A Syllabus for Introducing MBA Students to Procedural and Object-oriented Programming (Object Rexx)

Rony G. Flatscher

Vienna University of Economics and Business Administration, Austria
Department of Management Information Systems

ASSOCIATION FOR INFORMATION SYSTEMS, AMERICAS CONFERENCE AUGUST 13-15, 1999, MILWAUKEE, WISCONSIN

Abstract

MBA students are exposed to information systems which can be controlled by the means of macro-languages and are faced with a new type of information systems which employ the object-oriented paradigm (e.g. SAP, 1999). With no idea about (procedural and) especially object-oriented concepts, it is impossible for such students to comprehend and evaluate the new breed of ERP systems with OO-interfaces and the ongoing developments in the realm of „business objects“ (e.g. OMG BODTF, 1999). Therefore, in the summer-semester 1999 an experimental lecture was introduced, which teaches MBA students the fundamental concpets of procedural and object-oriented concepts. To allow them to experiment and build hand-on experience a language was chosen that is known for its simple and easy syntax, yet powerful object model: Object Rexx. The devised syllabus, which teaches the fundamental concepts of procedural and object-oriented programming is layed out and explained in this article.[1]

MBA Students

At the WU there are over 25,000 MBA students who learn the fundamentals of business administration. The curriculum and the structure of the studies were layed out at an IAIM conference. (Flatscher, 1993) Basically, the studies are divided into two parts, each lasting a minimum of two years. It is in the second part of the studies, where MBA students are allowed to choose up to two specialized fields of Business Administration like „Tourism“, „Finance“ or „Management Information Systems“. For successfully studying MIS a minimum of 12 hours must be taken followed by a comprehensive written and oral examination. Figure 1 sketches the available MIS-classes as of January 1999.

choice of	
Electronic Commerce (lecture, 2 hours)	Electronic Finance (lecture, 2 hours)
Business Process Modelling (lecture, 2 hours)	
choice of	
„ISD“: Information Systems Development with CASE (lecture, 2 hours, optional introduction)	„SPET“: Solving Problems with Enduser Tools (lecture, 2 hours, optional introduction)
ISD - Application (pro-seminar, 2 hours)	SPET - Application (pro-seminar, 2 hours)
at least two lectures („Electives“) out of IS in Finance and Accounting; IS in Marketing; IS in Commerce; Computer Law; IT Market and Information Management; Electronic Money, Payment Systems and Security	
MIS (seminar, 2 hours)	

Figure 1: MIS-Classes at WU (as of 1999-01).

Due to the business-oriented nature of the University, there was no need in the past years to teach object-oriented concepts in order for the students to understand and evaluate business-related information systems like ERP. Yet, in the past years the initiatives of OMG to devise object-oriented standards for business objects with the implementation of the BODTF („Business Object Domain Task Force“) has led - among other things - to a competition between ERP vendors like SAP and Baan, which offered their BO-solutions to OMG for standardization.[2]

It is this particular situation which makes it important for MBA-students to learn about object-oriented principles, because otherwise they are unable to comprehend the BO frameworks offered by ERP-vendors, let alone assessing them. With other words, the marriage between business adminstration domain knowledge and information technology (IT) is not possible for the students anymore, once object-oriented IT gets applied.

This uneasy situation led to the creation of a new (elective) MIS-class for the summer semester 1999, which attempts to teach the object-oriented paradigm with the help of an easy to learn programming language. The programming language should allow the students to apply what they learn, experiment further with the taught concepts and build experience on their own. In this context it is felt very important that a language be chosen, which is rather easy to learn: Rexx was built with this intention from the ground up and Object Rexx carried this idea further into the object-oriented world, built on a powerful object model. (Cowlishaw, 1984, 1990, 1994; Flatscher, 1996, 1997) Of course, the fundamental procedural and object-oriented concepts taught can be transferred to most other programming languages. For this class Object Rexx[3] was chosen due to its simple syntax, which should allow the students to concentrate on the concepts rather than the syntax or peculiarities of a specific programming language like C++ or Java[4].

It is expected that students electing the class „Introduction into the procedural and object-oriented programming (Object Rexx)“ will be able to learn the planned object-oriented concepts and as a result become able among other things to comprehend and follow the developments of business objects.

