Is OpenModelica Finally Coming of Age?

Francesco Casella

(francesco.casella@polimi.it)

OpenModelica

Historical Perspective

A Historical Perspective on the OMC Development

- Infancy: 2002-2007
  - A Compiler developer’s playground
  - Nearly impossible to use on practical applications applications

- Childhood: early OSMC years 2008-2013
  - OpenModelica gradually starts being usable for work in selected areas
  - Very strong limitations in terms of coverage, speed, GUI usability

- Adolescence: 2014-2020
  - Serious work possible in some areas, some OSMC members depend on OMC for their daily operation
  - Good coverage, speed and usability in some areas of interest for OSMC
  - Fairly solid performance in most aspects
  - Still falling short in many areas of applications and use cases

➡ What’s next?

Current Status

Coming of Age by Components

- Frontend
- Backend + Codegen
- Runtime
- FMI
- OMEdit
- MetaModelica → Julia
- OSMC

Coming of Age: Frontend

- Success story
- Frontend rewritten from scratch
  - More rational design
  - Using MetaModelica 3.0
  - Delayed scalarization, ready for array-based backend
- Development nearly complete after 4 years
- Much faster than old one (20 X on average)
- Better coverage and performance in nearly all cases
  provide the source code is strictly conforming to the Modelica Specification

- Default choice in OMEdit *and* command line from version 1.17.0

- Extra features coming ahead
  - Conversion scripts
  - Improved checking and diagnostics
  - Support for faster API and GUI performance

Coming of Age: Backend + Codegen

- The current backend has reached the end of useful life
  - Quite sophisticated capabilities
    - State-of-the-art solvers, including sparse ones
    - Tearing
    - Homotopy
    - Advanced index reduction
    - Dynamic state selection
  - Increasingly difficult to develop and maintain due to evolutionary design and development
- New backend rewritten from scratch
  - More rational structure
  - No repeated similar functionality for different phase
  - Can exploit new frontend features
- Existing functionality ported and improved within 1-2 years
- Will allow efficient handling of non-expanded arrays for large models with many instances of the same model
- Full-fledged support of non-expanded arrays will require extra resources than currently planned
- Code generation will need to be updated accordingly

Coming of Age: Runtime

- Several runtimes available in OMC
  - C runtime
  - “old” C++ runtime (kept for backwards compatibility)
  - “new” OMSI C++ (improved design)
  - FMI runtime
  - RT experimental runtimes

- The current C runtime is pretty sophisticated and currently includes bleeding edge developments
  - Dense and sparse algebraic solvers
  - Wide array of dense and sparse ODE solver
  - Dense and sparse IDA solver for DAEs
  - Good performance

- Development and maintenance is very inefficient

- Long-term goal: converge to a single (C++?) runtime.
  Requires additional resources

Coming of Age: FMI (and SSP)

- FMI export has been available for many years in OMC
- In practice, the quality of implementation was not particularly high

- Recent projects allowed the OSMC to focus more on
  - FMU generation
  - FMU use via SSP and OMSimulator

- FMI is a successful technology – OMC can also be successful
  - as convenient way to generate royalty-free FMUs
  - as a convienient way to simulate third-party FMUs

- On-going work to improve the overall quality of implementation of these features in the short term (2021)

Coming of Age: OMEdit

- OMEdit is currently used
  - For professional use (with some limitations)
  - For research
  - For teaching

- Overall perfomance and end-user experience has greatly improved over the last 2-3 years

- Several critical missing features for uncompromised professional use
  - Support for replaceable classes with parameters
  - Faster GUI response
  - Fully functional Duplicate feature
  - Diagnostics of unbalanced models based on Modelica 3.x rules
  - Full support of parameter-driven conditional connectors and dialogs
  - Parameter editing in hierarchically structured models
  - Code refactoring (changing names across opened libraries)
  - Library management support, including conversion scripts
  - Consistent handling of non-parameter modifiers
  - Array parameter input dialogs

How much work was that (2019-2020)?

