20 Years Xpress Mosel – Software design driven by application needs and technological advances

Susanne Heipcke[1], Yves Colombani[1]

Xpress Optimization, FICO, FICO House, Starley Way, Birmingham B37 7GN, UK
{SusanneHeipcke,YvesColombani}@fico.com

Mots-clés : *mathematical modeling software, optimization, language design.*

1 Introduction

At the occasion of the 20th anniversary of the first commercial publication of Xpress Mosel (2001) this contribution takes the audience through the major phases of its development, stressing as the two main drivers behind its design and evolution user requirements/usage patterns and technological (hardware) advances. Particular focus is directed at the phase preceding the first publication during which the fundamental questions of whether, why, and how to replace an established software (namely Mosel's precursor mp-model [1], commercialised by Dash Associates since 1983) had to be addressed.

2 Phases of software development

2.1 Phase 0 : Inception and design (1997–2001)

Target : match functionality of existing systems ; easy transitioning from precursor standalone command-line tool mp-model without any disruptive changes to user habits whilst switching to an entirely new architecture that provides openings for a large variety of future developments (in-memory data exchange [3], implementation of user modules [4]).

— *First step :* proof of feasibility — reading and executing existing mp-model files on the new architecture (replacing an interpreted/script language written in Fortran and in parts Assembly by parsing and compilation to a virtual machine implemented in C) without any loss of efficiency or functionality (specifically the handling of sparse data structures).

— *Second step :* creating a new language in close collaboration with expert academic and industrial users (most notably the group of Laurence Wolsey at CORE, Univ. Louvain-la-Neuve, and the team of mathematicians at BASF AG, Germany, led by Anna Schreieck).

— *Third step :* development of tools and interfaces : 'mod2mos' converter, library APIs (C, Java, VB), Xpress IVE (Interactive Visual Environment) as development environment on Windows.

— *Fourth step :* finding a name (prototype phase : Mmod2 for 'mp-model version 2' with file extension 'mm2', already using 'bim' for compiled files) ; the name 'Mosel' has no specific meaning and is easy to pronounce ; naming convention 'dso' for Mosel modules.

2.2 Phase 1 : New directions for modeling : debugger and profiler, multi-processor parallelism (2002–2008)

Target : make programming tools available for a modeling language ; exploit newly available multi-processor machines for parallelising (decomposition) algorithms and other computational tasks on the model level.

— Programming tools and functionality : debugger and profiler for the Mosel language ; new programming-style data structures ; packages (libraries written in Mosel).
— Multi-processor parallelism : concurrent model execution within a Mosel instance coordinated by event-based message queues ; also : multiple problems within a model.
— Modular building block : concept of generalized file handling (I/O drivers).
— New solvers and problem formulation paradigms (Nonlinear, Constraint Programming).

2.3 Phase 2 : Support for distributed computing (2008–2017)

Target : anticipate emerging new usage patterns, creating technology to enable working in distributed settings including cloud architectures.
— Mosel Distributed Framework [5] and remote launcher (XPRD) provided the underlying technical framework for Xpress Insight (platform for web-based apps with scenario handling based on Mosel models, first release in 2014).
— The newly introduced remote invocation protocol is used by a new browser-based development environment Xpress Workbench (first release July 2017) that replaces Xpress IVE.
— Support for cloud platforms : aec2, hadoop ; related features : Unicode, internationalisation (message catalog selection based on system language configuration).
— Connectivity : data exchange with/invocation of other languages (R, Java, Python).

2.4 Phase 3 : Advanced programming needs (2017–now)

Target : address programming functionality needs of increasingly large software development projects including connectivity, testing systems, and expectations of developers using mainstream programming langagues ; provide tooling (high-level packages and low level functionality for their implementation) for low-code development of end-user apps.
— End of 2017 Mosel was turned into free software in recognition of its increasing use as general programming language, also opening up the matrix manipulation routines of the Mosel Native Interface (NI) to provide access to alternative LP/MIP/NLP solvers [6].
— New advanced programming features for large-scale projects with multiple contributors : dynamic packages, definition of namespaces, shared data, union types, online doc [7].

Références

[1] R. W. Ashford and R. C. Daniel. LP-MODEL : XPRESS-LP's Model Builder. *IMA Journal of Mathematics in Management* **1**, 163–176, 1987.

[2] Y. Colombani, B. Daniel and S. Heipcke. *Mosel : a Modular Environment for Modeling and Solving Problems.* In : J. Kallrath (ed.) : Modeling Languages in Mathematical Optimization. Kluwer Academic Publishers, Norwell, 211–238, 2004.

[3] T. A. Ciriani, Y. Colombani and S. Heipcke. Embedding optimisation algorithms with Mosel. *4OR*, 1(2), 155–168, 2003.

[4] Y. Colombani, B. Daniel and S. Heipcke. *Mosel : a Modular Environment for Modeling and Solving Problems.* In : J. Kallrath (ed.) : Modeling Languages in Mathematical Optimization. Kluwer Academic Publishers, Norwell, 211–238, 2004.

[5] S. Heipcke. *Xpress-Mosel : Multi-Solver, Multi-Problem, Multi-Model, Multi-Node Modeling and Problem Solving.* In : J. Kallrath (ed.), Algebraic Modeling Systems : Modeling and Solving Real World Optimization Problems. Springer, Heidelberg, 81–114, 2012.

[6] Mosel Open Source repository. https://github.com/fico-xpress/mosel, 2017.

[7] Heipcke, S. and Colombani, Y. *Xpress Mosel : Modeling and Programming Features for Optimization Projects.* In : J. S. Neufeld et al. (eds.), Operations Research Proceedings 2019. Springer, 677–683, 2020.