# METHOD AND APPARATUS FOR LOADING SOFTWARE AIRCRAFT PARTS

## BACKGROUND INFORMATION

### 1. Field:

The present disclosure relates generally to an improved data processing system and in particular to a method and apparatus for managing software for aircraft. Still more particularly, the present disclosure relates to a computer implemented method, apparatus, and computer usable program product for managing loadable software aircraft parts.

### 2. Background:

Modern aircraft are extremely complex. For example, an aircraft may have many types of electronic systems on board. A particular electronic system on an aircraft also may be referred to as a line replaceable unit (LRU). A line replaceable unit may take on various forms. A line replaceable unit may be, for example, without limitation, a flight management system, an autopilot, an in flight entertainment system, a communication system, a navigation system, a flight controller, a flight recorder, and a collusion of boarding system.

A line replaceable unit may use software or programming to provide the logic or control for various operations and functions. The software and other information used in a line replaceable unit are commonly treated as parts in the airline industry. For example, a software application for use in the line replaceable unit on an aircraft may be tracked separately from the line replaceable unit and referred to as a loadable software aircraft part (LSAP) or as a software aircraft part.

Software aircraft parts may be loaded into a line replaceable unit as part of the delivery of the aircraft from the manufacturer or as part of a maintenance operation. Software aircraft parts may be loaded by various techniques. For example, a computer readable media, such as a floppy disk, a flash memory drive, or a digital versatile disk (DVD) may be taken to the aircraft and loaded into the computing or avionics system for the aircraft.

Other techniques may involve transmitting the software aircraft part to the computing system through a communications link established between the computing system and the source of the software aircraft part. In yet other techniques, a portable data processing system, such as a laptop, may be carried to the aircraft to transfer the software aircraft part.

Different aircraft may require different software aircraft parts. For example, different types of aircraft may require different software aircraft parts. As another example, specific aircraft of the same type also may require different software aircraft parts. For example, a Boeing 777 may have different versions. With this example, one version of this aircraft may use General Electric engines while another version of the aircraft may use Rolls Royce engines. A software aircraft part designed for a line replaceable unit used to control parameters, such as thrust settings, for engines is different for these different versions of a Boeing 777. As a result, software aircraft parts may be specific for a particular aircraft even within the same type or model.

Currently, operators verify that a software aircraft part is appropriate for a particular target aircraft based on an operator checking documents about the software aircraft part to ensure that the target aircraft is an appropriate aircraft for the software aircraft part. Once the appropriate software aircraft part has been identified for a target aircraft, the software aircraft part may be placed on a media for transport to the aircraft. This type of process is cumbersome and requires careful checking of software aircraft parts to ensure that the proper software aircraft part reaches the appropriate target aircraft.

## SUMMARY

The advantageous embodiments provide a computer implemented method, apparatus, and computer usable program code for managing a software aircraft part. In one advantageous embodiment, a computer implemented method identifies an aircraft from a set of target aircraft to form a target aircraft. Software for use in the aircraft is identified. A determination is made as to whether compatibility exists between the software and the target aircraft. The software aircraft part is created comprising the software and metadata relating usability of the software aircraft part in the target aircraft

in response to a determination that compatibility exists between the software and the target aircraft.

In another advantageous embodiment, a software aircraft part is received for use in an aircraft data processing system on an aircraft, wherein the software aircraft part comprises a set of software components and metadata identifying a set of target aircraft on which the software aircraft part can be used. A determination is made as to whether the software aircraft part can be used in the aircraft data processing system on the aircraft using the metadata. The software aircraft part is loaded into the aircraft data processing system in response to a determination that the software aircraft part can be used in the aircraft data processing system.

In yet another advantageous embodiment, a computer program product comprises a computer readable media and contains program code stored on the computer readable medium. Program code is present for identifying an aircraft from a set of target aircraft to form a target aircraft. The program code also is present for identifying software for use in the aircraft. Program code is present for determining whether compatibility exists between the software and the target aircraft. Program code is present to create the software aircraft part comprising the software and metadata relating usability of the software aircraft part in the target aircraft in response to a determination that compatibility exists between the software and the target aircraft.

In still yet another advantageous embodiment, a computer program product computer program product comprises a computer readable media and contains program code stored on the computer readable medium. Program code is present to receive a software aircraft part for use in an aircraft data processing system on an aircraft, wherein the software aircraft part comprises a set of software components and metadata identifying a set of target aircraft on which the software aircraft part can be used. Program code also is present to determine whether the software aircraft part can be used in the aircraft data processing system on the aircraft using the metadata. Program code is present to load the software aircraft part into the aircraft data processing system in response to a determination that the software aircraft part can be used in the aircraft data processing system of the aircraft.

