A Plugin System for Charlie

Jan-Thierry Wegener¹  Martin Schwarick²  Monika Heiner²

¹ Otto-von-Guericke-Universität
   Institute of Biology
   Universitätsplatz 2, 39106 Magdeburg, Germany

² Brandenburg University of Technology
   Department of Computer Science
   Postbox 10 13 44, 03013 Cottbus, Germany

CS&P 2011, Pułtusk, Poland
Outline

1. Introduction
2. Charlie’s Main Features
3. Requirements on Charlie’s Plugin System
4. Charlie’s Plugin System
5. Conclusion
Charlie is a tool for analyzing Petri nets that

- is written in Java;
- has an intuitive and easy to use GUI;
- is able to analyze several properties;
- is able to conclude further results from results (rule system).
Charlie

Charlie is a tool for analyzing Petri nets that
- is written in Java;
- has an intuitive and easy to use GUI;
- is able to analyze several properties;
- is able to conclude further results from results (rule system).

Charlie is not a tool for
- drawing Petri nets;
- simulating Petri nets.
Charlie is not a tool for
- drawing Petri nets;
- simulating Petri nets.
Charlie

Charlie is not a tool for
- drawing Petri nets;
- simulating Petri nets.

Charlie can read following file-formats:
- Snoopy;
- INA;
- APNN.
Introduction

History of Charlie

2003  last version of INA
2006  first version of Charlie (Master Thesis by Martin Schwarick)
2009  added reachability graph for time-dependent Petri nets (Diploma Thesis by Ansgar Fischer)
2009  new GUI and redesign of Charlie’s code (Diploma Thesis by Andreas Franszke)
2011  integration of a plugin system (by Jan-Thierry Wegener)
INA and Charlie

The net has a nonempty clean trap.
The net has no transitions without pre-place.
The net has no transitions without post-place.
The net is connected.

ORD HOM NBM PUR CSV SCF CON SC Ft0 tf0 Fp θ pF0 MG SM FC EFC ES
Y Y Y N N Y N N Y Y N N N N N N N
DTP SMC SMD SMA CPI CTI B SB REV DST BSt DTr DCF L LV L6S

Analysis menu:
Decide structural boundedness........................................B
Non-reachability test of a partial marking using the state equation........N
Compute the symmetries of the net....................................Y
INA and Charlie

The net has a nonempty clean trap.
The net has no transitions without pre-place.
The net has no transitions without post-place.
The net is connected.

The net has no transitions without pre-place.
The net has no transitions without post-place.
The net is connected.

The net has no transitions without pre-place.
The net has no transitions without post-place.
The net is connected.

Analysis menu:
Decide structural boundedness .................................. B
Non-reachability test of a partial marking using the state equation ....... N
Compute the symmetries of the net .................................. Y
Main Features

The main feature of Charlie are:

- open files: Snoopy, INA and APNN file-format;
- analyzers:
  - incidence matrix based analysis;
  - siphon/trap computation;
  - reachability/coverability graph;
  - model checking (CTL/LTL);
  - path search;
  - visualization of properties;
- rule system.
Charlie’s Rule System

Rule $\supseteq$ Theorem: set of results (pre-conditions) and another set of results (post-conditions)

Advantage: retrieving results without heavy computations, e.g., CPI $\Rightarrow$ SB $\Rightarrow$ k-B
Charlie’s Rule System

Rule $\triangleq$ Theorem: set of results (pre-conditions) and another set of results (post-conditions)

Advantage: retrieving results without heavy computations, e.g., CPI $\Rightarrow$ SB $\Rightarrow$ k-B
Charlie’s Rule System

Rule ≜ Theorem: set of results (pre-conditions) and another set of results (post-conditions)

Advantage: retrieving results without heavy computations, e.g., CPI => SB => k-B
Charlie’s Rule System

Rule $\triangleright$ Theorem: set of results (pre-conditions) and another set of results (post-conditions)

Advantage: retrieving results without heavy computations, e.g., CPI $\Rightarrow$ SB $\Rightarrow$ k-B
Charlie’s Rule System

Rule $\triangleq$ Theorem: set of results (pre-conditions) and another set of results (post-conditions)

Advantage: rules are shown to user $\Rightarrow$ can be used in teaching
Advantages of a Plugin System (or Why do we need a Plugin System?)

