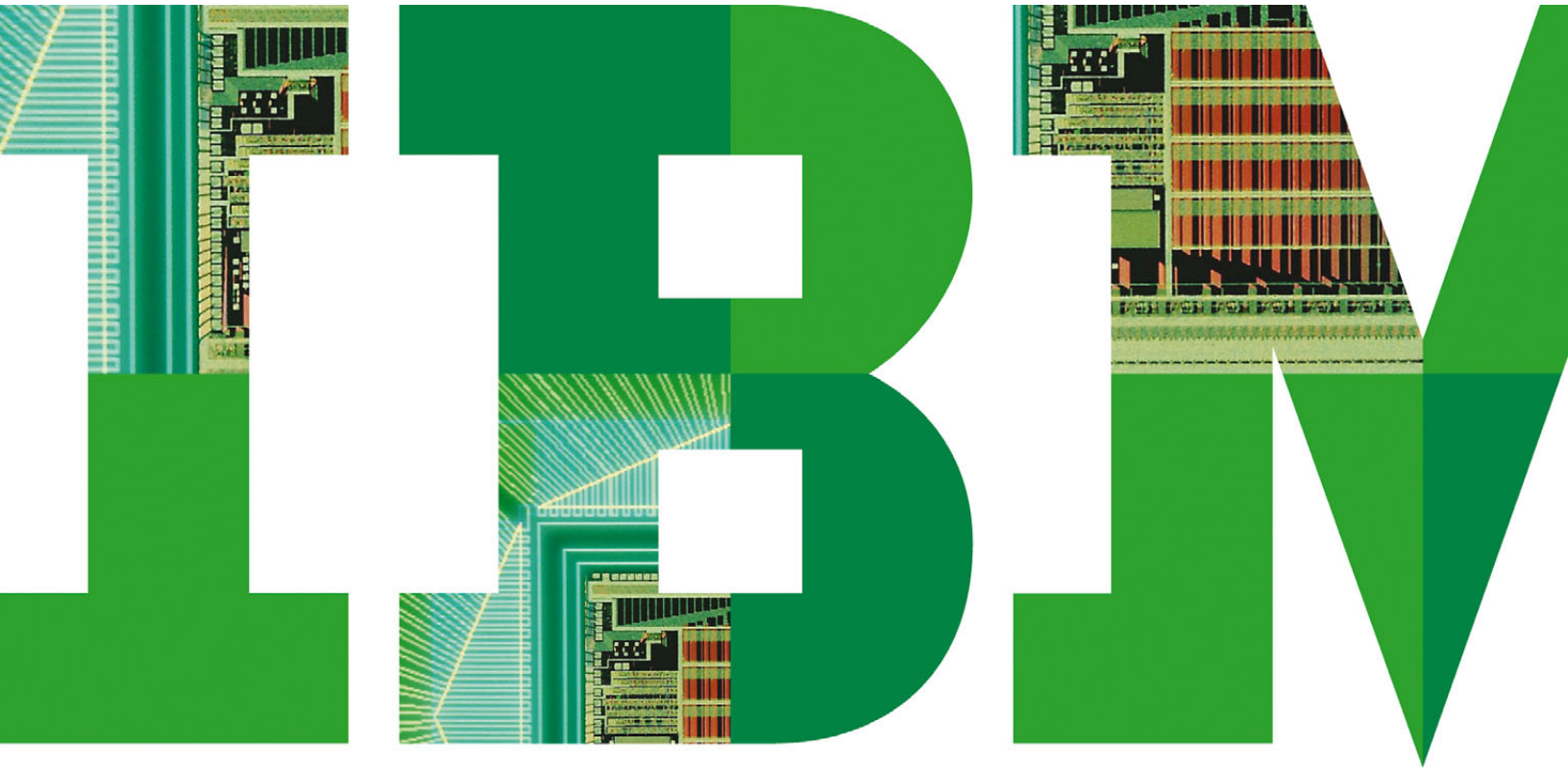
IBM Software
Thought Leadership White Paper

Accelerating product development through codevelopment of hardware and software



IBM

Introduction

The combination of hardware and software is driving the next wave of innovative products. For designers of electronics systems, rising product complexity and marketplace demands present major challenges. To stay competitive, tight coordination between the processes used by software and hardware engineers is no longer just an amenity—it's critical to optimizing product quality in the limited available time while controlling costs.

This white paper discusses key elements of rapid codevelopment. It also explores how sharing processes and tools between hardware and software development enables teams to codevelop smarter products more efficiently, resulting in shorter time to market and improved product quality.

Why codevelopment can't wait

Competitive pressures and shifting customer expectations are forcing manufacturers to develop and launch ever more complex products in shorter timeframes. To address this challenge, companies are looking at software to differentiate their products by delivering innovative functionality.

