An Object REXX Retrospective

Simon Nash

19th May 2009

Sheridan House, Winchester

My Office

Disclaimers

- Don’t expect:
  - Deep technical detail
  - A complete historical account
  - Thoroughly researched accuracy
- Do expect:
  - My personal perspective
  - A few things about Object REXX that you didn't know previously!

General Observations

- Object REXX was a wonderful experience for me
- It was also a difficult experience, and things didn't quite turn out as I had hoped
- I learned some extremely important things (mostly non-technical)
- If I could do it over again, I would do some parts of it very differently (mostly non-technical)
- I am amazed and humbled that Object REXX is still going strong after 20+ years

The Making of Object REXX

- In the beginning (1988)
- The essential core (1989)
- The REXX ARB (1989–94)
- Building the Oryx System (1990)
- SHARE (1990–93)
- CCOT and SOM (1990–92)
- Transfer to Endicott (1993)
- Transfer to Boeblingen (1995)
- IBM products for OS/2 (1996) and Windows (1997)

What's in a Name?

When preparing this talk, I came across documents describing:

Oryx, ORYX, Object–Oriented REXX,

O–REXX, OREXX, O–O REXX, Object REXX

From now on, I'll use the first of these names.

1988: In the Beginning

- Early ideas: Brian Payton, Bruce Lucas, Ian Brackenbury
- 10 July 1988 IB->SN: Are you future architect of VROOM?
- Random thoughts: Simon Nash
- The Oryx Workshop: 15 December 1988
- At the end of 1988, Oryx had:
  - Object-based encapsulation, message passing only
  - Object-based concurrency (early reply and guarded methods)
  - Everything (including runtime mechanisms) is an object
  - Classes and inheritance are programmed, not built-in

1989: The Essential Core

- People: Simon N, Aran Lunzer, Dave Mitchell
- Some things didn't survive: closures, keyword arguments, [] for pointers and reference arguments, assertions, coercion, ~* and ~&, ~()
- At the end of 1989, Oryx had:
  - `a~b(c)`
  - `~foo` for public environment objects
  - `method expose` for object variables
  - `start` method returning a proxy object
  - Unrestricted multiple inheritance
  - o-code: an object-based instruction set and assembly language
  - Work in progress for VISOR and Advisor

An o-code Example (Dec 89)

The Oryx method

```
method expose foo
a = 3
foo~bar(a)
```

could be translated to

```
          s_mvd  m_get   a_1     ;get variable object for "a"
          e_0    m_set   a_2     ;set variable "a" to 3
          s_ovd  m_val   a_3     ;get value of variable "foo"
          s_mvd  m_val   a_1     ;get value of variable "a"
          o_aput 1       a_4     ;put into args array
          o_starg e_2    a_5     ;send the message
          s_cont m_ret   *       ;return from this context
a_1       c_str  'a'
a_2       c_num  3
a_3       c_str  'foo'
a_4       c_array *
a_5       c_str  'bar'
```

1990: Building the Oryx System

- People: Simon N, Aran L, John Bennett, Dave Renshaw
- Translator: From source code to o-code
- VISOR: VISual Oryx
- Advisor: The Oryx IDE
- The great debate: ~ vs. ::
- At the end of 1990, Oryx had:
  - Availability within IBM on OS2TOOLS
  - Visibility outside IBM through SHARE (thanks, Linda!)
  - Leading edge technology far ahead of its time

1991: CCOT Extension Language

- People: Simon N, Aran L, John B, Dave R, Mike Conner, Nurcan Coskun
- My first REXX Symposium (thanks, Cathie!)
- At the end of 1991, Oryx had:
  - Integration points for use as an extension language
  - Designs for using Oryx with CCOT and SOM
  - Multiple masters and conflicting priorities
  - No product plans or executive support
  - Gradually decreasing funding

1992: SOM and Workplace Shell

- At the end of 1992, Oryx had:
  - A chapter in the REXX Handbook
  - Support for using and subclassing SOM objects
  - OS/2 Workplace Shell integration
  - SuperVisor instead of VISOR
  - Debugging support for BOB
  - No more CCOT funding
  - A decision to transfer Oryx from Hursley to Endicott

1993: The Transfer to Endicott

- People: Gary Cole, Steve Pritko
- May 1993: REXX Symposium, La Jolla
- On 30 June 1993, Oryx had:
  - `::class` and `::method`
  - `expose` instead of `method expose`
  - `guard` instead of `method guard`
  - `.foo` instead of `~foo` for public environment objects
  - Double twiddle
  - Square brackets for arrays
  - A startup image (at last!)
  - DAVE (Develop Applications Very Easily): a visual

1989–94: The REXX ARB

- People: Brian Marks, Mike Cowlishaw
- 2 June 1989: REXX Language Point 100
- 14 March 1994: REXXLD93 Final Draft
- On 14 March 1994, the Object REXX language had:
  - Message objects
  - A collection class hierarchy
  - Mixin-based inheritance
  - Property support

Some Lessons Learned

- Keep it simple
- Ship it while it’s still leading edge, even if not fully baked
- Design by committee is a mixed blessing
- Technical excellence is not enough
- Step back and look at the bigger picture

My Favourite Oryx Things

- Everything an object, including system primitives
- Fully dynamic semantics
  - no declarations, translation not compilation
- o-code (an object-based computer)
- Object-based concurrency
- The message object
  - the ultimate solution to asynchronous programming
- Shared memory objects with process affinity
- The little Oryx icon!