REXX: TECHNICAL ISSUES, TODAY AND TOMORROW

MICHAEL SINZ
COMMODORE

REXX

Technical Issues

Today and Tomorrow

Michael Sinz
Senior Systems Engineer
Commodore International - Technology Group

Today

The Good

- ## REXX is a computer language

  REXX is a easy language to learn do to the non-typed, non-declared nature of the language. MFC did a very good job in thinking about what the user of REXX needed rather than how languages are normally written.

- ## REXX is becoming a standard

  The X3J18 group is currently working on a draft ANSI standard for REXX.

- ## REXX is available across platforms

  REXX is now a standard part of a number of operating systems and is available in flavors for most others.

- ## REXX is part of solutions

  REXX is now seen as a standard tool in environments where REXX is installed. It has not only become part of the environment but has proven itself to be very useful. A great many example of this can be seen in the Amiga environment, where REXX has become the tool of choice for systems integration by VARs in many vertical markets.

Today

The Good

- ## REXX is very flexible

  Due to the design of REXX, it has turned out to be very flexible in adapting to more complex systems. For example, on the Amiga, REXX can communicate with any number of applications that have support for REXX. This makes it possible for users and systems integrators to pull together very powerful tools into what looks and acts like one very customized application. This makes the migration into vertical markets much easier and reduces the turn-around time to meet the demands of the changing markets.

- ## REXX has many good points

  After all, it took me two pages just to skim over the key points...

Today

The Bad

- ## REXX is a computer language

  While REXX is a easy language to learn, it is still a "computer language" and that is keeping some people from using it. Many users would easily be able to use REXX for "programming" if it did not feel like programming. A good example of this is the Lotus 1-2-3 macros which business people used all the time but did not realized that they were programming. (And if told it was programming, they suddenly stopped)

- ## REXX is a not up to date

  While REXX has many good points, it is currently not up to the task of some of the issues in today's computing environments. It is not so much that REXX can not be since any implementor of the language can choose to extend it in some ways; rather it is a problem of choosing a model that fits into the REXX model as well as addressing the requirements of complex multi-tasking, multi-user, multi-processor, networked, graphical, object oriented environments. (What a mouth full)

- ## REXX is not yet a standard

  While X3J18 is working hard on getting the standard done, it is not done yet and the various implementations of REXX are not fully interchangeable.

- ## REXX support in applications

  This will happen more as the market starts to demand it and as the utility of REXX becomes a major feature in products. A good example of this happening already is in the Amiga computer where productivity applications are almost required to support REXX due to public demand and feature requirements.

Today

The Ugly

- REXX is *NEVER* ugly...
- Well...
- ...almost never....

The implementation of a good REXX on many platforms is not as simple as the language seems. Part of this is due to the specification of the language and part of it is due to the way REXX is designed to interact with the operating environment of the system. Hopefully the specification of the language will help out, but the close interaction with the system will always be there for the developer to deal with. In addition, without work at getting REXX into new computing technologies such as GUIs, it can be rather "ugly" to code in REXX for such environments.

Tomorrow

- ## REXX and the future

In order for REXX to grow, the direction of the growth needs to be identified first. If the goal is to make REXX into the "user's" programming language, it is important that that goal is what drives development of the language.

- ## Multi-Tasking, Multi-User, & Networks

The current REXX model works great in simple environments. The fact that I/O is very simplistic make it easier for users to learn and use. However, this has also made a number of things rather difficult (if not impossible) to do in complex environments. Issues such as synchronization, semaphores, and shared access are all currently outside of the REXX model. While it would be simple to just use the models of other computer languages, it would be counter to the main goal of REXX: simplicity for the user. This means that a new model for such things as file locking, access control, and synchronization will be needed.

- ## Graphical User Interfaces

The world is moving into GUI environments. The reason for the growth of this interface model is due partly to the fact that computers are more powerful and that users find GUIs easier to learn and use. REXX, as a language, does not address any of these issue directly. External function libraries exist for a number of different GUIs but not having the language contain some fundamental support for GUI operation makes life more difficult for the person writing the REXX program that deals with the GUI. Research at a number of places, most notably IBM, have shown how REXX can be gracefully enhanced to gain these features. However, the amount of work involved for the implementor of the language processor is high.

Tomorrow

- ## REXX and Objects

As operating environments become more object oriented, REXX will need to learn about objects in order to fit in with the environments it is operating in. Last year, IBM showed some of their ideas on how this could be done. Work such as that will need to continue and will need to become standardized such that REXX continues to be a cross-platform language.

- ## REXX as a visual language

This is one of my goals for REXX. REXX has become a user's language. However, it is still very much like a computer language. With the Amiga (and soon to be the many OS/2 2.0 users) REXX has become a staple of application features. On the Amiga, over 140 REXX supporting applications are available with every new application having REXX support due to user demands. REXX has become both a systems integrators best friend and the advanced users power-tool. The next step would be to give this power to users who do not "program" a computer in the traditional sense. A visual interface to REXX programming that can be mastered by the business man and home computer user would be the ultimate goal. In a mature, REXX supporting platform, such a tool would give more users the power to combine their creativity along with the applications they have bought to produce something that is "what they want." Such a tool does not have to replace REXX but would just have to be able to sit on top of REXX. However, such a tool would require more standardization of the way applications support REXX and of the REXX language itself. (I am assuming that due to the complexity of such a tool that it would be "ported" to all the platforms that support REXX in such a way.)

- ## REXX in the future...

With the current growth of REXX as a user's tool and its inclusion as a standard part of a number of operating environments, the future for REXX looks bright. (And REXX developers can be assured of a number of tough problems that will need to be addressed.)

REXX

Going Strong Into the Future.