- Number of tickets fixed, for v.1.14.0 and later: **785**
- About one ticket every day on average by a team of about 8 people
- Note: most of these resources are mainly allocated on funded projects, not on fixing known issues.

The Developers' Dilemma
(a.k.a. the Director's Dilemma)

Development of fancy new features

Development & maintenance of mainstream features

Some Ticket Bookkeeping

- Number of tickets fixed, for v.1.14.0 and later (two years): **785**
- Number of valid tickets opened 2019-2020 (two years): **898** of which **403** were fixed in the same period.
- Number of pending tickets as of 1 Feb 2021: **683**

The situation is under control

Extra resources needed to overcome backlog and do what must be done

Order of magnitude:
10-20 full-time person-year

Coming of Age: MetaModelica → Julia

- The automatic translation of the OMC codebase from MetaModelica into Julia is currently under evaluation (John Tinnerholm's PhD)

- Goals
  - Reduce dependency on an exotic language known and used by a handful of programmers
  - Increase the chances of incoming contributions to the OMC codebase from the OS community
  - Leverage on the progress of the Julia language and OS community
  - Leverage on the Julia mathematical libraries for advanced features (e.g. variable-structure system support)

- Some interesting early results (see later talk)

- Feasibility of MetaModelica → Julia transition will be clearer in 1-2 yrs.

Coming of age: The OSMC

- Current OSMC members number: 52
  - 24 Companies and Institutes
  - 28 Universities

- Several OSMC member companies regularly use OMC in their operation and have long-term staks in the Consortium
  - ABB
  - Bosch Rexroth
  - Dynamica
  - EDF
  - RTE
  - ...
- Sustainable long-term OMC development and maintenance requires to at least double the number of Companies and Institute members in the Consortium

Coming of Age: Advanced Features

- Functional Debugger
  - Step-by-step execution of Modelica Functions
- Declarative Debugger
  - User-friendly visualization of solved equations
  - Solved equations traced back to source code
  - Integration with runtime for simulation-time debugging
- Sparse Solvers and daeMode
  - Increased efficiency in cases with large noninear implicit systems
  - Increased efficiency in cases with large numbers of state variables
- Sensitivity computation and analysis (OMSens)
  - Compute and rank sensitivities of variables along transients
- Custom extensions, e.g. data reconciliation
  - Much lower barrier to developer’s access, compared to commercial tools

The Wish List

The 2018 OSMC User's Survey

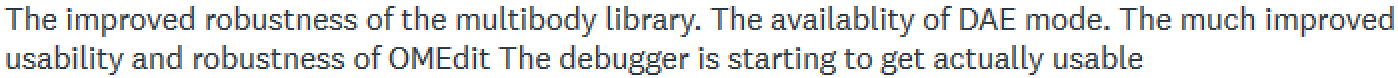
- At the end of 2017, we collected data from OSMC members with a survey on their use of OpenModelica

- The results were presented at the 2018 OpenModelica Workshop

- It is interesting to have a look at some of them in retrospective

Q9: Most Appreciated Improvements in 2017

Improved usability of OMEdit

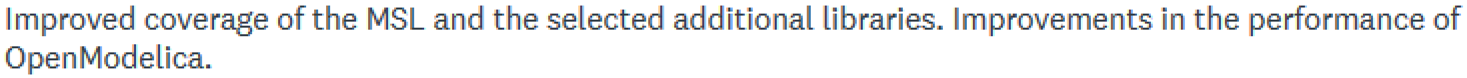

The improved robustness of the multibody library. The availablity of DAE mode. The much improved usability and robustness of OMEdit The debugger is starting to get actually usable

State Machine graph editing

More robustness, stability and faster operation

Line numbers in OMNotebook

Did not notice major changes - I am not a heavy user, of course.

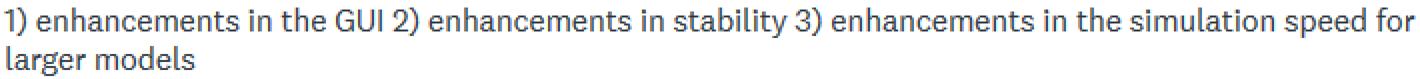

Improved coverage of the MSL and the selected additional libraries. Improvements in the performance of OpenModelica.

