The exponential growth in the use and cost of software in avionics systems has brought with it, both the advantage of reprogrammability and the resulting equivalent growth in the requirement to enhance the software to meet changing mission requirements and to be responsive to changing threats, while still maintaining the validity of the software. Distributed and highly integrated software now in design/development as well as parallel processing and artificial intelligence concepts put additional strains on the testability, instrumentation of software, dynamic support environment, and validation capabilities. This paper will describe the work being done under the Embedded Computer Resources Support Improvement Program to meet these challenges by addressing the development of an advanced low cost software support capability and developing a rapid turnaround capability for software. A modular, reconfigurable support environment has been demonstrated for the F-16, which incorporates off-the-shelf components where applicable, Ada simulation modules, non intrusive dynamic debug, and real-time networks. Techniques to accomplish a rapid turnaround of software in response to mission or threat changes will be discussed.

BACKGROUND

Mission Critical Software, defined here as avionics software resident on board a weapons platform, has escalated in size and functionality in the past decade to meet changes in the threat and changes in mission requirements. As a result, an increased burden has been placed on the post deployment software support (PDSS) process.

PROJECT

To keep pace with this dynamic environment, the Air Force has fielded systems with embedded computers that can be upgraded and reconfigured via software changes. This additional degree of freedom has resulted in a host of new support complications. Software modifiable systems have, in most cases, exceeded the capabilities of the institutions and infrastructure responsible for their support. It is not uncommon for an operational flight program (OFP) change or block update to take as long as one year before release to the field. Embedded computers in avionics will become even more pervasive; as time goes on, future systems and retrofits of existing systems will show a further migration of hardware functions to software and an increase in the integration of processes/resources. As the software for embedded avionics computers continues to escalate in size, function, and criticality, so too will the software support, maintenance, and enhancement requirements. The technology required by the Air Logistics Centers (ALC) and the operational commands to support this proliferation of vital software throughout the deployed life cycle has not kept pace. Without an infusion of new technology developments in methodologies, tools, and environment it will not keep pace in the future.

OBJECTIVE

This paper addresses two of the objectives set forth in the Air Force Embedded Computer Resources Support Improvement Program (ESIP), a joint Air Force Logistics Command (AFLC)-Air Force Systems Command (AFSC) program. The objectives being pursued by Air Force Systems Command, Wright Research and Development Center (WRDC), Avionics Laboratory, System Avionics Division, Avionics Logistics Branch (AAAF) are that of low cost Integrated Support Facilities for depot level software support and the issue of software rapid turnaround. The first objective is to investigate advanced techniques and technologies to reduce the support capability acquisition and life cycle cost; improve the expandability, flexibility and interchangeability of the software and hardware required by the support facilities; and to improve the development, test, and documentation capability of the facility. The second objective addresses the issue of software rapid turnaround, also known as software readiness or software availability. In this context, AAAF is examining software technologies and support methodologies in response to a TAC Statement of Operational Need requiring the capability to effect emergency OFP changes in 24 hours or less from identification of the software fault or change in mission requirements or threat environment to a fielded, thoroughly tested update. AAAF is investigating and developing methodologies for adding supportability mechanisms into the development cycle of the software, and streamlining existing evaluation, validation, and test capabilities. To date, post deployment software support has been largely overlooked in the development cycle. With the onset of highly integrated avionics suites, the idea of software 'built-in-test' becomes more critical. This second area addresses such technologies as software fault tolerance, software performance monitoring, software structure and methodology for maintenance, software adaptation to the environment, and ground based tools for pre-emptive engineering solutions. Resolution of these software issues will reduce the software life cycle cost as well as reduce the time required to enhance and maintain software.

