

Development of a R Graphic User Interface using Eclipse RCP platform

Víctor Llorens Vilella
Héctor López Sacanell
Juanjo Pardo Invernón

Escola Politècnica Superior - Universitat de Lleida

Setembre de 2006

Summary

- 1 Source and technology
- 2 Core
- 3 Data manager
- 4 Package manager
- 5 R code editor
- 6 Demo
- 7 Conclusions and To Do

Introduction

- R
- Eclipse RCP platform
- EclipseR

Introduction to R

What is R?

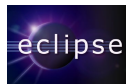
- Statistical analysis application and graphic engine
- Own code language programming: R
- Is GPL: you can see, improve, share and execute it without constraints
- Extensible: add new functionalities



Introduction to Eclipse RCP

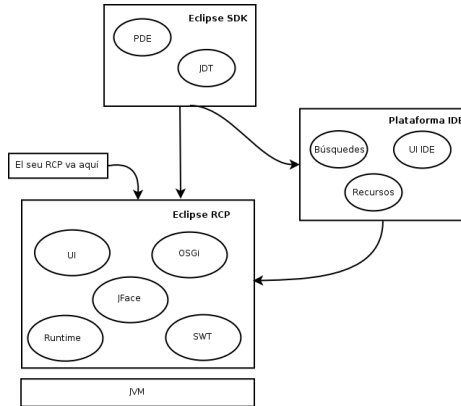
What is Eclipse RCP?

- Eclipse starting point: IBM splits functionalities
- Define a minimal application architecture
- Architecture based on plug-ins
- Scalability from Objects to Functionalities
- Multiplatform application



Eclipse technologies

- SWT
- JFace
- UI
- OSGi
- Runtime



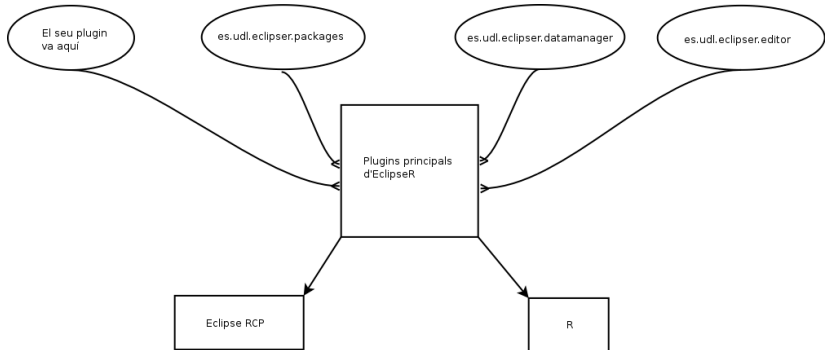
Introduction to EclipseR

What is EclipseR?

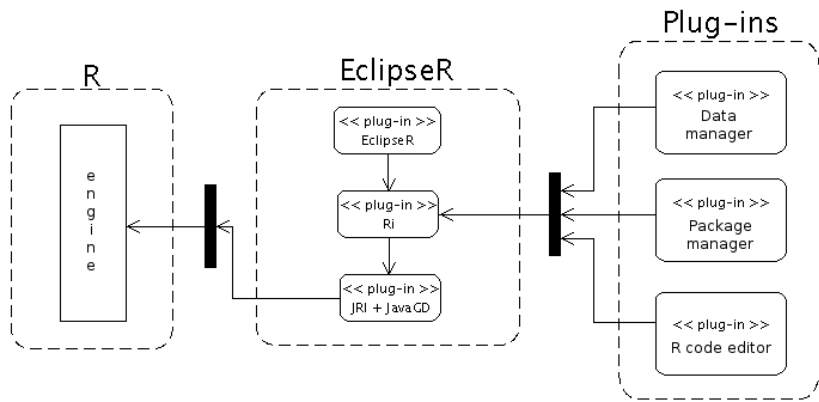
- Eclipse RCP + R = EclipseR
- One graphical user interface for R
- Is GPL: you can see, improve, share and execute it without constraints
- Add functionalities to R

