Modelica Evolution – From My Perspective

Hilding Elmqvist
Dassault Systèmes
Ideon Science Park, SE-223 70 Lund, Sweden
Hilding.Elmqvist@3DS.com

Abstract

This paper intends to tell the story of Modelica (www.Modelica.org) from the author's perspective. It is a fantastic saga that for me started in April 1976. The saga includes studying the needs, the original idea, the development of a solution, waiting for mature hardware technology, a start-up company, a fantastic collaboration, an automotive company caring for its software supplier, how to get momentum by standards collaboration, forming the right team, and the magic phone call from the right company.

Keywords: Simnon; Dymola; Modelica; Physical Modeling; DAE; Modelica Association

1 Introduction

In order to describe Modelica evolution it is important to start by describing Dymola, Dynamic Modeling Language, since many of the original features of Modelica are taken from Dymola.

The paper then describes forming the company Dynasim and the important initial research collaborations. The initial Modelica design work and the Modelica community are then discussed. The paper ends with a discussion about Modelica future. Most of the material is presented in chronological order.

For an overview of other simulation tools, see (Åström, et.al., 1998).

I propose that other key contributors to Modelica evolution, Martin Otter, Peter Fritzson and many others, write similar papers from their perspectives in particular since this paper has a focus on the work before Modelica and gives my perspective.

2 Dynamic Modeling Language

When looking for a master thesis project in autumn 1971, the Automatic Control Department, Lund Institute of Technology wanted to develop a simulation program where the user could input the models using mathematical expressions. I was just studying compiler technology, so I considered this project perfect for me. The result was Simnon (SIMulation of NONlinear systems): (Elmqvist, 1975, 1977a). A novel feature was that a mixture of continuous time and discrete time systems could be simulated, i.e. perfect for simulation of sampled data control systems which were becoming very important at that time.

However, I was not satisfied with the approach as a basis for my PhD thesis, since it was based on submodels with inputs and outputs. I therefore wanted to investigate the fundamentals of modeling which my professor Karl Johan Åström supported. In April 1976, I was studying a report with about 200 pages about a drum-boiler model written by Sture Lindahl (Lindahl, 1976). The report was very systematic, showing how the equations were organized for the drum, the super heaters, the turbines, etc., i.e. the approach was object oriented. The modeling started by stating the relationships as general equations. Then the derivations were shown how to put them into state space form in order to be able to simulate using Simnon. These derivations involved manual symbolic manipulation of individual equations to solve for the unknown, solving linear systems of equations (see excerpt from report below), unrolling Newton´s algorithm a fix number of times to solve nonlinear systems of equations, and differentiating certain equations when there were constraints between differentiated variables (nowadays called index reduction).

Just before Easter 1976, I realized what I needed to do: design a new language allowing general equations, having a class concept for object oriented modeling (I was used to Simula so that was natural) and having a structured feature to describe interaction between submodels (called cut). The language was called Dymola for DYnamic MOdeling LAnguage (Elmqvist, 1977a, 1978, 1979a, 1979b).

An example in the Dymola language of an electrical component is shown below. It describes a pin A characterized by an across variable Va and a through variable I. Pin B has correspondingly Vb. The through variable of B is also I but with a preceding minus sign showing that the currents of pins A and B sum to zero. In Dymola, all variables were of real type. The der operator is the same as in Modelica. The path concept was introduced in order to allow operators for series-, parallel- and loop-connections. Such a concept is not used in Modelica since connections are typically drawn graphically.

```
model type capacitor
  cut A (Va / I) B (Vb / -I)
  main cut C [A B]
  main path P <A - B>
  local V
  parameter C
  V = Va-Vb
  C*der(V) = I
end
```

Consider the following electrical circuit:
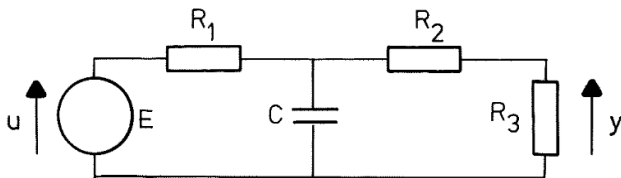
Using a library of "model types": resistor, capacitor, voltage and Common, the circuit could be described as follows;

```
model Network
  submodel(resistor) R1 R2 R3
  submodel(capacitor) C
  submodel(voltage) E
  submodel Common
  input u
  output y
  connect Common to E to R1 to (C par (R2 to R3))to Common
  E.V = u
  y = R3.Va
end
```

The "to" operator means series connection and "par" means parallel connection.

The trouble was then to figure out how to simulate such a model. I had used the symbolic package REDUCE and started to write symbolic manipulations in LISP. But how to know what variable to

solve for in each equation? The naïve approach is: look for equations with only one unknown, mark those as known and repeat. The problem is that there can be mutual dependencies, i.e. algebraic loops for which this approach does not work. And mutual dependencies were normal in the way Lindahl did his modeling, so the language could not be restricted to avoid algebraic loops. I invented my own algorithms using binary incidence matrices. However, these were O(n^3), so they would not scale.