Speed. Debugging capabilities

Improvement of: - OpenModelica API - Stability - Performance - Open Modelica Testframework

Backend: Improved handling of nonlinear systems of equations around external functions.

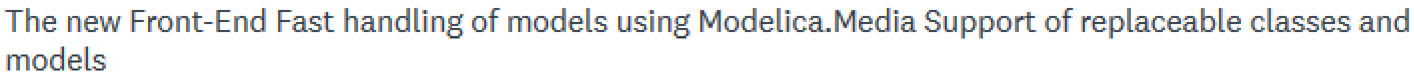
Stability OMEdit


1) enhancements in the GUI 2) enhancements in stability 3) enhancements in the simulation speed for larger models

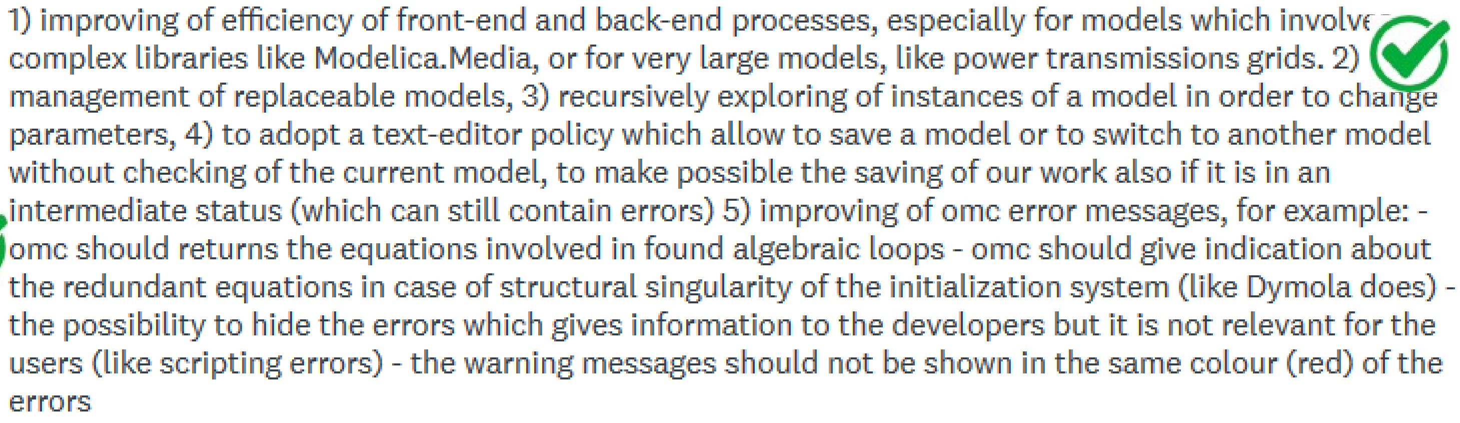
growing coverage of the MSL and other Libraries

Q10: Most Wanted or Missing Features

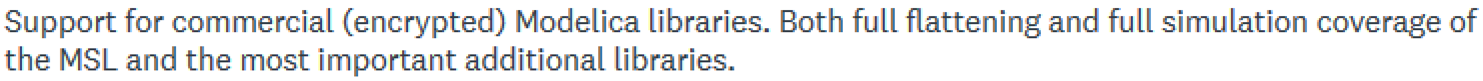
For us we would have needed more focus, rather than trying to solve everything for every sort of use case.


The new Front-End Fast handling of models using Modelica.Media Support of replaceable classes and models

Improve source code generation for embedded targets, specially that state machines will be supported.