Introduction

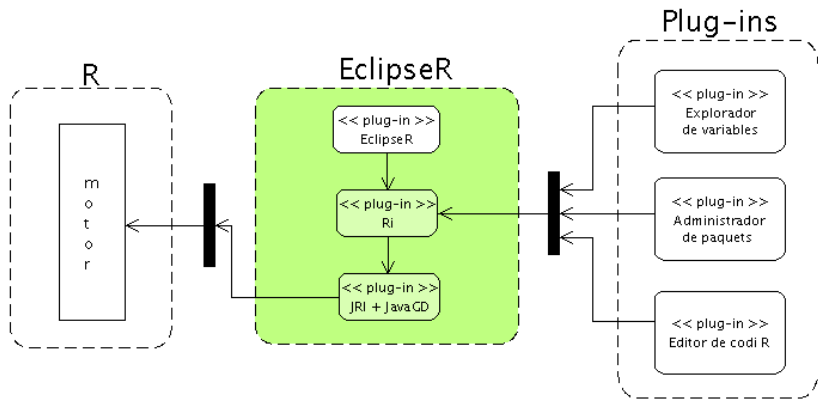
EclipseR architecture



Vista general



EclipseR: the application as a plug-in



EclipseR: The plug-in

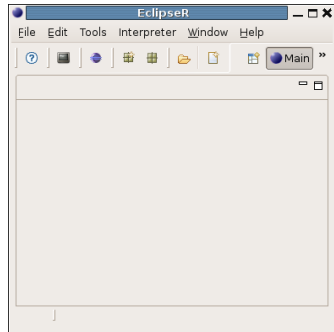
The EclipseR plug-in defines all basic elements to be a RCP application.

- Graphic user interface
- Multiplatform file system access
- Application customization
- I18n
- Help system
- ...

Main application aspects

Inherit elements from Eclipse platform

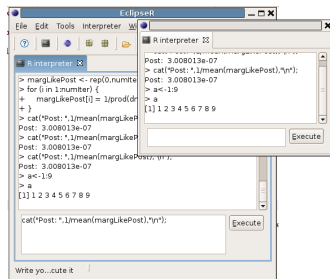
- Main window
- Tool bars
- Menu bars
- State bar
- Perspective bar



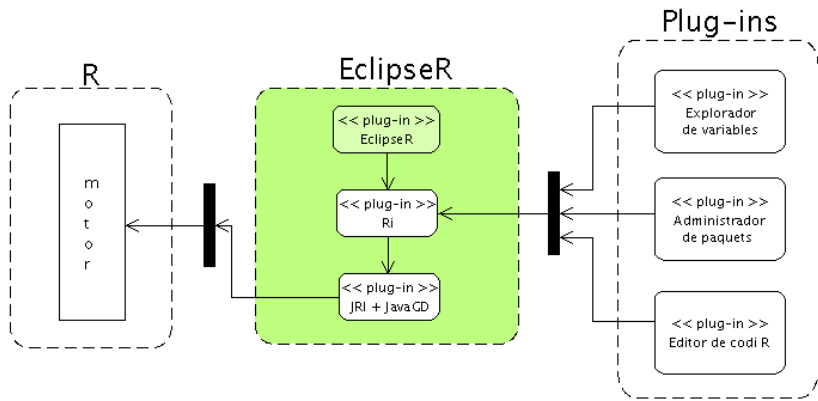
R console

We can:

- Send R-commands to R engine and show results
- Have more than one console in the same session



Engine and graphic system



Ri+JRI plug-ins

Ri plug-in

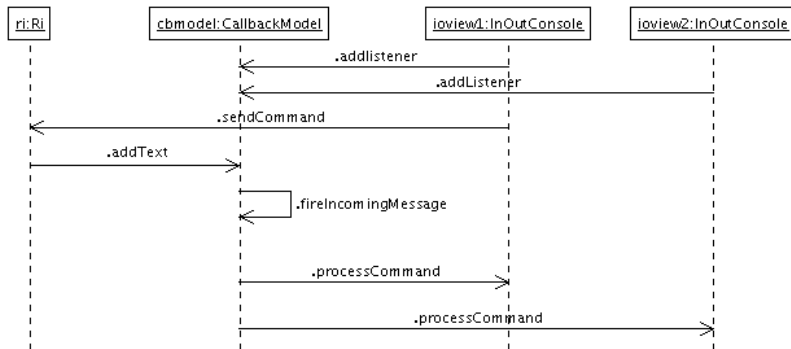
- Centralize information
- Define a communication model *plug – ins* \longleftrightarrow *JRI*
- Implements main functionalities to work with packages