The Devised Syllabus

In this section the contents of the concepts which ought to be taught each week are documented.

Procedural Concepts

The descriptions for the weeks 1 through 5 introduce the procedural concepts.

Week 1: Overview of the class, accepting of students, history of Rexx, latest developments: ANSI Rexx, Object Rexx, NetRexx (Java). This installment explains the intention of the author of Rexx, Mike F. Cowlishaw, to create a „human centric“ programming language which results in a simple and easy to learn syntax (Cowlishaw, 1994).

Week 2: Minimal Rexx program, interactive programming („RexxTry.cmd“), variables, constants, comments, statements, blocks, selections, iterations. The basic building blocks of a typical procedural programming language are explained.

Week 3: Labels, procedures, functions, searching order for functions/procedures, scopes. The concepts of subroutines and (depending on time even recursive) functions together with the scopes and their effects are introduced and discussed.

Week 4: Built-in Rexx-functions, „stem“ variables, „Rexx utility“ functions („RexxUtil“). „Stems“ can be regarded as asscociative arrays, whose index can be any (compound) string. Some utility functions use numeric indices as subscripts to stems by convention and need to be understood.

Week 5: Exception handling (errors, user exceptions), Object Rexx extensions: routines, argument references. The importance being able to raise and intercept exceptions is stressed.

Object-Oriented Concepts

The descriptions for the weeks 6 through 15 introduce the most important object-oriented concepts.

Week 6: Classes, methods, attributes, messages („twiddle“ = tilde ‘~’), scopes, creation (instantiation) of objects. This class introduces the basic concepts of the object-oriented paradigm starting out with a discussion of ADT’s (abstract data types). Object Rexx is backwardly compatible to (procedural) Rexx and retains its easy syntax. Internally any statement is transposed to its OO-form before execution.

Week 7: Inheritance, specializing, scopes, concurrent execution. The concept of the classification tree and its applications are introduced as well as concurrency issues with respect to invocation of methods on the same object at different levels of the classification tree and of different objects at the same level.

Week 8: Object Rexx classes and examples. An overview of the built-in classes and their application is given with examples stressing different issues like message-invocations, definition of alarms, usage of monitors.

Week 9: Object Rexx collection classes and examples. An overview of the built-in collection classes and their application is given together with examples.

Week 10: Class methods, class attributes, class ‘Class’ (Object Rexx’ metaclass). The concept of ‘metaclass’ is introduced and explained. Of the Object Rexx metaclasses the founding class ‘Class’ is explained.

Week 11: The big picture (Object Rexx class hierarchy and its instantiation), definition of classes and methods at runtime, ‘one-off’ objects. The important role of the classes ‘Object’ and ‘Class’ are re-iterated, the methods for reflection explained and the logic of setting up a runtime system for representing the classification tree itself is introduced.

Week 12: Coupling with the help of the Object Rexx environment, ‘`.local`’- and ‘`.environment`’-directories. The runtime system sets up environments for modules and sessions for keeping objects available for retrieval. It is possible to couple different routines, methods and modules via these environments.

Week 13: Object Rexx Utilities ‘ORX7’ and ‘ORX8’ (Flatscher 1996, 1997). A set of miscellaneous utility modules, routines, classes and programs for Object Rexx are introduced and some of them are explained in detail in order to ease the understanding and the creation of such utilities.

Week 14: Concurrent execution between (‘inter’) objects and within (‘intra’) objects, access management (‘Guard’), multi-threading (‘Reply’statement, exploiting class ‘Message’). This installment exposes particular concepts to the students, which might help them to understand even complex concurrency issues.

Week 15: Wrap-up, Outlook: Security manager, multiple inheritance, ‘Forward’- and ‚Unknown’-statements, DSOM/SOM-, OLEAutomation/ActiveX-support. This is aimed at pointing the students to different possible applications of Object Rexx, whereby it should become clear how to address OO environments of the respective operating environments.

