Leveraging the Model Driven Development and Software Product Line Engineering Synergy for Success

Charles W. Krueger, Ph.D., Big Lever Software
Martin Bakal, Telelogic

Version 1

05 May 2008
# Table of Contents

- **Preface**  
  3
- **Introduction**  
  4
- **Background on SPLs and BigLever Software Gears**  
  6
- **Background on MDD and Telelogic Rhapsody**  
  8
- **Rhapsody/Gears Bridge – Integrating MDD and SPL**  
  10
- **A Handset Product Line Example**  
  12
- **Conclusions**  
  16
- **About the Author**  
  17
- **About Telelogic Rhapsody**  
  18
- **About BigLever Software Gears**  
  18
- **About BigLever**  
  19
Preface

Developing software for a portfolio of similar products is a formidable task, but Software Product Line (SPL) engineering practices effectively address this challenge. Model Driven Development™ (MDD™) raises the level of programming abstraction with UML® to make software engineering more efficient. With the Telelogic Rhapsody®/BigLever Software Gears™ Bridge, engineers can now leverage the synergy of the industry’s first integrated MDD and SPL engineering solution, providing a simple, elegant approach that enables organizations to effectively manage product line diversity in MDD for faster development of more new products and features.
Introduction

The motivation for integrating Rhapsody and Gears via the Rhapsody/Gears Bridge is to capitalize on a strong synergy that results by combining Model Driven Development (MDD) and Software Product Line (SPL) engineering. MDD is well known for its ability to accelerate system and software development by leveraging the higher level of abstraction provided by UML and SysML™. SPL engineering provides the ability to efficiently and effectively create, maintain and evolve a portfolio of similar products by taking advantage of a very strategic and predictive form of software reuse. Combined, an entire product line can be expressed and engineered from a single configurable SPL/MDD model.

With MDD, creating software for a portfolio of similar products has traditionally relied on one of two different approaches, clone-and-own or one-size-fits-all. In the clone-and-own approach, the model for every new product is created by making a copy — or clone — of the model for a similar existing product and then modifies that model so that it implements the unique features and characteristics of the new product. There is 100% reuse at the time of the cloning, but similar to clone-and-own of conventional source code, the duplication can lead to divergence over time and require merging or replicated development among the different models, adding to the time and cost of maintenance and evolution.

Some organizations adopt the one-size-fits-all approach to avoid the overhead of clone-and-own. With the one-size-fits-all approach, the product features and product diversity for an entire product line portfolio are implemented in a single model, using meta-logic and configuration files to allow decisions about which product feature alternatives to include in any particular product to be made at runtime. This eliminates the need for cloned models. Similar to one-size-fits-all in conventional source code, this approach can lead to models that continue to grow larger over time as more and more products and features are added to the portfolio.

With SPL engineering, specialized tools and methods are provided for efficiently creating, maintaining and evolving software assets for a product line portfolio of similar products. A high degree of software reuse within a well-defined architecture of a product line is possible through explicit and encapsulated variation points at the software implementation level, a formal feature model that expresses the feature diversity that differentiates the products in the product line, and a product configurator that automatically instantiates products by configuring the variation points based on feature choices in the feature model.

In the past, SPL tools and methods have primarily focused on conventional source code, requirements and test cases. SPL support has not been available for variation points that capture product diversity in UML and SysML model elements in MDD. SPL support has not been available to allow MDD models to serve as reusable SPL core assets that can be automatically configured by an SPL product configurator.

In response to customer demand to combine the latest generation of SPL engineering technology with modern MDD tool support, BigLever Software and Telelogic teamed up to create a bidirectional integration between Telelogic’s Rhapsody MDD solution and BigLever’s Gears SPL tool. The result of this collaborative effort is the Rhapsody/Gears Bridge, combining the strengths of the MDD and SPL approaches.

With the Rhapsody/Gears Bridge, Rhapsody models can now be first-class SPL reusable core assets in a Gears software product line. These model core assets can be interspersed with other types of SPL core assets including requirements, conventional source code, documentation, test cases and so forth. The Gears variation point concept is extended to include Rhapsody model elements in order to express model diversity in the model elements. Automated product configuration in Gears now extends via the Bridge to include automated configuration of different model instances based on feature selections in the Gears feature profile for a particular product instance.

With the Rhapsody/Gears Bridge, UML and SysML models explicitly show both the common and varying parts of the product line design. The design of each SPL variation point — including the op-
tions, alternatives and the logical specification that differentiates them — is directly visible in the model.