At that time one had to go to the university library to get help in doing a computerized literature search (no Alta Vista or Google being available). I found that graph theoretical algorithms were O(n). The graphs are bipartite, i.e. consists of two types of nodes corresponding to equations and variables. An edge between a variable and an equation means that the variable appears in that equation. A report from the University of Umeå analyzed several algorithms for Assignment or matching which corresponded to finding out what variable to solve for in each equation. Furthermore, Tarjans algorithm would find strongly connected components in a directed graph corresponding to minimal systems of simultaneous equations and sort them into a sequence suitable for solving one system at a time, i.e. the incidence matrix becomes block lower triangular (BLT).

Lindahl sometimes differentiated equations when potential states were constrained. The Jacobian is then always singular. I designed an algorithm to find what equations to differentiate and tested it successfully on some small examples. Fortunately, Costas Pantelides and Sven Erik Mattsson later figured out suitable algorithms, (Pantelides (1988), (Mattsson, Söderlind, 1993).

To test the language ideas and the algorithms, I wrote a program in the object oriented programming language Simula, which is included in my thesis, (Elmqvist, 1978). I tested it successfully on a few examples from different domains: electronics, mechanical, thermodynamics, and electrical power distribution. As an example, the program found one system of equations for solving accelerations and forces of mechanical systems corresponding to the need to invert the mass matrix. The drum boiler model could now be formulated in 15 pages using original equations defined only once for each class and no need for manual symbolic manipulation of the equations. The total model had about 250 nontrivial equations and 11 systems of simultaneous equations.

But there was one big problem: lack of computer memory for symbolic manipulation. I was barely able to translate the drum boiler model on a Univac 1108 computer having only 64 kWords of 36 bits of

memory accessible for the user and the garbage collector had to work hard. So the hardware technology at that time was not mature enough for realistic industrial problems.

After my thesis in May 1978 (Elmqvist, 1978), I spent one year as a post doc at Computer Science Department at Stanford University. I took all the courses in languages and compilers given by John Hennessy (who later became President of Stanford University). Working in a compiler project regarding parametric types, (Hennessy, Elmquist, 1982) was a very useful experience.

I then returned to the Automatic Control department for a project on Languages for Implementation of Control Systems (LICS), (Elmqvist, 1985). In addition to language elements, we also considered the relations to graphics and were able to acquire a mouse. That gave valuable knowhow which later became meta comments in Dymola and adopted in Modelica as graphical annotations. In order to achieve sufficient speed for navigating in hierarchical structures, we developed special raster graphics hardware and a font generation program. The LICS program was developed on a VAX-11 computer and ported to an Apollo DN 600 workstation and demonstrated at the second IEEE Computer Aided Control Systems Design Symposium in 1983 at MIT with high interest since it showed several novel interaction principles such as "information zooming".

During this project, we also translated the Dymola program from Simula to Pascal since there was no Simula compiler on the VAX. This enabled handling larger models due to the linear address space of the VAX. Francois Cellier got a copy of the Pascal version. The importance will be clear later.

The LICS project got an industrial follow-up since I worked for the company SattControl 1984-1992. We developed a distributed control system called SattLine (Elmqvist, 1991a, 1991b, 1992). This experience later influenced the design of hybrid features of Modelica.

3   The Company Dynasim

In August 1991, an important thing happened. I was working in Toronto for two years and attended a seminar at the university by a colleague of Francois Cellier. Francois had moved to Tucson, Arizona and we had lost contact. During our renewed contacts I learnt that Francois was writing a book: Continuous Systems Modeling (Cellier, 1991). I got the manuscript and was overwhelmed! It described Dymola and used Dymola for modeling of many examples.

I also met Peter Fritzson in the summer 1991 at a computer language conference in Toronto and became aware of his work on designing a modeling language, ObjectMath, which combined features of Computer Algebra languages and Object Oriented Programming. I successfully converted some ObjectMath models to Dymola.

These were two of the tipping points that made me resume the work on Dymola since I understood that there was finally an interest in equation based modeling. The third tipping point was the release of Windows 3.0 since the memory barrier of 640 kBytes linear memory was removed.

Cellier and I started to work together and we wrote a paper for a CACE conference in Napa, California (Cellier, Elmqvist, 1992) which later became an article in IEEE Control Systems (Cellier, Elmqvist, 1993). I continued to make development on the Pascal version of Dymola for some time and used the p2c (Pascal to C) translator to facilitate using a C compiler. Sven Erik Mattsson had given me a copy of the article by Costas Pantelides about index reduction, (Pantelides, 1988). I incorporated this in Dymola for the paper.

