Generating R reference classes in rClr with software reflection

Jean-Michel Perraud¹

1. Commonwealth Scientific and Industrial Research Organisation, Australia *Contact author: jean-michel.perraud@csiro.au

Keywords: interoperability, R reference classes, .NET, CLR

rClr (http://rclr.codeplex.com) is a package for R to access arbitrary .NET code executing on a Common Language Runtime (CLR) implementation: Microsoft's implementation or the multi-platform runtime Mono. **rClr** complements and in part re-uses the *R.NET* library (http://rdotnet.codeplex.com) that makes *R* programmatically accessible to .NET programmers, mostly but not only from the C# and F# languages. Work has been done in the growing F# community to expose R functionalities with F# idioms [3]. rClr should similarly leverage the most appropriate R idioms to expose CLR objects. The style of object oriented programming supported by the CLR makes R reference classes (see help (ReferenceClasses) in R) a natural candidate to access CLR objects from R. R reference classes have been used by at least two packages for interoperability: **RCpp** (Eddelbuettel [2]) and **rJavax** (Danenberg [1]). **rClr** will have an updated public release with support reference classes by or around the time of this conference. Given the closer similarities of the CLR with the Java runtime than C++, the design and implementation handling reference classes naturally shares similarities. In particular the capacity to reflect on software types/classes in the CLR and Java is useful to generate R reference classes, with a minimum of custom code. There are of course language difference between R and the CLR that remain and require choices to find a balance between the accessibility from R and the faithful representation of the CLR objects, properties and methods. Using R reference classes with **rClr** is demonstrated in a case study, the calibration of hydrological models. The general programming workflow and techniques that generate R reference classes is presented.

References

- [1] Danenberg, P. (2011). rJavax, http://cran.r-project.org/web/packages/rJavax/index.html
- [2] Eddelbuetel, D. (2013). Seamless R and C++ Integration with Rcpp. Springer, pp 236.
- [3] Mansel, H. and contributors, (2014). An F# type provider for interoperating with R, https://github.com/BlueMountainCapital/FSharpRProvider