

Filling the Technology Void for Industrial Software Product Lines¹

Charles W. Krueger

BigLever Software, Inc., 10500 Laurel Hill Cove, Austin, TX, 78730, USA.

Tel: +1 (512) 426.2227. Fax: +1 (512) 795.9854.

ckrueger@biglever.com

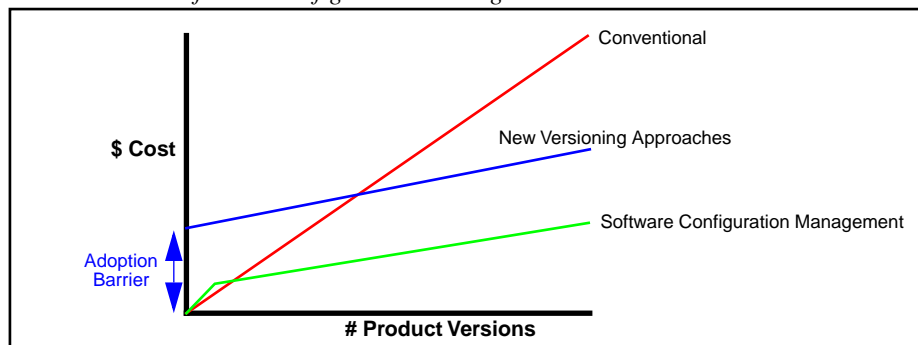
http://www.biglever.com

1 Position Statement

Imagine trying to engineer industrial grade software without configuration management technology. Each engineering organization would spend an inordinate amount of time contriving complex solutions based on combinations of file and directory naming conventions, preprocessor logic, runtime conditionals, special build scripts, special install scripts, and so forth. As more and more product versions were created for product evolution and parallel maintenance, the cost would grow sharply as illustrated by the *Conventional* line below.

To address the problem, researchers would come up with processes, methodologies, and techniques using conventional technology to address the system versioning problem. Although the long term operating costs would be less for each product version, the cost of adopting and deploying these new approaches would require considerable up-front effort and might require a significant paradigm shift, as illustrated by the *New Versioning Approaches* line below.

Eventually, someone would recognize that there was a void in the software engineering technology space and devise a true software configuration management technology, focused only on that separate concern, that could be easily adopted into existing practices without a paradigm shift and without the associated adoption barrier, as illustrated with the *Software Configuration Management* line below.



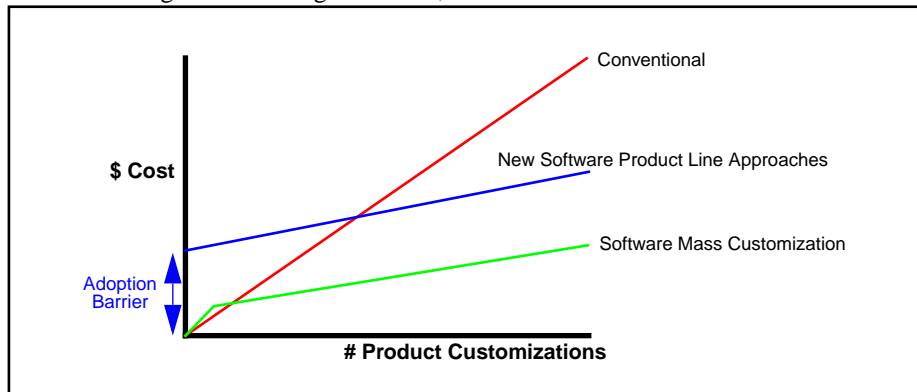
Of course, the moral to this imaginary tale is that we are in the same situation with regard to the industrial practices surrounding software product lines. We have learned

1. © Springer-Verlag

that conventional approaches are expensive and ineffective, that adopting new software product line approaches can be an expensive, long-term, high-risk endeavour, and that there is a void in the technology space for supporting true software mass customization.