Gone are the days when software developers would wait for the latest release of a chip and then determine how to use it; today software frequently drives hardware choices. As a result, companies are moving away from the traditional approach of software and hardware development teams working independently with multiple silos of teams, with different processes, separate tools and unique terminology. They recognize the need for interaction at each stage in development to understand the cascading effects that one change can have on another component, discipline or team.

Today's hardware and software disciplines need to evolve together to provide a high-quality working component, product or system. With a collaborative life-cycle approach to design and development, hardware and software teams can do the following:

- Build to a consistent set of requirements
- Evaluate system design early in the process through modeling
- Have visibility into all change requests while maintaining the traceability needed to relate the problem to a hardware problem, a software problem or both

The unified V process: a foundation for codevelopment

Currently, hardware and software teams use similar processes for development. The standard V diagram shows the typical stages both teams use for analysis, design, implementation and testing. The challenge is that the teams operate separately, limiting their ability to synchronize key steps. What's more, alignment is hindered by the use of different languages and tools. To achieve rapid codevelopment and the associated business benefits, the hardware and software processes need to be integrated into a unified process.

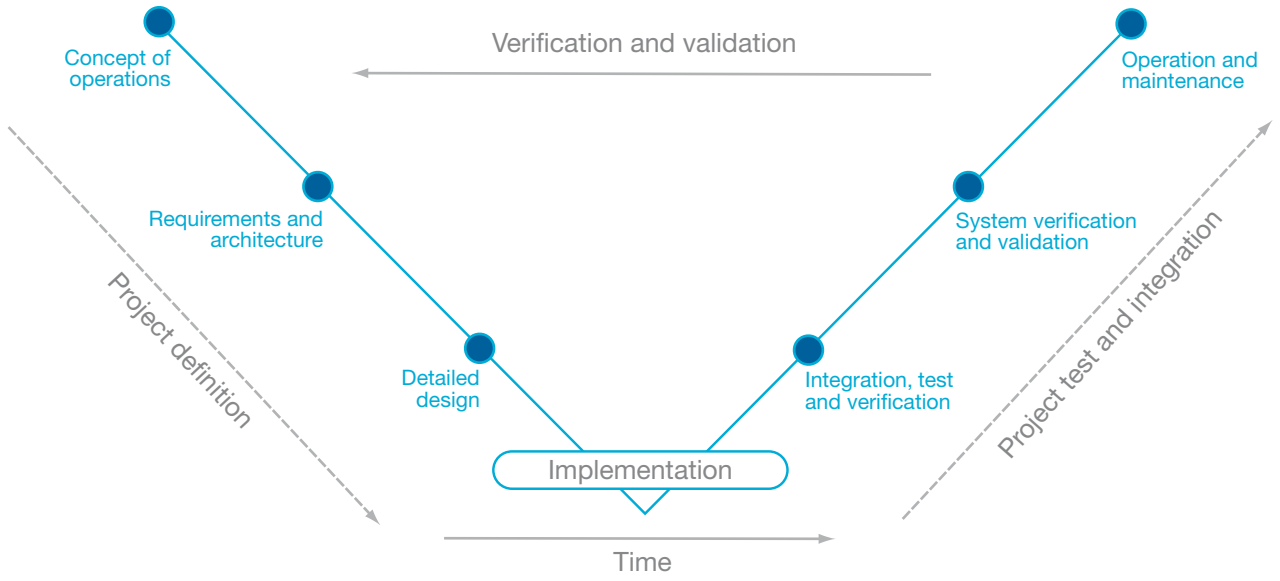


Figure 1: The steps in the standard V development process

Synchronization from the requirements phase through the testing phase will help teams gain efficiencies while improving quality to get better returns from short product life spans. Depending on your specific processes, there are many potential synchronization points, and exploring these points is key to enabling effective collaboration.

Requirements engineering: managing complex requirements relationships

Over time, the importance of requirements has become increasingly critical. Requirements provide the basis for stakeholders (customers) to interface with the engineering teams. They can also be linked to tests to prove that what stakeholders request is realized in the product. As such, project success can hinge on the ability of hardware and software teams to come together and develop the requirements correctly.

Typically high-level system requirements are thrown over the wall to hardware and software teams that work independently. After the teams receive the requirements, functionality may need to migrate between the teams, but that may not happen until the first sync point months down the road. This can lead to major integration and optimization issues. For example, if the software team decides to add functionality to the field-programmable gate array (FPGA), the hardware team may be unaware that the FPGA will need more capacity, which could

result in buffer overflows. As a result, it's important that these types of decisions are made early with involvement from hardware and software teams.

