decomposition approach. It does require, however, that for the implementation approach that can be chosen, the concepts and rules imposed by the architecture, be faithfully adhered to in the resulting implementation.

The first step in designing the architecture of the FMS as with any system is to define the problem being solved and to examine the context in which the system is to operate. Figure 1 depicts a notional representation of the FMS 11 and its surrounding environment. Although the context diagram presents a simplistic view of the entire system, it is not unrealistic to think of it in this manner. After all, the FMS 11 is, to a great extent, simply an interface between the aircrew 12 and the aircraft 13. Basically, using the interface provided by the FMS 11, the operator 12 invokes the services provided by the FMS 11. Depending on the service that is invoked, the FMS 11 can require information

from aircraft sensors 14 and can control aircraft avionics equipment 15. In addition, data can be loaded into and extracted from the FMS 11 using an interface device 16.

Even though the FMS 11 can be pictured in the simplistic manner presented by the context diagram, there are actually several complex tasks being performed within the FMS 11. Therefore, to understand the details of how the FMS 11 performs its functions, it is beneficial to review the contents of the FMS box shown in Figure 1. In describing the functionality of the FMS 101, its capabilities can be divided into just a few groups - called subsystems. Each one of these subsystems is generally dependent upon other subsystems in order to accomplish its function. Finally, the entire FMS 11 resides on a computing platform 21, shown in Figure 2, to accomplish its function.

A representation of layered subsystem architecture contained within the FMS 11 is illustrated in Figure 2. At this architectural level, the functions of the FMS 11 are classified in the broadest sense. Therefore, internally, there are four major subsystems that collaborate to accomplish the overall set of capabilities provided by the FMS 11.

The operator interface subsystem 100 collects inputs from the aircrew operator 12 and presents outputs to the aircrew operator 12. This can be accomplished via several means including without limitation, control display units (CDU), electronic flight instrument systems (EFIS), and various annunciators.

The communications subsystem 200 manages and interprets

communications to and from the aircraft 13 and the equipment used for that communication, including both voice and data transmissions. Interpretation of data-link information provided by a communications management unit (CMU), (not shown) can be performed in this subsystem.

The flight management subsystem 300 manages all functions associated with flying of the aircraft 13, including for example flight planning and aircraft guidance.

The database management subsystem 400 manages and provides access to the several databases housed within the overall FMS, most notably a worldwide navigation database (not shown).

The diagram in Figure 2 depicts each of the subsystems as a box, their dependencies upon each other as arrows, and the computing platform 21 as the surrounding box. In Figure 2, any dependency upon the computing platform 21 or the external environment (from the context diagram) is not shown. However, it should be clear that those dependencies actually do exist, both physically and conceptually. Conceptually, the FMS internal subsystems (100, 200, 300, and 400) have more of a dependency on the external environment than the computing platform 21. Physically, these subsystems (100, 200, 300, and 400) are dependent upon the computing platform 21 to access the external environment. This can be because the computing platform 21 has a physical dependency upon aircraft wiring to affect external aircraft systems.

Figure 3 shows not only the dependencies visible in Figure 2, but also the additional dependencies from each subsystem on the computing platform 21. Notice that an additional subsystem, equipment management subsystem 500, has been added. The purpose of the equipment management subsystem 500 is to provide a domain-level interface to external equipment upon which the performance of application functions, contained within the other subsystems

(100, 200, 300 and 400), can be based. For instance, the flight management subsystem 300 can depend upon inputs from an external inertial navigation system (INS) (not shown). While the logical, domain-level interface to the INS can be important to flight management subsystem 300, the low-level interface is not. Therefore, the equipment management subsystem 500 is responsible for implementing the domain-level interface to the INS using low-level interface facilities included in a computing platform services subsystem 600. This also shows that the equipment management subsystem 500 has a dependency on the computing platform services subsystem 600. The communication subsystem 200 is shown having dependencies on the database management 400, equipment management 500, and computing platform services 600 subsystems.

An important aspect of the layered subsystem architecture is illustrated in Figure 3. It should be noted that the dependencies between subsystems are illustrated in a "downward" direction. Subsystems can only depend upon inputs from other subsystems in the same or lower hierarchical layers. There is no dependency of any lower subsystem on those subsystems above it. The architecture can be totally predicated on the enforcement of what is known as the "subsystem-dependency rule." The subsystem-dependency rule states that, "A given subsystem may depend only upon another subsystem in the same or lower hierarchical layers." Although the "same layer" portion of the rule is not illustrated in Figure 3, its use will become apparent in subsequent views of the system. The implication of this rule is that higher-level subsystems can be dependent upon lower-level subsystems to accomplish their function, but not

vice-versa. Enforcing this dependency rule establishes a client-server scheme where the clients are always above their available servers. This architectural feature makes it possible to ensure that the servers are designed such that the client is mostly irrelevant. It is important in preserving the integrity of the system architecture that the subsystem-dependency rule be enforced at all subsystems of the system.

The importance of enforcing the subsystem-dependency rule is well illustrated in the following example. Ideally, all of the application functions, contained in the various subsystems within the FMS, are independent of the operator interface subsystem 100. Therefore, the operator interface subsystem 100 can be modeled as the ultimate client, located at the top of the layered architecture, while the rest of the application functions are contained in subsystems in lower layers within the hierarchy. By following the subsystem-dependency rule, no dependency can exist between these subsystems and the operator interface subsystem 100. Therefore, the introduction of a new type of operator interface subsystem, perhaps required by a specific aircraft type, will have no effect on the underlying FMS subsystems, just as one would expect.

With a slight variation in diagramming, the architecture can be viewed in slightly different manner. In Figure 4, the same subsystems depicted in Figure 3 are represented, but their dependencies are not. This can be the only distinction between the two representations, except for the obvious graphical difference. The difference, however, can be quite important in laying the ground rules for system development. Indeed, the layered subsystem diagram of Figure

4 shows all the potential dependencies that exist in the system. This view of the system has the benefit of providing implicit guidance for design decisions during development and future system upgrades because of the subsystem-dependency rule. By using the layered subsystem diagram in Figure 4, one can verify that the dependencies in the diagram of Figure 2 are valid. If a dependency turns out to be invalid, then there is a basis for questioning whether the overall system representation is correct, the system functionality is well understood, and whether the subsystem diagram is truly representative of the potential dependencies. It can be important that the subsystem diagram be kept at a real-world domain level of understanding. By keeping it at that level, it should be clear what the set of valid potential dependencies actually is. Then, again because domain language is being used, when consulting domain experts, such as navigation equipment engineers, regarding the validity of dependencies, the diagram can be interpreted and understood by both the FMS system architects and these domain experts.