Charlie has a good list of analyzers and rules.
Advantages of a Plugin System (or Why do we need a Plugin System?)

Charlie has a good list of analyzers and rules. However, since Charlie is (currently) closed software writing new analyzers and deploying a new version

- takes some time;
- maybe is not wanted to be included by Charlie’s authors;
- usability suffers from too many unneeded options.
Advantages of a Plugin System (or Why do we need a Plugin System?)

Charlie has a good list of analyzers and rules. However, since Charlie is (currently) closed software writing new analyzers and deploying a new version takes some time;
- maybe is not wanted to be included by Charlie’s authors;
- usability suffers from too many unneeded options.

With a plugin system, users can
- easily extend a program by writing a plugin or by loading a plugin;
- implement algorithm that they think are useful;
- activate the features they want to have activated.
Advantages of a Plugin System (or Why do we need a Plugin System?)

Developers of a tool profit from a plugin system by:
- standardizing several features;
- implementing a good framework;
- force themselves to follow a design.
Advantages of a Plugin System (or Why do we need a Plugin System?)

Developers of a tool profit from a plugin system by:

- standardizing several features;
- implementing a good framework;
- force themselves to follow a design.

Therefore

- it is easier for developers, new to the project, to get started;
- overall productivity increases.
We stated the following requirements to a plugin system:

1. possibility of accessing Charlie’s core, e.g., starting analyzers, evaluating results, adding information to the log file, . . .;
2. basic Petri net framework;
3. handle external libraries in plugin file;
4. possibility of sharing code between plugins;
5. dependency check;
6. start and stop plugins during runtime.
Basic Requirements

We stated the following requirements to a plugin system:

1. possibility of accessing Charlie’s core, e.g., starting analyzers, evaluating results, adding information to the log file, . . . ;
2. basic Petri net framework;
3. handle external libraries in plugin file;
4. possibility of sharing code between plugins;
5. dependency check (Future Work);
6. start and stop plugins during runtime (Future Work).
With a plugin one can extend the following features of Charlie:

- the set of analyzers;
- the set of rules;
- the GUI;
- the set of supported file formats.
With a plugin one can extend the following features of Charlie:

- the set of analyzers;
- the set of rules;
- the GUI;
- the set of supported file formats.

Plugins do not need to provide any of the features above, e.g., if a plugin shall share common code only.
Charlie’s Plugin System

With a plugin one can extend the following features of Charlie:

- the set of analyzers;
- the set of rules;
- the GUI;
- the set of supported file formats.

Plugins do not need to provide any of the features above, e.g., if a plugin shall share common code only.

Note: there is no dependency check yet.
Plugin: GUI & Analyzer
Plugin: GUI & Analyzer

- marking editor
- IM-based analysis
- siphon/trap computation
- reachability/coverability graph
- model checking
- path search
- net properties

- marking editor
- IM-based analysis
- siphon/trap computation
- reachability/coverability graph
- model checking
- path search
- conflict graph
- net properties
Plugin: Rule & Net Properties

Hint: Rules are easily implemented ⇒ quick check for contradictions
Plugin: Rule & Net Properties

Hint: Rules are easily implemented ⇒ quick check for contradictions

Wegener, Schwarick, Heiner (OVGU, BTU)
Plugin: Rule & Net Properties

Hint: Rules are easily implemented ⇒ quick check for contradictions
Plugin: Readers
Plugin: Readers

![Screen shot of Charlie's Plugin System]

- **File**
  - Open...
  - Reload
  - Load Session...
  - Save Session...
  - Recent Files
  - Exit
- **Show**
  - Working Editor
- **Preferences**
  - Model Checking
  - Path Search
  - Net Properties
- **Help**
  - Strg-O
  - Strg-R
  - Strg-L
  - Strg-S
  - Strg-Q

**Import** button highlighted in red.
Conclusion

We have seen that Charlie’s plugin system:
- are useful for users;
- increases usability;
- can extend important features;
- are powerful due to code sharing;
- can use external libraries.

Future Work:
- dependency check;
- start/stop plugin during runtime.
Charlie’s Website

Download
Charlie can be downloaded free of charge at

http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Charlie

Bug Report
Bugs can be reported at

http://www-dssz.informatik.tu-cottbus.de/DSSZ/Bugs/CharlieBugList

There is also a link on Charlie’s website.
Thank You