

# Layout In Domain-Specific Visual Modelling

Denis Dubé

August 23, 2004

Domain-specific visual modelling enables working directly with domain concepts at a high level of abstraction. Meta-modelling specifies the syntax of a domain-specific modelling formalism explicitly (in the form of a model). A meta-modelling tool thus allows domain experts to build a meta-model and synthesize a domain-specific modelling environment from it. One such tool, AToM<sup>3</sup> (A Tool for Multi-formalism Meta-Modelling), developed by the Modelling, Simulation and Design Lab, was the basis of this summer's work on visual layout.

Visually building meta-models for formalisms with a visual syntax necessitates the ability to quickly and easily specify not only the syntactic relationships between various entities, but also the graphical representation to assign to the entities and to the relationships between them. Since the previous build of AToM<sup>3</sup> was neither quick nor easy, the Icon-Editor originally created by NSERC student François Plamondon, with expanded capabilities, was fully integrated in AToM<sup>3</sup>.

The visual modelling part of AToM<sup>3</sup> has both static and dynamic layout needs.

Static layout involves finding an aesthetically pleasing and intuitive layout given an unchanging model. To this end an extensive review of the literature and existing tools was done. One such tool, the publicly available yED, provides solid implementations of the most important layout algorithms for static models. Thus, the ability to export AToM<sup>3</sup> models to several common graph representation languages as well as the ability to import the language recognized by yED was added. As external tools can never be seamlessly integrated, a few important layout algorithms were adapted and directly implemented in AToM<sup>3</sup>. The most important of these are a spring-based physical system simulation layout algorithm and a force transfer based algorithm.

The spring-based approach considers every pair of nodes with an edge between them to be connected by an ideal physical spring with a given rest length. Moreover, each node is considered to have an electrical charge and thus exerts an electrostatic force on all its nearby neighbours. This force is limited only by a simulated friction force.

To satisfy the dynamic layout needs, the force transfer algorithm was used. The force transfer based algorithm provides a means of adding new nodes, or a cluster of nodes, into an existing model, by moving all overlapping nodes (and optionally, edge control points as well) enough that no overlapping occurs.

A plethora of small but useful improvements were made to AToM<sup>3</sup>. These include a snap grid system for producing cleanly aligned models quickly and easily, an interactive edge manipulation system that makes creating and editing edges quick and precise, and various useful capabilities including: scaling, text scaling, zooming, label dragging, copy&paste, and undo&redo.

A fundamental contribution lies in the fact that the behaviour of the graphical user interface (GUI) which implements the above myriad of capabilities, was explicitly modelled, in AToM<sup>3</sup> in the Statechart formalism and automatically converted into code (using DCharts/SCC developed in Thomas Feng's M.Sc. thesis).

A significant refactoring of AToM<sup>3</sup>'s entire graphical user interface, enabling faster and easier manipulation of visual models, including their layout, was accomplished this summer.

The dynamic aspects of visual layout, in particular, the modelling and resolving of graphical constraints in a meta-model, were not satisfactorily solved however and must be addressed in future work.