At the same conference, Sven Erik presented the dummy derivative method including automatic state selection after index reduction based on Pantelides algorithm (Mattsson, 1992).

In Napa, I met Georg Grübel, DLR (Deutsches Zentrum für Luft- und Raumfahrt) and learnt that Cellier would spend the summer 1992 at DLR in Oberpfaffenhofen, Germany. It was decided that Cellier would introduce the Dymola technology at DLR. Martin Otter immediately got hooked and started to implement a library of MultiBody components which was later published in (Otter, et.al., 1996). Francois remembers: "Four weeks into my visit, Martin came to my office on Monday morning, telling me that he now could throw away his program "myRobot" that he had previously developed for the simulation of tree-structured robots. Over the weekend, he had re-implemented his robot model in Dymola, which turned out to be much more elegant and flexible."

I moved back to Sweden and started the company Dynasim in August 1992. DLR realized the potential of the Dymola technology but also the needs to extend the Dymola language and the tool. For example, Dymola did not have **matrix support** then which is essential for MultiBody modeling. Fortunately, DLR could support further development financially which gave Dynasim important initial funding. Martin Otter and I have worked in close collaboration since then. We met for the first time at DLR Oberpfaffenhofen in August 1993 after one year of collaboration via e-mail.

We made a design for **modeling hybrid systems** in order to be able to model power electronics and phenomena such as friction, (Cellier, et.al., 1993) and (Otter, 1994). The concepts of instantaneous equations and event iterations were then introduced.

In July 1993, Dag Brück was employed especially to develop a **graphical user interface** enabling creating components and connecting them by drag and drop. Dag had worked in the Omola project at the Automatic Control Department, Lund.

A missing piece of the technology was to handle that the blocks corresponding to systems of simultaneous equations often got large but sparse. Tearing is a static technique to handle this problem. Our first attempt was to help the translator by introducing hints in the form of a residue operator, (Elmqvist, Otter, 1994). Shortly afterwards, Martin Otter and myself figured out an algorithm to automatically perform tearing, and the algorithm was introduced in Dymola. Dymola would then, for example, for a tree structured multibody model find a large linear system of equations involving accelerations, forces and torques and discover that it could be reduced to inverting a linear matrix of size equal to the number of degrees of freedom.

The efficiency of the code could be enhanced if structural and symbolic processing is made not only on the model equations but also on the discretization equations of the numerical integrator. We called it **inline integration**, (Elmqvist, et.al., 1995). This became very important for hardware-in-the-loop simulation. For example, this technique allows efficient use of implicit methods for multibody models since it combines inverting the mass matrix and solving the non-linear systems of equations.

The **early adopters** of this equation and object based modeling technology can be seen from the Dymola license numbers:

101: Francois Cellier, University of Arizona
102: Martin Otter, DLR
103: Rafael Huber, Dept. Ing. Sistemas, ESAII, Barcelona, Spain
104: Sebastian Dormido, Depto. De Informatica y Automatica, UNED, Spanien
105: Michael Grimsberg, Lund Institute of Technology
106: Dieter Kraft, Dept of Mechanical Engineering, University of München
107: Jochen Seibold, Rechenzentrum University of Stuttgart

Ingrid Bausch-Gall early understood the potential of Dymola and Bausch-Gall GmbH became **distributor** in German speaking countries in September 1995. This was very important for growing the interest in the technology in Germany.

Bengt Jacobson, Machine and Vehicle Design, Chalmers Technical University was among the early adopters and started to use Dymola for powertrain development in a collaborative project with Volvo and Saab in 1995.

Jan Tuszynski at Sydkraft Konsult in Malmö, Sweden deployed Dymola for thermal power systems in 1995.

Sebastian Dormido organized a workshop in Avila, Spain summer 1996. Francois Cellier and I presented the principles of object-oriented equation based modeling.

Akira Ohata and Yutaka Hirano, Toyota Motor Corporation heard about Dymola during a presentation in Japan by Georg Grübel, DLR. They got Dymola licenses in middle of 1996 and started to use it for the **Toyota Prius development**. The multiengineering modeling capability was necessary for the hybrid car development. However, they realized that the maturity of the Dymola software was not sufficient for large scale deployment and provided funding during 2 years starting end of 1997 regarding general GUI enhancements, GUI for Modelica, symbolic solver capabilities, etc. This was an essential funding allowing for expansion of Dynasim. In the end of 1998, Toyota Techno Service Corporation became distributor of Dymola in Japan with Takashi Matsuba and Rui Gao as technical support. In 2007, Rui Gao came to work for Dassault Systèmes.

