Google Summer of Code 2008 Idea

Roxygen

Roxygen is an extension of the R language and provides a documentation and development assistance system. The documentation system is in the style of Doxygen or Javadoc, and the development assitance system is (somehow) like a pre-processor with its directives.

A Roxygen block is marked as

```
#'
```

and consists of one ore more lines. The blocks are processed by Roclets (in the style of Doclets), which are small R-programs, specifying the content and format of the output.

An outline

A roxygen block must precede a class, method, generic function or function declaration. It is made up of two parts: a description followed by block tags.

An exemplar S4 class with a generic function and the corresponding method:

```
#' This class represents a person.
#'
#' @slot fullname The full name of the person
#' @slot birthyear The year of birth
#' @prototype Prototype person is named John Doe
#'     and born in the year 1971}
setClass('Person',
        representation = representation(
                fullname='character',
                birthyear='numeric'),
        prototype = prototype(
                fullname='John Doe',
                birthyear='1971'))
```

```
#' The naming of an object.
#'
#' @param object A object which gets a name
setGeneric('name', function(object, ...){[...]})

#' Name a person, the baptism.
#'
#' @param object A Person object
#' @param ... Not used
#' @export
setMethod('name', signature('Person'),
function(object, ...) {
        ...
})
```

The class description says the purpose of the class, and the tags describe slots and the prototype object. The generic and method arguments are described by `param` tags. The standard Documentation-Roclet generates `Rd` files out of these information.

The `export` tag in the method Roxygen block is a development assistance tag; it defines that this method is exported. Another tag is the `import` tag, which defines the package, where a generic function or class is imported from (`@import Package`). A Namespace-Roclet handles these information and creates the package namespace file.

A Collate-Roclet handles multiply source files. At the beginning of every file, the `@include` tag describes which other source files must be visible to this one. The order of the source files is calculated based on that tags, and the collate field of the package description file is set.

Project attributes

Sophisticated project; student needs to know `R` very well, especially the structural elements like functions, S3/4 classes, generics, ..., and the student needs some basic knowledge about writing a parser. At best the student has worked with Doxygen, Javadoc or another documentation system.

The project goals and milestones, respectivly, are

1. Define a list of tags which are meaningfull in conjunction with `R`.

2. Develop code to parse `R` source files: `R` code and Roxygen block comments.

3. Define the behavior of a Roclet; write the documentation Roclet which produces `Rd`-files.