Is UML Dead? The New Future of Executable Modeling

Name Leon Starr

Abstract text The history of software modeling, particularly in embedded systems, has a mixed record of success and failure. Mostly failure. Yet, in other fields of engineering, modeling is accepted as a necessary and valuable aspect of the design process. Despite the rosy promises of tool vendors and standards organizations, attempts to do the same with UML and SysML in software engineering have been disastrous. Why? Some would argue that software development has a creative, magical quality to it that makes modeling unnecessary or impractical. Those of us who build software systems for safety or mission critical systems might argue otherwise. So, if models are essential to engineering and software should truly be engineered, lack of widespread software modeling suggests that we have been doing it wrong. Very wrong.

Fortunately, a number of factors are putting productive, agile modeling back in practice for embedded systems and software systems in general. These include the advent of executable modeling so that abstract models can be run and debugged like programs, domain engineering to bridge the gap between systems and software design, platform independent modeling for portability and rapid adaption to evolving implementation technology, and a range of newly available open source modeling and code generation tools.

The SAAB Gripen E, in particular, is benefiting from this renaissance in systems and software modeling throughout the aircraft. So, what is being done differently here? How does UML and SysML distract us from solving real problems? How is executable modeling and open source changing the game? What is the real point of modeling and how can you unleash its power without wasting your time pushing meaningless boxes and rectangles around? Finally, is UML dead and if so, how dead is it?