Model-Based Engineering: A New Era Based on Papyrus and Open Source Tooling

Francis Bordeleau

Ericsson
8500 Decarie Blvd, Town of Mount Royal, QC, H4P 2N2, Canada
francis.bordeleau@ericsson.com
http://www.ericsson.com

Abstract. Model-Based Engineering (MBE) has proven to be highly successful in many different contexts in large software organizations such as Ericsson over the last decades. However, the broad adoption of MBE has been significantly limited by the fact that existing tools have failed to provide for better customizability and support for Domain-Specific Modeling Language (DSML) and to deliver capabilities to cover for the broad range of development aspects that are considered critical by end-users. Moreover, the lack of evolution of commercial modeling tools in the recent years has led several development units to seriously re-consider the use of modeling tools. We believe that the emergence of Papyrus as an industrial-grade open source modeling UML tool has the potential to be a real game changer and provide the required cornerstone of a new MBE era that will enable collaboration between industry and the research community to develop a complete MBE Integrated Development Environment (IDE) that will provide support for the broad set of capabilities required by the end-users.

Keywords. Model-Based Engineering (MBE), Open Source, Eclipse, Papyrus

1 Introduction

Companies like Ericsson rely on software development technologies and tools to maximize developers' productivity to reduce product development time and cost and get to market faster. The use of leading-edge technologies and tools is key to maximize the capability to innovate and develop key product differentiators.

In this context, Model-Based Engineering (MBE) has proven to be highly successful in many different contexts over the last decades. As a result, MBE, and modeling in general, is now used for a broad range of different aspects of overall product development, including software design, system modeling, architecture modeling, network modeling, information and data modeling at different levels, and business process modeling. MBE is now considered a key component of the overall development process.

However, one of the key issues we are currently facing is that, in spite of the promises, existing commercial modeling tools are much too complex to customize for domain-specific (and user/role) needs, and have failed to deliver capabilities to cover development aspects that are considered critical for end-users. The list of key aspects that are not properly covered by exiting commercial tools is long and includes, among others: model-based testing, model-based tracing & debugging, model validation, model executability, variability modeling and product line management, deployment/allocation analysis and architectural exploration, and team support in general. Moreover, the lack of evolution of the modeling tools in the recent years has led several development units to seriously re-consider the use of modeling tools in spite of the all of the benefits they provide.

So, as a result, we are facing a situation where the modeling tools themselves are the key limiting factor to the broad adoption of MBE. What we need at this point is better tools. Tools that provide for a more complete set of the required capabilities and that can be customized to support different modeling contexts and Domain-Specific Modeling Languages (DSML).

We believe that the only way we can get access to a more complete MBE tool suite (and a growing set of capabilities) is through the use of a "standard/de facto standard" MBE tooling platform and the establishment of a vibrant MBE community, that brings together the industry and the research community to collaborate on the basis of this tooling platform. In this context, the emergence of Papyrus [1] as an industrial-grade open source UML [9] modeling tool provides the potential cornerstone we need for the establishment of a new MBE era. We also need to start thinking as a community and see how we can combine efforts to develop a complete MBE solution instead of investing resources and efforts in developing more or less the same solutions over-and-over again (or competing solutions that are offering more or less the same things in a slightly different manner). We need a real MBE Integrated Development Environment.

In this paper, we will first discuss some of the key lessons learned from the last 25 years of MBE, and then discuss why we believe open source and Papyrus can be used as a basis for the development of a complete industrial-grade MBE tool suite.

2 Main Issues with Commercial Tools: Lessons Learned From the Last 25 Years in MBE

2.1 It is not possible for a single company to develop all of the required MBE capabilities

As stated in the introduction, MBE tools need to provide for a broad range of development aspects. Considering the amount of money that has been invested in the development of modeling tools in the last 25 years by different companies and the current status of the existing tools regarding the support they provide for these aspects, we can safely conclude that no single company can have the expertise to develop the complete set of required capabilities. If no company was able to do it so far, even after all of the investments and acquisitions over the last 15 years, no single company will ever be able to do it.

So, the only hope is to put in place a development context that will allow the MBE community to collaborate on the development of complete MBE tooling solution.

2.2 The use of commercial proprietary tools does not allow end-users to focus on their core business

In the 80's, most companies[1] involved in the development of telecom equipment were developing their own operating systems, programming languages, compilers, debuggers, versioning control systems, and so on. During the 90's, the main argument used by the emerging software development tool companies to convince telecom companies to move away from their own internally developed solutions was that the use of commercial off-the-shelf solutions would allow them to focus on their core business and that they would be able to use their resources to focus on developing key innovations and main product differentiators. A main underlying promise of this business model was that it would ultimately also allow telecom companies to have access to better development tools as the software development tool companies would themselves have more dedicated resources to focus on the development and evolution of a broad range of commercial off-the-shelf products.