In still yet another advantageous embodiment, an aircraft data processing system comprising a bus, communication unit, a storage device, and a program unit. Program

code is located on the storage device. The processor unit executes the computer usable program to receive a software aircraft part for use in an aircraft data processing system on an aircraft, wherein the software aircraft part comprises a set of software components and metadata identifying a set of target aircraft on which the software aircraft part can be used; determine whether the software aircraft part can be used in the aircraft data processing system on the aircraft using the metadata; and load software aircraft part onto the aircraft data processing system in response to a determination that the software aircraft part can be used in the aircraft data processing system.

The features, functions, and advantages can be achieved independently in various embodiments of the present disclosure or may be combined in yet other embodiments in which further details can be seen with reference to the following description and drawings.

## BRIEF DESCRIPTION OF THE DRAWINGS

The novel features believed characteristic of the advantageous embodiments are set forth in the appended claims. The advantageous embodiments, however, as well as a preferred mode of use, further objectives and advantages thereof, will best be understood by reference to the following detailed description of an advantageous embodiment of the present disclosure when read in conjunction with the accompanying drawings, wherein:

**Figure 1** is a pictorial representation of a network of data processing systems in which the advantageous embodiments may be implemented;

**Figure 2** is a diagram of a data processing system in accordance with an advantageous embodiment;

**Figure 3** is a diagram illustrating components used to create software aircraft parts in accordance with an advantageous embodiment;

**Figure 4** is an example of an aircraft data processing system in accordance with an advantageous embodiment;

**Figure 5** is a diagram illustrating components in an avionics system used to load a software aircraft part in accordance with an advantageous embodiment;

**Figure 6** is a high level flowchart of a process for creating a software aircraft part in accordance with an advantageous embodiment;

**Figure 7** is a high level flowchart of a process for loading a software aircraft part in accordance with an advantageous embodiment;

**Figure 8** is a flowchart of a process for creating a software aircraft part in accordance with an advantageous embodiment; and

**Figure 9** is a flowchart of a process for loading a software aircraft part in accordance with an advantageous embodiment.

## DETAILED DESCRIPTION

With reference now to the figures and in particular with reference to **Figures 1-2**, exemplary diagrams of data processing environments are provided in which the advantageous embodiments may be implemented. It should be appreciated that **Figures 1-2** are only exemplary and are not intended to assert or imply any limitation with regard to the environments in which different embodiments may be implemented. Many modifications to the depicted environments may be made.

With reference now to **Figure 1**, a pictorial representation of a network of data processing systems is depicted in which the advantageous embodiments may be implemented. Network data processing system **100** is a network of computers in which embodiments may be implemented. Network data processing system **100** contains network **102**, which is the medium used to provide communications links between various devices and computers connected together within network data processing system **100**. Network **102** may include connections, such as wire, wireless communication links, or fiber optic cables.

In the depicted example, server **104** and server **106** connect to network **102** along with storage unit **108**. In addition, clients **110**, **112**, and **114** connect to network **102**. These clients **110**, **112**, and **114** may be, for example, personal computers or network computers. In the depicted example, server **104** provides data, such as boot files, operating system images, and applications to clients **110**, **112**, and **114**. Clients **110**, **112**, and **114** are clients to server **104** in this example. Aircraft **116** also is a client that may exchange information with clients **110**, **112**, and **114**. Aircraft **116** also may

exchange information with servers **104** and **106**. For example, aircraft **116** may receive software aircraft parts from servers **104** and/or **106**. Aircraft **116** may exchange data with different computers through a wireless communications link while in-flight or any other type of communications link while on the ground. In these examples, server **104**, server **106**, client **110**, client **112**, and client **114** may be computers in different geographic locations. Network data processing system **100** may include additional servers, clients, and other devices not shown.

In these examples, server **106** may be located in manufacturer **118**. Manufacturer **118** may create software aircraft parts for loading or distribution to aircraft **116** from server **106**. As another example, airline **120** may store software aircraft parts on server **104**. These software aircraft parts may be created by airline **120** or may be received from manufacturer **118**. The software aircraft parts stored on server **104** may be distributed or transmitted to an aircraft, such as aircraft **116**. The distribution of the software aircraft parts may occur through the transmission of these software aircraft parts through network **102**. In other advantageous embodiments, software aircraft parts may be transferred through the use of portable devices, such as data loaders, or the use of media, such as a digital versatile disk or a flash memory drive.

