Tenth Int. Workshop on Generalized Nets
Sofia, 5 December 2009, 70-75

GN IDE – A Software Tool for Simulation with Generalized Nets

Dimitar G. Dimitrov
Centre of Biomedical Engineering, Bulgarian Academy of Sciences
105 Acad. G. Bonchev Str., 1113 Sofia, Bulgaria
e-mail: *mitex@gbg.bg*

Abstract: Generalized Nets Integrated Development Environment (GN IDE) is a software tool, developed as a client to the GN simulation server, GNTicker. The product allows users to load and save XGN files, run simulations and edit GN models. GN IDE is written in Java, hence, it is platform independent. It connects to a simulation core via the GNTP protocol. In this paper, certain features of GN IDE are described, along with technical information and information about future work.

1 Introduction

Generalized nets are an instrument for modelling and optimization of parallel and competitive processes in complex systems, as well as solving problems where other tools like flowcharts, Petri Nets, systems of differential equations, etc., turn to be inapplicable or inefficient.

Many generalized net models in different areas are created during the years. A complete software tool for modeling with GNs will practically demonstrate the advantages of this instrumentation, notwithstanding the theoretical evidences for these benefits, given with purely mathematical means.

Over the years several software simulators for generalized nets were developed (see [3, 4, 5]). The first attempt dates back to 1987–1988. In 2003, GNTicker was released, featuring many advantages over the previous tools. It represents a GN simulation server without a graphical user interface that communicates via a GNTP protocol with a client. GN IDE is developed to be user-friendly integrated modeling and simulation environment, which can be used as a client to GNTicker and any other GNTP-compatible simulation core.

It is expected that the reader is familiar with GN concepts (see [1, 2]).

The remainder of this paper is organized as follows. In Section 2 technical details of the software product are described briefly. In next sections the features of GN IDE's graphical user interface are described, supported with technical information. Finally, conclusion remarks about current and future work are given**.**

2 Technical Details

The major stipulated architectural requirement was the platform independence of the product. This allows GN modeling to be performed in a heterogeneous environment. Java programming language was chosen, so that the software can run on any platform with Java

Runtime Environment (JRE) installed (see [10]). JRE is freely available for many platforms. GN IDE can be run from a browser too, using the Java Web Start technology.

Another main goal pursued in GN IDE's development has been the extensibility of the end product. New requirements arise such as support for new GN operators. Also new GN tools have been written and they can be integrated into the modeling environment.

GN IDE is intended for release as an open source application, which eases its extensibility and customizability.

3 Loading Models from Files

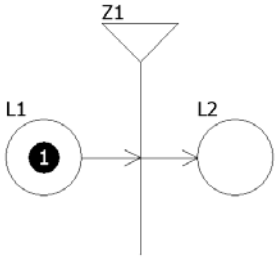
While previous GN software tools used various non-standard file formats, GN IDE fully supports the new XGN file format (see [4]). XGN defines a standard for storing and interchanging GN models. The file format is XML-based, which has several advantages. It provides platform independence. There are many software tools for parsing, transforming and manipulating XML content. Also XML is easily extensible.

XGN files should contain visual information, such as coordinates of places, transitions and arcs, in order to be displayed visually. Various visual details such as sizes, metrics, colours, etc., can also be specified, giving the user full control of GN models' appearance. That visual information is ignored by GNTicker, but it represents an essential part of the visual modeling.

GN models with no visual information can be handled, as well. An external tool such as Gnvis (see [3]) can be used to infer graphical structure by a given XGN file, which contains only structural information.

Current version of the software supports parsing of function and predicate definitions written in GNTCFL (GNTicker Characteristic Function Language, see [4]). Future versions will also support JavaScript functions. General GN users are more familiar with JavaScript syntax. Also JavaScript provides more powerful constructs than GNTCFL (see [4]). XGN does not impose any format for function and predicate definitions. They are specific for the interpreter that is used for simulation. GN IDE can easily be extended to support any specific function language.

A very simple GN model and its corresponding XGN file are shown on Figure 1.




```
<?xml version="1.0" ?>
<gn xmlns="http://www.clbme.bas.bg/GN" name="MinimalGN" time="256">
  <transitions>
    <transition id="Z1" positionX="100" positionY="100" sizeY="100">
      <inputs>
        <input ref="L1">
          <arc>
            <point positionX="50" positionY="150"/>
            <point positionX="100" positionY="150"/>
```

```
            </arc>
          </input>
        </inputs>
        <outputs>
          <output ref="L2">
            <arc>
              <point positionX="100" positionY="150"/>
              <point positionX="150" positionY="150"/>
            </arc>
          </output>
        </outputs>
        <predicates>
          <predicate input="L1" output="L2">true</predicate>
        </predicates>
      </transition>
    </transitions>
    <places>
      <place id="L1" positionX="50" positionY="150"/>
      <place id="L2" positionX="150" positionY="150"/>
    </places>
    <tokens>
      <token id="alpha1" host="L1">
        <char name="Default" type="double" history="1">1</char>
      </token>
    </tokens>
    <functions/>
  </gn>
```

Figure 1: sample GN model and its corresponding XGN file

4 Displaying GN Models

GN IDE offers several views for a given GN model. Each of them has different purposes. No single view itself can display the whole information about the GN model, but the main window is a combination of several views. GN IDE has single document interface.