While the top-level subsystem diagram of Figure 1 illustrates potential dependencies between the FMS and the external environment, it is at too high of a level to adequately address the total set of capabilities generally provided by the FMS. As long as the subsystem-dependency rule is followed, however, it is valid to further decompose each of the major FMS subsystems. The decomposition of each major subsystem further refines its functionality and clarifies the potential internal dependencies. Further decomposition, for example into sub-subsystems, can be performed as long as the resulting

diagram conveys a reasonable level of understanding. Again, each of the sub-subsystems should be defined using domain language. For illustration purposes, the flight management subsystem 300 has been further decomposed and the result is shown in Figure 5.

In Figure 5, conceptual dependencies among five sub-subsystems, representing real-world aircraft flight operations, are shown contained within a decomposed flight management subsystem 300, according to our invention. At this point in the decomposition, the focus is on determining how the capabilities of the overall FMS 11 are broken down and what the relationships between those capabilities are. To determine the proper breakdown, it is useful to remember that the purpose of the FMS is simply to automate various functions that the flight crew 12 had traditionally performed. Therefore, it is not unreasonable to expect the resulting FMS design to directly model functions that it is automating. Again, it is important to stay in the domain language as long as possible during system development so that domain expertise can easily be used to evaluate its correctness. In Figure 5, the aircraft guidance subsystem 310 is shown to have dependencies on the flight engineering 320, navigation 330, and flight planning 340 subsystems. The navigation radio tuning subsystem 350, on the other hand, is dependent on the flight planning 340 and navigation 330 subsystems.

In the development of the decomposition shown in Figure 5, no attention was paid to the structure of dependencies. Rather, it was modeled strictly from a "what-really-happens-in-the-real-world" perspective. If, however, the functions

are simply rearranged on the diagram, it becomes clear that even the real-world representation of the system follows the subsystem-dependency rules that have been set forth. This view of the decomposed flight management subsystem 300 is illustrated in Figure 6. Merely the fact that the subsystem-dependency rule falls out of the design provides some assurance that the model is correct.

Of course, just as for overall FMS system, the hierarchical diagram of the decomposed flight management subsystem 100 in Figure 6 can be represented using a layered subsystem diagram. The layered subsystem view of the decomposed flight management subsystem 100 can then be integrated into the layered subsystem diagram for the overall FMS to get a more detailed understanding of the internal dependencies within the overall FMS.

Referring to Figure 7, the composite layered subsystem diagram shows the effect of decomposing the flight management subsystem 100 into sub-subsystems (310, 320, 330, 340, and 350) on an overall FMS system architecture. Similar decompositions can be performed on the other major subsystems such that the complete subsystem diagram for the entire FMS 11 can be derived. It is important to realize that the subsystems within the architecture are never etched in stone. As the FMS system evolves, the change in subsystems can be inevitable. The architecture, however, must always reflect domain level language and potential dependencies that reflect the real-world view of the FMS 11. A layered subsystem view of the entire FMS 11 is shown in Figure 8. Notice that the complete layered subsystem architecture shows each of the other major subsystems in their decomposed states. In fact, upon

further analysis of the FMS subsystems, it was discovered that there is yet another class of functions, that act as an 'automated crewmember', that must be performed. Therefore, an automated-crewmember subsystem 360 has been added to our inventive FMS, as shown in Figure 8. The fact that this automated-crewmember subsystem 360 was added following a subsequent design review of the FMS illustrates that the position and number of both hierarchical layers and subsystems within each layer can remain fluid during development. As the FMS product line design progresses, the position and number of both hierarchical layers and subsystems within each layer will solidify, but there will be additional modifications, during initial FMS development for the first several specific aircraft types.

Each subsystem of one representative FMS 11, within a planned FMS product line, that embodies our invention is described below. Descriptions include the purpose of the subsystem (at a very high level), the relationship of that subsystem to the rest of the subsystems contained within the FMS, and the design concepts and architectural philosophies of the subsystems. Each subsystem can be described in broad terms with respect to the inputs, outputs, and processing that are a part of that subsystem.

The only responsibility of the operator interface subsystem 100 is to manage all aircrew 12 interfaces to the FMS. This primarily includes control display units (CDU) and electronic flight instrument systems (EFIS) but can include any other device or system that the aircrew 12 can use to affect the operation of the FMS. While it may seem trivial, there are only two parts to

operator interface management - collecting and interpreting inputs from the aircrew 12 and gathering, formatting, and presenting system information for output to the aircrew 12. This can affect the architecture of the FMS because it is important that the operator not be concerned with the details regarding the underlying operation. For example, when the aircrew makes an input to the FMS requesting that some operation be performed; the operator interface subsystem 100 is only aware that some application function exists to support that operation and which input parameters are required. For outputs from the FMS 11, the operator interface subsystem 100 is aware that some application function is providing data for presentation to the aircrew 12, but is not involved in any significant manipulation of that data.

It may be desirable to decompose the operator interface subsystem into a CDU management subsystem 110 and an EFIS management subsystem as shown in Figure 8. The more complex of these two subsystems contained within the operator interface subsystem 100 is the CDU management subsystem 110 because of the complex data entry methods employed by a typical CDU. This typical CDU has a general-purpose interface comprising a keyboard and several buttons where a functional meaning of each button can change depending on an operational context, such as communication radio tuning. This general-purpose interface, which entails a large number of possible entry combinations, can be contrasted against specific purpose interfaces such as cockpit switches or dials, where there are a limited number of possible inputs. The generic nature of the resultant CDU interface drives the complexity of the CDU management

subsystem 110 because aircrew 12 data entry actions and error handling must be implemented within a specific operational context.

Although the general-purpose nature of the typical CDU makes it difficult to manage, its flexibility makes it a very attractive man-machine interface for the aircrew 12. Advantageously, our inventive architecture will allow the entire operation of the FMS 11, from the viewpoint of the aircrew 12, to change without any change to the underlying applications within the FMS. Because of this, the interfaces to the underlying applications, contained within the various subsystems within the FMS architecture, should be well defined and the applications themselves must never directly reference the operator interface subsystem 100.