JRI plug-in

- Communication bridge unidirectional with R (java -> R)
- Encapsulating org.rosuda.JRI package
- SO dependent libraries
- Graphic system

Sequence chart for a single R command

Tracking a single R command



Introduction

- Introduction to R graphic system
- JavaGD device
- Integrating JavaGD in EclipseR

R graphic system

Before everything

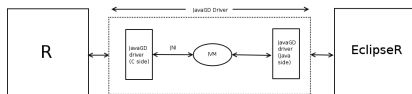
- What is a device in computer science?
- Graphic device is the same than a graphic format
- Devices for R: Ins



The JavaGD

What is?

- A R graphic device using Java
- Provided by Rosuda (<http://rosuda.org>)
- GPL: free to use
- Java AWT graphic technology



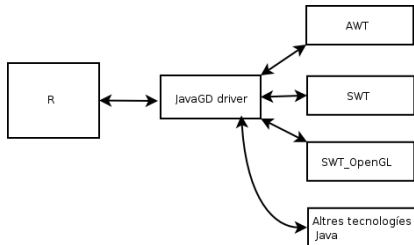
Integrating JavaGD in EclipseR (I)

Why we reuse it?

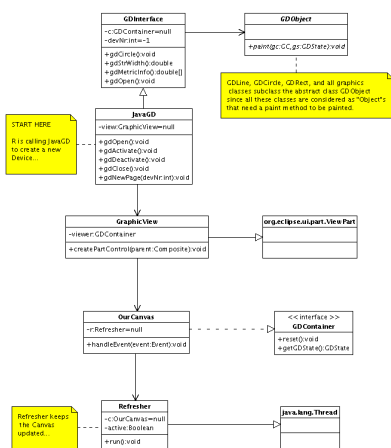
- Both are developed in Java
- Why reinvent the wheel?
- Scalability
- No changes in future R released

Problem:

- We have a strong dependency with Rosuda team



Integrating JavaGD in EclipseR (II)

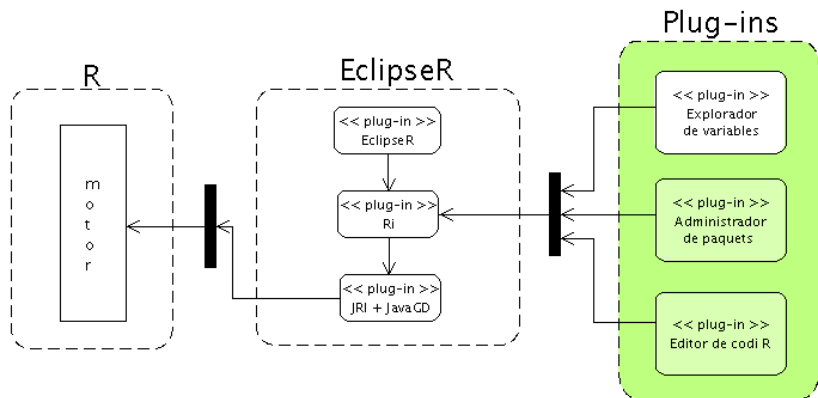


The screenshot displays the EclipseR IDE interface with the following components:

