R.W. Witty

My DIC note

24 SE

SCIENCE AND ENGINEERING RESEARCH COUNCIL
RUTHERFORD APPLETON LABORATORY

INFORMATICS DIVISION

DISTRIBUTED INTERACTIVE COMPUTING NOTE 992

issued by
A J Kinroy

Software Engineering using Executable Specifications

17 September 1984

DISTRIBUTION: SEG
F R A Hopgood
Workshops/OBJ file

(see next page)

The course was held at UMIST on 10-12 September. The lecturers were Robin Gallimore and Derek Coleman of the Department of Computation.

Twenty-three people attended the course. The participants came from both academic and industrial institutions (see list attached). It was obvious that the course members had a wide range of experience in specification methods. Some were obviously up to date with developments in the field, others were complete beginners (myself included).

Course Content

The course was split into lectures and practical work. Each morning there were 3 lectures, followed each afternoon by practical work - solving examples from a problem sheet. Typed copies of the lecture material were handed out and bound together at the end of the course.

Throughout the course a rigorous approach to formal program design was described. Ideally this starts with a formal specification of the system requirements formulated in a high level mathematical notation and at a high level of abstraction from the intended implementation.

Formal specification languages are used to describe the intended behaviour of a program. Mathematical proof techniques can be used to check the specification is well-formed.

The course described the formal specification language OBJ which allows description of abstract data types and operations over those data types (using equations) together with the capability to execute an expression to see if it evaluates correctly when reduced using the defining equations as rewrite rules.

The description of an abstract data type (object) in OBJ includes object descriptions imported, new types defined (sorts) operator definitions (which include name, function domain and function range), variable definitions (for variables used in the equation definitions) and equation definitions (which define in a functional manner the action of the various operators. Note that constructor operators have no defining equations - they act as denotational term building operators for objects in the range (carrier) of the sort.). An example of an OBJ object is given below:

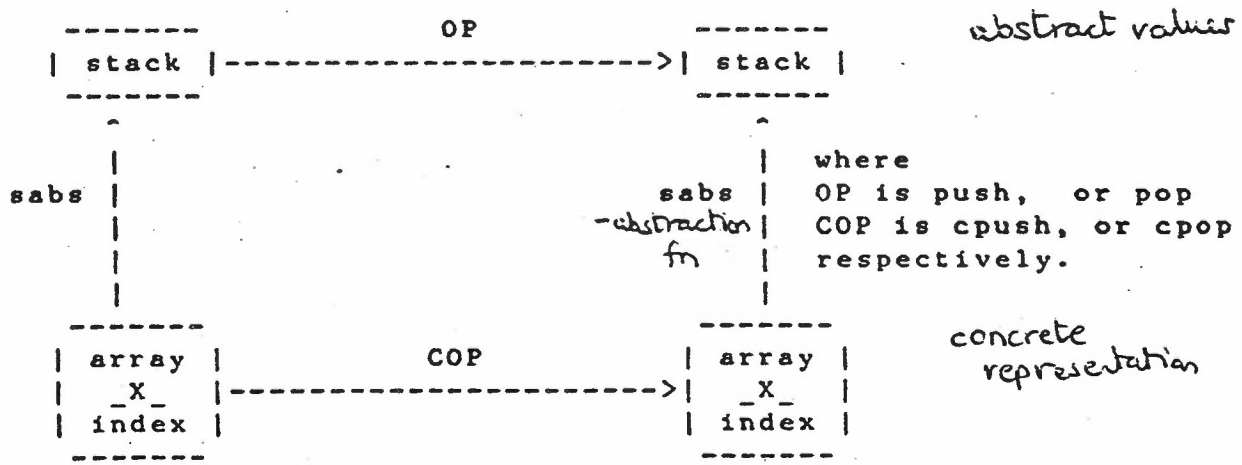
```
OBJ Stack/Item
SORTS stack
OPS
     create   :                  -> stack   ) constructors for the
     push     : item  stack     -> stack   ) carrier associated with
     pop      : stack           -> stack     sort stack
     top      : stack           -> item
     isempty  : stack           -> BOOL
VARS
i:  item
s:  stack
EQNS
  ( pop(create) = create )
  ( pop(push(i,s)) = s )

  ( top(create) = underflow_item )
  ( top(push(i,s)) = i )

  ( isempty(create) = T )
  ( isempty(push(i,s)) = F )
JBO
```

The defining equations for an abstract data type may be recursive. It is important to ensure that there is a base case for the recursion and the equations correctly pattern match all arguments of the domain type so that a correct value of the range type is always evaluated.

Program specification can be modelled as a transition between an initial and a final state. This can be given as a set of pre-execution and post-execution predicates. In OBJ these conditions can be given definitions in terms of operations on abstract data types.