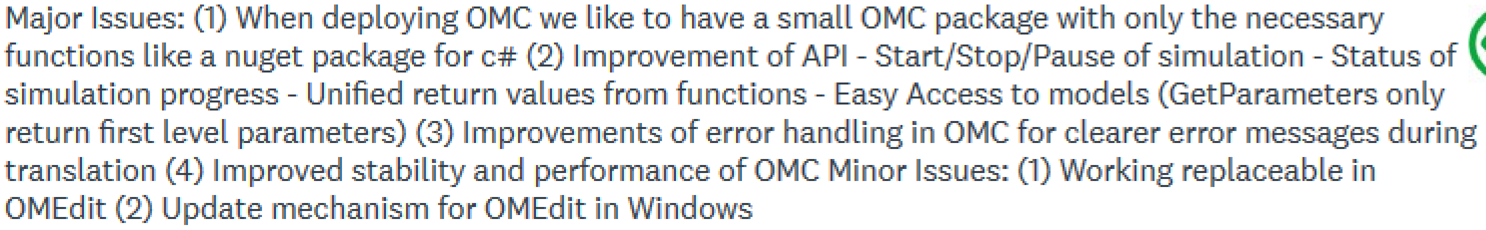

1) improving of efficiency of front-end and back-end processes, especially for models which involve complex libraries like Modelica.Media, or for very large models, like power transmissions grids. 2) management of replaceable models, 3) recursively exploring of instances of a model in order to change parameters, 4) to adopt a text-editor policy which allow to save a model or to switch to another model without checking of the current model, to make possible the saving of our work also if it is in an intermediate status (which can still contain errors) 5) improving of omc error messages, for example: - omc should returns the equations involved in found algebraic loops - omc should give indication about the redundant equations in case of structural singularity of the initialization system (like Dymola does) - the possibility to hide the errors which gives information to the developers but it is not relevant for the users (like scripting errors) - the warning messages should not be shown in the same colour (red) of the errors

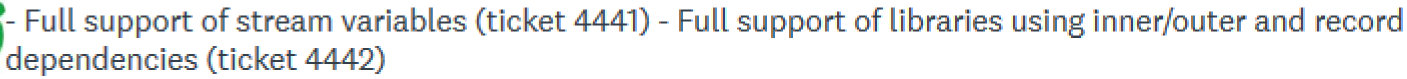
Web-based interface for server deployment; database integration; I/O libraries.


Support for commercial (encrypted) Modelica libraries. Both full flattening and full simulation coverage of the MSL and the most important additional libraries.

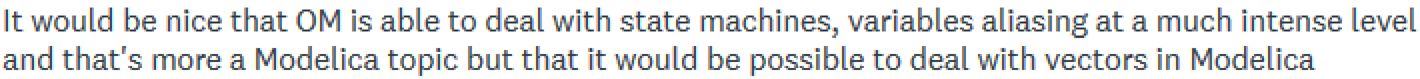
Still problems to run large models with high speed

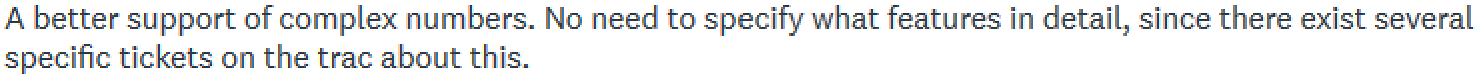
Q10: Most Wanted or Missing Features - cont'd

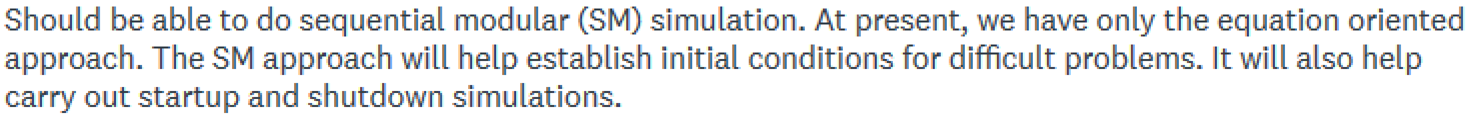

Major Issues: (1) When deploying OMC we like to have a small OMC package with only the necessary functions like a nuget package for c# (2) Improvement of API - Start/Stop/Pause of simulation - Status of simulation progress - Unified return values from functions - Easy Access to models (GetParameters only return first level parameters) (3) Improvements of error handling in OMC for clearer error messages during translation (4) Improved stability and performance of OMC Minor Issues: (1) Working replaceable in OMEdit (2) Update mechanism for OMEdit in Windows ✓

