# ESUG'04 Innovation Technology Awards Submission: StarBrowser

Roel Wuyts
roel.wuyts@ulb.ac.be

## Description

The goal of the Star Browser is to allow you to browse your Smalltalk environment and classify anything you encounter while doing so. Therefore it has full drag&drop support to allow you to classify software elements such as classes and methods while you are browsing.

As researched and described in Koen De Hondt's dissertation this allows you to document your system in a flexible way. For example, while browsing a new piece of code you might store some classes and methods that look interesting in a classification, that you can then later use to remember these places. Therefore you can drag&drop the items in an extentional classification. Or, if you know better what you are looking for, you can describe this in the Smalltalk block that defines an intentional classification. Then the items will be calculated by the block. For example, you might want to add all classes from certain applications or all classes starting with a certain prefix. Besides a description with a Smalltalk block, we can also add descriptions using Soul, a reflective logic programming language. Then you can for example calculate a classification consisting of all the participants in a certain composite pattern. You can keep the intentional classification as it is, or convert it to an extentional classification. This makes it easier to remove or add singular items to it. If you keep it as an intentional classification you can recalculate it, for example to bring it up to date with a changed implementation.

Currently, the Star Browser has the following noteworthy functionality (see also the screen shots on the website):

- The classifications are on the left of the browser, in a Tree Widget. Classifications can contain anything (any object, see the screenshot for an example where we add a singleton instance to a classification for easy access). Note however that no items can be added to intentional classifications; therefore their definition has to be changed Extentional classifications can be used to drag&drop any item in.
- The children of an item are also defined by a service. By default only classifications have children (as defined by the ItemChildren service). When you

use the ItemChildrenExtended service, classes get as children their protocols (that have as children their methods). That gives a more Dolphin-like browser.

- Icons indicate the different kinds of items. This is defined by the current #icon service in the configuration menu.
- The text displayed for the item is defined by the #label service. This makes it for example really easy to add a specific labeler that shows classes belonging to your applications or parcels in a specific color, or all abstract classes in red, or shows the number of methods of a class etc.etc.etc
- When you click on an item in the tree view, an editor for that item is displayed. For example, clicking an object opens an inspector on that object. Clicking on a class or a method opens a Refactoring browser. Which editor is used for an item is defined by the #editor service in the configuration.In the screenshots we show some different examples of editors (using the refactoring browser, or using the Advance2 UML editor, etc.).
- The items in the Tree View have a context sensitive menu, as defined by the #menu service.
- Which visitor is used for which action can be chosen at any time in the Services menu. That lists all the registered services. For example, we registered several #editor actions. Which set is used can just be selected by clicking the appropriate visitor in the Configuration menu. The next item will be opened with the editor defined by the current edit service.
- Using the button with the light-blue background the Classifications view can quickly be removed, focussing only on the currently selected editor. Clicking the button once more brings back the classifications view where it was.
- Classifications can be imported and exported in binary format so that you can transfer classifications between images. Items that cannot be loaded are replaces with placeholders as needed.

## Screenshots

To illustrate we provided two scaled down screenshots: one showing the integration with external editors like the Advance UML tool (Figure 1) and one showing the Popular Classes intentional classification (Figure 2).

Much more screenshots, describing a small usage scenario, can be found on the StarBrowser website.

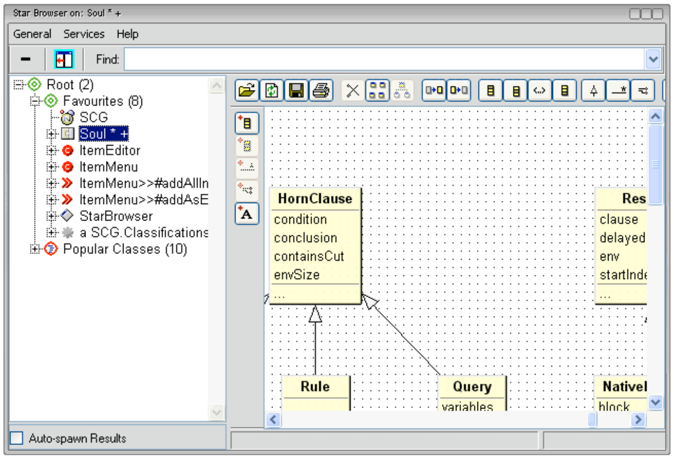
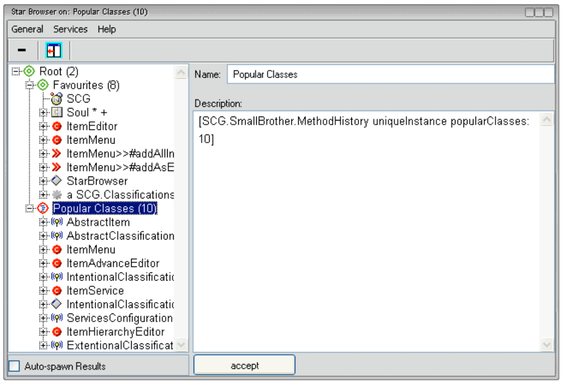


Figure 1: StarBrowser that uses the Advance UML plug-in to show a bundle.



Name: Popular Classes

Description:
[SCG.SmallBrother.MethodHistory uniqueInstance popularClasses: 10]

Figure 2: Editing the definiton of the "Popular Classes" classification.

# Administrative Information

- **Keywords.** Development environments, browsers.
- **Platforms.** VisualWorks, and a spin-off for Squeak.
- **License.** Free. Beer or whiskey donations accepted at conferences.
- **Developers.** Roel Wuyts, Université de Bruxelles, Belgium.
- **URL.** http://homepages.ulb.ac.be/~rowuyts/StarBrowser/index.html