Martin Otter visited Dynasim for two months in September and October 1996 and gave a lecture series on object oriented modeling of mechanical systems at the Automatic Control department at Lund Institute of Technology. We worked in particular on real-time simulation of hybrid systems such as automatic gearboxes. Martin was the main author of the **Dymola standard library** containing Blocks, Electrical, Rotational, and MultiBody components.

Hans-Jürg Wiessmann and Bernhard Bachmann, ABB, Baden-Dättwil in Switzerland became users of Dymola in 1996 for **power electronics** modeling and HILS (Hardware-In-the-Loop Simulation). Bernhard spent 4 weeks at Dynasim in March 1998. He then proposed a simple state selection algorithm which was used in Dymola before the dummy derivative algorithm of Sven Erik Mattsson was included.

Peter Beater, then at University of Paderborn, started to develop a **Hydraulics library** beginning of 1997. This was later converted to Modelica and sold to Modelon in 2006.

Rüdiger Franke (then at University of Ilmenau) started to use Dymola with partial Modelica support in 1997 for optimization of thermodynamic systems.

I started to completely rewrite the **symbolic engine** of Dymola in July 1997. Until then the old version converted to C from my thesis had been used. I used C++ Standard Template Library successfully on several hundred thousands of equations and Dag Brück implemented counted pointers to handle allocation and de-allocation of memory automatically.

Martin Otter held his first modeling course based on Dymola at Technical University of Munich for students of electrical and mechanical engineering in the year 1997.

Mike Dempsey (then at Rover) started to evaluate Dymola in 1998 for powertrain modeling.

Sven Erik Mattsson was employed in August 1998. He had a perfect background from the Omola project (Andersson, 1994) since he implemented the symbolic engine of OmSim.

Hans Olsson was employed in January 1999 and had a complementing background regarding numeric and GODESS solvers. So now, the **dream team** was formed with Dag, Sven Erik, Hans and me together with collaboration with Martin Otter.

Sven Erik, Dag, Hilding, Hans

Roger Larsson was utilized as a consultant from June 2004 to accelerate building the business side of Dynasim.

4 Modelica Language Design

In spring 1996, I had lunch with Prof Karl Johan Åström and Sven Erik Mattsson from the Automatic Control department, Lund University. Sven Erik was the project leader of the Omola project (Andersson, 1994). We discussed the need for unification of our efforts on Omola and Dymola.

In May 1996, I attended a workshop in Brussels regarding simulation methodologies organized by Hans Vangheluwe. I then, for the first time, met Per Sahlin, Bris Data AB, Stockholm (later renamed to Equa AB) and Alexandre Jeandel, Gas de France, Paris who both were working on equation oriented modeling languages: NMF and Allan respectively.

I then realized that it was time for a **global unified language design initiative**. Modeling requires reuse of stored knowledge, i.e. there must be a standard language. It does not make sense that various tool vendors invent their own language and that a new language is created for every Ph.D. thesis on modeling.

So, I initially phoned Sven Erik Mattsson, Martin Otter, Per Sahlin and Alexander Jeandel to discuss starting a collaborative language design effort. We agreed to meet in Lund on September 2-4, 1996. The participants are seen below, from left to right: Martin Otter, Alexandre Jeandel, Per Sahlin, Sven Erik Mattsson, Bernt Nilsson and me. Dag Brück participated partially and took the photo.

The picture below shows participants of the 75th Modelica design meeting in Lund May 2012.

You might wonder about the name Modelica. The name was invented during one of the lunches of the first meeting. I don't recall who said it first. We quickly checked the usage in Alta Vista. It is used in Spanish for a role model.

Sven Erik Mattsson organized a COSY workshop in Lund the week after with about 20 participants.

Our language design ideas were then presented, i.e. it got some immediate attention. We met Costas Pantelides at this workshop and convinced him to arrange the 3rd design meeting in London. One of the goals was to get in closer contact and cooperate together. Unfortunately, Costas decided later to not join the Modelica development and continued with his own modeling language, gPROMS.

The language design effort initially got a small travel funding from the ESPRIT project "Simulation in Europe Basic Research Working Group (SiE-WG)". It was later organized as the EuroSim Modelica Technical Committee 1 and the Technical Chapter on Modelica within Society for Computer Simulation. I was the chairman for 3 years and then handed over to Martin Otter, the present chairman.

In the year 2000, the non-profit Modelica Association was formed in Linköping, Sweden to formalize the continually evolving Modelica language and the development of the free Modelica Standard Library.

Modelica 1.0 was released after one year design in September 1997. It had classes, inheritance, class parameters (replaceable classes and components) equations, blocks, connectors, functions, etc. I remember an intense debate at the release meeting when Peter Fritzson, who joined the Modelica effort early 1997, argued for including functions and I argued to wait until the next release. Thank you Peter for pushing. Hybrid modeling features were included in version 1.1 released in December 1998.