The Rhapsody/Gears Bridge provides the combined simplicity and benefits offered by MDD and SPL approaches — including significant gains in productivity, reduction in defect density and faster time-to-market with new products — as well as synergistic benefits that dramatically increase the scalability of a product line portfolio.
Background on SPLs and BigLever Software Gears

The software product line problem can be characterized as a mismatch problem. Customer demand requires most companies to create a product line portfolio of similar products rather than just a single product. The software for these products must be developed to support the feature and function diversity within the product line. Examples of software product lines are seen in all markets, including telecom, computer printers, automotive, industrial automation, avionics, defense, medical systems, data storage systems, e-commerce and so forth. However, as illustrated in Figure 1, most of the software development methods, tools and techniques today take a product-centric approach, with a focus on how to take a single product through the development lifecycle — from requirements through design, implementation, and test — and do not provide explicit and effective support for developing a product line portfolio.

Case studies show that using product-centric tools and methods for SPL engineering can account for 50%-90% of the total software development cost and effort. SPL approaches provide new capabilities for eliminating this overhead waste.

Figure 2 illustrates how the SPL approach with Gears differs from the product-centric approach in Figure 1. The Gears software product configurator takes two types of inputs — reusable core assets and product feature profiles — in order to automatically create product instances within a product line portfolio. The reusable core assets contain variation points to encapsulate implementation-level diversity needed for the product line. Core assets can include requirements, architecture and design, source code, test cases, product documentation, and so forth. The product feature profiles are concise abstractions that characterize feature differences and selections for the different product instances in the product line portfolio, analogous to an options list for optional and alternate features that are available on a new car.
The Gears SPL methodology eliminates the duplication, divergence and merging of the product-centric approach by consolidating all the development effort into the single collection of reusable core assets. This results in a dramatic increase in productivity. The organizational structure of core asset teams is much more stable and focused compared to the product-centric approach, particularly as new products and features are introduced into the portfolio. This results in a significant overall increase in portfolio and feature quality. The business and engineering teams in a company can manage the evolution of the product line portfolio based on the concise and formal representation of the feature profiles, allowing new products and features to be deployed with significantly less time and effort. This yields major reductions in time-to-market and large extensions to portfolio scalability.
Background on MDD and Telelogic Rhapsody

MDD technology enables professionals to achieve unparalleled productivity gains over traditional document driven approaches by enabling users to specify the system design and architecture graphically, simulate and automatically validate the system as it is being built. This allows engineers and developers to ultimately produce a quality systems specification that is correct, non-ambiguous and completely satisfies original requirements.

As illustrated in Figure 3, Rhapsody provides systems engineers and software developers with UML/SysML compliant products that can be extended for domain-specific modeling, providing a truly collaborative development environment that allows both large and small teams to communicate effectively and productively. Integrated requirements management and traceability features ensure that the design always meets the requirements. Design for Testability (DFT) capabilities reduce defects early in the process and always validate against the requirements. Rhapsody accelerates development by generating full applications, rather than just code frames. These technologies, packaged in an easy-to-use format, make Rhapsody the ideal solution for software and systems engineers.

Figure 3. Model Driven Development with Rhapsody
Through Rhapsody's Model Driven Architecture (MDA) support, development teams can rapidly target the Platform Independent Model (PIM) to a real-time embedded operating system in seconds. Rhapsody lends itself to a design approach where the software can be constantly executed and validated on the host environment, then brought down to the embedded target for target based testing. By fully integrating the specific demands of the systems engineer and the software developer, Rhapsody places a powerful, feature-loaded tool in the user's hands so that high quality systems and software can be developed in a shorter timeframe.
Rhapsody/Gears Bridge – Integrating MDD and SPL

The Rhapsody/Gears Bridge is a dual plug-in between Rhapsody and Gears, shown in Figure 4. The Bridge provides SPL capabilities in MDD and extends the set of supported SPL core assets to include MDD models.

On the Gears side of the plug-in bridge, Rhapsody MDD models can be included as Gears SPL modules, or core assets. The Gears product configurator can automatically instantiate Rhapsody models based on the feature selections in Gears feature profiles, along with the configuration of other Gears core assets such as requirements, documentation, conventional source code and test cases.

On the Rhapsody side of the plug-in bridge, Rhapsody model elements can be converted into first-class Gears variation points. This allows optional elements and alternative behaviors to be specified for model elements, where the options and alternatives reflect the feature diversity that needs to be supported at the model level. Gears SPL operations, editors and power tools are available directly from the Rhapsody menus, providing seamless interoperation.