To meet the AFSC goal of developing methodologies, tools, and environments for PDSS, AAAF reviewed the PDSS cycle through visits to software support facilities and discussions with software maintainers to identify technology deficiencies that must be addressed to meet the ESIP goals. Technologies identified affect all aspects of the PDSS cycle. These include process automation; identification and capture of software anomalies as they occur; elimination of costly point designed dynamic test environments; lack of design/process traceability; time consuming incomplete test

ABSTRACT

The exponential growth in the use and cost of software in avionics systems has brought with it, both the advantage of reprogrammability and the resulting equivalent growth in the requirement to enhance the software to meet changing mission requirements and to be responsive to changing threats, while still maintaining the validity of the software. Distributed and highly integrated software now in design/development as well as parallel processing and artificial intelligence concepts put additional strains on the testability, instrumentation of software, dynamic support environment, and validation capabilities. This paper will describe the work being done under the Embedded Computer Resources Support Improvement Program to meet these challenges by addressing the development of an advanced low cost software support capability and developing a rapid turnaround capability for software. A modular, reconfigurable support environment has been demonstrated for the F-16, which incorporates off-the-shelf components where applicable, Ada simulation modules, non intrusive dynamic debug, and real-time networks. Techniques to accomplish a rapid turnaround of software in response to mission or threat changes will be discussed.

BACKGROUND

Mission Critical Software, defined here as avionics software resident on board a weapons platform, has escalated in size and functionality in the past decade to meet changes in the threat and changes in mission requirements. As a result, an increased burden has been placed on the post deployment software support (PDSS) process.

PROJECT

To keep pace with this dynamic environment, the Air Force has fielded systems with embedded computers that can be upgraded and reconfigured via software changes. This additional degree of freedom has resulted in a host of new support complications. Software modifiable systems have, in most cases, exceeded the capabilities of the institutions and infrastructure responsible for their support. It is not uncommon for an operational flight program (OFP) change or block update to take as long as one year before release to the field. Embedded computers in avionics will become even more pervasive; as time goes on, future systems and retrofits of existing systems will show a further migration of hardware functions to software and an increase in the integration of processes/resources. As the software for embedded avionics computers continues to escalate in size, function, and criticality, so too will the software support, maintenance, and enhancement requirements. The technology required by the Air Logistics Centers (ALC) and the operational commands to support this proliferation of vital software throughout the deployed life cycle has not kept pace. Without an infusion of new technology developments in methodologies, tools, and environment it will not keep pace in the future.

OBJECTIVE

This paper addresses two of the objectives set forth in the Air Force Embedded Computer Resources Support Improvement Program (ESIP), a joint Air Force Logistics Command (AFLC)-Air Force Systems Command (AFSC) program. The objectives being pursued by Air Force Systems Command, Wright Research and Development Center (WRDC), Avionics Laboratory, System Avionics Division, Avionics Logistics Branch (AAAF) are that of low cost Integrated Support Facilities for depot level software support and the issue of software rapid turnaround. The first objective is to investigate advanced techniques and technologies to reduce the support capability acquisition and life cycle cost; improve the expandability, flexibility and interchangeability of the software and hardware required by the support facilities; and to improve the development, test, and documentation capability of the facility. The second objective addresses the issue of software rapid turnaround, also known as software readiness or software availability. In this context, AAAF is examining software technologies and support methodologies in response to a TAC Statement of Operational Need requiring the capability to effect emergency OFP changes in 24 hours or less from identification of the software fault or change in mission requirements or threat environment to a fielded, thoroughly tested update. AAAF is investigating and developing methodologies for adding supportability mechanisms into the development cycle of the software, and streamlining existing evaluation, validation, and test capabilities. To date, post deployment software support has been largely overlooked in the development cycle. With the onset of highly integrated avionics suites, the idea of software 'built-in-test' becomes more critical. This second area addresses such technologies as software fault tolerance, software performance monitoring, software structure and methodology for maintenance, software adaptation to the environment, and ground based tools for pre-emptive engineering solutions. Resolution of these software issues will reduce the software life cycle cost as well as reduce the time required to enhance and maintain software.