Dag Brück started to implement a Modelica compiler immediately after the first design meeting. It initially output Dymola code in order that the Modelica models could be simulated. Dymola has been used as a platform for the design group to test all features during Modelica evolution. The first version of **Dymola which officially supported Modelica** was version 4.0 released July 1, 1999. We worked all night until 6.10 A.M in the morning since we wanted it to be released first half of 1999 (in at least some part of the world). The meaning of Dymola then had to be changed to DYnamic MOdeling LAboratory. It also had a conversion option to convert Dymola models and packages to Modelica. Martin Otter was the main author of the **Modelica Standard Library (MSL).** Later, many people contributed to the development of MSL and the current version of MSL has Modelica models from about 30 people.

Michael Tiller, then at Ford, became aware of Dymola (the language) but got really interested in Modelica due to the open specification. He became the first Dymola/Modelica power user in beginning of 1999 and helped us test our Modelica translator. He also wrote the first book about Modelica: Introduction to Physical Modeling with Modelica, (Tiller, 2001). We provided a demo version of Dymola which was shipped with the book. Ford was also interested in converting Adams models, so we made a translator from adm-files to Modelica MultiBody. Andreas Möller (now my son-in-law) developed a supporting Modelica library. To handle these large models, we had to enhance the internal data structures of Dymola. In May 2000, we could simulate an entire vehicle, (Bowles, et.al., 2001).

Sven Erik Mattsson invited Hubertus Tummescheit in 1998 to make a Ph.D. thesis at the Automatic Control department, Lund. Hubertus and Jonas Eborn developed a comprehensive Modelica library for thermo-dynamical systems which broadened the applicability of Modelica. Hubertus organized the first Modelica Workshop in Lund in 2000 with 80 participants. Gerhard Schmitz and his group in Hamburg utilized this library for building a library for air-conditioning modeling. DaimlerChrysler, BMW, Volkswagen and Audi wanted to standardize on a modeling format to allow them to exchange models with their suppliers. During the autumn 2003 there was intense work including benchmarking various tools. However, we needed to form an organization that could supply and support these customers. Hubertus and Jonas then worked at UTRC, USA. I called Hubertus in December 2003 and asked him to come back to Lund and help organizing the effort. He was back before Christmas. This led to founding of Modelon AB by Hubertus Tummescheit, Jonas Eborn, Magnus Gäfvert, Johan Andreasson and me in 2004.

Peter Fritzson made a formal specification of Modelica in 1998 which later led to the open source implementation OpenModelica, (Fritzson, 2002).

Hans Olsson took the responsibility in 2000 of being the editor of the Modelica specification, i.e. to organize the process of resolving all change requests.

Peter Fritzson wrote the book **Principles of Object Oriented Modeling and Simulation with Modelica 2.1** which was published in 2003, (Fritzson, 2003).

5 Dassault Systèmes

Francois Bichet, Dassault Systèmes called in November 2005 and asked if he and Dominique Florack, VP of R&D could visit Dynasim. I made a presentation of the company and the technology and got invited to Paris for presentations to a larger group at Dassault Systèmes and for further discussions. I then got the question if Dynasim was for sale. I considered such a deal as a perfect marriage since I believe systems engineering, modeling and

simulation must be married to 3D and CAD. An intense period of meetings and discussions led to a letter of intent signed before Christmas 2005. The deal was closed on April 2, 2006 and announced at the Dassault Systèmes DevCon Conference in June, 2006.

The first years of joining such a big company were somewhat turbulent for us. Dymola and Modelica represented a new world for some of the managers at Dassault Systèmes HQ and the Dynasim team was too remote from HQ. The situation was corrected in the beginning of 2010 with new manager for CATIA Systems, I was appointed as Chief Technology Officer for Systems reporting to Philippe Laufer, now CEO of CATIA. Dan Henriksson became manager of the Dassault Systèmes Lund team and Martin Malmheden relocated to Dassault Systèmes in Paris to help in the technology transfer. This enabled me to focus on technical aspects and in particular, together with Sven Erik Mattsson, to work on new features such as the synchronous extensions to Modelica and multi core simulation.

6 Modelica Community

The Modelica community is now growing rapidly. There are 9 tools having at least partial support for the Modelica language, see www.Modelica.org/tools. There are more than 1300 models in the Modelica Standard Library. A Modelica compliance test suit with more than 1000 tricky models is available for checking tool compliance. It is planned that a web page with compliance status will be available.

We are now organizing the 10th Modelica Conference. More than 22 Ph.D. theses, 800 papers and articles have been published related to Modelica.

Several large EU projects have been activated: RealSim, EuroSysLib, Modelisar, OpenProd, and Modrio with more than 100 man-years of effort.

The Modelica Association is also growing, now having more than 100 individual members and more than 10 industrial members. People from the synchronous language community have joined after Philippe Laufer suggested I should attend a meeting with them in Paris in October 2010. The synchronous Modelica features were developed with the help of Albert Benveniste, Marc Pouzet, Francois Dupont and others, (Elmqvist, et.al., 2012).