While both teams use requirements to trace their artifacts, they typically use different tools. The use of these separate tools means they don't have a single set of requirements they can use to trace through both components. As a result, troubleshooting requirements change requests is usually time-consuming and often requires a guessing game as to where the issue originated from. Moreover, many requirements are met by both hardware and software, so the ability to show and follow links between the different components is important. To overcome requirements-related challenges, hardware and software teams need a common tool that can provide traceability to all other tools.

System modeling for earlier insight into potential issues

Given the complexity of today's products, the earlier you can look at a system and understand it, the more you can reduce risks and save time and money in the long run. Not only is it important to make sure that all system parameters are captured, but learning where architectural trade-offs between hardware and software are necessary is also key.

One of the most effective ways to reduce ambiguity and capture and evaluate a system early in the development process is through modeling. Yet hardware and software teams usually lack the means to efficiently create comprehensive product models. Systems Modeling Language (SysML), which is an industry-standard language that systems engineers use to design systems, can fill this gap. Software engineers who use Unified Modeling Language (UML) can read SysML because they are simply different dialects of the same base language. Typically hardware developers rely on other languages such as SystemC. Today, however, tools are available that can generate SystemC code used by hardware engineers directly from SysML. As a result, SysML provides an ideal bridge between the hardware and software teams and systems engineers for modeling.

SysML-based models provide teams with a number of ways to streamline development. They can use models to simulate the design to understand its dynamic behavior, whether requirements are complete and correct, and where architectural trade-offs are needed. Moreover, teams can automatically generate test vectors from models for use during system integration.

Improving efficiency with common configuration and change management

Siloed software and hardware tools present another difficult set of challenges related to managing configurations and changes. Separate development infrastructures are costly to maintain and administer given the need to create multiple (usually brittle) integrations between the toolsets. Couple that with the fact that hardware and software teams use different tools and terminology, and you have a recipe for redundant work, guesswork or miscommunication.

For example, a change to a configuration may need to be entered into multiple tools, increasing the likelihood of errors and wasting valuable time. Or consider change request submissions. Team members often don't know whether they are related to hardware or software and have to make judgment calls, making it possible that they will submit the change request to the wrong tool. Moreover, because multiple tools are used to track change requests, there is no easy way for project managers to pull metrics on all outstanding issues. Overall, this situation makes team synchronization virtually impossible.

In addition to impeding the change request process, siloed tools also create problems when integration engineers are ready to build the next version of a product. To optimize efficiency and the chances of project success, engineers need easy access to the right versions of all components. When traceability is not available and change requests are entered into the wrong tool, key players may not have an opportunity to comment on why something did or did not happen, compromising engineers' abilities to fully understand the issue. This can have a serious effect because without shared visibility by hardware and software teams, the wrong components may be chosen.

Case study: IBM Systems and Technology Group

In 2003, the IBM Systems and Technology Group was trying to create a chip with a converged layer of firmware that would work across multiple brands. Going into the project, the teams had a disparate set of tools for change request tracking, source control and other key areas that made efficient program management virtually impossible. For example, program managers had to merge change request information from multiple groups into a single set, but by the time they finished, their work would already be outdated.

To overcome this issue for future projects, the group implemented IBM Rational® ClearQuest® software. The implementation identified a high level of commonality across the teams, which led to much more efficient processes and significant cost savings. To the surprise of IBM, system hardware developers became the largest user base of the solution. Upon seeing the success of the solution, the integrated supply chain group also requested convergence of its change request tracking solutions, enabling clearer insight into problems within manufacturing and between the supply chain and the hardware and software development labs. This solution has been deployed within IBM across more than 25,000 engineers and programmers spread across 35 labs and 17 countries. Overall, the benefits include the following:

- Significant reduction in development costs
- Virtual elimination of abandoned projects
- Reduction of warranty costs
- Large increase in standard parts reuse
- Increase in systems development efficiency
- Greater program management efficiency

The IBM solution for codevelopment

IBM offers a comprehensive set of open tools and capabilities to support rapid and efficient codevelopment processes. Based on the IBM Rational Platform for Systems and Software Engineering solution, Rational tools can integrate with existing electronics design automation and software development tools to allow an integrated hardware and software process flow across the development life cycle.