The CDU management subsystem 110 can be based around "CDU pages". A CDU page is a set of information that is displayed at the same time on the display surface of the CDU. Generally, related information is shown on a CDU page such that there is a set of displayed parameters and a related set of options that define the operations that a set of buttons on a CDU bezel will perform. When an operation is invoked via a bezel button, that operation typically requires a typed-in entry where contents of a CDU scratchpad are used to communicate the data. These basic concepts of CDU operation should be provided as general-purpose services to all pages defined in the system, thus forming a framework for the management of CDU pages.

Finally, there should be no other FMS subsystems that are concerned with the CDU as an input/output device. The CDU management subsystem 110,

contained within the operator interface subsystem 100, has the sole responsibility for interpreting CDU inputs and converting them into internal data representations and commands to application functions contained within other FMS subsystems. Conversely, the CDU management subsystem 110 has the responsibility of making service requests to application functions contained within other FMS subsystems and converting the resultant internal data representations into formats suitable for presentation on the CDU display.

The input portion of the CDU management subsystem 110 provides the following functions.

(a) A data collection function gathers the various aircrew 12 inputs to the CDU, such as bezel button presses and scratchpad inputs.

(b) A data validation function can compare the aircrew 12 inputs against an expected input format based on a CDU operational context and invokes error-handling routines as required.

(c) A data interpretation function converts the aircrew 12 inputs into specific service requests based on a CDU operational context and provides these service requests to appropriate applications within the FMS.

It is generally important that CDU management subsystem 110 only interprets aircrew 12 inputs to the point that they can be appropriately dispatched to other FMS subsystems. For example, the CDU management subsystem 110 can be responsible for convert a text-based input of geographic latitude, such as "N35.12.45", into an appropriate internal numerical

representation, such as "+35.2125", and then passing this converted input to other FMS subsystems. In this case, either the CDU management subsystem 110, or alternatively, the operator interface subsystem 100, knows that a latitude entry is expected and therefore needs to recognize the format of a valid latitude entry. Knowing this information, the operator interface subsystem 100 is able to determine the validity of the entry and is able to convert the entry from a human-readable form (a text string) into the internal FMS representation of latitude. This level of data manipulation can be acceptable, in fact required, of the operator interface subsystem 100. It generally does not make sense for the other FMS subsystems that contain application functions, such as flight trajectory determination, to do the conversion of text strings to internal forms of representation. Once the entry has been validated, the operator interface subsystem 100 will determine which operation is being requested and invoke the required application functions. In a contrasting example, the CDU management subsystem 110 should not be responsible for converting the distance between two latitude/longitude pairs and then sending the result to the application subsystem. Instead, the FMS includes a domain service subsystem 610 that contains an application function that takes two latitude/longitude pairs as inputs and computes a distance between them. Only the CDU management subsystem 110 should be allowed access to the inputs provided by the CDU.

The operator interface subsystem 100 is responsible for the collection of FMS computed data from the other FMS subsystems and the subsequent formatting and presentation of that data such that it can be understood by the

aircrew 12. For instance, CDU management subsystem 110 (or any other part of the operator interface system 100) should not be processing aircraft headings and velocities to determine wind speed and direction. For example, a dedicated 'wind computation' application function within the navigation subsystem 330, could determine wind parameters in whatever way it deems best, and the operator interface subsystem 100 should simply use this application function to retrieve wind parameters. Once data is retrieved, the operator interface subsystem 100 can be responsible for converting internal representations to text strings, for instance, for subsequent presentation on the CDU.

There can be occasions where it is certainly appropriate for the operator interface subsystem 100 to perform more complex processing. For instance, other FMS subsystems may internally represent all aircraft headings with relation to true north. The operator interface, however, can provide the option of displaying headings referenced either to true north or to magnetic north. In this case, it can be entirely appropriate for the true/magnetic conversion to be performed by the operator interface (in addition to keeping track of the current magnetic/true selection). It is advantageous to limit such processing to that required for display purposes only. There are very few instances where raw data must be manipulated solely for the purpose of generating displays.

The EFIS management subsystem 120 collects various data, from the other FMS subsystems, that is destined for display on the EFIS. The data is then converted from FMS internal representations into the formats required by the EFIS and then sent to the EFIS. A typical EFIS can have several modes into

which it can be commanded. Often, the FMS will be the entity that commands the EFIS into the appropriate mode, but may be several ways for the aircrew to select a desired EFIS mode. For instance, there can be control panel switches that are read by the FMS or there can be a CDU page on which the EFIS mode can be selected. Regardless of the method employed, the EFIS subsystem must provide a service that can be used to set the EFIS mode. This service can then be used as required by other subsystems (or, possibly, the same subsystem) to effect a change on the EFIS displays. The point is that the EFIS management subsystem 120 should not be dependent upon another FMS subsystem simply to determine the EFIS mode. Rather, that can be something that should be communicated by some other FMS subsystem to the EFIS management subsystem 120. The output processing of the EFIS Management subsystem can be straightforward. Basically, lower level FMS subsystems contain application functions that are used to collect FMS data, such as flight director steering bars. Then, that data can be reformatted as required for communication with the EFIS. Finally, the services of the equipment management subsystem 510 are used to transmit the data to the EFIS.

With respect to the overall function of the FMS, the data-link management subsystem 210 can be nothing more than a remote user of the FMS application functions. Remote user inputs and outputs are communicated to the aircraft via radio transmissions instead of via direct connection of a device in the cockpit. This detail, however, does not preclude viewing of the data-link management subsystem 210 in the same light as CDU Management or EFIS

Management. Just as the CDU Management subsystem 802 is the only subsystem aware of the presence of a CDU; the data-link management subsystem 210 is the only FMS subsystem that is aware of the presence of a data-link. The data-link management subsystem 210 will interpret a message from the data link and convert this message into a service request to an application function contained within another FMS subsystem. However, the other FMS subsystems have no idea of the source of these data requests. They could just as easily be coming from the CDU management subsystem 110. The isolation of the data-link servicing can be a crucial point in the design and architecture of the FMS product line.

Although no other entity in the FMS needs to be aware of the data-link per se, other FMS subsystems that reside above the data link management subsystem 210 clearly know of its existence and can use that knowledge as required. Generally, aircrews 12 do not appreciate some automatic changes such as modification of the flight plan. Since the aircrew 12 has ultimate responsibility for the safe operation of the aircraft 13, they should have the final say regarding any commands that can affect their flight trajectory. Therefore, when an uplink is received, the aircrew 12 should manually accept the proposed change. Since this can be accomplished via the CDU, the CDU management subsystem 110 must access the data link management subsystem 210 to collect the pertinent information. It can be important to remember, though, that only those details of the data-link management subsystem 210 that can be important to higher subsystems should be exported. For instance, it may be appropriate

to export services that can indicate whether an uplink is pending and also indicate what type of data can be uplinked. It may not be appropriate to export services that interpret raw message data or that communicate with the external data-link computer.