The scope of Modelica Association is also growing to also incorporate the FMI development.

7 Modelica Future

This section discusses a few possible directions for the future of Modelica. In most cases it is a matter of both tool support and various extensions to Modelica.

As mentioned before, systems engineering also need to take **3D aspects** into account, such as mass and inertia calculated from actual shape and materials, collision handling, etc. It is also important to have consistent parameterization of the behavior model and the shapes. It should be possible to initially use a simplified model for concept studies and gradually replace submodels when CAD parts or a kinematic model become available as illustrated below:

Modelica gives the possibility to compose more and more detailed models since model components can be reused. This means that simulation needs to be faster. One possibility is then to use multi-core technology. Recent advances with more than 1000 cores (www.Kalray.com) show the potential. The problem is then how to utilize this enormous processing power in a user friendly way. Partitioning needs to be made automatically. Modelica gives good possibility to automatically partition the model equation execution into separate threads since it is a declarative language based on equations. Early developments are described in (Aronsson, et.al. 2002). A new paper (Elmqvist, Mattsson, 2014) describes a method to **automatically parallelize model equations** implemented in Dymola. A speed-up of 3.4 times has been achieved using 4 cores/8 threads.

Modelica models and simulations is one part of systems engineering. It is important to **enable various tool chains** including sensitivity analysis, optimization, Monte Carlo simulations, linear analysis, code generation, etc. It should be possible to define a process flow that can be reused if some early requirements or designs need to be changed. It is important with standardized interfaces. A subgroup of Modelica Association is working on defining an XML format for various levels of Modelica representation, source, flattened, solved, etc. This will enable various backend tools such as fault tree analysis and code generation. Another subgroup is working on how to store meta information in a standard-

ized way, for example, for optimization (Zimmer, et.al., 2014).

A subtask within the MODRIO project is to enable **modeling of multi-mode and stochastic systems**, for example, to model failing components. A generalization of synchronous state machines to continuous time and that the number of states of a system can change at events is proposed, (Elmqvist, et.al., 2014).

Modelica now has quite limited data structures, i.e. only arrays and records are allowed. It is not possible to define recursive data structures or maps. Such **advanced** d**ata structures** are, for example, needed for advanced controllers, scripting, and meta information for work flows. MetaModelica has successfully incorporated such features already 2005 (Fritzson, 2005).

We in the Modelica Association also want to define more features of Modelica using a **Modelica Kernel language**. This would enable, for example, defining connection semantics, state machine semantics, etc. in the Modelica Kernel language. A prerequisite for this approach is that more advanced data structures are available in the Modelica Kernel language.

Most Modelica tools allow several views of a model or library such as Diagram, Icon, Modelica Text, Documentation, Package tree view. However, **SysML offers several additional useful views** such as inheritance tree, component hierarchy, etc. Such views can be automatically generated from Modelica. For example a Block Definition Diagram (bdd) of SysML for the CoupledClutches example might look as shown below using a Layered Digraph layout algorithm.
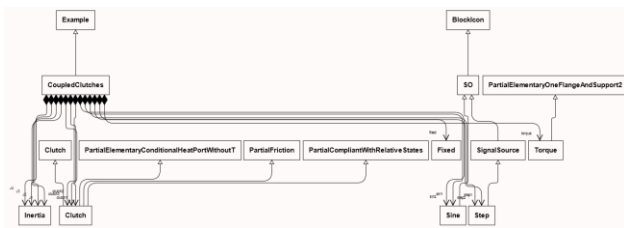
The question about **standardization** of Modelica as ANSI or IEEE standard is frequently raised. The members of Modelica Association want to wait because we feel there are some unifications and enhancements that need to be made first, such as simplification and extension of generics (redeclare) and Modelica Kernel language.

8 Conclusions

The Modelica effort is a truly multi-disciplinary challenge involving compiler technology, graph theory, computer algebra, numerical mathematics, etc. Furthermore, Modelica library development requires skills from many engineering domains such as electronics, mechanics, and thermo-dynamics. The beauty of Modelica is that it enables them all to express themselves in the same language.

An important driving force behind Modelica is to enable storing engineering knowledge in a form that can be reused. This enables more realistic simulations of the complex systems now being designed. It gives the engineers a more interesting work place since tedious manual model development is not needed. This also has the effect that many manual errors are avoided and the simulation results can be better trusted.

Another driving force has been and is the joy of being part of a community with the desire to change the design methodology to become model based. Many hard problems have had to be solved which means that there has been many exciting moments of success.

It seems fair to state that Modelica has already been proven as a means for storing engineering knowhow in a reusable form. The equation orientation means that equations from engineering books can often be reused directly. There is no need to manually transform the equations since the tools do that for you. It is a neutral form which already is supported by many tools. So a good stable platform has been established.

