NetRexx Compiler Project Scope Statement

René-Vincent Jansen

I-BIZZ IT Services and Consultancy
Amsteldijk 14
1074 HR Amsterdam.

February 3, 2004

Abstract. This document states the initial project scope of the NetRexx compiler project initiated by the NetRexx.org foundation. It is the baseline for starting development of an Open Source version of a compiler for the NetRexx language.

1 Goals of the project

1.1 Primary Goals

The primary goal of the project is to protect the continuity of the NetRexx programming language. NetRexx is an object oriented programming language of the Rexx family, currently targeted to the Java Virtual Machine platform. It distinguishes itself from other programming languages by the level of attention that is spent on making it an easy language for the expression of designs in a humanly understandable form, and on the easy integration with its platform, like every language in the Rexx tradition did. This is a great accomplishment of it's inventor, Mike Cowlishaw.

Another primary goal of the project and the foundation that backs it is to further the use of NetRexx in open source projects, integrate it with major distributions of open source platforms and present it as a programming environment that would be in the unique position of being simultaneously suitable for educational purposes and the execution of large enterprise projects. The low astonishment factor behaviour of the language syntax is already there; our goal is to tune compiler performance as well as runtime performance to be on par with the best there is in compiler technology. For this, and to attain a widespread acceptance, the target platform may be widened to include other environments, starting with other runtime language environments and binaries for different operating platforms on different instruction set architectures.

1.2 Future Goals

In order to keep NetRexx a significant contender for programming in the research and educational environments, new programming paradigms will be evaluated for suitability to be supported by the compiler. The current thoughts include the Aspect Oriented paradigm and runtime metaprogramming. The exact form of these extensions will be subject of research when the first release of the compiler, supporting the current NetRexx language, is available to the community.

Around the time of the first release work will also be started for bringing in support for retroactive continuity for the Classic Rexx language and the Object Rexx language into the Java VM world. For this phase in the activity, *integration* and *extension* are the main goals, as there are already several well functioning compilers for nearly every important computing platform. There is also a working Classic Rexx to NetRexx converter. The number of compilers, though, seems to be limited to the NetRexx compiler and the IBM MVS/VM S/390 compiler. [1] In the *Planning* section of this document, the exact sequence activities and their relative priority is presented.

1.3 The language definition

Throughout this project, the term *NetRexx* denotes the definition of the programming language as stated in [Cowlishaw, 1997], and supplemented in the NetRexx supplement[Cowlishaw, 2000]. For Classic Rexx, the ISO standard will be leading. For Object Rexx, the current IBM commercial implementation will be leading. As a target environment, The Java Virtual Machine Specification[Lindholm and Yellin, 1999] is the authorative specification.

1.4 Form of the project

The project will be an Open Source project overseen by the NetRexx foundation. The contributed source code will be available under an Open Source license such as BSD or the LGPL[2]; this is implied by the stated primary goal of the project. The board will invite known experts to participate in its discussions and seek his advice on issues of implementation and

[1] Here we want to distinguish between compilers and tokenizers.

[2] See the list of approved Open Source Licenses at http://www.opensource.org/licenses/

language change. The foundation has an open attitude towards participation of members of the IBM labs with responsibility for the Object Rexx product, and would welcome contributions, other forms of assistance and integration plans. The eventual integration of all Rexx dialects would, in the views of the NetRexx foundation, be very beneficial to the user community. The board of the NetRexx Foundation will consist of at least five members with yearly rotating chairmanship, with decisions to be taken only with full approval of all the members, and after seeking advice of outside experts. The foundation strives towards an ISO standard for NetRexx, and will welcome alternative implementations, and will accept contributions as long as they do not subvert the open source nature of the project.

2 Infrastructure and tools

2.1 Project Infrastructural Environment

2.2 Platform

Java 2 will be the lowest supported level of Java to host and target the NetRexx.org compiler. The compiler code and the resulting code will be continually tested on Linux on IA32, MacOSX on PPC and Windows on IA32. When resources on other platforms become available to the implementation team this set can be extended. In later stages of the project the CLR and Mono can be employed as host and target for the NetRexx language.

2.3 Grammar Support

Antlr will be the parser generator that wil be used to generate the NetRexx grammar implementation. An Antlr backend NetRexx code generator will be written and used by the project.

2.4 Byte Code Generation

For the Java VM target environment a bytecode generation framework will be employed. BCEL and SERP are being evaluated as candidates for the project standard.

