COLEGIO OFICIAL DE **PILOTOS** DE LA AVIACIÓN COMERCIAL

AVIADOR

JULIO-AGOSTO-SEPTIEMBRE 2015. Nº77

Un compromiso con la calidad y la excelencia en la formación

Imágenes de altura, el retrato de una profesión

LCI: Ser parte de la solución o sálvese quien pueda

COPAC

Seguridad aérea e informática

Alberto Prieto. *Miembro de la Academia de Ciencias Matemáticas, Físico-Químicas y Naturales de Granada*

A finales del pasado mes de mayo diversos medios de comunicación publicaron la siguiente noticia: "Un fallo en el software del ordenador que controla los motores fue la causa del accidente que el pasado día 9 sufrió un avión A400M en Sevilla y que provocó la muerte de cuatro tripulantes y heridas a otros dos, según los primeros resultados de la investigación. Eso explica que el avión de transporte militar perdiera empuje poco después de despegar en un vuelo de prueba y que se precipitara a pesar de contar con cuatro motores, que no pueden fallar simultáneamente" *(El País 19 de mayo 2015)*. Según dicha investigación interna uno de los errores estuvo en los ordenadores que transmiten las órdenes a los propulsores que pudieron transmitir órdenes contradictorias y provocar que los pilotos no pudieran controlar la aeronave *(ABC, 19 de mayo 2015)*.

Después de leer estas noticias, la pregunta que inmediatamente muchos nos podemos hacer es la siguiente ¿es seguro el software?

La informática es un recurso que se está convirtiendo poco a poco en imprescindible para la práctica totalidad de la actividad humana, y concretamente para los avances de las ciencias y de las ingenierías.

En una aeronave moderna la mayoría de los sistemas de control se basan en la utilización de pequeños computadores "empotrados" que están diseñados para realizar funciones muy concretas y cuya actuación se monitoriza por medio de programas software. Esto tiene diversas consecuencias prácticas como son: abaratamiento de los costes de diseño y fabricación, versatilidad en cuanto a que las prestaciones o funciones a realizar se pueden actualizar sin más que cambiar algún programa, el cambio radical en cuanto a la comunicación piloto-avión, y la importancia extrema que adquiere la fiabilidad de los programas.

Me voy a referir a continuación únicamente a los dos últimos aspectos.

La informatización está provocando que la comunicación piloto-avión (interfaz hombre-máquina) esté cambiando radicalmente. De tener volantes, palancas, ruedas y otras piezas mecánicas se pasa a interactuar a través de pantallas táctiles, ratones, pequeñas palancas *(joysticks)*, etc. Esto para los pilotos supone un cambio muy profundo de manipulación, aunque las interfaces puedan estar dotadas de elementos adicionales para darles la apariencia de elementos mecánicos.

Sin duda en aviones altamente informatizados, para algunas operaciones o cuando se produce un fallo, el piloto echará en falta la existencia de volantes, palancas o engranajes mecánicos (y no virtuales) sobre los que actuar. Este problema se soluciona con un adecuado entrenamiento de los pilotos y con las nuevas generaciones de los mismos que llegarán acostumbradas desde la infancia a una comunicación hombre-máquina más virtual que mecánica. También sería muy oportuno, hasta que un sistema esté cabalmente probado, poder pasar de un entorno de funcionamiento en control automático a control manual de forma tal que el piloto utilizase su conocimiento, afianzado en su experiencia, para monitorizar eventualmente la conducción del avión. Debería ponerse especial cuidado en determinar la conveniencia de dejar impotente o no la actuación directa del piloto sobre el control del avión en las distintas fases del vuelo.

Está ampliamente demostrado, en multitud de aplicaciones de ingeniería, que el control por software es sumamente eficiente reuniendo ventajas indudables. No obstante, la realización de programas requiere un conocimiento completo y exhaustivo de la aplicación o proceso a informatizar en el sentido de que el programador debe prever todos los casos posibles para que el programa reaccione correctamente ante cualquier eventualidad. Con frecuencia los bloqueos que se producen en nuestros ordenadores (afortunadamente cada vez menos probables) se producen debido a llegar a una situación o estado no previsto por los diseñadores del sistema operativo o programa de aplicación. Esto es especialmente importante en los llamados sistemas críticos, que en informática se definen como aquellos en los que un fallo puede dar lugar a pérdidas humanas o pérdidas ambientales o económicas significativas. Este sería el caso de los sistemas de control de una aeronave: en las especificaciones del diseño se debe prever cualquier situación posible (o incluso, se podría decir "imposible") para que los informáticos las incluyan en sus programas y así el sistema reaccione correctamente ante cualquier eventualidad.