To meet the AFSC goal of developing methodologies, tools, and environments for PDSS, AAAF reviewed the PDSS cycle through visits to software support facilities and discussions with software maintainers to identify technology deficiencies that must be addressed to meet the ESIP goals. Technologies identified affect all aspects of the PDSS cycle. These include process automation; identification and capture of software anomalies as they occur; elimination of costly point designed dynamic test environments; lack of design/process traceability; time consuming incomplete test
The first project initiated addressed the development and demonstration of a new support architecture for the dynamic test and evaluation of operational flight software. The goal was to develop a low cost, modular and extendable, dynamic software support environment which had all the existing capability of in-use simulation support facilities and also provided the capability for future expansion to meet future upgrades and routine enhancements using off-the-shelf hardware and software wherever possible. The support environment developed is called the Advanced Multi-Purpose Support Environment (AMPSE). The architecture concept was demonstrated in July 86. A prototype was then developed to provide software support for the F-16 A/B Fire Control Computer operational flight program. This same architecture has been adopted by the F-16 System Program Office and is being expanded to provide the support for the F-16 C/D. The capability of this architecture to provide support for more than one version of the aircraft through software simulation model sets and not hardware simulation equipment brings to the support area tremendous cost savings. Replication costs and facility space savings have also been realized. An objective of the AMPSE was to use off-the-shelf hardware and software where feasible, thus providing mature, maintainable systems. Digital Equipment Corporation microVAXes were used for the F-16 A/B prototype. A distributed architecture was demonstrated which does not rely on a single mainframe host computer. For the F-16 A/B, a network of six microVAXes serve as the simulation host and the interface host. A review of existing simulation networks determined that no existing commercial network would meet the speed requirements for this type of architecture. Therefore, to successfully demonstrate this new concept, a network had to be developed. This network, which is now a commercially available product, was conceived and demonstrated by an Air Force engine and proven in the Avionics Laboratory as meeting all of the requirements for the simulation environment and would provide for future growth of the simulator. The simulation model set was translated to or developed in Ada to take advantage of the maintainability features of Ada. Both hardware and software are modular and extendable to enable the support environment to grow with the weapon system throughout its life cycle. The AMPSE has been transitioned to Ogden Air Logistics Center for the support of the F-16A/B. Follow-on work is now being accomplished to provide the F-16 C/D capability for the F-16 A/B. Development costs were reduced by a factor of two, replication costs were reduced by a factor of eight, and floor space reduced by a factor of six.

The area of testing was identified as a time consuming yet never complete task for software. Testing is a significant user of simulation resources as well as manpower and time. Automating the test procedures and the test case generation could significantly decrease the time required to validate a software change, eliminate operator induced errors, and utilize simulation resources during non-duty hours. The Automated Operational Flight Program Validation (AUTOVal) was devised to automate the test procedures execution. AUTOVal will flag anomalies during execution of the test suite. At that point, the test engineer will further evaluate the test results. As the next phase of AUTOVal, the ability to generate a new test suite as a result of requirements changes will be developed. The ability to automatically generate test cases to validate operational flight program changes will insure the update of the test suite with each change. Another problem which will be addressed during this phase is determining when enough testing has been accomplished. It is impossible to comprehensively test all weapon system software as we now know it. To test all the software expected to be resident on such platforms as the Advanced Tactical Fighter and the B-2 Bomber would take years. This completely negates the block cycle change procedures for software which attempts to make routine changes in 12 months, urgent changes in 10 days and emerging changes in 24 hours. AUTOVal will initially reduce the time of the test cycle to 1/3. This occurs from the immediate ability to run the test suite around the clock. If weekends and human operator breaks are factored in, there is a larger savings. An area yet to be explored is what are the effects of running operational flight program software faster than human interface speeds. The effects of this speed increase must be assessed as the software is designed and written with pilot interaction via switches on the aircraft controls and panels.