Continuing with the view of the data-link as a remote user, in real world terms, the user on the other end of the data-link is probably a human. In the CDU management subsystem 110, data can be converted to and from a form that can be understood by the user, such as a text string. Although the remote user on the other end of the data link is probably a human, in the context of the FMS, it can also be a machine. Nevertheless, the data-link management subsystem 210 acts similarly to the CDU management subsystem 110 with respect to the user. The data-link subsystem 803 also converts data to and from a form that can be understood by the user, but the user in this case can be a machine.

The bottom line with respect to the architectural view of the data-link management subsystem 210 is that it should be viewed exactly as either of the two subsystems in the operator interface layer. All the details regarding the physical interface should be processed internally to the subsystem. Operations requested over the data-link should be converted internally to the set of application service requests that are required to perform that operation. Therefore, given the same operation commanded via the data-link as would be commanded via the CDU, the exact same application service requests would be invoked.

The input processing required for the data-link management subsystem 210 comprises collecting transmission requests (uplink and downlink) from the data-link equipment and also collecting of uplink data from the data-link equipment. The collected data is then used to invoke application services to accomplish the appropriate request. Additionally, this subsystem receives inputs from higher subsystems in the FMS such as operator mode settings, from the operator interface subsystem 100, or requests that are commanded from the CDU management subsystem 110.

The outputs of the data-link management subsystem 210 comprise primarily messages to the data-link equipment. Data to be downlinked can be first collected from the application functions within other FMS subsystems, then formatted and sent to the data-link equipment using the equipment management subsystem 510.

The flight management subsystem 300 is generally responsible for performing all those capabilities of the system that deal with the actual flight of the aircraft 13. These capabilities are generally functions that used to be performed by a member of the aircrew, but that are now automated by the FMS 11 or new functions that assist a crewmember in performance of his/her task. The typical, basic functions that are performed by the FMS 11 are, not surprisingly, almost one to one with the crewmembers on a typical aircraft 13. Therefore, the flight management subsystem 300 can be decomposed into the following sub-subsystems.

a) The pilot determines control inputs to guide the aircraft along its

desired path - this corresponds to the guidance subsystem 310.

b) The flight engineer monitors aircraft performance and predicts aircraft situation along the flight path - this corresponds to the flight engineering subsystem 320.

c) The navigator determines current position and state of the aircraft - this corresponds to the navigation subsystem 330.

The existence of the automated-crewmember subsystem 360 can be necessitated by the fact that the FMS has some capabilities that relate two sets of data that are otherwise unrelated. These types of functions are best illustrated using a non-limiting example.

The FMS provides the aircrew operator 12 with the capability to manage a set of custom waypoints that are physically contained in the database management subsystem 400. The aircrew operator 12 can define a custom waypoint via the entry of a waypoint name and the position of the waypoint. The operator can also modify or delete any of the custom waypoints. Once defined, a custom waypoint can be used just as any other entity that defines a position, including as one of the endpoints of a flight plan leg.

Assume that a flight plan has been entered and one of the legs is defined using a custom waypoint. Then, the operator decides to delete the custom waypoint from the database. FMS system requirements, however, do not allow the deletion of a custom waypoint if it is part of a flight plan. If this requirement is implemented simply by having the custom waypoint processing, in the database management subsystem 400, check the flight plan, the subsystem-

dependency rule will be broken. Further analysis of this requirement, however, will reveal that this can be a higher-level system requirement. The system could surely be defined to allow that operation to occur. Certainly, the custom waypoint processing does not have any problem deleting a custom waypoint at any time. Therefore, the proper way to implement this requirement can be to place a custom 'waypoint delete' application function within an automated-crewmember subsystem 360 that is above both the database management 400 and flight planning 340 subsystems.

This is a pretty specific example in the scheme of the entire FMS architecture. There are, however, other capabilities that fall into this general category. In fact, the auto-tuning function for the navigation radios could be classified as an application function within the automated-crewmember subsystem 360. That function uses the services of the navigation 330, flight planning 340, and database management 400 subsystems to accomplish a more complex, higher-level function. The navigation tuning subsystem 350 is broken out, as a separate subsystem, only because of its significance as a feature of the overall FMS and because of its complexity.

The inputs to this subsystem are really varied. The most important aspect to the inputs for this subsystem 801 is that the inputs come from two or more lower subsystems. If a particular capability of this subsystem requires inputs from only one lower subsystem, then the capability should probably be implemented in that lower subsystem.

As with the inputs, the outputs of this subsystem are widely varied and

dependent on the higher level function being implemented.

In the available legacy systems, this subsystem is virtually non-existent. In fact, the absence of this subsystem can be one of the primary causes of the implementation complexity found in those systems. In fact, in the custom waypoint example given above, the legacy FMS systems implemented an equivalent function by simply making a database management function dependent upon a flight planning function. Multiply that artificial dependency by the number of these types of higher-level functions, implemented in a typical FMS and the complexity becomes virtually incomprehensible.

The approach to the development of the automated-crewmember subsystem 360 is to identify those functions that belong to the class of higher-level functions within the existing systems. Once identified, it can be simply a matter of moving the function from where it is currently implemented into the automated-crewmember subsystem 360. By moving the function, the artificial dependencies from the original subsystem are eliminated.

The navigation radio tuning subsystem 350 can have two general functions. First, the subsystem can provide a general set of services to tune each of the on-board navigation radio receivers to selected radio navigation aids (navaid), that are located at predetermined geographic positions. Second, the subsystem can provide a function that determines an optimum set of radio navaids to which the receivers should be tuned. By using these two navigation radio tuning subsystem 350 functions, the navigation subsystem 330 is able to compute a very accurate aircraft position. By accomplishing this function in an

automated fashion, the navigation radio tuning 350 and navigation 330 subsystems can work together to relieve the aircrew 12 of both radio navaid determination and radio tuning tasks.