Otro problema susceptible de presentarse son los errores de programación. En la década de los 1970 un satélite artificial

La realización de programas requiere un conocimiento completo y exhaustivo de la aplicación o proceso a informatizar en el sentido de que el programador debe prever todos los casos posibles para que el programa reaccione correctamente ante cualquier eventualidad

Cockpit de A350. Foto: Airbus

se perdió en el espacio debido a un sencillo error de programación. El programa se había desarrollado en el lenguaje FORTRAN, y el error consistía en que en una sentencia para realizar un bucle no se había dejado un espacio en blanco entre dos caracteres y el compilador interpretó erróneamente que el significado de esa instrucción era sencillamente asignar un valor a una variable. En definitiva, la falta de un espacio en blanco entre dos caracteres de una instrucción supuso la pérdida de millones de dólares.

Hay que darse cuenta de que los errores mecánicos o electrónicos de un sistema son relativamente fáciles de percibir, resultando más evidentes o visibles, pero las instrucciones de un programa son intangibles y es mucho más sutil detectar posibles errores.

En este entorno también se pueden producir errores debidos a la **retrocompatibilidad**. Este fenómeno da lugar a que con frecuencia no se esté utilizando el hardware para el que originalmente se creó el software. Normalmente pasa una gran cantidad de tiempo desde que se introduce cualquier tipo de hardware nuevo, por ejemplo un microcontrolador, hasta que se verifica y valida a través de los protocolos apropiados para su uso en aviónica. Una vez certificada la seguridad de los nuevos dispositivos, es normal **emular** (con programas) en ellos el comportamiento de los antiguos dispositivos para que el costoso software original siga siendo válido hasta tanto se rediseñen los nuevos programas. En definitiva, la introducción de nuevo hardware no va sincronizada con la introducción de nuevo software, presentándose la retrocompatibilidad como otra fuente potencial de posibles errores.

Pero, después de lo dicho anteriormente, no hay que alarmarse: en Ciencia y en Ingeniería cuando se identifica un problema se busca y, por lo general, se encuentra la solución. Así, el problema del error de programación en el control de un satélite, citado anteriormente, se solucionó completamente con el desarrollo de nuevos lenguajes de programación (denominados "fuertemente tipificados"), como es el caso del lenguaje Ada. También existe una disciplina dentro de la informática denominada ***Ingeniería del Software*** que trata de la producción de programas y aplicaciones de forma sistemática, utilizando muchas de las técnicas utilizadas y ampliamente probadas en ingeniería para el diseño, fabricación y mantenimiento de productos industriales. Especial énfasis se hace en la calidad del producto final (en nuestro caso software) teniendo en cuenta tres factores: ***confiabilidad*** ("reliability") o aptitud del sistema para proporcionar servicios tal y como se han especificado en la fase previa al diseño, ***inocuidad*** ("safety") o garantía del sistema para funcionar sin fallos (que en algunas aplicaciones podrían ser catastróficos), y ***seguridad*** ("security") o habilidad del sistema para protegerse así mismo frente a intrusiones ya sean accidentales o deliberadas.

> Siempre debería preverse que un experto pudiese ser capaz de imponerse al sistema automático, en tanto que es extremadamente complejo predecir absolutamente todo

Por último los sistemas diseñados se someten a dos procesos: uno de ***verificación***, por el que se hace un chequeo de si el sistema construido se ajusta fielmente a las especificaciones, y otro de ***validación*** a través del cual se comprueba si el sistema se ajusta a las necesidades y expectativas del cliente.

Obviamente el proceso de verificación es exhaustivo en el caso de sistemas críticos.

En definitiva, como cualquier construcción de ingeniería, no se puede asegurar al cien por cien el funcionamiento correcto del software, pero existe una metodología (Ingeniería del Software) ampliamente estudiada y conocida por los profesionales informáticos para obtener programas de muy alta calidad, teniendo dentro de ella una especial consideración los sistemas críticos en los que el principal criterio de diseño es la seguridad. Aun así, siempre debería preverse que un experto pudiese ser capaz de imponerse al sistema automático, en tanto que es extremadamente complejo predecir absolutamente todo. ■

Fuente: "Seguridad aérea e informática" Alberto Prieto, Academia de Ciencias de Granada (Spain), 10 junio 2015