- Dispositiu2Actiu3:** A grid plot with 'connect\$season' on the y-axis (1, 2, 3) and 'connect\$group' on the x-axis (A, B, D, E, F). It shows a grid of colored cells (red, blue, white).
- Dispositiu3:** A scatter plot with 'Index' on the x-axis (0.6, 0.8, 1.0, 1.2, 1.4) and 'a' on the y-axis (30, 35, 40, 45, 50, 55, 60). It shows a single data point at (1.0, 45).
- Dispositiu4:** A line plot titled 'presentacio' with 'Index' on the x-axis (1, 2, 3, 4, 5, 6, 7) and 'a' on the y-axis (0, 20, 40, 60, 80, 90). It shows a line connecting points at approximately (1, 45), (2, 55), (3, 35), (4, 55), (5, 80), (6, 95), and (7, 5).
- Consola R:** An R console window showing the following code:

```
V V  
V V plot(tmp, matrix=FALSE, lines=TRUE, col=c("green", "blue"),  
+ pointsArg=list(pch=c(15, 19), cex=2), linesArg=list(lwd=2))  
V V V V  
V V V V plot(tmp, scale=FALSE, lines=TRUE, linesSet=1,  
+ linesArg=list(col="black", lwd=2))  
V V V V  
V V V V plot(tmp, set=2, col=c("gray"),  
+ plotArg=list(xlab="Group", ylab="Season"))
```

Plug-ins: Data manager



Why?

Reasons to use a data manager:

- Move away not-friendly R interface for beginners
- Simplify hard processes
- Usability
- Integration

Contributions

- Data manager
 - Show all R variables sorted
- Complex data editor
 - To show variables value
 - To edit variables value
- Easy to add new functionalities

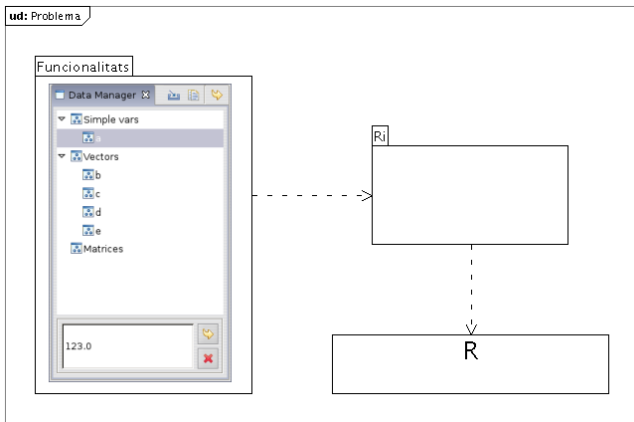
Problem

We need R to evaluate variables.

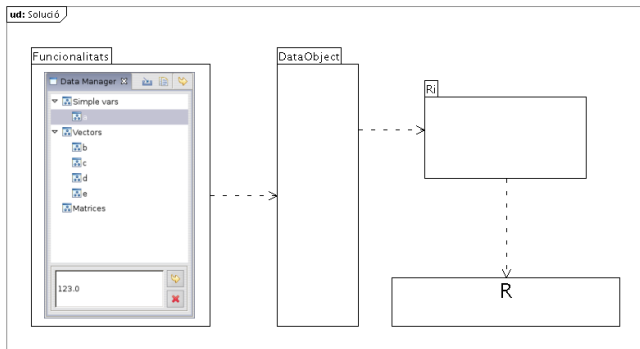
It implies:

- Repeat code
- NOT Java object oriented programming
- Code has to be changed when new R versions will release

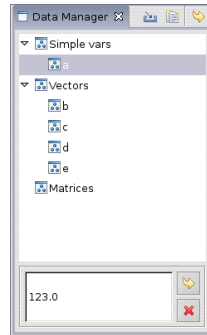
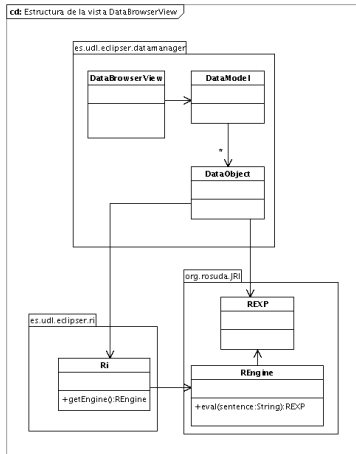
Problem