We can now state that this model did not deliver on its promises and that it even had negative impact on many different aspects. The fact is that the use of proprietary technologies directly results in vendor lock-in and loss of control on tool evolution and lifecycle. In this context, the capacity to innovate (of telecom companies) becomes directly dependent on the desire and capacity of the commercial tool vendors to develop new required capabilities. In many cases, telecom companies have been powerless in observing declining investments in key existing tools and development of new products to replace existing ones. This has forced them to face the harsh reality of product end-of-life situations and forced tool migrations.

As a result, the use of proprietary development tools has led to a situation where telecom companies have lost control of their own destiny regarding development tools and where they had to dedicate significant resources on trying to resolve undesired tooling issues, as oppose to focus on core business.

2.3 The use of modeling tools that are based on proprietary tooling platforms is a main obstacle to technology transfer and the development of a complete MBE tool suite

We believe that a key factor responsible for the lack of evolution of modeling tools over that last 10 years is the number of existing tools (with very little additional value for end-users) and the fact that these tools are essentially all based on incompatible tooling platforms (which differ at one level or another, even if for example they are based on a same underlying platform such as Eclipse) that prevents being able to use technologies/capabilities developed for one tool with another tool. Also, the proprie-

[1] In this section, we are using the telecom industry as an example, but the reasoning applies to other application domains.

tary and closed nature of most of these tooling platforms forces researchers to develop their own tools to give them the flexibility they need to experiment and develop new capabilities.

So, the current situation is that we have on one side the industrial developers that are using commercial modeling tools that are based on proprietary tooling platforms, and on the other side researchers that are developing new technologies/capabilities based on their own research tooling platforms. The key problem with this situation is that technology transfer becomes almost impossible both for technical reasons (porting technologies from one platform to another is never simple) and for business and legal reasons. Even if we can solve the technical issues, technology transfer quickly becomes too complex for what it's worth when we add the business and legal aspects.

As a result, end-users can not benefit from the innovations done outside of the commercial vendors' direct environment and, in spite of all of the MBE research done based on UML over the last years, covering a broad range of the required development aspects, very little new technologies/capabilities have been made available to the industrial end-users using commercial modeling tools.

We believe that to solve the problem of technology transfer and enable the development of a complete MBE tool suite, we need an open source industrial-grade modeling tool platform that can be shared by both the industry and the research community. This would provide the research community with the open tooling platform they need to conduct research, enable open collaborations between the two communities, and allow much easier (or even seamless) technology transfer.

2.4 Despite UML as an open standard, UML tools are still a main issue

While it is far from being perfect, the establishment of UML as a de facto standard has allowed building a much broader community of users and researchers using the same modeling language and technology base, compared to the situation that existed before, where the ecosystem was composed of a large set of different notations and modeling languages. UML provides a solid foundational semantic layer that is used to develop a broad set of domain specific languages. This foundational layer is also used as a basis for the development of technologies to support the different required system development aspects.

For a company like Ericsson, the establishment of UML as a standard has allowed converging to a common modeling language, focusing development efforts, simplifying the integration of modeling tools with other tools and underlying technologies, and reducing cost associated with training and integration of new hires. Also, while tool migration is still costly (and painful), UML has contributed to greatly simplify it.

So, in the current context, we believe that the main problem is not UML, but the UML tools. As discussed in Section 1, existing modeling tools are too difficult to customize, do not provide sufficient support for DSML, and they have failed to deliver a required tooling capabilities. So, we can conclude that while the existence of a standard modeling language is a necessary condition for the development of a complete MBE tool suite, it is not a sufficient one. To succeed with MBE, we now need to converge on a common open source tool platform/technology to enable real technology innovations.

3 Industrial-Grade Open Source Solution based on Papyrus

3.1 Why Open Source?

The emergence of open source solutions over the last decade has deeply transformed the software industry. Open source solutions have clearly demonstrated their key benefits in many different industrial contexts and are now used at the core of many large industrial products and development processes. From a software development perspective, many open source products are now broadly used to support a wide range of development aspects, including code editing (Eclipse CDT), version control (CVS, SVN, Git), debugging (GDB), tracing (Lttng), and code review (Gerrit). From a tool perspective, Eclipse has become the de-facto standard platform.

For a large software development organization like Ericsson, the emergence of industrial strength open source solutions opens new possibilities as it eliminates vendor lock-in, reduces our dependency towards commercial vendors, allows increasing agility and ability to get required product features and improvements faster, facilitates customization for specific domains, and reduces overall cost.