In the depicted example, network data processing system **100** is the Internet with network **102** representing a worldwide collection of networks and gateways that use the Transmission Control Protocol/Internet Protocol (TCP/IP) suite of protocols to communicate with one another. Of course, network data processing system **100** also may be implemented as a number of different types of networks, such as for example, an intranet, a local area network (LAN), or a wide area network (WAN). **Figure 1** is intended as an example, and not as an architectural limitation for different embodiments.

Turning now to **Figure 2**, a diagram of a data processing system is depicted in accordance with an advantageous embodiment. Data processing system **200** is an example of a data processing system that may be used to implement servers and clients, such as server **104** and client **110**. Further, data processing system **200** is an example of a data processing system that may be found in aircraft **116** in **Figure 1**.

In this illustrative example, data processing system **200** includes communications fabric **202**, which provides communications between processor unit **204**, memory **206**,

persistent storage **208**, communications unit **210**, input/output (I/O) unit **212**, and display **214**.

Processor unit **204** serves to execute instructions for software that may be loaded into memory **206**. Processor unit **204** may be a set of one or more processors or may be a multi-processor core, depending on the particular implementation. Further, processor unit **204** may be implemented using one or more heterogeneous processor systems in which a main processor is present with secondary processors on a single chip. As another illustrative example, processor unit **204** may be a symmetric multi-processor system containing multiple processors of the same type.

Memory **206**, in these examples, may be, for example, a random access memory or any other suitable volatile or non-volatile storage device. Persistent storage **208** may take various forms depending on the particular implementation. For example, persistent storage **208** may contain one or more components or devices. For example, persistent storage **208** may be a hard drive, a flash memory, a rewritable optical disk, a rewritable magnetic tape, or some combination of the above. The media used by persistent storage **208** also may be removable. For example, a removable hard drive may be used for persistent storage **208**.

Communications unit **210**, in these examples, provides for communications with other data processing systems or devices. In these examples, communications unit **210** is a network interface card. Communications unit **210** may provide communications through the use of either or both physical and wireless communications links.

Input/output unit **212** allows for input and output of data with other devices that may be connected to data processing system **200**. For example, input/output unit **212** may provide a connection for user input through a keyboard and mouse. Further, input/output unit **212** may send output to a printer. Display **214** provides a mechanism to display information to a user.

Instructions for the operating system and applications or programs are located on persistent storage **208**. These instructions may be loaded into memory **206** for execution by processor unit **204**. The processes of the different embodiments may be performed by processor unit **204** using computer implemented instructions, which may be located in a memory, such as memory **206**. These instructions are referred to as, program code, computer usable program code, or computer readable program code

that may be read and executed by a processor in processor unit **204**. The program code in the different embodiments may be embodied on different physical or tangible computer readable media, such as memory **206** or persistent storage **208**.

Program code **216** is located in a functional form on computer readable media **218** and may be loaded into or transferred to data processing system **200** for execution by processor unit **204**. Program code **216** and computer readable media **218** form computer program product **220** in these examples. In one example, computer readable media **218** may be in a tangible form, such as, for example, an optical or magnetic disc that is inserted or placed into a drive or other device that is part of persistent storage **208** for transfer onto a storage device, such as a hard drive that is part of persistent storage **208**. In a tangible form, computer readable media **218** also may take the form of a persistent storage, such as a hard drive or a flash memory that is connected to data processing system **200**. The tangible form of computer readable media **218** is also referred to as computer recordable storage media.

Alternatively, program code **216** may be transferred to data processing system **200** from computer readable media **218** through a communications link to communications unit **210** and/or through a connection to input/output unit **212**. The communications link and/or the connection may be physical or wireless in the illustrative examples. The computer readable media also may take the form of non-tangible media, such as communications links or wireless transmissions containing the program code.

The different components illustrated for data processing system **200** are not meant to provide architectural limitations to the manner in which different embodiments may be implemented. The different illustrative embodiments may be implemented in a data processing system including components in addition to or in place of those illustrated for data processing system **200**. Other components shown in **Figure 2** can be varied from the illustrative examples shown.

For example, a bus system may be used to implement communications fabric **202** and may be comprised of one or more buses, such as a system bus or an input/output bus. Of course, the bus system may be implemented using any suitable type of architecture that provides for a transfer of data between different components or devices attached to the bus system. Additionally, a communications unit may include

one or more devices used to transmit and receive data, such as a modem or a network adapter. Further, a memory may be, for example, memory **206** or a cache such as found in an interface and memory controller hub that may be present in communications fabric **202**.