Process automation is being addressed through the automation of the report generation and tracking systems in place for the emergency reprogramming center at Warner-Robins Air Logistics Center (WR-ALC). This effort will centralize and consolidate the process control for the center. The effort, called the Emergency Reprogramming Center Information Manager will be delivered to WR-ALC in September this year.

The area of dynamically debugging software non-intrusively is key to the success of delivering error free software. To use any other method to debug real-time software typically interrupts the timing and busing of the software during debug, resulting in changing through an interrupt service routine to service the debugger and changes in the busing to allow data to be retrieved. As timing and busing are key to the integrity of the software, a dynamic debugger must have no effect on the software. A Performance Monitor and Control (PMAC) device which accesses internal busses, memory, and registers through an external port is now commercially available for current generation embedded processors was developed by AAAF. Further work is being done to address Reduced Instruction Set Computers (RISC), Very High Speed Integrated Circuit (VHSC) processors and multiple processors used in a distributed/integrated architecture. For those processors which do not have external access to the internal busses and registers, another approach has been taken, that of a chip resident on the processor card which gains access to the internal processor information and brings that information out to pins. This capability is necessary to enable software debugging and performance monitoring in a non-intrusive manner for highly distributed/integrated architectures. Correlation of the available information then becomes paramount to enable the test engineer to interpret and analyze the data.

A great deal of effort has been expended to collect data on hardware failures through built-in-test. This effort explores how and what data needs to be collected to capture software anomalies as they occur. Capturing the proper data which can be used to duplicate the anomaly in the dynamic test facility and be provided to the analyst can significantly reduce the time required to respond to the anomaly. Providing this capability within the space and weight constraints of embedded systems presents a significant challenge. This
A need occurs to automate and aid in the analysis of flight test data and analysis of the data produced in the dynamic debug process. Reams of data are available to the analyst, however, only very small portions contain significant information. An effort to search this data and flag and correlate anomalies in the data for the analyst is now in the concept stage. Methods of flagging, correlating and presenting the data to the analyst are being explored.

The most time-consuming portion of most block cycle changes occurs in the documentation trail which must be completed prior to release of the enhanced OFP. This comes from the manual nature of this task, the enormous amount of documents that must be updated, and the lack of correlation between the documents. A recently started task under ESIP is exploring the application of Hypermedia technology to the documentation problem. By building the links between the documentation and the source code, each place that the documentation must be changed to reflect the software enhancement is flagged, enabling the documenter to address all the documents electronically. This streamlines the process and ensures a complete cross-referenced document will be shipped with the OFP. Another benefit of this technology would be the availability of the cross-referenced documentation to the analyst and software engineer in the beginning stages of the change cycle. This would enable them to have at their fingertips all the documentation that is applicable to the enhancement that they are processing.

If the documentation also included a traceability of the design, thought processes, and software paths pursued during initial software development, the software maintainer/enhancer would have the insight of the original designer/programmers. This would eliminate the possibility of repeating previous unsuccessful paths which were taken to arrive at the OFP now being enhanced. Complimentary projects have been instituted to establish traceability during the software development process. Initial prototype demonstrations are scheduled for late 1990.

To prevent repeats of the PDSS shortfalls when highly integrated/distributed software comes on line, a new software development methodology with the appropriate tools and environments must be in place. A modular embedded computer methodology has been proposed which addresses the development of software and provides for the mission/system partitioning required by the maintenance community. Prototype resource mapping and scheduling tools have been developed. Continued work is planned in the methodology arena.

SUMMARY

The above described efforts under the ESIP program have attempted to address many of the issues facing PDSS responsible organizations. The highly successful completion and prototype demonstrations to date as well as user acceptance and support will insure that the technologies are incorporated into the PDSS process. Numerous other efforts are in the concept development stage and will bring to the PDSS process additional methodologies, tools, and environments to improve the process, resulting in both cost and time savings.