Solution



Data manager

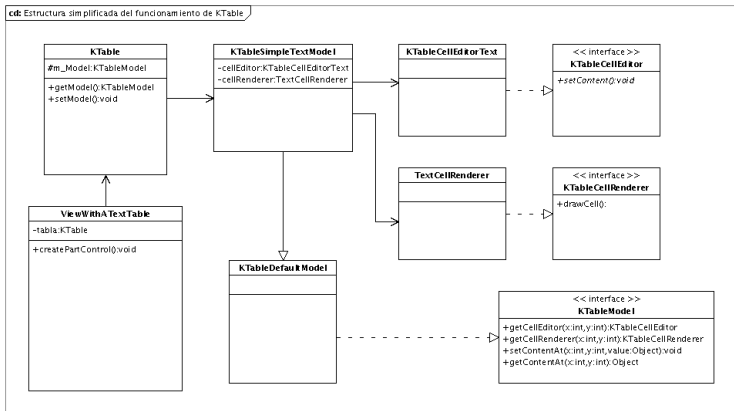


Complex data editor

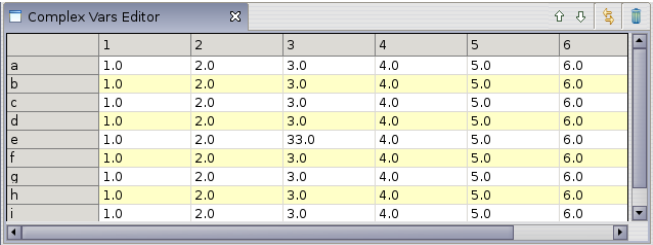
- Previous version
 - SWT table
 - JFace table
 - SWT modified table
- KTable
- One view for two models

KTable

cd: Estructura simplificada del funcionamiento de KTable



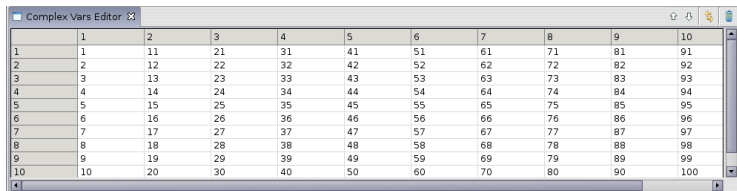
Vector edition



The screenshot shows the 'Complex Vars Editor' window in Eclipse. It contains a table with 10 rows (labeled a through j) and 6 columns (labeled 1 through 6). The data in the table is as follows:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|-----|-----|------|-----|-----|-----|
| a | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 |
| b | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 |
| c | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 |
| d | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 |
| e | 1.0 | 2.0 | 33.0 | 4.0 | 5.0 | 6.0 |
| f | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 |
| q | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 |
| h | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 |
| i | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 |

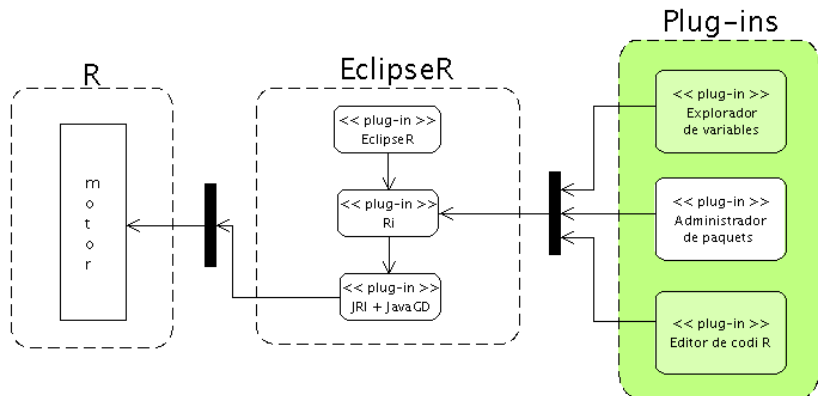
Matrix edition



The screenshot shows the Eclipse IDE's 'Complex Vars Editor' window. The window title is 'Complex Vars Editor'. The editor displays a 10x10 matrix of numbers. The numbers in the matrix are arranged in a grid where each cell contains a value that is the sum of its row and column indices (e.g., row 1, column 1 is 1; row 10, column 10 is 100). The window has a standard toolbar with icons for home, refresh, and search, and a scroll bar on the right side.

