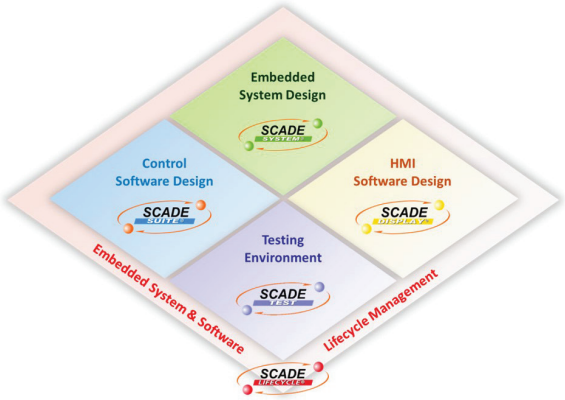
ANSYS

White Paper

Deterministic Modeling and Qualifiable Ada Code Generation for Safety-Critical Projects

Ada is a time-tested, safe and secure programming language that was specifically designed for large and long-lived applications where safety and security are essential. Traditionally, Ada development teams have faced the challenge of a lengthy, time-consuming development process; the most difficult part has typically been proving compliance with standards, such as DO-178C, used to qualify high-integrity software. A model-based design approach that involves the creation of an executable model in a block diagram design environment improves the efficiency of Ada coding, but the potential for ambiguities between the system requirements and software implementation in existing modeling environments still requires a long and tedious qualification process.

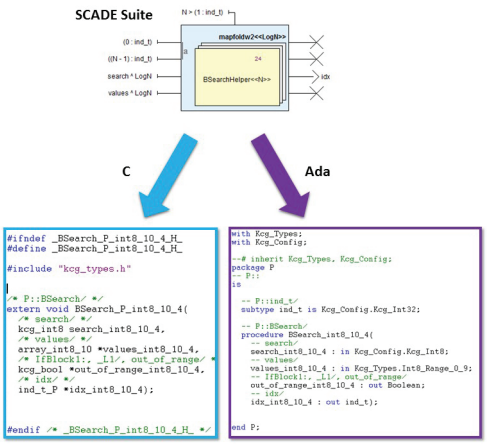
ANSYS SCADE products provide a complete solution for development of high integrity Ada applications, including a formally defined modeling environment that provides all of the inherent benefits of the model-based design approach while avoiding the ambiguity inherent in other Ada modeling environments. The SCADE KCG Ada Code Generator is qualifiable for DO-178C and certified for IEC 61508 and EN 50128 to ensure that the generated code behaves exactly as the design with respect to software requirements. The new tools are part of a complete development environment that substantially reduces the development time for safety-critical applications built with the Ada language.



The SCADE Product Family

Ada Language Addresses Safety-Critical Applications

Human life often depends on the performance of mission-critical, real-time applications that control systems such as commercial and military airplanes, submarines, mass transit systems, railway systems, rockets, etc. Developers of these systems have to meet a standard, such as DO-178C, that defines a process for proving that the software does what it is supposed to do and does not do anything else. The Ada programming language is unique in that it was designed under contract to the United States Department of Defense in the late 1970s and early 1980s to address these types of applications and replace the hundreds of different programming languages that were then used in mission-critical software projects. Some of the key features of Ada include compressive support for object oriented programming, explicit concurrency, real-time programming, strong typing and synchronous message passing. Ada has been used to develop many well-known and highly critical applications, such as the Patriot Missile Command and Control Center, U.S. Navy Submarine Combat System, Boeing 777 Airplane Information Management System, United States Federal Aviation Administration Conflict Detection Tool, French High-Speed Rail (TGV), Delta II and Delta IV commercial rockets and many others.



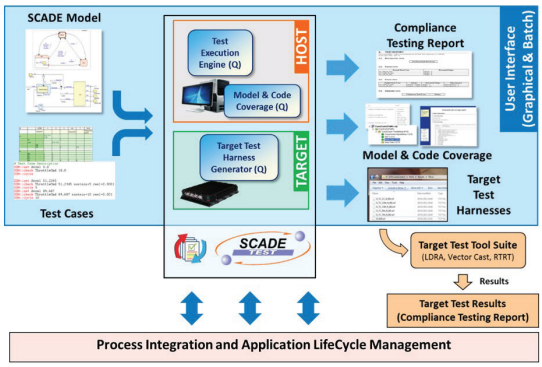
SCADE Suite KCG automatically generates C code and Ada code directly from the model.