The different advantageous embodiments recognize that currently existing systems for loading software aircraft parts onto a computing system for an aircraft require many tedious checks to ensure that errors do not occur when a software aircraft part is selected for loading onto a computing system of an aircraft.

Turning now to **Figure 3**, a diagram illustrating components used to create software aircraft parts is depicted in accordance with an advantageous embodiment. In this example, software management environment **300** is an example of a software management environment that may be located at an entity, such as manufacturer **118** or airline **120** in **Figure 1**.

Additionally, software management environment **300** may be found at locations for other types of entities involved with managing software aircraft parts for aircraft. For example, software management environment **300** also may be located at a maintenance facility for a maintenance organization. Software management environment **300** includes software aircraft part process **302**, software components library **304**, metadata repository **306,** software aircraft part library **308**, policy **310**, and software aircraft part **312**. Software aircraft part process **302** creates software aircraft parts, such as software aircraft part **312** in these examples.

Software aircraft part **312** contains a set of software components **314** and metadata **316**. A set as used herein refers to one or more items. A set of software aircraft components **314** is one or more software components. As used herein, a software component may be any information in a computer usable form. In particular, this information is in a functional form that is usable by a computer, such as an aircraft data processing system.

A software component may be, for example, an executable file, a configuration file, a data file, or a text file. Metadata **316** describes a set of aircraft on which software aircraft part **312** may be used. In the different advantageous embodiments, software aircraft part **312** differs from other currently available software aircraft parts in that this part includes metadata **316**.

Software aircraft part process **302** manages software aircraft parts, such as those found in software aircraft part library **308**. The management of software aircraft parts may include, for example, creating software aircraft parts, organizing software aircraft parts, sending software aircraft parts for loading on to an aircraft data processing system, or any other suitable manipulation of software.

Software aircraft part library **308** contains a set of software aircraft parts, such as software aircraft part **312**. Software components library **304** contains software components that may be used to create and/or modify a software aircraft part, such as software aircraft part **312**.

Metadata repository **306** contains information describing target aircraft on which a software aircraft part may be installed. Software aircraft part process **302** may be used to define metadata within metadata repository **306**. This metadata defines the particular aircraft on which different software components within software components library **304** may be used.

In constructing a software aircraft part, an operator of software aircraft part process **302** may select one or more target aircraft. The target aircraft may be based on the model and type of aircraft as well as a particular aircraft identified by a tail number. Based on the selection of the target aircraft, an operator selects a target aircraft for a particular part. Policy **310** determines what software components may be used from software components library **304** to create software aircraft part **312** for the target aircraft.

In some advantageous embodiments, some software components, tools, or features in the software components within software components library **304** may not be usable, depending on the selection of the target aircraft. Further, if the operator resets or selects a different type of target aircraft, software aircraft part process **302** uses policy **310** to determine whether the selected software components can still be used with the new target aircraft.

If the components cannot be used with the new target aircraft, the session is reset. In other advantageous embodiments, certain features may be selectively disabled within the software components or certain software components may be removed from the software aircraft part being created instead of resetting the session.

After a set of software components have been selected from software components library **304** for software aircraft part **312**, metadata **316** is added to software aircraft part **312** from metadata repository **306**. The content of metadata **316** is selected based on the target aircraft selected by the operator of software aircraft part process **302**. Metadata **316** is a software component that is part of software aircraft part **312**.

Further, checksum **318** is generated based on a set of software components **314** and metadata **316**. This checksum ensures that software aircraft part **312** has not been modified. The modification may occur, for example, from corruption or errors in a software component during transfer and/or loading of software aircraft part **312**.

Once software aircraft part **312** is completed, software aircraft part **312** may be added to software aircraft part library **308**. Further, software aircraft part **312** may be sent to an aircraft for use in a set of line replaceable units on the target aircraft.

Turning now to **Figure 4**, an example of an aircraft data processing system is depicted in accordance with an advantageous embodiment. In this example, electronic flight bag **400** is an example an aircraft data processing system that may be found in an aircraft, such as aircraft **116** in **Figure 1**. Electronic flight bag **400** may be a portable device. In other advantageous embodiments, electronic flight bag **400** may be integrated into the aircraft and connected to the aircraft network.

In this example, electronic flight bag **400** includes data loader **402**, mass storage **404**, and line replaceable unit **406**.