The set of radio tuning services provided by the navigation radio tuning subsystem 350 are designed for use by any other FMS subsystem that has a requirement to tune the navigation radios. It is reasonable to ask: "Why cannot a pilot just tune the navigation radio directly and not have to go through this set of services?" The answer is that, for a navigation radio to be useful, it must be tuned to a navaid whose position on the earth is known. If a navigation radio is simply tuned to a frequency, the resulting information is useless. Therefore, the set of services provided by the navigation radio tuning subsystem 350 exist solely to determine the radio navaid (more specifically, the position of the navaid) associated with the desired tuning operation. Of course, a radio could be tuned using a navaid identifier, in which case, the operation can be easy. When tuning via just a frequency, however, the services will search the set of navaids located geographically near the current aircraft position for a navaid that has the given frequency. If no compatible navaid is in the vicinity, the radio is still tuned, but the resulting navigation information received from the radio, if any, will not be generally useful in determining a radio position.

The navigation radio tuning subsystem 350 also provides a function to automatically select the particular navaids to which the radios should be tuned that will result in the best position computation given the available navaids. This auto-tuning function is designed to execute without aircrew 12 intervention.

Whenever the navigation radios are tuned manually, via input provided by upper subsystems, the navigation radio subsystem 350 attempts to correlate the tuning request with a radio navaid that geographically located near the current aircraft position. The radio navaid geographic position information is gathered from the database management subsystem 400. When the auto-tuning function is active, the navigation radio tuning subsystem 350 periodically searches for a better set of navaids to which the navigation radios should be tuned. Inputs required for the auto-tuning function include a list of navaids from the navigation database 403 and the current aircraft position from the navigation subsystem 330. The accuracy of the aircraft position provided to the navigation radio tuning subsystem 350 determines whether the subsystem will select those navaids that will result in the optimal aircraft position calculation.

The outputs of the navigation radio tuning subsystem 350 are simply radio tuning commands that go to the navigation radios, which are modeled in the equipment management subsystem 510. It is important to the design of the FMS that no other subsystem can tune the navigation radios in the equipment management subsystem 510 directly.

Both existing military and commercial FMS systems have navigation radio tuning functions. Commercial aircraft typically have very high frequency omni-direction radio (VOR) receivers and a scanning distance measuring equipment (DME) transceivers. Military aircraft typically have VOR receivers and tactical air navigation (TACAN) transceivers.

The vertical guidance subsystem 312 contains other high-level functions

of the FMS. Its purpose is to provide commands that will cause the aircraft 13 to maintain an intended vertical flight-path. These commands can be in several forms such as climb/descend cues or control surface deflections. Regardless of the output form, the details of vertical guidance should remain invisible to users of the subsystem.

It is desirable that the vertical guidance subsystem 312 should be removable from the FMS with minimal functional impact to the remainder of the system, because certain specific aircraft do not use the functions contained within this subsystem. It is expected that such removal can impact the flight engineering subsystem 320, so it can be acceptable to be able to remove the flight engineering 320 and vertical guidance 312 subsystems as a pair.

The vertical guidance subsystem 312 should be designed such that it is able to compute steering commands based simply on the altitude-related information for each leg in an aircraft flight plan. It should have no knowledge of complex structures or patterns that can be used during the construction of the flight plan.

The vertical guidance subsystem 312 gathers the intended vertical trajectory of the aircraft from the flight planning subsystem 340 and the current aircraft-state vector from the navigation subsystem 330. The flight engineering subsystem 320 will provide some data that will act as constraints on the operation of vertical guidance subsystem 312. The vertical guidance subsystem 312 can be responsible for gathering these constraints, such as maximum climb rate, from the flight engineering subsystem 320.

The primary output of the vertical guidance subsystem 312 is a set of commands that will cause the aircraft 13 to follow the intended vertical trajectory. The vertical guidance subsystem 312 can provide these commands in the form of speed and altitude targets or in the form of pitch commands, depending on what is required by a specific aircraft autopilot. It is desirable that the vertical guidance subsystem 312 should be capable of producing both forms so that it may be reused in several specific aircraft types. The vertical guidance function 312 will also output commands that are capable of driving an autothrottle in case one is installed on the host aircraft.

The lateral guidance subsystem 311 contains another set of the high-level functions of the FMS. Its purpose is to provide commands that will cause the aircraft to maintain an intended lateral flight-path. The commands could be in several forms including turn left/right cues or control surface deflections. Regardless of the output form, the details of lateral guidance should remain invisible to users of the subsystem.

The lateral guidance subsystem 311 should be designed such that it simply requires the flight plan legs with no regard to their source. The lateral guidance subsystem 311 should treat legs that are manually built exactly the same as those that are part of a complex structure such as a standard instrument departure (SID), standard terminal arrival (STAR), or holding pattern. The lateral guidance subsystem 311 only requires the parameters of the current leg and the next two legs in the flight plan. The lateral guidance subsystem 311 gets navigation data directly from the navigation subsystem 330 and computes

guidance commands to follow the intended lateral flight-path.

The lateral guidance subsystem 311 gathers the intended lateral trajectory of the aircraft from the flight planning subsystem 340 and the current aircraft-state vector from the navigation subsystem 330. The flight engineering subsystem 320 can provide some data that acts as constraints on the operation of the lateral guidance subsystem 311. The lateral guidance subsystem 311 is responsible for gathering these constraints, such as maximum bank angle, from the flight engineering subsystem 320. The primary output of the lateral guidance subsystem 311 is a set of commands, such as roll commands, that will cause the aircraft 13 to follow the intended lateral trajectory.

The flight engineering subsystem 320 automates many of the functions that the aircraft crewmember known as a 'flight engineer' has typically performed in the past.

(a) A weight and balance function monitors and computes actual and predicted aircraft 13 gross weight and center of gravity (CG).

(b) A V-speed function computes a set of aircraft performance airspeeds, such as minimum power airspeed, based on current conditions.

(c) A flight information function computes and predicts various parameters at each point along the flight plan, such as estimated time of arrival (ETA), fuel consumption, and fuel remaining.

The flight engineering subsystem 320 resides in one of the uppermost

hierarchical layers, within the FMS architecture and uses services from subsystems residing in lower hierarchical layers to perform its duties.

Most of the algorithms used by the various functions, within the flight engineering subsystem 320, remain the same regardless of the specific aircraft type for which they are being performed. The flight engineering subsystem 320 is designed so that aircraft specific data, for example parameters used to calculate distance required for takeoff, can be loaded into an internal database, without requiring modification of the operational software. The data from the performance database 402 can be gathered by the flight engineering subsystem 320 by accessing services provided by the database management subsystem 400.

