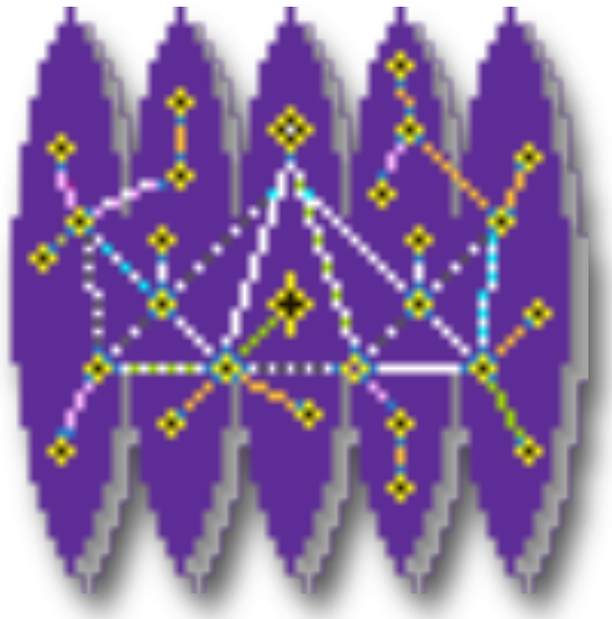

Open-Sourcing NetRexx

A fifteen minute update

René Vincent Jansen RexxLA 2008

Hello from a nightly Amsterdam.
I am sorry for not being there but I hope, in this way, I can share some of the excitement about the latest developments regarding NetRexx.
NetRexx has been my main production language for the last ten years and it is the way I like to think about programs and algorithms. It gives quick access to the Java Class libraries, writes like a scripting language and executes as quickly as a compiled language.
I never have worried about its popularity because I know what we have here. I have worried about continuity and survival, and that is what this talk is about.

In short

- It is our intention to have Open Sourced NetRexx sometime this year (2008)
- We are striving for the same governance structure as chosen for Open Object Rexx
- Working on this project within IBM are Tracy Dean (from July 1st onwards), Mike Cowlishaw and yours truly.

The title gave it away on the last slide, but it is our strong intention to have an Open Source NetRexx sometime this year. This is the place where I have to stop for a moment and thank Mike Cowlishaw, for not only giving us the language twice (once in having the vision that languages are for users and not for language implementors, and one more time in recognizing the significance of the Java virtual machine).
It only took a few years of slight prodding before Mike saw the point of preserving the future of NetRexx by offering its source to a community of developers. We hope to execute this task in the course of this year, and have a governance structure in place that enables it to develop and grow.
With guidance from Mike, Tracy Dean and I are working on steering the language translator to freedom.

IBM NetRexx
From Wikipedia, the free encyclopedia

NetRexx is IBM's implementation of the Rexx programming language to run on the Java virtual machine. It supports a classic Rexx syntax along with considerable additions to support Object-oriented programming in a manner compatibile with Java's object model. The syntax and object model differ considerably from Open Object Rexx, another IBM object-oriented variant of Rexx which has been released as open source software.

NetRexx is free to download from IBM[1].

A short wikipedia article, but nevertheless very to the point. It mentions that it is Rexx, has additional support for OO development, is compatible with Java's object model, differs considerably from Open Object Rexx, is not open source but freely downloadable.

This is what we should work on. And I do not only mean the Wikipedia article here.

Steps

- Follow IBM's open sourcing process - OSSC
- Prepare source code for release
  - Tidy up & Package, build procedure, arrange testing suite
- Formal handover to RexxLA
  - Form language committee & development team
  - Source code to repository

IBM has an Open Source Steering committee that defines the processes of using Open Source in IBM projects, or opening up the source of IBM Projects. The code base needs some small preparation, and this is a process that is being executed in parallel. Next to some tidying up, we have to make sure that it builds everywhere and has an integrated test suite. I am happy to say that most of this is there already, but the product is worked on exclusively within IBM. Before handover to the Rexx Language Association, there is ample time to form a language committee and a developer team, in charge of planning and releases, respectively. After handover the code will be available in a public repository, much like ooRexx.

OSSC

- IBM's Open Source Process
- ooRexx went through this in 2005-2006
- Strict and thorough approval process

The OSSC process is strict and well defined, and we will have to address every phase of this. There is some experience with this, and for this reason we will be in touch with the people who have driven ooRexx through it.