9 Acknowledgements

I am thankful to all persons mentioned in this paper and to many others that enabled this saga to take place.

In particular, I want to thank my thesis advisor and mentor Prof. Karl Johan Åström for guidance and for believing in me that this direction would be possible and useful. He wrote in a certificate about me: "He walks his own ways". It took several years until I understood how positive that was.

Thanks to Sture Lindahl for writing such a well-structured and comprehensive report about power systems modeling which could be formalized.

Francois Cellier was the first user to really understand the principles and power of Dymola, introducing Dymola in his courses in Tucson and in his book. Thank you, Francois.

I want to thank Georg Grübel for helping financing the early developments and for introducing Dymola in Japan.

Without the devoted work by Martin Otter, Modelica would not have succeeded. Thank you, Martin, for your dedication, hard work and fantastic collaboration during 22 years.

Yutaka Hirano and Akira Ohata understood the industrial significance of Dymola but also understood that in order to succeed, financial support was needed which I am very grateful for.

Francois Bichet followed the Modelica evolution for a long time before calling me. Thank you, Francois, for promoting Modelica and Dymola internally in Dassault Systèmes.

I want to thank my manager Philippe Laufer for believing in me and letting me use my own judgment for advancing the Modelica, Dymola and FMI technologies.

Dag Brück, Sven Erik Mattsson and Hans Olsson were absolutely essential for our success. Thank you for the creative, innovative and fun collaboration. I also want to thank all the other past and present members of the Dynasim/Dassault Systèmes Lund team for a great collaboration.

And most importantly, thanks to my wife Iréne Lind Elmqvist who has supported and encouraged me for more than 40 years and for coping with the uncertainties of a start-up company. Thank you, Iréne, for also taking me away from virtual reality to reality in places such as Namibia, South Africa, Tanzania, Botswana, Antarctica, Galapagos, New Zealand, Cook Islands; finding tigers in India and skiing in Åre.

References

(In chronological order.)

Elmqvist H. (1975): **SIMNON— An interactive simulation program for nonlinear systems — User's manual**. *Technical Report TFRT-7502. Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.*

Lindahl S. (1976): **A Non-linear Drum-Boiler Turbine Model**. *Technical Report ISRN LUTFD2/TFRT--3132--SE, Department of Automatic Control, Lund University, Sweden, March 1976.*

Elmqvist H. (1977a): SIMNON - **An interactive simulation program for nonlinear systems**. *Proc. Simulation '77, Montreux, Switzerland.*

Elmqvist H. (1977b): **A New Model Language for Continuous Systems**. *LUTFD2/(TFRT-7132), Department of Automatic Control, Lund Institute of Technology, Sweden.*

Elmqvist, H. (1978): **A Structured Model Language for Large Continuous Systems**. *PhD Thesis ISRN LUTFD2/TFRT--1015--SE, Department of Automatic Control, Lund University, Sweden, May 1978.*

Elmqvist, H. (1979a): **DYMOLA - A Structured Model Language for Large Continuous Systems**. *Proc. Summer Computer Simulation Conference, Toronto, Canada.*

Elmqvist, H. (1979b): **Manipulation of Continuous Models Based on Equations to Assignment Statements**. *Proc. Simulation of Systems '79, IMACS Congress 1979, Sorrento. North Holland Publ. Comp.*

Hennessy, J.L., Elmquist, H. (1982): **The Design and Implementation of Parametric Types in Pascal.** *Software: Practice and Experience 12:169-184, 1982.*

Elmqvist H. (1985): **LICS - Language for Implementation of Control Systems**. *Technical Report ISRN LUTFD2/TFRT--3179--SE, Department of Automatic Control, Lund University, Sweden, December 1985.*

Pantelides C. (1988): **The Consistent Initialization of Differential-Algebraic Systems,** *SIAM J. Sci. Stat. Comput., 9(2), pp. 213-231.*

Cellier F. E. (1991): **Continuous System Modeling**. *Springer-Verlag, New York, USA.*

Elmqvist H. (1991a): **Cooperating Distributed Control objects.** *IFAC Symposium on Distributed Intelligence Systems, August 13-15, 1991, Arlington, Virginia, USA.*

Elmqvist H. (1991b): **A Uniform Architecture For Distributed Automation.** *ISA/91 International Conference & Exhibition, Anaheim, USA, October 28-31, Instrument Society of America.*

Elmqvist H. (1992): **An Object and Data-Flow based Visual Language for Process Control**. *ISA/92-Canada Conference & Exhibit, Toronto, Canada, Instrument Society of America.*

Cellier, F.E., and H. Elmqvist (1992), **The Need for Automated Formula Manipulation in Object-Oriented Continuous-System Modeling,** Proc. *CACSD'92, IEEE Computer-Aided Control System Design Conference*, Napa, CA, pp.1-8.