Because of the variety of tasks performed by the flight engineering subsystem 320, the inputs to this subsystem are widely varied. Many of the functions require parameters that are typically provided by an operator. Typical inputs include values such as gross weight, zero fuel weight, and outside air temperature. These types of input values are used to perform operations such as calculation of takeoff and landing data (TOLD) parameters. Certain complex functions that are performed by flight engineering subsystem, such as calculation of a top of climb point, require knowledge of the flight plan and the current state of the aircraft. The flight engineering subsystem 320 gathers these inputs from lower layer subsystems of the FMS as required. The performance database 402 provides inputs generally required by the flight engineering subsystem 320.

Two categories of outputs are provided by the flight engineering subsystem 320. The first category includes calculation results, such as TOLD, that are made available to higher subsystems. The other category includes calculation results, where these results are inserted into the flight plan, such as the calculation of ETA at every remaining waypoint. These results are inserted into the flight plan using the services of the flight planning subsystem 340.

Aircraft navigation can be described simply using the phrase, "Where am I?", as contrasted with aircraft guidance, which can be described by the phrase, "Where do I want to go?" This simple phrase gives the entire purpose of the navigation subsystem 330. Using available navigation sensors, the navigation subsystem 330 determines an 'aircraft-state vector', that includes three-dimensional geographic position and aircraft lateral and angular positions, velocities, and accelerations. The aircraft-state vector can include the basic parameter set of aircraft position, altitude, heading, course, airspeed, ground speed, winds, vertical speed, attitudes, attitude rates, and magnetic variation. This aircraft-state vector is made to all other FMS subsystems and their associated functions in accordance with the subsystem-dependency rule.

For each of the data quantities within the navigation subsystem 330, there can be several possible units of measure. For instance, position could be provided in degrees/minutes/seconds, such as earth latitude and longitude, or in radians and heading could be provided with respect to either true or magnetic north. The overall FMS design philosophy is to provide internal data in a single measurement system, such as the metric system, and to provide conversion

routines to allow the use of other units, such as nautical miles (nm).

The navigation subsystem 330 can be capable of generating the aircraft-state vector using several different navigation solutions; the actual solutions provided for a particular aircraft depend upon the suite of navigation sensors that are installed on that aircraft. The sensor suite can comprise such equipment as navigation radio, inertial navigation system (INS), and global positioning system (GPS). Regardless of the sensor(s) used to determine the aircraft-state vector, the navigation subsystem 330 must provide the basic parameters enumerated above. The navigation subsystem 330 can use complementary sensors to supplement a sensor that does not provide the entire set of parameters. In addition, the navigation subsystem 330 can use blending or filtering algorithms to refine or augment the raw sensor outputs. Regardless of the methods used, the navigation subsystem 330 should be structured such that no other FMS subsystem has visibility into the computational methods used to determine the aircraft-state vector.

One function of the navigation subsystem 330 can be to determine which navigation sensors to use in generating the aircraft-state vector, based on aircrew 12 selection. The aircrew 12 has the ability to deselect navigation sensors through the operator interface subsystem 100. The navigation subsystem 330 determines the sensor or combination of sensors that will provide the most accurate aircraft-state vector.

During normal operation, the navigation subsystem 330 generates the aircraft-state vector based on either a manually or an automatically selected

navigation sensor set. The aircraft-state vector can be populated by either acquiring data directly from sensors such as a GPS receiver, or by computation, such as by using range and bearing information from a TACAN.

Minimally, the navigation subsystem will provide aircraft-state vector using the GPS, INS, and navigation radios (using either one or more navaids). The aircraft-state vector can be derived by blending information from two or more navigation sensors.

During simulated aircraft operation, such as in an aircraft flight simulator the navigation subsystem 330 generates the aircraft-state vector parameters based upon simulation commands. Once the navigation subsystem 330 has populated the aircraft-state vector, it serves as the navigation interface to the other FMS subsystems residing at the same or higher hierarchical layers within the FMS architecture.

Functions in the flight planning subsystem 340 are responsible for building and maintaining the list of legs defining an aircraft flight plan. The flight plan can be composed of three basic parts - the origin, the destination, and one or more flight plan legs in between. A flight plan leg is defined as the path between two waypoints. Legs can be entered into the flight plan simply by specifying waypoints. In addition, complex, predefined sequences of legs, such as SID, STAR, and holding patterns can be entered into the flight plan using a single entry or a set of definition criteria. The rules for building the flight plan and the various types of flight plan legs are specified in engineering standards released by ARINC Incorporated. Specialized military flight patterns not

addressed by ARINC standards, such as aerial refueling and airdrops, can be standardized within the FMS product line.

The aircraft flight plan can be critical to the operation of the aircraft and several FMS subsystems may require access to the flight plan. It is important that a single consistent copy of the flight plan is maintained for all users of this data. Therefore, other subsystems within the FMS requiring access to the flight plan must use only those services provided by the flight planning subsystem 340. In order to avoid possible contention amongst flight planning subsystem 340 clients, requests for flight plan modifications are prioritized and queued.

One function of the flight planning subsystem 340 is to process incoming requests to either add or delete flight legs or to modify the data associated with existing legs. These requests can come from a variety of sources, including external sources such as the aircrew 12 and data-link operators, as well as other FMS subsystems. These requests are dispatched in a standard and consistent manner by the flight planning subsystem 340 regardless of the source. The dispatching of any particular request remains the same whether it was instigated by the crew or by another FMS subsystem, such as the flight engineering subsystem 320.

The flight planning subsystem 340 contains functions that allow other FMS subsystems to access information from the flight plan. All other subsystems within the FMS must use the flight planning subsystem 340 in order to obtain flight plan data. When certain components of leg data are modified, the flight planning subsystem 340 indicates, to other FMS subsystems, that the

flight plan has been modified that these modifications must either be accepted or deleted.

In prior art FMS systems, functions that are equivalent to the flight planning subsystem 340 and the aircraft guidance subsystem 310 have been tightly coupled, which made flight plan related changes unnecessarily complex and time-consuming. Advantageously, in our inventive layered subsystem architecture, the aircraft guidance subsystem 310 depends upon the flight planning subsystem 340, which has absolutely no knowledge of the aircraft guidance subsystem 310, as it resides in a higher hierarchical layer. The purpose of the aircraft guidance subsystem 310 is to guide the aircraft flight trajectory by producing steering commands and targets, while the purpose of the flight planning subsystem 340 is to maintain a list of flight plan legs. This disciplined separation of concerns should make future enhancements to the flight planning subsystem 340 much simpler and less time-consuming then such changes would have been to prior art FMS systems. Indeed, if this separation of concerns is strictly adhered to, future changes to either the flight planning 340 or aircraft guidance 310 subsystems should be minimal.