Some History

- NetRexx is in existence since 1996 as a compiler/translator
- Version 1.00 saw the light in 1997, as did the book
- An Interpreter was added in 2000
- NetRexx has been stable for a number of years

NetRexx started out as being a language with a reference implementation that was a translator that translates NetRexx source into Java source, and start the compiler. Later, methods were added to interpret clauses and execute them immediately. This was the last large change of this implementation, and after 2000 NetRexx was fairly stable. Stable, where stability means that there is nothing that needs fixing.

Publications

- Mike Cowlishaw, **The NetRexx Language**, Prentice Hall, ISBN 0-13-806332-X
- Heuchert, Haesbrouck,Furukawa, Wahli, **Creating Java Applications Using NetRexx**, IBM 1997, http://www.redbooks.ibm.com/abstracts/sg242216.html
- Mike Cowlishaw, **The NetRexx Interpreter**, RexxLA/WarpTech 2000, (rexxi.pdf)

For documentation, these are the main sources we are going to draw from. There is the standard book, like the TRL (we could call it the TNRL), and there is a very useful Redbook, that although a bit dated now, should form the base of a new edition that supports the open source product.
The RexxLa presentation about the Interpreter contains the best description of the inner working, on which can be built for the technical language translator implementation document.

Planning

- Preparation of source code has started
- Starting IBM Legal process July 1st, 2008 (Tracy Dean has agreed to be our NetRexx champion)
- RexxLA will plan for custodianship of NetRexx
- Development and code base will be separate from ooRexx, but with open communication lines
- Support and access will follow ooRexx's example

The current state of affairs is like this: the source code, the building process and the packaging are being worked on. The second half of this year we will begin the legal process. We have to speak to RexxLA for the official embedding of the project. Of course we will have open lines to ooRexx and might do some shared planning efforts when needed.

NetRexx is written in ~

- The language is ‘bootstrapped’ (starting in Rexx btw)
- A working compiler is needed to compile the compiler
- Disadvantages
  - Some problems become slightly non-trivial, like changing package name - opportunities to shoot oneself in the foot
- Advantages (other than not having to maintain C++)
  - It can be built everywhere where there is a running Java
  - Structure of the language translator is very clear - interpreter like

ooRexx is written in C++, and mainframe Rexx is written mostly in Assembler. NetRexx made the jump to a bootstrapped, self-hosted language, and is aptly written in NetRexx. This is excellent for maintainability, as the source code has all the clarity of Rexx itself. Also, because it only needs itself and a running Java to build, there is not a lot in terms of setup that needs to be done, and all the platform dependent problems can be skipped. There are a few dependencies on the Java environment, and there is, for example, some support to avoid elaborate path settings on most Java VM’s.

.. continued ...

- .. and will have even more clarity in a number of months
- will try to put together a group for maintenance and new releases
- this announcement is also meant to stir up the NetRexx community and form a small group
- Please:
  - suggestions - send them
  - help is welcome

This is as much an announcement of ongoing work, as a rally for supporters. We will try to mobilize the NetRexx users and form a community that cares about NetRexx and likes to write its software in it. Please consider to join and write in with suggestions and support.

Resident option

- As the Scala compiler does, the compiler will have an option to stay resident
- This is my old compiler server, now integrated in the main compiler source
- Will solve the problem of changed interfaces captured in compiler server state

Some of the work performed now is aimed at translator performance, an issue with more JVM languages, as the Scala compiler shows. I am now integrating the compiler server into the compiler, while solving an old problem with changing interfaces while the compiler server is running, while preserving the performance gains delivered by this approach.

Short and long term plans

- Open sourcing as-is (only small improvements and packaging)
  - Catch up on Java developments
    - annotations, generics
  - alternative backends
- Effort to come closer to Open Object Rexx
- Committee for language enhancements including Mike C.

Release 3 will be the current release with slight improvements to the build process, testing and packaging, but essentially no changes to the language.
After that we plan to look at recent Java developments in a cautionary way, and we might look at alternative runtime environments. We do not rule out that we might end up closer to ooRexx than we are now. Each decision needs to be documented and approved by the language committee before it is implemented.

Contact

- www.netrexx.org will be reactivated - starting with serving tools, documents and other information
- rene.vincent.jansen@nl.ibm.com
- Let’s hope for an Open NetRexx V3 in 2008 (or as soon as possible)

- Thanks for your attention!

We will reactivate the website. It will contain information and tools before the OSSC process has run its course. I would like to finish up with expressing the hope that we have made progress before the end of this year, and certainly before the next Rexx LA symposium, in which I hope to give you a detailed account of what has been accomplished. I thank you very much for your attention.
Are there any questions?