Figure 5 illustrates the integrated MDD and SPL approach of the Rhapsody/Gears Bridge. In contrast to Figure 3, Rhapsody MDD Models can now be included as first-class reusable SPL Core Assets with variation points.
Figure 5. Gears Software Product Line with Rhapsody Core Assets
A Handset Product Line Example

The simple handset product line shown in Figure 6 illustrates by example the integration of MDD and SPL via the Rhapsody/Gears Bridge. This product line comprises three products and four features. Each feature column has an enumerated set of choices:

- **Call Connectivity** models the types of phone calls that are supported on the handset, Voice only or both Video & Voice.

- **Memory** models three levels of flash RAM that can be configured on the handset, Low, Medium, or High.

- **Call Recording** models which types of calls can be recorded for replay, None, Voice, or Voice & Video.

- **LED Flashlight** models a Boolean choice of whether or not the handset contains a built-in LED flashlight.

The three different product instances have feature profiles that can be seen across the rows:

- **Low-End** handsets are oriented towards developing regions around the world, where low cost is important and intermittent power grids make the flashlight feature valuable.

- **Mid-Range** handsets provide full call connectivity functionality, but the moderate amount of flash RAM makes video call recording impractical.

- **High-End** handsets with the highest configuration of flash RAM can support voice and video call recording.
Rather than using conventional hand-coded source level development to create the software for the handsets, Rhapsody allows the system structures and behaviors to be expressed using the higher level and visually rich abstractions of SysML and UML. The firmware source code can then be automatically generated and compiled directly from the Rhapsody models.

As described earlier, the traditional approaches for creating MDD models for the three handsets in this product line are:

- **Clone-and-own.** The MDD model for one device would be initially created. A copy (clone) of this model would then be made and modified for the next device. Similarly for the third clone. While there is 100% reuse at the time a cloned copy is made, there is 0% subsequent reuse since enhancements and bug fixes must be made repeatedly to each of the three copies.

- **One-size-fits-all.** One MDD model would be created for all three handsets. Feature and behavioral variations among the three devices would be managed through runtime variations in the model and controlled by different static configuration settings installed on each device. While this eliminates the need for duplicate model copies, the resulting model can become large and complex to maintain. Furthermore, the large footprint executable needed for the high end device must also fit on the low end device, driving up hardware requirements and cost for the low end device.

The integrated MDD/SPL approach offered by the Rhapsody/Gears Bridge provides the benefits of consolidation offered by the one-size-fits-all approach and the benefits of precise customization offered by the clone-and-own approach, without the associated drawbacks. Referring to Figure 5, the reusable Rhapsody Core Assets for the handset product line contains all the feature variations required for the three handsets, encapsulated in Gears variation points. These variation points are automatically con-
figured by the Gears product configurator, based on a Gears feature profile, to produce the precisely customized MDD model needed for each of the three handsets.

A Gears variation point in a Rhapsody MDD model is illustrated in Figure 7. In the upper-left are two model elements, In Call and Toggle Recording, which are part of an activity diagram for managing the handset call recording feature. In Call is a common model element that is the same in each of the three handset devices. Toggle Recording is a variation point model element, as indicated by the gear annotation, that implements the alternate behaviors needed for the Call Recording feature on the different handsets (see the fourth column in Figure 6).

The implementation of the Toggle Recording variation point is composed of the two white boxes in Figure 7, the Gears Variants and the Gears Logic. There are three Gears variants, indicated with ‘V’ icons in the Variants section. These are the three alternate code generation fragments needed by Rhapsody to generate the different call recording variants in the handsets:

- **ActionNoRecording.** Empty code variant for when no call recording is supported in handset.
- **ActionRecordVoice.** Model element code fragment for voice-only call recording in handset.
- **ActionRecordVoiceVideo.** Model element code fragment for voice and video call recording in handset.

The Gears Logic is executed by the Gears Product Configurator (see Figure 5) to instantiate the Toggle Recording variation point, differently for each handset. The When clauses refer to the values in the feature profile for a handset, in order to Select the appropriate Variant based on the feature settings.
When Gears configures all of the variation points in a Rhapsody model to instantiate the firmware for a particular handset, the result is the precise firmware footprint needed for that device – nothing more, nothing less.
Conclusions

Companies face complex challenges in creating and maintaining the embedded software needed to support a rapidly expanding product line portfolio. To better address this challenge, the Telelogic Rhapsody/BigLever Gears Bridge solution is the industry’s first convergence of the synergistic MDD and SPL technologies. Compared to conventional hand coding approaches, SysML and UML provides a powerful enabler for the rapid development of individual products within a product line, as well as better visualization for the maintenance and evolution of those products over time. SPL engineering methods and tools are specifically designed to provide the essential capabilities of expressing, encapsulating and managing the feature diversity within a product line portfolio.

