Problems and Issues Writing REXX Compilers

Markus Pelt-Layman
Pelt Industries
Problems and Issues writing REXX Compilers
Pelt Industries

History

- 1976 **TEACH** language interpreter (Honeywell)
- 1979 **DPS** command language (JES2 look-alike)
- 1985 **Intercept** aka **AF/Operator** command language
  
  *(TSO CLIST language look-alike for operations automation)*
- 1986 **OPS/MVS** REXX interpreter contract
  
  *(ADDRESS ISPEXEC, automation rules, cmd/response API)*
- 1987 Windows **Net*Edit** and **Elan Workstation** contracts
- 1989 **OPS/MVS** REXX **External Product Interface**
- 1992 **OS/2** Smalltalk GUI front-end (**REXX EHLLAPI**)
- 1994 **REXXANNE** full-time
WHY REXXANNE?

- I love REXX’s simplicity and power
- Serious development requires a compiler
- I need
  - Speed
  - Portability
  - High-level GUI framework
SOFTWARE FORCES

Speed

More hardware/software platforms

Portability

More APIs

© Copyright 1994 Pelt Industries. All Rights Reserved.
I LANGUAGE DEFINITION
TRL/ANSI REXX/Object REXX

II DEVELOPMENT ENVIRONMENT
Portable interpreter/compiler, IDE

III RUNTIME ENVIRONMENT
Portable run-time lib (built-in functions)
Portable ADDRESS environments (cmd/response)

IV OBJECT CLASS HIERARCHY
Portable base class library, GUI class library
REXXANNE
DESIGN GOALS

➢ Shorten development through portability
  (no new learning on new platform)
  • Cross-platform compiler
  • Cross-platform IDE
  • Cross-platform run-time library
  • Cross-platform base class library
  • Cross-platform GUI class library

➢ Shorten execution time
  through compilation and optimization
REXXANNE

DESIGN ISSUES

» Runtime licensing (INTERPRET)
» Run “interpreted” during development, compiled in production
» GUI portability versus platforms’ native look and features (synthesis problem)
REXXANNE
DEVELOPMENT ENVIRONMENT

Pro Edition
Standard Edition
Princess Edition

REXXANNE Compiler
RXN
Source
Compare
DIFF
Source
Update
UPDATE
Assembler
RXNASM
Object Code
Librarian
RXNLIB
Class
Hierarchy
Browser

Editor
Debugger
LINKER
RXNLINK
Project
Manager
RXNMAKE

Legend
Description
Good line name

© Copyright 1994 Pelt Industries. All Rights Reserved.
include pr0log.i

$$ START

i-h bp ; save caller bp
mw bp,sp - bad caller's stack pointer
adl
rxini ; htidization

====== Line 1 ========

say "Hello World!"

string MS errrors - MS warnings

0 Copyright 1994 Pelt Industries. All Rights Reserved.
10
REXXANNE
WINDOWS IDE

HELLO.REX

say "Hello World!"

HELLO.ASM

 Generated by RexxAnne compiler

 include prolog.i ; $START
 RX_TEXT segment byte public 'CODE'
 assume cs:RX_TEXT

 _RXPGM proc far
 push bp ; save caller bp
 mov bp,sp ; load caller's stack pointer to bp
 mov ax,sp ; adjust sp
 call _refx ; for intermediate
 mov sp,ax ; results
 call _rinit ; initialization

Line 1

 mov bx,offset DCNUMPL1 ; $RVALUE; string
 PUSHARG ds ; $SAY1
 call far ptr _RXSAY ; $S MALT
 add sp,ElemSize ; restore stack pointer
 push bp ; restore bp
 call _rinit ; return

 _RXPGM endp ; $END

REXXANNE Compiler Version 2.0
Copyright (C) 1991-1995 Markus Pelt-Layman. All Rights Reserved.

No errors - No warnings

© Copyright 1994 Pelt Industries. All Rights Reserved.
We are **WAY** behind schedule
- under-staffed
- under-capitalized
- too ambitious a vision?

Lack of user interest/support in REXX
Task out work

- Lenny Koff - online tutorial & doc
- John Kastner - OS/2 IDE
- Billy Jack - runtime library
- Find Windows/Windows NT/Windows 95 guru

Cut down on vision
Generate revenue
  ◆ INTER REXX subscriptions
  ◆ InfoREXX sales
  ◆ Enterprise REXX sales
  ◆ Miscellaneous REXX product sales
  ◆ Beta sales (free upgrade)

Look for business partners

Look for capital
REXXANNE
LACK OF USER INTEREST

➤ Generate interest in REXX language
  ◆ INTER REXX newsletter
  ◆ Get articles published in other magazines
  ◆ Prod IBM and others to better marketing

➤ Publish information on REXX
  ◆ InfoREXX multimedia help file
  ◆ Novice tutorial

➤ Generate interest in REXXANNE
  ◆ comp.lang.rexx newsgroup participation
  ◆ MSJ advertisements
REXXANNE
FUTURE

➤ Continue REXX evangelism
➤ Release “early bird” version (free upgrade)
➤ Share development of new features with other vendors (IDE, APIs)
➤ Get users involved in development
➤ Continue long-term commitment to REXX