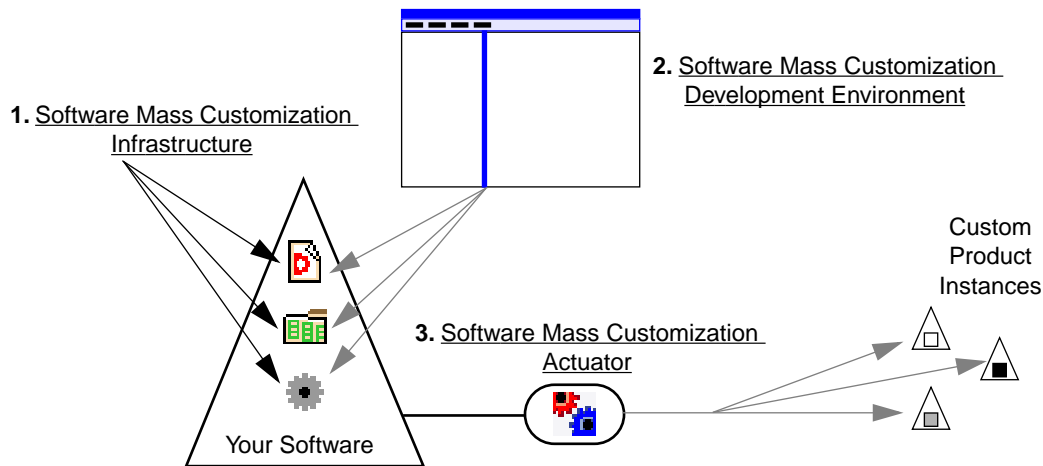
To address this problem, we have created a commercially available technology that allows existing one-of-a-kind software, tools, techniques, and organizational structures to be incrementally transitioned into a *Software Mass Customization* practice. Reuse of existing software, tools, techniques, and organizations removes the adoption barrier associated with re-architecting, re-designing, re-implementing, re-tooling, re-organizing, and shifting paradigms.

The leverage of this approach comes from *separation of concerns*. Existing software engineering technology and techniques are sufficient for most of the software product line engineering tasks except for managing variation. We add a separate technology to the conventional technology mix that is specifically focused on managing product line variation at the artifact level. The cost curves, analogous to our imaginary software configuration management tale, are shown below.



Our approach is illustrated in the diagram on the following page. It shows a *software mass customization production line* that generates different custom software products by assembling different collections of files, according to a feature model that express the possible customizations. There are three technology components to this approach to software mass customization:

1. **Software mass customization Infrastructure.** The software mass customization infrastructure consists of special-purpose files and directories that are added to your software in order to create a software mass customization *production line*, or the generator. The infrastructure supports feature modeling, variation points at the source file level, logic to map from feature model to variation point instantiation, and product definition.
2. **Software mass customization Development Environment.** The software mass customization development environment is used to browse, create, organize, and maintain the infrastructure for your software mass customization production line.
3. **Software mass customization Actuator.** The software mass customization actuator will activate your software mass customization production line in order to generate custom instances of the software product.



2 Experience

The fundamental ideas described above evolved from the author's PhD thesis research[5]. These concepts have been researched, developed, and applied in practice over the last 15 years[2][3][4]. The approach is currently being commercialized and actively deployed into practice by BigLever Software, Inc. as a collection of technology and services to aid software development organizations make the transition to software product line practice[1].

3 Expected Contributions

Software mass customization, as interpreted in this work, offers a unique perspective on software product lines that can eliminate the adoption barrier. We take an extremely pragmatic approach to sound computer science principles to place software mass customization within the grasp of the software industry.

References

- [1] BigLever Software, Inc. Austin, TX. www.biglever.com
- [2] Krueger, C. Using Separation of Concerns to Simplify Software Product Family Engineering. *Proceedings of the Dagstuhl Seminar No. 01161: Product Family Development*. April 2001. Wadern, Germany.
- [3] Krueger, C. Easing the Transition to Software Mass Customization. *Proceedings of the 4th International Workshop on Product Family Engineering*. October 2001. Bilbao, Spain. Springer-Verlag, New York, NY.
- [4] Krueger, C. Software Reuse. 1992. *ACM Computing Surveys*. 24, 2 (June), 131-183.
- [5] Krueger, C. 1997. *Modeling and Simulating a Software Architecture Design Space*. Ph.D. thesis. CMU-CS-97-158, Carnegie Mellon University, Pittsburgh, PA.