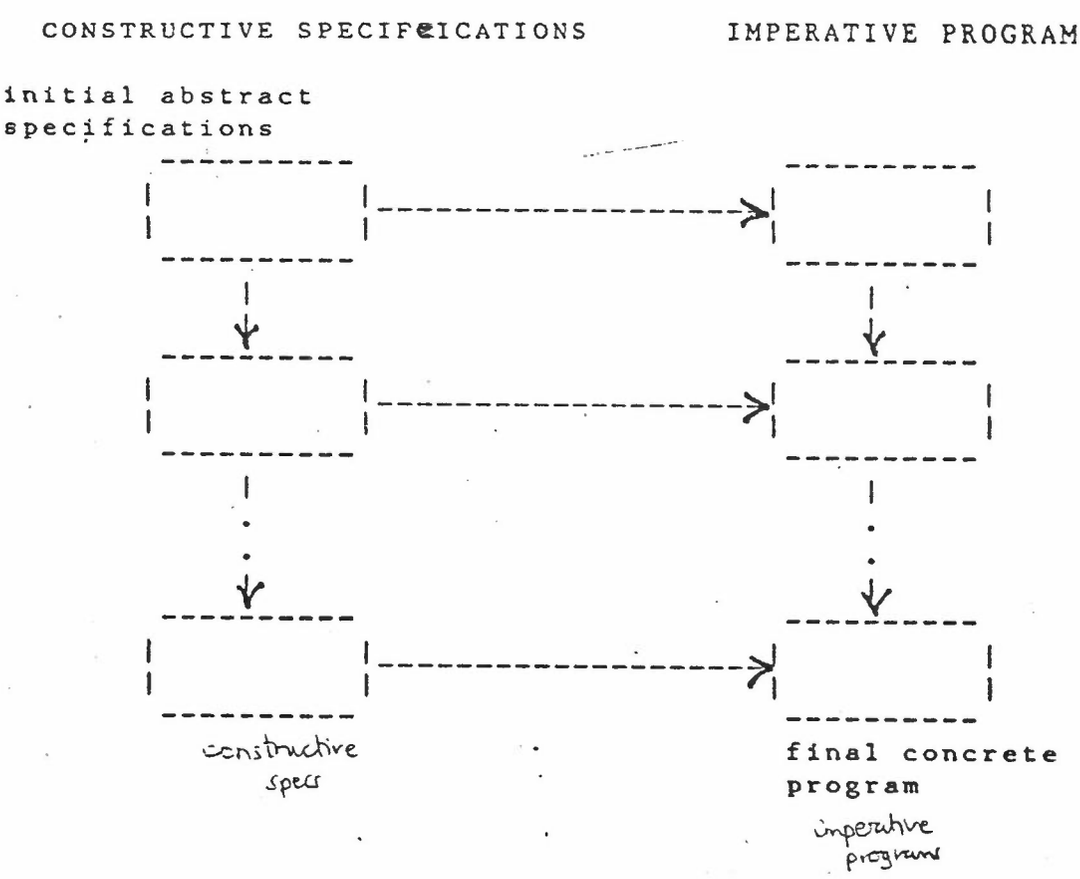
A complete model of a requirements specification in OBJ consists of a requirements specification (non-algorithmic), a problem domain (of basic OBJ objects), a constructive specification (enhanced OBJ environment) and a validation section (which contains sample test examples and output produced from executing the examples).

The refinement of abstract data types to concrete data types and the use of an abstraction function was illustrated with an example of how to implement a stack as an (array, integer) pair.



An example of how to transform an OBJ specification for a list selecting function into Pascal was given. Some optimisation of the Pascal may be necessary if for example there is a lot of inefficient recursion.

The refinement process is illustrated by the following diagram.



To enable program validation it is best to perform refinements on the specifications (left side of the diagram) and delay the transformation to the imperative programming language.

The final example in the lecture notes shows how to construct an OBJ model for a pattern matcher.

Practical Sessions

These were run using Gallimore & Coleman's currently implemented subset of OBJ. The implementation runs on a VAX 750 under VMS. It is implemented in Pascal.

Each course member had use of a VT100 terminal connected to the VAX.

The problem sheets given out gave the chance to learn how to describe OBJ objects. The examples got progressively more complex and difficult over the 3 days.

It was very difficult not to feel that one was programming (albeit functionally) rather than specifying. I had particular conceptual difficulty in distinguishing constructors from other operators.

Conclusions

The lecturers gave a well integrated, well presented course. Some background in logic and domain theory was necessary to gain full benefit from the course (eg to understand the underlying semantics of OBJ). The practical sessions were useful though it was difficult to remember one was supposed to be specifying not programming.

UMIST

The University of Manchester Institute of Science and Technology
PO Box 88, Manchester M60 1QD, United Kingdom
Telephone 061-236 3311
Telex 666094
D H McWilliam, *BA, Barrister (Inner Temple)*
Secretary and Registrar

SOFTWARE ENGINEERING USING EXECUTABLE SPECIFICATIONS
10 - 12 September 1984

List of Participants:

Dr Stephen P Bear	Central Electricity Generating Board, London
Mr Mark J Benson	Oriel Computer Services (1982) Limited
Mr M Diss	Plessey Electronic Systems Research Limited
Mr Miles R Doubleday	Oriel Computer Services (1982) Limited
Mr P Forrow	Racal I.T.D. Limited
Dr Alex D Hill	Central Electricity Generating Board, London
Mr Timothy S Hoverd	STC Idec Limited
Mr Alan J Kinroy	Science & Engineering Research Council, Oxon
Mr Graham P Martin	British Telecom, Ipswich
Mr Joseph M McCaig	Kingston Polytechnic, Surrey
Mr Paul McGrath	Leeds Polytechnic
Mr Gordon Nichols	The Hatfield Polytechnic, Herts
Mr Brian Passingham	Software Sciences Limited
Mr Roger M A Peel	University of Surrey
Mr John Reed	Oriel Computer Services (1982) Limited
Mr Christopher J Sadler	Polytechnic of South Bank, London
Mr Inderjit S Sandhu	Marconi Underwater Systems. Wembley
Mr Ian H Smith	Oriel Computer Services (1982) Limited
Mr Richard Tavendale	Marconi Research Centre, Essex
Dr Kenneth J Turner	I.C.L. Stoke-on-Trent
Dr Robert W Witty	Science & Engineering Research Council, Oxon
Dr U Webb	Computation Department, UMIST
Dr C Walter	Computation Department, UMIST