Data loader **402** may be used to receive software aircraft parts that may be transferred to electronic flight bag **400**. When a software aircraft part, such as software aircraft part **408**, is received, data loader **402** stores software aircraft part **408** in mass storage **404**. Software aircraft part **408** is a software aircraft part similar to software aircraft part **312** in **Figure 3**. Mass storage **404** may be, for example, any storage device, such as a hard disk drive and/or a solid state drive. Software aircraft part **408** is not used or loaded into line replaceable unit **406** until a determination is made as to whether software aircraft part **408** is usable with line replaceable unit **406**.

Data loader **402** is also used to load a software aircraft part, such as software aircraft part **408**, into line replaceable unit **406** for use by line replaceable unit **406**.

A data loader is a device that is used to load software aircraft parts into an aircraft data processing system. More specifically, data loaders may be used to load a software aircraft part into a line replaceable unit. A data loader may be constructed based on a data loader standard from Aeronautical Radio, Incorporated (ARINC). In these examples, the standard is ARINC 615, which is a set of standards covering data loading and for transferring software and data to and from various avionic devices. A data level may be a functional component or a hardware device, depending on the particular implementation.

In these advantageous embodiments, data loader **402** includes processes to ensure that software aircraft part **408** is the correct or appropriate software aircraft part for use by line replaceable unit **406**.

The metadata within software aircraft part **408** is checked by data loader **402** against information about the aircraft. Data loader **402** determines whether the metadata matches to the information located in configuration file **416**. This information about the aircraft may be obtained directly by interrogating other components within the aircraft.

If data loader **402** determines that software aircraft part **408** can be used in line replaceable unit **406**, data loader **402** then loads software aircraft part **408** into line replaceable unit **406** for use. If software aircraft part **408** is not suitable for use based on the comparison of the metadata to configuration information, then software aircraft part **408** is not loaded into line replaceable unit **406**.

In these examples, the different components within electronic flight bag **400** are illustrated as functional components. These components may be implemented in hardware and/or software, depending on the particular implementation. For example, data loader **402** and line replaceable unit **406** may be different electronic components within electronic flight bag **400**. In other advantageous embodiments, data loader **402** and line replaceable unit **406** may be software components or software parts executing within a single data processing system within electronic flight bag **400**.

The components illustrated are depicted for purposes of illustrating different features of the advantageous embodiments. Other components may be present in addition to or in place of these in the electronic flight bag **400**. For example, electronic flight bag **400** also may include a display, an output for a display device, user input

controls, a communications unit to communicate with other data processing systems, and other suitable components.

Turning now to **Figure 5**, a diagram illustrating components used to process the loading of a software aircraft part is depicted in accordance with an advantageous embodiment. In this example, avionics system **500** is an example of an aircraft network that may be found within aircraft **116** in **Figure 1**. Avionics system **500** includes aircraft network **502**, aircraft data processing system **504**, line replaceable unit **506**, line replaceable unit **508**, and line replaceable unit **510**.

Avionics system **500** is a network that provides communications between aircraft data processing system **504** and a different line replaceable unit within avionics system **500**. In these examples, aircraft data processing system **504** includes storage device **512** and data loader **514**. Storage device **512** stores software aircraft parts, such as software aircraft part **516** and configuration file **518**. Storage device **512** also may be referred to as a mass storage in some examples.

When a software aircraft part, such as software aircraft part **516**, is received by aircraft data processing system **504**, a part is stored within storage device **512**. In these examples, software aircraft part **516** is not loaded into a line replaceable unit until an operator, such as a mechanic, initiates the loading of the software aircraft part. In response to such a request, data loader **514** checks the metadata within software aircraft part **516** to determine whether software aircraft part **516** may be loaded into the selected line replaceable unit.

The determination of whether software aircraft part **516** can be loaded into a line replaceable unit is made, in these examples, by comparing the metadata within software aircraft part **516** to information about the aircraft. This information is stored in configuration file **518**. Of course, the information about the aircraft may also be obtained by interrogating various components within avionics system **500**. For example, data loader **514** may interrogate the line replaceable unit on which software aircraft part **516** is to be loaded. If the comparison indicates that software aircraft part **516** can be loaded, software aircraft part **516** is loaded into the line replaceable unit by the data loader. Otherwise, software aircraft part **516** remains unloaded in response to the request.

Avionics system **500** may include other devices and software components, depending on the particular implementation. Only some of the components actually present in avionics system **500** are depicted for purposes of illustrating features of the different advantageous embodiments. For example, avionics system **500** also may include an electronic flight bag connected to aircraft network **502**. This electronic flight bag may be, for example, electronic flight bag **400** in **Figure 4**. Other examples of components that may be found in avionics system **500** include, for example, wireless communication units, user input devices, displays, and other suitable components.

