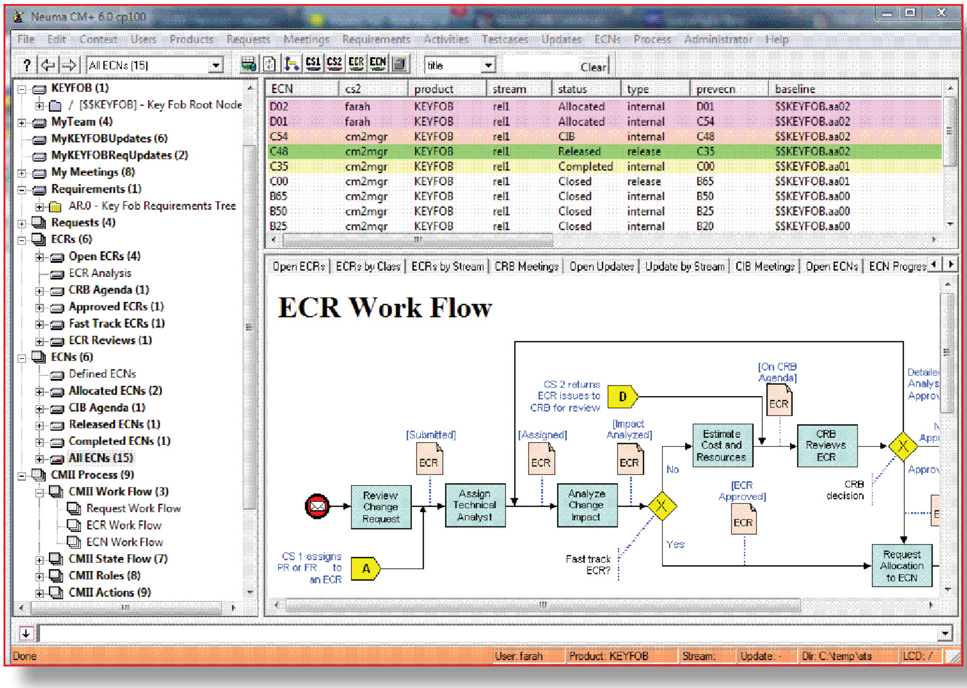
CMII for Software

by Joe Farah and Rick St. Germain

Lift your eyes from this page and take a look around you. Chances are you'll see many things that use software to make them work. A cell phone. Your MP3 player. Maybe even a computer like the one I'm using to write this article. The fact is, all but the most trivial consumer products we commonly use have software in them. Software is pure functionality – a form of knowledge – expressed in an executable way that allows us to control the behavior of hardware.



ECN	cs2	product	stream	status	type	prevecn	baseline
D02	farah	KEYFOB	rel1	Allocated	internal	D01	$$KEYFOB.aa02
D01	farah	KEYFOB	rel1	Allocated	internal	C54	$$KEYFOB.aa02
C54	cm2mgr	KEYFOB	rel1	CIB	internal	C48	$$KEYFOB.aa02
C48	cm2mgr	KEYFOB	rel1	Released	release	C35	$$KEYFOB.aa02
C35	cm2mgr	KEYFOB	rel1	Completed	internal	C00	$$KEYFOB.aa01
C00	cm2mgr	KEYFOB	rel1	Closed	release	B65	$$KEYFOB.aa01
B65	cm2mgr	KEYFOB	rel1	Closed	internal	B50	$$KEYFOB.aa00
B50	cm2mgr	KEYFOB	rel1	Closed	internal	B25	$$KEYFOB.aa00
B25	cm2mgr	KEYFOB	rel1	Closed	internal	B20	$$KEYFOB.aa00

Software development teams generate vast quantities of rapidly changing information. What we often consider to be "software" – the "code" – is just the final work product, or more precisely, a by-product of applying acquired knowledge.

Like any other systems generation activity, software development acquires and applies knowledge. It is strongly driven by specifications (requirements). Whether this knowledge is expressed as software or as hardware is a matter of implementation detail.

Properly structured information linked to their corresponding work products and the management of change to both is what CM is all about. That's what CMII is all about.

In the CMII community, we've seen plenty of examples of applying CM to the hardware development process. Not so much to the software process. And although there are many similarities between software and hardware CM, such as the need for requirements management and various product baselines, there are also a number of differences that have to be addressed when applying CMII to software.

Over the past two years, Neuma Technology, a leading-edge software CM (SCM) company, teamed up with the CMII expertise of Nouvella Consulting Services, with the support of the National Research Council of Canada (NRC), to address the application of the CMII process to software development. Some of the key issues that had to be resolved included:

- Hardware items have both functional and physical properties, while software is pure function delivered to a hardware system(s) outside the software system. Because software must be designed to a specific hardware/product context, requirements traceability is a crucial capability.

- Hardware production involves a significant investment in time, effort and cost compared to software production that is a relatively quick and automated process. This has a major impact on the way software systems are developed and tested.
- Executable software, and the source code specifications that guide their production, have both physical and logical properties. A software CM tool must support the changing physical location of software files and their logical relationships (where used, who do I call, who calls me, etc…) in order to effectively conduct change impact analysis.
- The “physical” (file) hierarchies of software systems tend to be much shallower but with more components to manage, typically in the tens of thousands.
- Re-use of designed hardware items in production is by manufacturing duplicate items in the desired quantities and schedule. Software item re-use is by reference (calling) the original. “Quantity” has little relevance in software (always ONE).
- Software changes are organized logically by function, rather than by physical parts (files). Each impacted function is assigned to a single developer or small team in a manner similar to Fast Track. All physical components related to the function are handled as a unit called a change package or “update”.
- All artifacts of the software development process, including executable code, are stored in the repository. Each artifact migrates through a series of states, defined in the CMII workflow that indicates its “readiness” leading up to production release.

Industrial strength performance, reliability, scalability
NEUMA CM+
VERSION 7.0 - CMII© PROFILE

©Copyright 2008, Neuma Technology Inc., the CM+ company.

The result of this joint Neuma/Neuvella effort was emergence of Neuma’s CM+ CMII Profile Edition as the first CMII-certified SCM application specifically designed for advanced support of software development. This is strategically important because it allows both hardware and software teams to work from the same CMII process. The resulting interaction between the two teams improves communication and resource flow. And although differences still remain when using CMII for hardware and for software, working from a common CMII page allows both sides to understand, rather insulate themselves from, the differences.

If you’d like to learn more about the application of CMII to software development or about Neuma’s CMII-certified CM+ SCM application, visit the Neuma booth at the CMII World 2008 conference in Orlando or contact Neuma at sales@neuma.com.

Joe Farah *is President and CEO of Neuma Technology based in Ottawa, Canada and is a regular contributor to the “CM Journal” (www.cmcrossroads.com). Joe has been involved in software CM tool development for over 30 years. You can contact Joe at farah@neuma.com*

Rick St. Germain *is President and Managing Director of Nouvella Consulting Services Ltd., a CM consulting firm based in Ottawa, Canada, that assists companies worldwide in defining and establishing sound configuration management practices. He holds the CMII Professional and Instructor certifications. You can contact Rick at rstgermain@rogers.com*