Challenges of Developing Mission-Critical Ada Applications

In the traditional method of Ada code development, specifications are conceptualized by systems architects, captured as text documents, and passed to project teams that specialize in areas such as algorithm development and analog electronics. These teams write software and perform tests to verify that the code matches the specifications. Verifying the code requires assembling the hardware, including the target computing device and the electromechanical components that are being controlled. A key problem with this approach is that errors often remain undetected until all the modules can be tested together at the prototype stage. Because the cost of fixing a problem generally increases by an order of magnitude as the design progresses, late-stage problems quickly drive up development costs. As the size and complexity of safety-critical systems increases at a rapid rate, the cost, time and risk involved in manually producing, testing and verifying tens of millions of lines of code is increasing at an astronomical pace.

Many Ada development teams have addressed this challenge by moving to model based development methods that provide engineers with the capability to quickly build a graphical model of safety-critical software and systems using prebuilt components without requiring manual code writing, although hand-written code can be incorporated when required. Engineers can simulate the behavior of the model and immediately view the results, making it possible to gain critical insights early in the systems design process and to rapidly improve the model's performance. Engineers also can link the predicted behavior to specific customer requirements. Later in the design cycle, the model can be used to automatically generate software that can be downloaded to an embedded hardware system to evaluate the prototype in real time.

But model-based design environments used until now for Ada are non-deterministic, so they open the door to ambiguity, which creates the potential for misinterpretations of system requirements and software implementations. Both the systems architects and the software developers have their own specific terminology that might not be familiar to the other team, making it difficult to prove that the automatically generated code meets the requirements. During the qualification process, the development team must overcome the issues of ambiguity or non-determinism in the model to prove that the code can be trusted. Addressing this concern to the degree of certainty required for DO-178C and other certification standards is a long and difficult process that offsets a substantial portion of the savings that would otherwise be achieved through model-based design and automatic code generation.



The SCADE development process

Deterministic Modeling Environment Ada Code Development

The ANSYS SCADE Suite provides a new approach to Ada code development that overcomes these challenges. SCADE Suite offers a deterministic modeling environment that prevents potential ambiguities by defining formal unambiguous semantics that avoid misinterpretations between systems architects and developers. The SCADE modeling environment provides formal data-flow and control-flow constructs. Hierarchy is supported via user-defined operators and hierarchical state machines. Most checks such as strong type-checking, namespace analysis, causality analysis, clock analysis and initialization analysis can be performed at the model level during design.

SCADE Suite supports software reliability throughout the entire development process from requirements definition to code generation, software verification and validations. Operational requirements can be written in tools including IBM® Rational® DOORS®, Rational RequisitePro® or Borland® CaliberRM®, and linked together using the SCADE Requirement Management Gateway.
Test cases can be developed with SCADE Test, and can be seamlessly run either on host or on target using IBM® Rational Test RealTime, LDRA TestBed or Vector Cast, with structural coverage ensured at model and code levels.

SCADE LifeCycle Requirement Management Gateway provides a complete and interactive solution to manage traceability links between high level requirements (HLRs) to low level requirements (LLRs) and from HLRs to test cases and procedures. This tool automates the process of validating that 100 percent of HLRs have been implemented and 100 percent of HLRs are covered by test cases and procedures.

Block Diagrams Represent Architecture and Software

SCADE System software uses block diagrams based on the SysML standard to represent software architecture components and connect them through ports and connectors. Once the software functional decomposition is available, the next step is to produce a software architectural design that implements the requirements using software blocks. The design includes an explanation of how the requirements have been allocated to the architecture. Consistency of the architecture can be validated with the SCADE System checker, and data propagation can be analyzed through all the views. Software engineers model the software components associated with the software architecture, often using state machines and data flows to model the logic and control laws. Software requirements can be traced through the development process to ensure that they are met. The architecture components and the design models can be synchronized, ensuring a complete consistency in the workflow. SCADE Suite incorporates a reusable symbol library that promotes reuse and commonality of design within and across software projects.