Mattsson S.E. and G. Söderlind (1992): **A New Technique for Solving High-index Differential-algebraic Equations**. Proc. *CACSD'92, IEEE*

Computer-Aided Control System Design Conference, Napa, CA, pp. 218-224.

Cellier, F.E., and H. Elmqvist (1993), **Automated Formula Manipulation Supports Object-Oriented Continuous-System Modeling,** *IEEE Control Systems*, **13**(2), pp.28-38.

Elmqvist, H., F.E. Cellier, and M. Otter (1993), **Object-Oriented Modeling of Hybrid Systems,** *Proc. ESS'93, SCS European Simulation Symposium*, Delft, The Netherlands, pp.xxxi-xli.

Mattsson, S.E. and G. Söderlind (1993): **Index reduction in differential-algebraic equations using dummy derivatives**. *SIAM Journal of Scientific and Statistical Computing, Vol. 14 pp. 677-692, 1993.*

Otter, M. (1994): **Object-Oriented Modeling of Drive Trains with Friction, Clutches and Brakes**, *Proceedings of the European Simulation Multiconference, ESM'94, Barcelona, Spain, June 1-3, 1994, pp. 335-339. SCS, The Society for Computer Simulation.*

Elmqvist, H., and M. Otter (1994): **Methods for Tearing Systems of Equations in Object Oriented Modeling**, *Proc. ESM'94, European Simulation Multiconference, Barcelona, Spain, June 1-3, 1994, pp. 326-332.*

Andersson M. (1994): **Object-Oriented Modeling and Simulation of Hybrid Systems**. *PhD thesis ISRN LUTFD2/TFRT--1043--SE, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.*

Elmqvist H., M. Otter, and F.E. Cellier (1995), **Inline Integration: A New Mixed Symbolic/Numeric Approach for Solving Differential-Algebraic Equation Systems,** *Proc. ESM'95, SCS European Simulation MultiConference*, Prague, Czech Republic, pp.xxiii-xxxiv.

Otter, M., H. Elmqvist, and F.E. Cellier (1996), **Modeling of Multibody Systems with the Object-Oriented Modeling Language Dymola**, *J. Nonlinear Dynamics*, **9**(1), pp.91-112.

Åström K.J., Elmqvist H., Mattsson S.E.: **Evolution of Continuous-Time Modeling and Simulation**, *The 12th European Simulation Multiconference, ESM'98, June 16-19, 1998, Manchester, UK.*

Tiller M. (2001): **Introduction to Physical Modeling with Modelica**, *The Springer International Series in Engineering and Computer Science.*

Bowles P., M. Tiller, H. Elmqvist, D. Brück, S.E. Mattsson, A. Möller, H. Olsson, M. Otter (2001): **Feasibility of Detailed Vehicle Modeling**, *SAE 2001 World Congress.*

Fritzson P., P. Aronsson, P. Bunus, V. Engelson, H. Johansson, A. Karström, and L. Saldamli. (2002): **The Open Source Modelica Project**. In *Proceedings of the 2nd International Modelica Conference*, Oberpfaffenhofen, Germany, Mar. 18–19, 2002.

Aronsson P., and P. Fritzson (2002): **Multiprocessor Scheduling of Simulation Code from Modelica Models**. In *Proceedings of the 2nd International Modelica Conference*, Oberpfaffenhofen, Germany, Mar. 18–19, 2002.

Fritzson, P. (2003): **Principles of Object Oriented Modeling and Simulation with Modelica 2.1**, *940 pages, ISBN 0-471-471631, Wiley-IEEE Press. Sept. 2003.*

Fritzson P., A. Pop, and P. Aronsson (2005): **Towards Comprehensive Meta-Modeling and Meta-Programming Capabilities in Modelica.** In *Proceedings of the 4th International Modelica Conference,* Hamburg, Germany, March 7-8, 2005.

Elmqvist H., Otter M., and Mattsson S.E. (2012): **Fundamentals of Synchronous Control in Modelica**. *Proceedings of 9th International Modelica Conference, Munich, Germany, September 3-5.*

Zimmer D., Otter M., Elmqvist H., and Kurzbach G. (2014): **Custom Annotations: Handling Meta-Information in Modelica**. *Proceedings of 10th International Modelica Conference, Lund, Sweden, March 10-12.*

Elmqvist H., and Mattsson S.E. (2014): **Parallel Model Execution on Many Cores**. *Proceedings of 10th International Modelica Conference, Lund, Sweden, March 10-12.*

Elmqvist H., Otter M., and Mattsson S.E. (2014): **Modelica extensions for Multi-Mode DAE Systems**. *Proceedings of 10th International Modelica Conference, Lund, Sweden, March 10-12.*