Turning now to **Figure 6**, a high level flowchart of a process for creating a software aircraft part is depicted in accordance with an advantageous embodiment. The process illustrated in **Figure 6** may be implemented in an environment, such as software management environment **300** in **Figure 3**. In particular, the different processes may be implemented within software aircraft part process **302** in **Figure 3**.

The process begins by identifying an aircraft from a set of target aircraft to form a target aircraft (operation **600**). The selection of the set of target aircraft may take different forms. For example, the selection may be specific aircraft identified by tail numbers. In other embodiments, the selection may be a set of aircraft types. In yet other embodiments, the selection of the set of target aircraft may be a combination of specific aircraft and types of aircraft. This step selects an aircraft as the target for the software for aircraft part. Of course, more than one aircraft may be selected as the target aircraft, depending on the particular implementation. The process identifies software for use in an aircraft (operation **602**).

A determination is made as to whether compatibility exists between the identified software and the target aircraft (operation **604**). The determination in operation **604** may be made using a policy, such as policy **310** in **Figure 3**. The policy contains rules as to what software components may be used with particular aircraft. If compatibility exists, a software aircraft part is created containing the software and the metadata relating to usability of the software aircraft part in the target aircraft (operation **606**), with the process terminating thereafter. If compatibility does not exist, the process terminates without creating the software aircraft part.

Turning now to **Figure 7**, a high level flowchart of a process for loading a software aircraft part is depicted in accordance with an advantageous embodiment.

The process illustrated in **Figure 7** may be implemented in an aircraft data processing system or avionics system on an aircraft. For example, the process illustrated in **Figure 7** may be implemented in electronic flight bag **400** in **Figure 4** or avionics system **500** in **Figure 5**.

The process begins by receiving a request to use the software aircraft part in an aircraft data processing system for a selected aircraft (operation **700**). The request may take different forms, depending on the particular implementation. For example, if the software aircraft part is already located on a mass storage or storage device on the aircraft, the request may come from an operation to load the software aircraft part into an aircraft data processing system, such as a line replaceable unit on the selected aircraft. In other advantageous embodiments, the request may take the form of a receipt of the software aircraft part in an avionics environment or an aircraft data processing system containing a line replaceable unit.

A determination is made as to whether the software aircraft part is suitable for use in the selected aircraft using the metadata in the software aircraft part (operation **702**). In operation **702**, the determination is made by comparing the metadata with information about the aircraft. This information may be, for example, an identification of configurations or components within the aircraft.

These configurations or components may be, for example, a particular engine type, a type of line replaceable unit, an operating system located on a line replaceable unit, or a type of hydraulic system on the aircraft. The information may be obtained through various mechanisms. For example, a configuration file containing the information may be located on the aircraft. In other advantageous embodiments, various systems within the aircraft may be interrogated to obtain information.

If the software aircraft part is suitable for use, the software aircraft part is loaded for use in the aircraft data processing system (operation **704**), with the process terminating thereafter. In other examples, the software aircraft part may be loaded into a line replaceable unit in the aircraft data processing system. With this type of implementation, the line replaceable unit may be a software component, and/or a hardware device. Turning back to operation **702**, if the software aircraft part is not suitable for use on the selected aircraft, the process terminates.

This determination also may be performed during other management operations with respect to the software aircraft part. For example, a determination of the compatibility of the target aircraft and the software aircraft part may be made when an operator at an airline or manufacturer decides to send the software aircraft part to a particular aircraft. This determination also may be made during the transfer of the software aircraft part. For example, when the software aircraft part is transferred from a manufacturer to an airline, a check may be made as to whether the software aircraft part is suitable for the target aircraft.

Turning now to **Figure 8**, a flowchart of a process for creating a software aircraft part is depicted in accordance with an advantageous embodiment. The process illustrated in **Figure 8** may be implemented in a soft environment, such as software management environment **300** in **Figure 3**. In particular, the process illustrated in **Figure 8** may be implemented in a software component, such as software aircraft part process **302** in **Figure 3**.

The process begins by receiving a selection of a set of aircraft (operation **800**). The set of aircraft is a set of one or more aircraft for which the software aircraft part is to be used. The process then receives a selection of a software component (operation **802**). In these examples, the selection may be, for example, an executable file, a dynamic link library, a configuration file, an electronic manual, or some other data for use in the software aircraft part.