Conclusions

This article attempted to explain the motivation for creating a syllabus for introducing the procedural and object-oriented paradigm to MBA students. In order for demonstrating and allowing the students to experiment interactively with the concepts the programming language Object Rexx was chosen due to its simple syntax, which was devised to be „human centric“. Subsequently, the contents of each week of a total of 15 weeks were briefly introduced in order for the reader to grasp their contents.

It is expected that MBA students who are taking this class for the first time in the summer-semester of 1999 develop an understanding of the fundamental procedural and most important object-oriented concepts. This should allow them to understand the principles, which underpin object-oriented modeling languages like OMG’s MOF (meta-meta-model; MOF, 1997), OMG’s UML (meta-model; UML, 1997), „business objects“, „business components“ and „business frameworks“.[5]

References

COWLISHAW M. (1984) *The design of the REXX language*. IBM Systems Journal, Volume 23, No. 4 (1984).

COWLISHAW M. (1990) The REXX Language. Prentice Hall (second edition), 1990.

COWLISHAW M. (1994) *The Early History of REXX*. IEEE Annals of the History of Computing, Vol 16, No. 4, Winter 1994, pp15-24.

COWLISHAW M. (1997) The NetRexx Language. Prentice Hall, 1997.

FLATSCHER R.G. (1993) *The Curriculum for Teaching „Developing Information Systems with CASE“ at the Vienna University of Economics and Business Administration and the Experience with two Semesters of Teaching It.* Proceedings of the 8th Annual Conference of the International Academy for Information Management (IAIM), Orlando/Florida, 1993.

FLATSCHER R.G. (1996) *Local Environment and Scopes in Object Rexx. Object Classes, Meta Classes and Method Resolution in Object Rexx. „ORX_ANALYZE.CMD“ - a Program for Analyzing Directives and Signatures of Object Rexx Programs.* Proceedings of the 7th International Rexx Symposium (Rexx Language Association), Austin/Texas, 1996.

FLATSCHER R.G. (1997) *Utility Routines and Utiltiy Classes for Object Rexx.* Proceedings of the 8th International Rexx Symposium (Rexx Language Association), Heidelberg/Germany, 1997.

OMG BODTF (1999) URL (as of 1999-02-28): `http://www.omg.org/techprocess/sigs.html#bomsig`.

OMG MOF (1997) *Meta Object Facility.* OMG standard, 1997. URL (as of 1999-02-28): `ftp://ftp.omg.org/pub/docs/ad/97-10-02.pdf`.

OMG UML (1997) *Unified Modeling Language.* OMG standard, 1997. URL (as of 1999-02-28): `ftp://ftp.omg.org/pub/docs/ad/97-08-04.pdf`.

SAP (1999) URL (as of 1999-02-28): `http://www.sap.com/` searching for „business objects“, yielding „BAPI“-documents.

VENESKEY G., TROSKY W., URBANIAK J. (1996) *Object Rexx by Example.* Aviar Co., Pittsburgh 1996.

WAHLI U., HOLDER I., TURTON T. (1997) *Object Rexx for Windows NT and Windows 95.* Prentice-Hall, 1997.

W3REXX (1999) WWW-homepage of Rexx with links to Object Rexx. URL (as of 1999-02-28): `http://www2.hursley.ibm.com/rexx/`.

[1] The gathering of experiences with this proposed syllabus happens at the time of writing during the summer-semester 1999 which ends in June 1999.

[2] In the wintersemester of 1998/99 a group of seven seminar-students had to work on a paper which was aimed at introducing MBA students to business objects. Neither the paper, nor a well-prepared presentation at the end of January 1999 helped the group of remaining seminar-students to really understand concepts like „object“, „instance“, „class“, „instance methods“, „messages“, „frameworks“ and the like. As a result the business object proposals by SAP and Baan were not understood either. The same group of students was able to understand concepts which are tought in the business modelling class like „entity“, „entity type“, „super/subtypes“.

[3] Object Rexx is available (in alphabetical order) for AIX, Linux, OS/2 and Windows 95/98/NT/2000.

[4] By the same token teaching NetRexx (Cowlishaw, 1997) would be preferred to teaching pure Java.

[5] For the first time this course started in March 1999 and ended in June 1999. Therefore at the time of this writing no experiences with respect to the apperception of the MBA students can be reported. It can be expected that an appropriate report can be given in the summer.