To achieve the objective of developing a complete MBE Integrated Development Environment (IDE), we believe that open source is a required condition to provide for the following key aspects

- Open technology required to independently develop new capabilities
- Open collaborative environment required to enable fruitful collaborations between different parties
- Open community required to enable contributions from different sources

The use of Eclipse over the last years has allowed making significant progress toward the development of an integrated software development IDE. From a modeling perspective, the establishment of key components like EMF [3] as a foundation for Eclipse modeling has allowed the development of an important set of technologies and capabilities that can be shared/developed/used by different modeling tools. However, this is not sufficient. We need to go one step further and standardize on a UML tooling platform that can be used as a basis for collaboration and development of new MBE capabilities.

3.2 Why Papyrus?

Open source is not new. Many other open source modeling tools [10, 11, 12] have been developed over the years and none of them has had a real impact in the industry. Why would it be different now with Papyrus?

The key difference is that Papyrus is the result of a close collaboration between CEA, which initially started the development of Papyrus in the context of MBE research projects and still lead the project, the industry and the research community.

This collaboration has allowed evolving Papyrus from a research modeling tool to an industrial-grade modeling tool.

In the last years, Ericsson and other industrial partners have worked closely with CEA to improve both the quality of Papyrus, regarding its level of industrialness, and the overall development process and project management around Papyrus. This close collaboration has allowed making very significant progress to reach the required level of stability and functionality now provided by Papyrus 1.0 that was recently released as part the Eclipse Luna release [8]. While improvement is a never-ending story, we can now confidently say that Papyrus has reached a level of maturity that allows for industrial deployment.

Beside the investment made in the development of the core Papyrus modeling tool itself, investments are also made in a number of required aspects, including: team support (Git/EGit [5] integration,, model diff/merge with Eclipse EMF Compare [4], document generation, and model review), Papyrus-RT tooling (which will provide modeling support for UML-RT), development of a UML-RT C++ Runtime and C++ Code Generator, development of advanced customization and DSML capabilities, model-based testing, model-based tracing & debugging, code-centric model-based development (which focuses on integrating modeling environment for C programming and UML), and model validation and architectural conformance. These different aspects are developed in collaboration with different suppliers and researchers in Europe and North America.

3.3 The Key Importance of the Community

To succeed with any open source solution, the community is crucial. We need a strong and vibrant community that brings together the industry (end-users), technology/product suppliers, and research/academia; not three distinct communities, but a single strong and vibrant community that creates synergy to bring value and satisfy the needs of its different members.

- End-users and suppliers can get better access to innovations developed by research/academia through technology transfer
- Research/academia gets input and feedback from end-users and suppliers – there are many aspects for which research/academia have developed potential solutions but can not currently be evaluated in an industrial context because of the IP issues
- Suppliers need to have a viable open source business model to ensure their growth – they obviously depend on the end-users for this purpose
- End-users need a strong community of suppliers to ensure the existence of commercial offerings around the open source solutions
- Research/academia needs partners and funding for research projects

4 Summary

Model-Based Engineering (MBE) has proven to be highly successful in many different contexts over the last decades, but the current generation of modeling tools has been a main limiting factor to its broad adoption.

In this paper, we discussed some of the lessons learned from the last 25 years of MBE. We believe that one of the main issues is that existing tools are based on proprietary tooling platforms and that this prevents proper collaborations between the industry and research/academia and that it makes technology transfer almost impossible.

To succeed with MBE, we need a new generation of modeling tools based on open source technology that will enable collaborations between the industry and research/academia to foster innovations.

We believe that the emergence of Papyrus as an industrial-grade open source modeling UML tool has the potential to be a real game changer and provide the required cornerstone of a new MBE era that will enable the development of a complete MBE Integrated Development Environment (IDE).

References

1. Papyrus UML Modeling tool, http://www.papyrusuml.org
2. Eclipse Foundation, https://www.eclipse.org
3. Eclipse, Eclipse EMF, http://www.eclipse.org/modeling/emf/
4. Eclipse, Eclipse EMF Compare project, http://www.eclipse.org/emf/compare/
5. Eclipse, Eclipse EGit, http://www.eclipse.org/egit
6. Eclipse, Eclipse Papyrus project, http://www.eclipse.org/papyrus/
7. Eclipse, Eclipse Public License (EPL), http://www.eclipse.org/legal/epl-v10.html
8. Eclipse, Eclipse Luna release 1.0, https://www.eclipse.org/luna/
9. OMG, Unified Modeling Language (UML)
10. Modeliosoft, Modelio, http://www.modelio.org
11. MKLab, StarUML, http://staruml.io
12. Grandite, Open ModelSphere, http://www.modelsphere.org