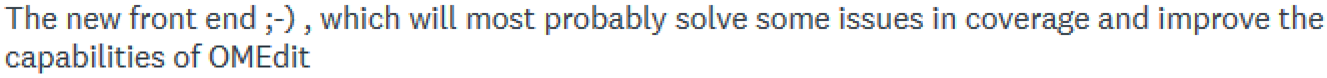
✓ 
- Full support of stream variables (ticket 4441) - Full support of libraries using inner/outer and record dependencies (ticket 4442)

Bug free FMU Export in OMEdit, Replaceable support in OMEdit, Copy&Paste of models from sheet to sheet in OMEdit, Embedded Code Generator

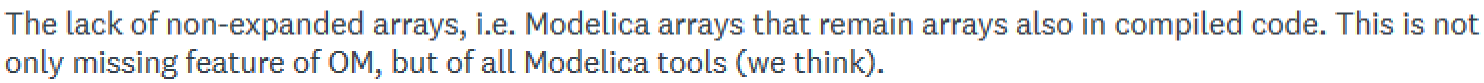

It would be nice that OM is able to deal with state machines, variables aliasing at a much intense level and that's more a Modelica topic but that it would be possible to deal with vectors in Modelica

✓ 
A better support of complex numbers. No need to specify what features in detail, since there exist several specific tickets on the trac about this.


Should be able to do sequential modular (SM) simulation. At present, we have only the equation oriented approach. The SM approach will help establish initial conditions for difficult problems. It will also help carry out startup and shutdown simulations.

✓ 
The new front end ;-) , which will most probably solve some issues in coverage and improve the capabilities of OMEdit

Q10: Most Wanted or Missing Features - cont’d


The lack of non-expanded arrays, i.e. Modelica arrays that remain arrays also in compiled code. This is not only missing feature of OM, but of all Modelica tools (we think).

Proper array/tensor support, i.e., do not expand them.

Better plotting facility Better handling of discrete/continuous equations

Compatibility to the Modelica language standard / covering all standard libraries.

Trust in the overall quality and usability.

faster OMEdit, reliable and performant FMU generation

Who Are Our Users, and Who Could Contribute More?

- User type A
  - Wants a decent Modelica tool, free of charge
  - School and University particularly important to spread the use of Modelica and OpenModelica
- User type B
  - Wants a good open-source Modelica tool, willing to contribute
    - Becoming a member of OSMC and paying the membership fee
    - Providing bug reports with MWE
    - Contributing to the code base
    - Contributing with DFD / MSA contract
- User type C
  - Wants a free Modelica tool for his/her customers/partners/users
  - Possibly willing to contribute to enhance user experience
- User type D
  - Needs some customized extensions
  - Lower entry barrier to developers than commercial tools

Who Are Our Users, and Who Could Contribute More?

Further quality increase required to get more type B, C, D users!

The Digital Twin Opportunity

- What is a Digital Twin?
  (it depends on who you ask…)
  - A repository of product documentation
  - A repository of operation data with AI to operate on it
  - A simulation model (or set of models)
- A very, very big market
- Modelica is a very good candidate to provide an open, standardized way to represent executable models in digital twins
- OpenModelica could provide an open-source engine to run and manipulate them
- Significant investment required to improve speed, robustness, reliability and ease of use

➡ OpenModelica’s chance of a lifetime?

Conclusions

- OpenModelica is 18+ years old
- From CS playground to serious modelling and simulation environment
- Solid background, on-going work on many fronts
- Some parts now coming of age, others still need to grow
- More resources required in the next 2-5 years to reach adulthood

Thank you for your kind attention!