2.5 Source code management

Version management will be on CVS or Bitkeeper, depending on the chosen hosting facility for the project. Contributers are free to use other local mechanisms as long as they take care to synchronise to work on current versions and deliver their updates in standard patch format.

2.6 Project building support

The GNU Make utility will be the standard build tool. When time allows Ant project files will be supplied and maintained, as is support for the makepp tool.

2.7 Editing

Any editor that can handle plain text format will be suitable. Emacs with netrexx-mode is recommended for those who are still looking for the one true editor. JEdit with NetRexx mode is recommended for those who do no like Emacs. There are no initial source code formatting rules or standards. They can be introduced when deemed necessary but preferably only if they are editor-enforcable.

2.8 Documentation

Javadoc style-documentation for methods and global properties is mandatory. All other documentation is welcome but optional. The set of Javadoc on the project site needs to be automatically generated from the source code. There will be an effort to provide a JavaDoc-like functionality for NetRexx source without going through the generation of Java source code.

2.9 Package Structure

Package names will start with org.netrexx. The domain name is acquired by I-Bizz BV and donated to the NetRexx Foundation. The package structure of the compiler will be decided later during the design stage of this component.

2.10 Testing

JUnit will be employed to define a testing framework, in which hopefully the original testcases for the reference NetRexx compiler can be embedded. Tests and builds will be daily and automatically scheduled.

2.11 Other Tools

Other tooling for dealing with NetRexx will be placed on the netrexx.org site, that acts as a repository of all NetRexx related materials. Available are

1. Emacs netrexx-mode.el (syntax coloring, class template, documentation template, compilation and debugging support, method indexes, automatic indentation support)
2. The Antlr extention to generate lexers and parsers in NetRexx source
3. The NetRexx Compiler Server to speed up the build process
4. A NetRexx mode for the LaTeX listings package to format NetRexx source code automatically for LaTeX documents.

3 Planning

As indicated earlier the goals of the project are classified in primary goals and future goals. Here are all top level activities presented in that sequence, where the activities stated under Phase 2 goals will be addressed when the Phase One goals are attained.

3.1 Phase 1 Goals

1. To produce a correct compiler for the defined source and target languages;
2. Produce a faster running compiler than the reference; implementation, utilizing incremental compile technology and other techniques;
3. Develop new or modified tools to meet the above goals;
4. Provide continuity for the NetRexx language by having an alternative, maintainable, open source implementation;
5. Provide continuity for the software projects built using NetRexx;
6. Deliver a compiler that mimics the reference implementation;
7. Deliver a compiler that is extensible and maintainable;
8. An integrated compiler server to avoid reloading during large builds;
9. Close-to human Java source generation;
10. Relaxed cross language cross dependency checks.

3.2 Phase 2 Goals

1. NetRexx Language Extensions
   (a) Following the evolving Java language;
   (b) Bitwise operators;
   (c) Java 1.5 generics;
   (d) Runtime Type conversions without subverting type checking.
2. Language support tools
   (a) A testbed for language innovations;
   (b) Deliver a compiler that can be retargeted to other runtime environments;
   (c) Aspect Oriented NetRexx;
   (d) Runtime Type conversions without subverting type checking;
   (e) Classic and Object Rexx compatibility to target Java VM.

4 Founding Members of the Project

The founding members of this project are Chad Slaughter from the United States, René Vincent Jansen and Arjan Bos from the Netherlands, and Thomas Schneider and Rony Flatscher from Austria. The initial development of the grammar and the design will be done by Chad Slaughter, in continuation of a previous project. After that the group will work to fill in the framework until it is complete. Arjan Bos and Rene Jansen will have Java source code generation as their primary focus. Thomas Scheider and Rony Flatscher will work on the integration of Classic Rexx and Object Rexx. This does not imply that anybody cannot contribute to other components of the project, nor that there will not be other developers involved.

References

M.F. Cowlishaw. *The NetRexx Language*. Prentice Hall, New Jersey, 1997. ISBN 0-13-806332-X.

M.F. Cowlishaw. *The NetRexx Language Supplement*. IBM UK Laboratories, Hirsley, 2000. ISBN pdf file of August 23,2000.

Tim Lindholm and Frank Yellin. *The Java Virtual Machine Specification*. Sun Press/Addison-Wesley, 1999. ISBN 2021432943.