| | | | | | | | | | | |
|----|----|----|----|----|----|----|----|----|----|-----|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 1 | 11 | 21 | 31 | 41 | 51 | 61 | 71 | 81 | 91 |
| 2 | 2 | 12 | 22 | 32 | 42 | 52 | 62 | 72 | 82 | 92 |
| 3 | 3 | 13 | 23 | 33 | 43 | 53 | 63 | 73 | 83 | 93 |
| 4 | 4 | 14 | 24 | 34 | 44 | 54 | 64 | 74 | 84 | 94 |
| 5 | 5 | 15 | 25 | 35 | 45 | 55 | 65 | 75 | 85 | 95 |
| 6 | 6 | 16 | 26 | 36 | 46 | 56 | 66 | 76 | 86 | 96 |
| 7 | 7 | 17 | 27 | 37 | 47 | 57 | 67 | 77 | 87 | 97 |
| 8 | 8 | 18 | 28 | 38 | 48 | 58 | 68 | 78 | 88 | 98 |
| 9 | 9 | 19 | 29 | 39 | 49 | 59 | 69 | 79 | 89 | 99 |
| 10 | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |

Plug-ins: Package manager



Why do we need it?

Why does EclipseR need a graphic system to manage packages in R?

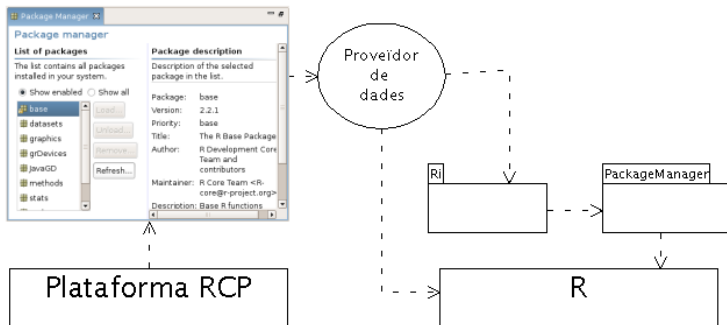
- Complex commands to get package description for:
 - installed packages
 - and new packages
- Get all available r-cran servers
- Install and remove packages
- ...

What does plug-in add in?

- Look up CRAN servers
- Get package information
- Install/Uninstall packages
- Enable/Disable packages

Architecture

Relationships between entities involve in to get R package information.

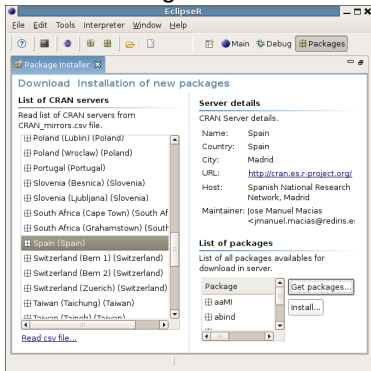


Definitive Views in plug-in

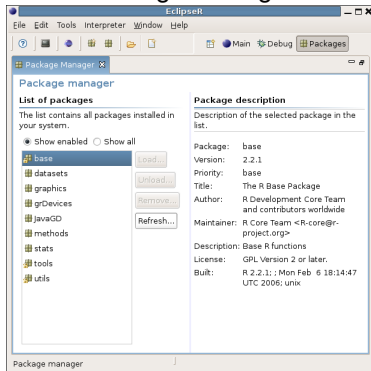
- Two views
 - Package installer
 - Package manager
- A perspective to join both: Packages