SCADE Syntactic and Semantic Checker perform an in-depth analysis of the model consistency, including detection of missing definitions, warnings on unused definitions, detection of non-initialized variables, coherence of data types and interfaces, detection of causality issues on data-flow dependencies, and coherence of the production/consumption rates of data. The rules that are automatically verified by this tool are part of the formal language definition. Compliance with these rules ensures that the model fully respects the semantics, which ensures its determinism.

Detecting Functional Faults Early in Design Process

Model simulation provides an efficient method of detecting functional faults at the earliest possible moment. Test cases can be run and validated in the host environment long before they are run in the much more expensive and complicated target hardware environment. SCADE Test Environment for Host allows developers using SCADE Suite to automate the creation and management of test cases. The coverage of tests can be analyzed using the Qualified Model Test Coverage tool. Complete test coverage ensures that the design fully complies with its software requirements. SCADE Test Environment for Host automates the process of running these test cases and generating conformance and model coverage reports, resulting in significant time and cost savings relative to manual testing. The tool has been qualified as a verification tool for DO-178C/DO-330, so the results prove the compliance of a SCADE model with its HLRs. The same functional test execution is provided on host and targets independent of code generation.

The generated code fulfills embedded application constraints such as avoiding dynamic allocation, bounded loops, etc., and is readable, commented and easily traceable back to the requirements. Integration is made easy by a Python API that gives access to generated objects. Consistency of module integration is verified at the model level before generating the Ada code, eliminating the need for integration verification at the code level. The SCADE KCG Ada code generator is qualified as a DO-178C development tool (cf. section 12.2 of DO-178C), so the conformance of the code to the input model is trusted, eliminating the need for verification activities related to the coding phase as the objectives are automatically fulfilled. The SCADE Ada code generator is the only Ada code generator developed according to key high-integrity standards including DO-330 (DAL A) TQL1, EN 50128 (SIL 3/4) and IEC 61508 (SIL 3) T3 and ISO 26262 TCL3. The same SCADE model can be used to generate Ada or C code so that it's easy to change from Ada to C or vice versa, and/or to generate both Ada and C code so the results can be cross-checked for resiliency in a high-safety environment.

Substantial Reductions in Development and Verification Cost

SCADE users have achieved substantial reductions in development and verification cost. Users report that they have doubled the average number of executable lines of code developed per person per day, from five for manual coding to 10 with model-based design, while also reducing execution time. Software certification cost is reduced by an average of 50 percent. Coding, review and testing cost are reduced by 70 percent to 90 percent. The software update cycle time is shortened by 65 percent to 75 percent. SCADE tools provide additional savings by eliminating manual coding errors and the need for low-level testing. The cost of design changes and associated testing throughout project lifecycle and testing cost is reduced by 70 percent to 90 percent.

Conclusion

ANSYS SCADE Suite provides a complete solution for development of high-integrity Ada applications, supporting the complete development cycle from system level, design level and testing. A formally defined modeling environment provides all the inherent benefits of the model-based design approach while avoiding the ambiguity inherent in other Ada modeling environments. Modeling capabilities that rely on a formal notation ensure accuracy and determinism of the model behavior and enable automatic coding and automatic verification of model consistency. A deterministic model-based development process supported by a qualified tool chain delivers the following benefits:

- Syntactic and semantic consistency checks of the model are automatic
- Coding is automatic and source code review is not required
- Design and verification are done at the model-level, thus identifying problems early in the development cycle
- Verification is primarily high-level and requirements-based
- The same SCADE model generates ready-to-embed Ada or C code qualified according to DO-178C
- SCADE Suite has a long track record of successfully certified projects

ANSYS, Inc.
Southpointe
2600 ANSYS Drive
Canonsburg, PA 15317
U.S.A.
724.746.3304
ansysinfo@ansys.com

If you've ever seen a rocket launch, flown on an airplane, driven a car, used a computer, touched a mobile device, crossed a bridge or put on wearable technology, chances are you've used a product where ANSYS software played a critical role in its creation. ANSYS is the global leader in engineering simulation. We help the world's most innovative companies deliver radically better products to their customers. By offering the best and broadest portfolio of engineering simulation software, we help them solve the most complex design challenges and engineer products limited only by imagination.
Visit www.ansys.com for more information.