SCIENCE AND ENGINEERING RESEARCH COUNCIL
RUTHERFORD APPLETON LABORATORY

INFORMATICS DIVISION

SOFTWARE ENGINEERING GROUP NOTE 151

Trip to USA - October 1986

issued by
R W Witty
27 October 1986

DISTRIBUTION: F R A Hopgood
R W Witty
D A Duce

(see next page)

DEC SPITBROCK

1. Non disclosure about VAX Forms.

So called 'declarational' (sic) language to describe forms and their interaction with user on one side and applications program on the other.

Will enable different devices to be handled by one form definition. Transmit over network as Ascii. Will work on 8-bit multinational character set.

2. GEN

Power tool for experienced Cobol programmers. Diagrammatic based, added text, to enable rapid generation of Cobol source. Runs over GKS. State of the art or just behind. Inconsistent interface with other DEC tools. No integration with compiler.

3. GKS

Does RAL know John McConnell? DEC using GKS as graphics for several products. Looking forward to certification.

4. Tom Harris

Just got 15 minutes. Twisted cartledge at DECUS dance. Ada and Environments in another part - no contact between his group? Knew about System Designers deal. Not saying much. Clear that DEC have some king of environment project based on relational database but no details forthcoming.

5. Other

DEC producing ADT (Analysis and Design Tool). To be 'methodology independent'. Some kind of language (MIF) in which user can describe his own method. Has a diagram editor but what else I could not discover. Did not seem to produce source code. Bruce - said that DEC needed to put some thought and effort into integrating its tools from now on! As described by Bruce they are classic USA 'built it to see if it can be done', ship prototype to customers if it works. then build the second, real product.

6. Spitbrock

DECs software products people being housed all together. About 4,000 people. DEC has 80,000 world wide. DEC wondering how to grow at 10-15% - means new areas of business (hence attacking IBM in DP area) and acquisitions.

CORNELL

1. General

Cornell is a medium sized department which has grown up from a fairly theoretical basis. It has around 5 full professors out of a faculty of about 25 staff. 80 undergraduates graduate per year. They have about 70 postgraduate students. The dept ranks about 5th after Stanford, CMU, MIT and Berkeley.

Cornell is a private university, quite well endowed. The CS dept is about to expand into a new building. David Gries has been Chairman from 1982-June 87. Bob Constable has been at Cornell since 1968. Wife and 2 daughters. Tietelbaum, Gries and Constable are all in their mid forties.

2. NU-PRL

Bob Constable described the NU-PRL system which is a theorem prover based on constructive type theory. It uses ML and tactics extensively due to the collaboration with Edinburgh. Proofs are constructed 'by refinement' ie top down goal directed. (This led to the proof editor idea when combined with Tietelbaum's project). NU-PRL has a reasonable library system. It runs on VAX and Symbolics with a multi window interface which is quite nice but not excellent. It is in FranzLisp, Zeta Lisp and now Common Lisp. It is well described in the recent book, Implementing Mathematics with the NUPRL Proof Development System.

Constable is interested in mathematics being done with mechanical tools to give brain amplification and checking etc. He wants to see theories about maths, physics developed in machine manipulable forms.

NU-PRL seems to me to be one of the most advanced systems in the world. It is very expressive, being based on constructive type theory, so all kinds of high order logics can be expressed. Its tactic driven top down refinement style proof development is a natural development of ML/LCF due to collaboration with Edinburgh. NU-PRL, being constructive, has a new feature the EXTRACTOR function, which enables an executable function (in this case LISP code) to be generated from a valid proof, thus guaranteeing that the code is 'correct'. I have not seen this before. NU-PRL will allow actual data to be supplied enabling conventional execution, or expressions can be given whence execution is symbolic.

Constable views his system as a problem solving tool. The theorem is a statement of the problem. The solution is the 'extracted' program and the proof is an explanation (cf expert systems jargon) of why the solution is guaranteed (because of proof) to solve the problem. The window based tree browsing is to allow the user to navigate around the proof as part of the explanation process.

3. EDITOR GENERATOR

Tim Tietelbaum has built an editor generator.

Given grammars describing

1. concrete input text

2. abstract syntax tree

3. pretty printing of abstract syntax tree to screen representation and output device

4. attributed for abs. syn. tree which are to be kept invariant by the editor ie uses attribute grammars.

The system allows

1. input into an edit buffer so can type in anything (this can be changed to be as small or large a part of the tree as required)

2. context free grammar is enforced as soon as user triggers it (usually unknowingly like space at end of word or carriage return, mouse click etc). User cannot proceed if error found so syntax always must be correct

3. attribute processing then takes place. Attribute invariants violation causes warning message not stopping editing process, so can have 'undeclared variable' message and then add variable declaration to rectify situation.

Editing has template driven mechanism which is context sensitive.

I saw it running on 4MByte SUN2 and had demonstration of Pascal, Guarded Commands Language with theorem proving of pre and post conditions, word processor in which chapters, paragraphs, sentences etc were explicitly in abstract syntax tree, and Tim Griffins work (see below). The editors ran very well with good response.

This is obviously a quick and easy way to produce editors for VDM, Z etc. Much better than the VDM editor I saw at the Alvey Conference. Cliff Jones has a copy of the editor generator. It is about 20,000 lines of C.

4. Tim Griffin

Tim a PhD student of Constables. He has visited Edinburgh recently. Tim spans the Tietelbaum/Constable groups. He built a proof editor with the Editor Generator. He is working on a language (Logical Framework with Edinburgh) in which to describe the syntax of logics and the type checking and inference rules (as attributes) thus enabling the generation of proof editors. I did not understand all of the theory but it clearly is very advanced work.

Tim's work probably could be used to produce an editor/type checker/proof editor for VDM and Z type languages.

5. Super Computers

Ken Wilson, a Nobel Laureate, has won for Cornell one of the Supercomputer Initiative Contracts. (Cornell is a big place for Physics). Cornell will have a production machine (an IBM vector processor?) and an experimental project (Hypercube).

seg5