4.1 Graphical view

This is the most important view. It displays the graph of all transitions, places, tokens and arcs. Each place or transition is labeled with its identifier. Tokens are displayed in their host places. When GN IDE is not in simulation mode, tokens are displayed in the places where they will enter during simulation. If a token has a color characteristic, it is colored accordingly to its value. The default characteristic, if defined, is displayed too. If there are more than four tokens in a single place, only their count is displayed.

The graphical view looks similar to Gennete's document windows, so Gennete users will easily migrate to GN IDE (see [5]).

4.2 Tree view

The tree view displays GN structural data in hierarchical structure. The root node represents a generalized net. The children of the root are transitions, the children of the transitions are places (a node for the input places and a node for the output places), the children of the places are tokens, the children of the tokens are characteristic names, the children of the characteristic names are values from the history. There are other nodes too – for functions, predicate and capacity matrices, etc.

This view is not available in Gennete. It is very useful for complex GN models with many input and output places, many tokens in a place, etc. Nodes of the tree can be expanded or collapsed, allowing the user to display only specific information.

Users with programming skills are familiar with such views – for example, Eclipse's Outline view (see [11]).

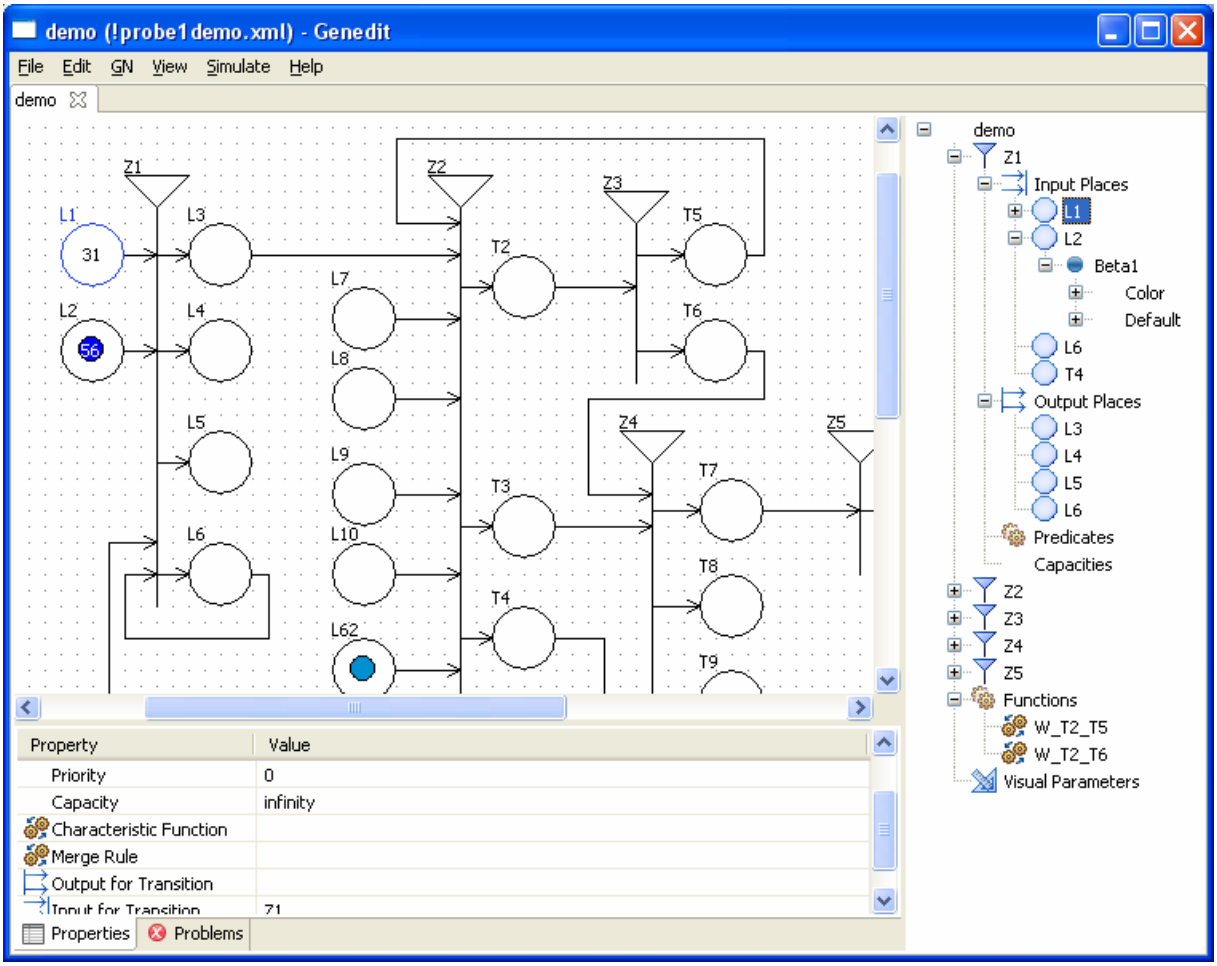


Figure 2: GN IDE main window

4.3 Properties view

For a single selected element, this view displays all element's properties in a two-column table. Capacity and predicate matrices of transitions can also be shown by this view.