Expected flight plan related enhancements include additional specialized flight patterns such as the previously mentioned military aerial refueling and airdrop patterns. In the case of many of these specialized patterns, it may be desirable to install the new functionality in the automated-crewmember subsystem 360, which has knowledge of the entire FMS. Adding new functionality to the automated-crewmember subsystem 360 will not affect other

existing core FMS subsystems, thereby minimizing side effects and required regression testing.

The navigation database subsystem 403 is primarily responsible for management of a worldwide navigation database. Generally, in the context of aircraft operations, when one references the 'database', the reference is actually being made to this worldwide navigation database. There are several other databases that are important to the overall functioning of the FMS. These include the communications database subsystem 401 and the performance database subsystem 402. The communications database subsystem 401 primarily contains functions to access frequency lists used during communications radio tuning. The performance database subsystem 402 contains functions to access information regarding the characteristics of the airframe and engines for a specific aircraft type.

The navigation database subsystem 403 is responsible for retrieving data from the worldwide navigation database as requested. The navigation database management subsystem must be capable of retrieving a single entity from the database or a list of related entries from the database. For instance, a function can be provided to retrieve information associated with a single navaid. Another function can be provided to retrieve a list of the navaids within fifty (50) nautical miles of a given position. In other examples, the entity requested could normally be in the form of a list; as in the case of an airway, that contains several waypoints. The key to successful development of the navigation database subsystem 403 can be the construction of a relatively standard interface that is

targeted toward the domain requirements. The implementation, then, of that interface can easily vary with the database source data and no other parts of the system would be affected. The navigation database subsystem 403 will typically contain navaids, SID, STAR, predefined waypoints, airways, airports, runways, military training routes (MTR), and refueling routes.

The input processing required of the navigation database subsystem 403 is, simply, the interpretation of database requests and the retrieval of the corresponding data from the internal data structures. The output processing in the navigation database subsystem 403 is, simply, to provide the data that was requested.

The performance database subsystem 402 is responsible for retrieving data from the performance database as requested. This performance database subsystem 402 can be characterized by a general set of data that is required by the functions within the flight engineering subsystem 320. The format and structure of the actual performance data will be hidden by the standard interface that is available to other FMS subsystems. As with the navigation database subsystem 403, the key to successful development of the performance database subsystem 402 can be the construction of a standard interface that is targeted toward the domain requirements. The implementation, then, of that interface can easily vary with the database source data and no other parts of the system would be affected. The performance database subsystem 402 will contain functions to retrieve airframe characteristics such as drag, V-speeds (values and limits), altitude limits, and lift coefficients. The performance database subsystem

402 will also contain functions to retrieve engine characteristics such as thrust-to-power settings and fuel-to-power settings.

The input processing associated with the performance database subsystem 402 includes interpretation of database requests and retrieval of data from internal data structures corresponding to those requests. The output processing in the performance database subsystem 402 is, generally, to provide the data that was requested.

The communication database subsystem 401 is responsible for retrieving data from the communication database as requested. The communication database primarily contains lists of frequencies to be used during the tuning of communication radios. This database is somewhat different than either the performance or the navigation databases, in that other FMS subsystems have the ability to both read and write to the communication database.

The input processing required of this subsystem comprises retrieving communications data for other FMS subsystems and storing communications data from the other FMS subsystems into the database structures. The output processing in this subsystem includes of the return of data that was requested by higher layer subsystems.

The computing platform services subsystem 600 contains sub-subsystems that are dependent upon the computing platform in some way. The subsystems within the computing platform services subsystem 600 exist entirely to isolate the true applications of the FMS product line from the underlying computing platform. Therefore, the sub-subsystems contained within the

computing platform services subsystem must be implemented such that the interface to those subsystems can remain unchanged even when the subsystems are modified to accommodate a specific computing platform.

The equipment management subsystem 500 contains models of all external equipment that interface with the FMS, such as radios, navigation sensors, aircraft autopilot, and CDU. All communication with the external equipment is to be managed using the services provided by the equipment management subsystem 500.

The overriding philosophy of the FMS with respect to equipment management is that it should provide a standard interface to a standard piece of equipment. The virtual equipment interface subsystem 510 contains the services that are typically provided by equipment of this type and the interface is considered to be a part of the core FMS product line. The implementation of this virtual interface is then based on the physical equipment that is actually a part of the system.

There is one other aspect of the services provided by a standard interface that merits discussion - required versus optional services. Because the philosophy of the core FMS product line is to be reused verbatim, it will require that certain data be available. This necessitates the concept of standard interfaces to standard equipment. The fact that the standard interface provides services that are typical to that type of equipment, however, does not imply that every instance of that type of equipment actually does provide that service. For instance, there can be an INS that does not provide magnetic variation even

though we consider that a typical service of an INS. In order to account for an INS that does not provide a magnetic variation, the core FMS product line will provide a computed magnetic variation as a standard service. Therefore, there are two classes of services provided by standard interfaces - required services and optional services. When the implementation of a standard interface is performed, for a specific project, there are some services within the standard interface that must be implemented and others that can be left unimplemented. Each standard interface will enumerate those services that are required and those that are optional. The determination of whether a service can be required or optional can be simply based on whether or not it can be used by the core product line.

It is important to note that the interfaces to standard equipment are generally logical models of that piece of equipment. Therefore, the virtual equipment interface subsystem 510 might specify not only services that are provided by the equipment, but also higher-level services that are related to that equipment. One example of this additional functionality occurs within the VOR receiver interface. Generally, a physical VOR receiver is tuned by simply providing a frequency. Another model of the ideal VOR includes the identifier and position of the navaid to which the radio is tuned, which allows the use of range and bearing information returned from the receiver to determine an aircraft position. Therefore, the service to tune the VOR receiver in the standard interface would include frequency, position, and identifier.