A determination is made as to whether the selected component is suitable for use in all of the aircraft in the set of target aircraft (operation **804**). If the component is suitable for use with all of the aircraft in the set of target aircraft, the process saves the selection of the software component (operation **806**). A determination is then made as to whether additional software components will be used in the software aircraft part (operation **808**). If additional software components will be used, the process returns to operation **802**.

Otherwise, an identification of metadata for the selected set of software components is made (operation **810**). This metadata provides an identification of the target aircraft on which the software components may be used. The process then generates a checksum (operation **812**). This checksum is a checksum for the metadata and the set of software components.

The checksum may be used to ensure that a software aircraft part is now loaded into a line replaceable unit that has been altered or modified. Modifications also may occur through the transmission of the software aircraft part if errors in transmission occur.

The process then saves the metadata, the set of selected software components, and the checksum as a software aircraft part (operation **814**), with the process terminating thereafter. In these examples, the software aircraft part is saved in a library, such as software aircraft part library **308** in **Figure 3**.

With reference again to operation **804**, if the selected software component is not suitable with all of the aircraft in the set of target aircraft, an error is generated (operation **816**). This error may indicate that one or more of the aircraft within the set of target aircraft cannot use the particular software component that has been selected. A determination is then made as to whether to terminate the process for generating the software aircraft part (operation **818**). If the process is not to be terminated, the process returns to operation **802** to select another component. Otherwise, the process terminates.

In other advantageous embodiments, rather than requiring the user to select another software component, the error generated in operation **816** may indicate that certain features in the software aircraft part may not be usable for the selected target aircraft and allow the user to continue with the same selected software component.

Turning now to **Figure 9**, a flowchart of a process for loading a software aircraft part is depicted in accordance with an advantageous embodiment. The process illustrated in **Figure 9** may be implemented using a component, such as electronic flight bag **400** in **Figure 4** or avionics system **500** in **Figure 5**. In particular, these processes may be implemented in a data loader, such as data loader **402** in **Figure 4** or data loader **514** in **Figure 5**.

The process illustrated in **Figure 9** is performed when a request is made to load the software aircraft part into the line replaceable unit. This check could be made at other times, but making the check at this point ensures that a modification to the software aircraft part has not occurred just before the part is loaded into the line replaceable unit.

The process begins by receiving a request to load a software aircraft part into a line replaceable unit (operation **900**). In these examples, the request is made by a human operator, such as a maintenance technician. In other advantageous embodiments, this request may be received electronically. The process then verifies a checksum for the selected aircraft part in the storage device (operation **902**).

Thereafter, a determination is made as to whether the software aircraft part has been modified (operation **904**). The software aircraft part has not been modified for use if the checksum has been verified. If the software aircraft part has not been modified, the process obtains information about the aircraft (operation **906**). This information may be obtained from a configuration file stored in the storage device on the aircraft or by interrogating components within the aircraft. Next, metadata is retrieved from the software aircraft part (operation **908**). The information about the aircraft is compared to the metadata from the software aircraft part (operation **910**).

A determination is made as to whether the software aircraft part is suitable for use on the aircraft from the comparison (operation **912**). If the software aircraft part is suitable for use, the software aircraft part is loaded into the line replaceable unit (operation **914**), with the process terminating thereafter.

With reference again to operation **912**, if the software aircraft part is not suitable for use in the aircraft, the process terminates. The process also terminates in operation **904** if the software aircraft part has been modified for use from the checksum verification performed in operation **902**.

Thus, the different advantageous embodiments provide a computer implemented method, apparatus, and computer usable program code for managing software aircraft parts. This management of software aircraft parts may include the creation of software aircraft parts in a manner that the software aircraft parts may be checked for suitability for use on a particular aircraft during a loading process. The different advantageous embodiments also provide a feature in which a check may be made of a software aircraft part before that part is actually used in an aircraft data processing system.

In this manner, a tedious process of checking the parts prior to sending the parts to an aircraft may be avoided. Also, the different advantageous embodiments allow for checks to be made when the software aircraft part is actually loaded. These types of

features are currently unavailable with the current systems for loading software aircraft parts into an aircraft.

The flowcharts and block diagrams in the different depicted embodiments illustrate the architecture, functionality, and operation of some possible implementations of apparatus, methods and computer program products. In this regard, each block in the flowchart or block diagrams may represent a module, segment, or portion of computer usable or readable program code, which comprises one or more executable instructions for implementing the specified function or functions. In some alternative implementations, the function or functions noted in the block may occur out of the order noted in the figures. For example, in some cases, two blocks shown in succession may be executed substantially concurrently, or the blocks may sometimes be executed in the reverse order, depending upon the functionality involved.