When clicking on a property, the user can edit its value. GN IDE provides easy-to-use editor controls, specific for the type of the property. For example, for characteristic functions, a combo box is opened, which lists all characteristic functions in the model, plus the option of creating a new function.

4.4 Function view

It allows users to define and edit characteristic functions and predicates. Each function is opened in a new tab in the main window.

This view is in its initial state and many new features are currently in development. For example, snippets will be available to help both beginner and advanced users to write standard functions and predicates.

5 Simulation

The most important purpose of GN IDE is to help users play simulations of GN models they created. The actual simulation is performed by an external interpreter. By default GNTicker is used, but any GNTP 0.1-compliant simulation core can be used instead (see [4]). It can be installed on the same machine as the graphic client, but also can be running on a remote machine. In both cases a TCP/IP connection is established between the client and the interpreter.

GN IDE provides controls for:

- starting a simulation: attempts to connect a running simulation core, then removes all tokens from the model (they will enter at some step of the execution of the model).
- performing a single step: queries the simulation server to perform a step and visualizes the result of this operation (for example, token entrance / movement / leaving, along with changes in characteristic values).
- autoplay the simulation ("Go"): performs all steps, with an optional delay after each one.
- pause/resume a running simulation.
- stopping the simulation: disconnects from the server and restores all tokens.

During simulation, the user can observe all parameters of the model, using the already described views (graphical, tree and properties). A very useful feature is the visual tracing of characteristic values. GN IDE can display real-time line charts for given characteristics. When the simulation is paused, the user can switch back to previous states of the model.

Another useful feature, which is currently in development, is the ability to record a simulation so it can be played again later.

6 Editing GN Models

Currently the GN Lite package provides Gennete for creating and editing GN models. Its last version should be able to export models with visual information to XGN files, which can later be imported in GN IDE and used for simulation. However, the future goal of GN IDE is to be complete tool for GN modeling.

GN IDE supports saving models to XGN file format. Some of the editing capabilities were already described – editing properties and functions.

GN IDE, like Gennete, has an error checker for GN models. It can be invoked manually at any time, but it is performed automatically before a simulation is started. Validating against the XML schema for GNs helps users to find problems such as missing inputs for a transition. GN IDE performs also checks that are outside of the possibilities of XML schemas.

7 Conclusion

GN IDE is essential part of the GN modeling software. The release of this preliminary version makes possible the mass usage of GNTicker. User documentation will be available online at http://www.ifigenia.org and also included in the software package for offline use. The author's ambition is to make GN IDE a complete fully functional integrated modeling and simulation environment for generalized nets, which also provides a software framework for building new GN tools.

References

[1] Atanassov K. Generalized nets. World Scientific, Singapore, New Jersey, London, 1991.

[2] Atanassov K. On generalized nets theory. "Prof. Marin Drinov" Academic Publishing House, Sofia, 2007

[3] Trifonov T., K. Georgiev, K. Atanassov. Software for modelling with Generalised Nets. *Issues in intuitionistic fuzzy sets and generalized nets*, Vol. 6, 2008, 36-42

[4] Trifonov T and K. Georgiev. GNTicker – A software tool for efficient interpretation of generalized net models. *Issues in Intuitionistic Fuzzy Sets and Generalized Nets*, Vol. 3. Warsaw, 2005.

[5] Aladjov H., N. Nikolov, P. Georgiev, K. Atanassov. Software for generalized nets. *Annual of Technical University*, Sofia, Vol. 50, No. 3, 1999, 125-132 (in Bulgarian).

[6] Atanassov K., Janev K., Atanassova L., Theory of Generalized nets (a programming aspect). *Proc. of II International Symp. "Automation and Scientific Instrumentation"*, Varna, May 1983, 397-399.

[7] Atanassov K., Christov R. Program Package for Generalized Nets (PPGN), *Petri Net Newsletter* 42, Aug. 1992, 23-26.

[8] Atanassov K., Christov R., About generalized nets and their program realization, *Automation and Informatics*, Vol. 5/6, 1993, 28 (in Bulgarian).

[9] Atanassov K., Christov R., About the program realization of the generalized nets, *Advances in Modelling & Analysis*, AMSE Press, Vol. 17, No. 1, 1993, 13-24.

[10] Kramer D. et al, The Java™ Platform. A white paper. 1996, from http://java.sun.com/docs/white/platform/javaplatformTOC.doc.html

[11] Eclipse documentation, http://www.eclipse.org/documentation/