Each service provided by the standard interfaces should contain not only

the data to be conveyed, but also an indication of the validity of the signal. This can be especially important for services that provide data to the FMS product line applications from external equipment. For instance, an air data computer (ADC) interface can provide true airspeed to the applications. In addition to providing the airspeed value, the ADC interface should provide an indication of the validity of that airspeed value. The validity can be based on such criteria as data freshness, physical equipment condition, and physical interface condition.

There are two aspects of input processing within the equipment management subsystem 500 corresponding to those inputs that are received from higher subsystems and to those that are received from the interfaced equipment. For the former, data is generally destined for output to the interfaced equipment. Since this subsystem contains logical models of interfaced equipment, it is not necessary that each input, from another FMS subsystem, results in an output to equipment. For the latter, input data is gathered from the interfaced equipment and made available for access via the logical interface provided for each piece of equipment.

As with the input processing, there are two aspects of output processing. Those aspects are the reverses of the two aspects of input processing. Output data are sent to the interfaced equipment as required. The fact that data are made available to other FMS subsystems, as described above, is another way of saying that the data is output from this subsystem.

The redundancy management subsystem 520 provides a backup contingency in the event that there is a failure of the processor hosting the FMS

system. The FMS product line operates in an active/standby (hot spare) configuration whereby one processor is actively executing the functionality of the system whilst the other processor is standing by and listening. The standby processor will to take over the workload of the active processor should it become necessary. The standby processor is listening and ready by virtue of periodic transfers of data from the active processor that will enable the standby processor to take over for the active processor and resume its work. The active/standby configuration assumes that the aircraft is wired such that the necessary data transfers can take place and that the processors can swap roles.

The redundancy management subsystem 520 takes in indications of processor condition to determine whether the active processor is fit to continue in its role. It also gathers up the data necessary to effect a reversal of processor roles and prepares it for transfer to the standby processor.

In addition to the transfer data, the redundancy management subsystem 520 outputs an indication as to whether the processors should exchange roles. The present invention takes advantage of both hot-spare and dual philosophies on various FMS programs for specific aircraft types.

The domain services subsystem 610 provides a collection of common type definitions and primitive utilities that are used throughout the FMS product line system. The definitions provided by the domain services subsystem 610 are specifically related to the domain in which the system operates. It can be important to realize that the entities exported from this subsystem are not simply a collection of software data types that make it easy for implementers of the

system. Rather, this subsystem provides services that are provided in the application domain. Examples of elements within the domain services subsystem 610 are software data types defining latitude and longitude, utilities to compute a distance between two points on the earth, and utilities to convert angular values between radians and degrees. All other subsystems within the FMS are expected to use the functions provided by the domain services subsystem 610 rather than define their own. This subsystem should not contain anything that is either platform or aircraft dependent; thus preventing the client systems from becoming dependent on the aircraft and platform. The domain services subsystem 610 accepts input parameters such as bearings, distances, and latitudes/longitudes and performs the requested processing on them, but none of the processing produces values that become resident in this subsystem. Outputs of the domain services subsystem 610 are software type definitions, which are universal to the FMS product line domain and computed values from utilities. The computed outputs are not resident in this subsystem, but are merely returned to clients for their use.

The operating system services subsystem 620 can buffer the rest of the FMS subsystems from the host-computing environment such that the core FMS will not be dependent on any particular host or platform.

The input to the operating services subsystem 620 can be, for example, an Ada tasking model for the FMS system. This model is translated into the underlying equivalent model of the platform, which can be independent of Ada tasks. An output of the operating services subsystem 620 can be a condition

indication that is the result of built-in-tests of the host-computing environment. The FMS subsequently uses this condition indication as a component in other condition indications, for instance, the status of data bus communications.

The embodiments and examples set forth herein are presented to best explain the present invention and its practical application and to thereby enable those skilled in the art to make and use the invention. Those skilled in the art, however, will recognize that the foregoing description and examples have been presented for the purpose of illustration and example only. Other variations and modifications of the present invention will be apparent to those of skill in the art, and it is the intent of the appended claims that such variations and modifications be covered. The description as set forth is not intended to be exhaustive or to limit the scope of the invention. Many modifications and variations are possible in light of the above teaching without departing from the spirit and scope of the following claims. It is contemplated that the use of the present invention can involve components having different characteristics as long as the principle of layered subsystem architecture for a flight management system is followed. It is intended that the scope of the present invention be defined by the claims appended hereto.

CLAIMS:

1. A method of controlling and coordinating communication between diverse modules in a flight management system (FMS), wherein the diverse modules are configured as a layered subsystem hierarchical architecture, the method comprising the steps of:

coordinating communication between operational subsystems from the layered subsystem hierarchical architecture, incorporated into the FMS based on a predefined subsystem dependency rule, said subsystem-dependency rule comprising allowing a given subsystem to depend only upon another subsystem in a same or lower hierarchical layer; and

verifying communication based on the subsystem-dependency rule during implementation and execution of software associated with said modules.

2. The method of Claim 1, wherein module operation is continuously checked against the predefined dependency rule to ensure that module operation reflects module dependencies throughout the FMS, mirroring corresponding real-world dependencies as defined by FMS subsystem-dependency rules.

3. A flight management system (FMS) including a layered-subsystem architecture residing on a computing platform, the FMS architecture comprising:

a means for enforcement of a predefined subsystem dependency rule, said subsystem-dependency rule comprising allowing a given subsystem to depend only upon another subsystem in a same or lower hierarchical layer;

an operator interface subsystem for receiving input from an operator and presenting output to the operator;

a communications subsystem for managing and interpreting communications to and from aircraft and equipment used for communication;

a flight management subsystem for managing functions associated with operation of an aircraft; and

a database management subsystem for managing and providing access to a plurality of databases housed within the FMS architecture.

4. The system of Claim 3, wherein said communication, flight management and database subsystems each further comprise a plurality of application functions that are independent of the operator interface subsystem.

5. The system of Claim 3, wherein the operator interface, communication, flight management and database subsystems are dependent upon the computing platform to access an external environment.

6. The system of Claim 3, further comprising an equipment management subsystem that provides a domain-level interface to a plurality of equipment upon which the performance of application functions within the operator interface, communication, flight management and database subsystems is based.

7. The system of Claim 3, further comprising an equipment management subsystem that is responsible for implementing a high-level interface using a plurality of low-level facilities provided by the computing platform.

8. The system of Claim 3, wherein said navigation subsystem further comprises:

a flight planning subsystem to build and maintain a list of legs defining a flight plan; and

an aircraft guidance subsystem to guide an aircraft flight trajectory by producing steering commands and targets.