The integration of MDD and SPL technologies provides a simple, elegant new approach that enables companies to effectively incorporate the management of product diversity into their MDD processes. The integration of these highly complementary technologies allows development organizations to more effectively deal with software product line diversity across the entire portfolio development lifecycle.

With this integration, development organizations can achieve significant productivity gains and heightened efficiency by:

- Developing with Rhapsody MDD models rather than conventional source code.
- Using Gears SPL consolidation and automated production capabilities rather than creating cloned copies of MDD models for each product or building “one-size-fits-all” software for all products.

This new level of efficiency enables companies to deliver more new products and features faster, while reducing the development effort and optimizing product quality.
About the Author

Charles Krueger, PhD, is the founder and CEO of BigLever Software, a leading provider of software product line development framework, tools and services. He is a thought leader in the software product line development field, with 20 years of experience in software development practice. He has proven expertise in leading commercial software product line development teams, and helping companies establish some of the industry’s most highly acclaimed software product line practices including Salion, 2004 Software Product Line Hall of Fame inductee, and LSI Logic, 2006 Software Product Line Hall of Fame inductee.

Dr. Krueger is an author and speaker for over 30 articles, columns, book chapters, and conference sessions. He has co-chaired the International Conference on Software Reuse, is a frequent organizer and speaker for the International Software Product Line Conferences, and moderates the SoftwareProductLINES.com practitioner community website. Dr. Krueger has presented at numerous industry conferences including Dr. Dobb’s Architecture & Design World, SD Best Practices, Better Software and OOPSLA. He was an invited author for the Communications of the ACM special issue on software product lines and was featured in a Dr. Dobbs podcast spotlighting software product lines. He received his PhD in computer science from Carnegie Mellon University. Dr. Krueger can be reached at ckrueger@biglever.com.
About Telelogic Rhapsody

Rhapsody is the industry's leading UML 2.1 and OMG SysML 1.0-based Model Driven Development environment for real-time systems and software engineering. With advanced capabilities to extend UML 2.1, Rhapsody allows both function-oriented and object-oriented design techniques to co-exist in one environment. Rhapsody has won numerous awards including the Best in Show award at the Embedded Systems Conferences in San Francisco and Boston from VDC; the SD Times 100 for the third year in a row by taking top honors in the Modeling category; and the Model Driven Development Focus of the Embedded Development Arena award. Rhapsody is endorsed by Embedded Market Forecasters as the tool of choice for C developers, and has received the Embedded Award for Software at Embedded World 2007 for the Rhapsody AUTOSAR Pack.

About BigLever Software Gears

BigLever Software’s industry-leading software product line solution, Gears, addresses the unique challenges of engineering software for a portfolio of similar products. Gears provides an innovative, yet pragmatic, approach that shifts the development focus from a multitude of products to a single software production line capable of automatically producing all the products in a portfolio. Gears can be used in any or all stages of the portfolio development lifecycle, from requirements to development, testing and deployment.
About BigLever

BigLever Software, a leading provider of software product line development tools and services, dramatically simplifies the creation, evolution and maintenance of embedded or standalone software for a product line portfolio. BigLever’s patent-pending solution, Gears, enables software development organizations to reduce development costs and bring new product line features and products to market faster, enabling businesses to more reliably target and hit strategic market windows. BigLever is based in Austin, Texas. For more information, see www.biglever.com or e-mail info@biglever.com.
About Telelogic

Telelogic® is a leading global provider of solutions for automating and supporting best practices across the enterprise—from the powerful modeling of business processes and enterprise architectures to the requirements-driven development of advanced systems and software. Telelogic’s solutions enable organizations to align products, systems, and software development lifecycles with business objectives and customer needs to dramatically improve quality and predictability, while significantly reducing time-to-market and overall costs.

To better enable our customers’ drive towards an automated lifecycle process, Telelogic supports an open architecture and the use of standardized languages. As an industry leader and technology visionary, Telelogic is actively involved in shaping the future of enterprise architecture, application lifecycle management, and customer needs management by participating in industry organizations such as INCOSE, OMG, The Open Group, Eclipse, ETSI, ITU-T, the TeleManagement Forum, and AUTOSAR.

Headquartered in Malmö, Sweden, with U.S. headquarters in Irvine, California, Telelogic has operations in 20 countries worldwide. Customers include Airbus, Alcatel, BAE SYSTEMS, BMW, Boeing, DaimlerChrysler, Deutsche Bank, ELM Messaging, Ericsson, General Electric, General Motors, Lockheed Martin, Motorola, NEC, Philips, Samsung, Siemens, Sprint, Thales, and Vodafone.

For more information, please visit: www.telelogic.com.