IV

Software

Robert P. Lyons, Jr.
United States Air Force

Digital avionics allow much greater integration of functions, unprecedented flexibility, enhanced reliability, and ease of technology upgrade than can be achieved with more classical analog avionics. Indeed, given the cost, volume, weight, prime power, cooling, complexity, and safety constraints of both civil and military aircraft, there may really be no other option than to digitize, and even integrate, avionics functions in ways analog avionics could never achieve. Clearly, advances in electron devices, particularly digital integrated circuits of very large scale and high speed, give nearly unlimited possibilities to the designer of modern avionics. But without concomitant attention to the development and verification of highly complex software and its integration with the underlying hardware, the avionics implementation will simply produce heat, but no useful, much less safe, avionics functions at all.

Although software is an abstract thing dealing with data flows, control flows, algorithms, logical expressions, and the like, it is a building material of today's and tomorrow's aircraft as surely as are silicon, aluminum, titanium, plastics, and composites. And just as all the system and hardware aspects of an aircraft and its avionics require and are amenable to disciplined engineering analysis, synthesis, and verification, so is software. Certainly, the abstract nature of software makes its engineering and implementation less obvious than the fairly concrete methods available to the system and hardware engineering segments of the modern avionics development job. Fortunately, the tools to engineer software do exist.

The two chapters in this section describe two very important aspects of the software engineering applied to avionics. Chapter 26 discusses ISO/IEC 8652-95 Programming Language – Ada. Of all the software languages available to implement very large and very critical avionics functions, Ada-95 and, to a lesser extent, its predecessor Ada-83 have been shown to be the most supportive of software engineering concepts most likely to produce error free, flexible, reliable, complex software programs. One can write bad software in any language; it is just harder to do so with Ada than with other languages such as C and C++. Chapter 26 gives an excellent introduction to the development of the Ada language, its features, and the rudiments of its application.

Chapter 27 covers the important points of one of the most pervasive software standards applicable especially to civil aircraft avionics, but which subsumes other standards usually seen in military avionics developments. DO-178/ED-12 Software Considerations in Airborne Systems and Equipment Certification

is a well-coordinated consensus standard combining the best thinking of software experts as well as aircraft certification authorities. Although this standard does not prescribe any particular software development methodology, all elements of the software life cycle are covered under its aegis. DO-178B, the current instantiation of the standard, does for the software part of the avionics sy