The different advantageous embodiments can take the form of an entirely hardware embodiment, an entirely software embodiment or an embodiment containing both hardware and software elements. Some embodiments are implemented in software, which includes but is not limited to forms, such as, for example, firmware, resident software, and microcode.

Furthermore, the different embodiments can take the form of a computer program product accessible from a computer-usable or computer-readable medium providing program code for use by or in connection with a computer or any device or system that executes instructions. For the purposes of this disclosure, a computer-usable or computer readable medium can generally be any tangible apparatus that can contain, store, communicate, propagate, or transport the program for use by or in connection with the instruction execution system, apparatus, or device.

The computer usable or computer readable medium can be, for example, without limitation an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system, or a propagation medium. Non limiting examples of a computer-readable medium include a semiconductor or solid state memory, magnetic tape, a removable computer diskette, a random access memory (RAM), a read-only memory (ROM), a rigid magnetic disk, and an optical disk. Optical disks may include compact disk – read only memory (CD-ROM), compact disk – read/write (CD-R/W) and DVD.

Further, a computer-usable or computer-readable medium may contain or store a computer readable or usable program code such that when the computer readable or usable program code is executed on a computer, the execution of this computer readable or usable program code causes the computer to transmit another computer readable or usable program code over a communications link. This communications link may use a medium that is, for example without limitation, physical or wireless.

A data processing system suitable for storing and/or executing computer readable or computer usable program code will include one or more processors coupled directly or indirectly to memory elements through a communications fabric, such as a system bus. The memory elements may include local memory employed during actual execution of the program code, bulk storage, and cache memories which provide temporary storage of at least some computer readable or computer usable program code to reduce the number of times code may be retrieved from bulk storage during execution of the code.

Input/output or I/O devices can be coupled to the system either directly or through intervening I/O controllers. These devices may include, for example, without limitation to keyboards, touch screen displays, and pointing devices. Different communications adapters may also be coupled to the system to enable the data processing system to become coupled to other data processing systems or remote printers or storage devices through intervening private or public networks. Non-limiting examples are modems and network adapters are just a few of the currently available types of communications adapters.

The description of the different advantageous embodiments has been presented for purposes of illustration and description, and is not intended to be exhaustive or limited to the embodiments in the form disclosed. Many modifications and variations will be apparent to those of ordinary skill in the art. Further, different advantageous embodiments may provide different advantages as compared to other advantageous embodiments. The embodiment or embodiments selected are chosen and described in order to best explain the principles of the embodiments, the practical application, and to enable others of ordinary skill in the art to understand the disclosure for various embodiments with various modifications as are suited to the particular use contemplated.

**CLAIMS:**

1. A computer implemented method for managing a software aircraft part, the computer implemented method comprising:

identifying an aircraft from a set of aircraft to form a target aircraft;

identifying software for use in the target aircraft;

performing a first determining step of determining whether compatibility exists between the software and the target aircraft;

responsive to a determination in the first determining step that compatibility exists between the software and the target aircraft, creating the software aircraft part comprising the software and metadata relating to usability of the software aircraft part in the target aircraft;

responsive to a request to use the software aircraft part in an aircraft data processing system for a selected aircraft, performing a second determining step of determining whether the software aircraft part is suitable for use on the selected aircraft using the metadata and configuration information of the selected aircraft, wherein the configuration information of the selected aircraft comprising an identification of configurations or components within the selected aircraft; and

responsive to a determination in the second determining step that the software aircraft part is suitable for use on the selected aircraft using the metadata, loading the software aircraft part for use in the aircraft data processing system.

2. The computer implemented method of claim 1 further comprising:

defining the metadata relating to usability of the software aircraft part on the set of aircraft.

3. The computer implemented method of claim 2 further comprising:

generating a warning if the software aircraft part is incompatible with the selected aircraft.

4. The computer implemented method of claim 1, wherein the loading step comprises:

loading the software aircraft part in a line replaceable unit in response to the determination that the software aircraft part is suitable for use on the selected aircraft using the metadata.

5. The computer implemented method of claim 1, wherein the aircraft data processing system is a line replaceable unit in the aircraft.

6. The computer implemented method of claim 1, wherein the software aircraft part is received by a data loader in the loading step.

7. The computer implemented method of claim 1, wherein the aircraft data processing system is an electronic flight bag.