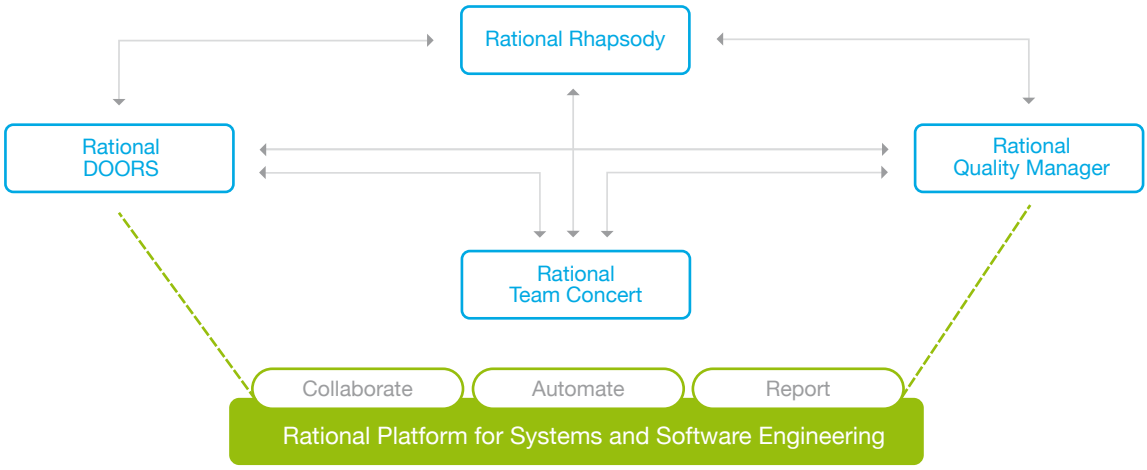


Figure 2: The Rational Platform for Systems and Software Engineering solution

Key Rational products include the following:

- **IBM Rational DOORS® software—**a family of requirements definition, management and engineering solutions that help hardware and software teams establish a common set of requirements; help ensure the final product is built to specification; and help improve traceability for compliance, increase reuse and streamline collaboration among teams
- **IBM Rational Rhapsody® software—**a visual development environment that enables diverse teams to abstract complexity in models using standard languages, including SysML, which can be used to generate SystemC code for earlier validation of functionality and better insight into where software and hardware trade-offs are needed
- **IBM Rational Quality Manager software—**a quality management solution that enables quality assurance teams to track and manage both hardware and software tests and provides a means to seamlessly share information, use automation to accelerate project schedules and report on project metrics across the life cycle
- **IBM Rational Team Concert™ software—**an integrated, collaborative development environment that can help teams build better products with the help of source code management, work item management, build management, better insight into project health, and integrated reporting and process support
- **IBM Rational ClearCase® software—**a configuration management solution that provides sophisticated version control, workspace management, codevelopment support and build auditing to improve productivity
- **IBM Rational ClearQuest software—**a comprehensive defect and change tracking solution with process automation that enables better visibility and control across hardware and software development

With these capabilities, you can better position yourself to more confidently and cost-effectively release complex products in tight timeframes.

Integrating with EDA tools

IBM is teaming with leading electronic design automation (EDA) vendors to build the integrations needed for streamlined codevelopment between hardware and software teams.

- **IBM Rational Rhapsody software's SysML support combined with technology from IBM Business Partners includes the ability to generate SystemC code that can be used by leading EDA tools.**
- **IBM Rational ClearCase software can be combined with Cadence Design Framework II software to provide configuration management for hardware and software designs.**
- **IBM and Cadence provide system-on-chip developers a comprehensive test and simulation environment with the enterprise verification management solution (EVMS).**

Improving business outcomes with unified processes

Business-as-usual product development has become a barrier to speed in the face of demanding marketplaces and rising product complexity. While hardware and software engineers use similar processes, they aren't staying in sync. This leads to product delays as well as usability and quality issues. Achieving effective codevelopment processes that support cross-discipline collaboration and cooperation requires better integration among all of the tools in the process—for hardware and software development, requirements, modeling, configuration management, and more. By adopting an integrated common tool chain, teams can reduce risks and be more confident that they are building to customer expectations.

Adopting the shared processes between hardware and software described in this paper can help your teams codevelop innovative products faster—at a potentially lower cost. When you're ready to explore codevelopment solutions, look no further than IBM.

For more information

To learn more about the integrated solution for codevelopment from IBM Rational and IBM expertise in electronics, contact your IBM sales representative or IBM Business Partner, or visit: **ibm.com**/software/rational/solutions/electronics

Additionally, financing solutions from IBM Global Financing can enable effective cash management, protection from technology obsolescence, improved total cost of ownership and return on investment. Also, our Global Asset Recovery Services help address environmental concerns with new, more energy-efficient solutions. For more information on IBM Global Financing, visit: **ibm.com**/financing

IBM Corporation
Software Group
Route 100
Somers, NY 10589
U.S.A.

Produced in the United States of America
October 2010

Please Recycle

Rational. software

RAW14227-USEN-00