Installer and package manager

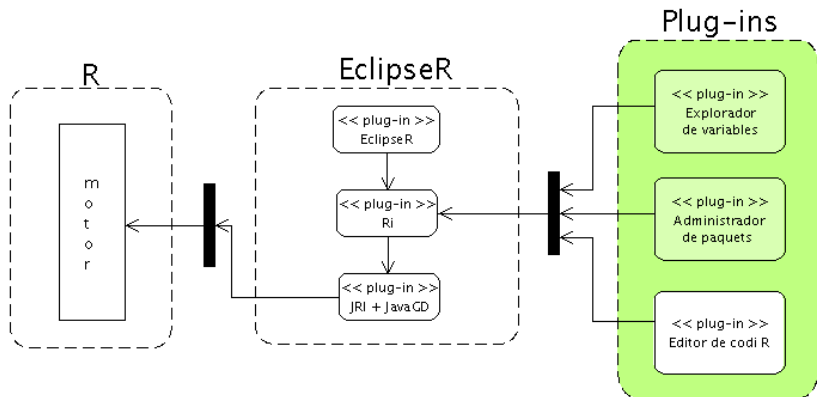
Package installer



Package manager



Plug-ins: R code editor



Why do we need a R code editor?

Why does EclipseR need a R code editor?

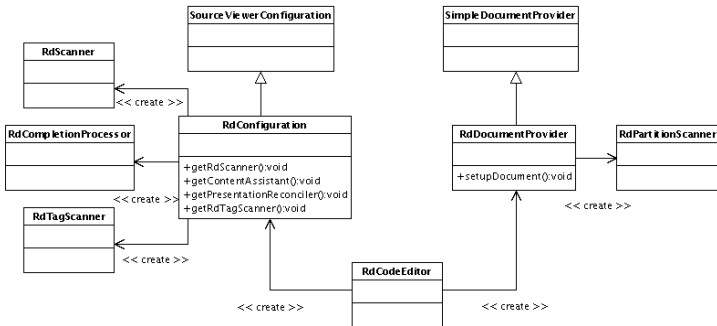
- Complex calculations to compute analysis
- We use to use scripts with R code instead single commands

Which is the contribution?

- View and edit text
- Common text actions: copy, paste and cut
- Syntax highlighting
- Contextual help
- Extra information in margins (line number, etc.)
- ...

Editor's architecture

Entities used to show/edit text:



Text editors defined in EclipseR

We have three text editors defined in:

- For commands: R code editor
- For documentation: Rd code editor
- For multiple purpose: Simple text editor

R and Rd text editor

R code

```
/home/beldog/volcan.R  /home/beldog/R/library/ade...
z <- 2 * volcano          # Exaggerate the relief
x <- 10 * (1:nrow(z))     # 10 meter spacing (S to N)
y <- 10 * (1:ncol(z))     # 10 meter spacing (E to W)

z0 <- min(z) - 20
z <- rbind(z0, cbind(z0, z, z0), z0)
x <- c(min(x) - 1e-10, x, max(x) + 1e-10)
y <- c(min(y) - 1e-10, y, max(y) + 1e-10)

fill <- matrix("green3", nr = nrow(z)-1, nc = ncol(z)-1)
fill[, i2 <- c(1,ncol(fill))] <- "gray"
fill[i1 <- c(1,nrow(fill)), ] <- "gray"

par(bg = "slategray",mar=rep(.5,4))
persp(x, y, z, theta = 135, phi = 30, col = fill, scale =
      ltheta = -120, lphi = 15, shade = 0.65, axes = FALSE)
```

Rd code

```
/home/beldog/volcan.R  /home/beldog/R/library/ade...
% --- Source file: man/amova.Rd ---
\name{amova}
\alias{amova}
\alias{print.amova}
\title{Analysis of molecular variance}
\description{
  The analysis of molecular variance tests the differences
  in a way similar to ANOVA. It includes evolutionary diste
}
\usage{
  amova(samples, distances, structures)
  print.amova(x, full = FALSE, \dots)
}
\arguments{
  \item{samples}{a data frame with haplotypes (or genotyp
  and abundance as entries}
  \item{distances}{an object of class \code{dist} compute
```

Demo

Time to demo!

Conclusions

- First application cycle done
- Flexibility and scalability in Eclipse
- Knowledge gained

To Do

- Improve graphic engine (current JavaGD) with OpenGL/DirectX?
- Printing engine for views, results and graphs
- Increase types compatibility in Data manager
- Define/use a communication channel from R to Java (rJava)
- Create and define Wizards (Cheet sheets)