Discussion with Rod Burstall about precompilation (30/6/83)

Rod looked briefly at the second draft, and we discussed three points:

(1) Should there be a directive

spec abstype {tyvar_seq}tycon and ... and {tyvar_seq}tycon
with {op}id:ty and ... and {op}id:ty ;

? Rod liked this because it allows some structure to be expressed in the directive. I had deliberately omitted it, requiring it to be done by

spec type {tyvar_seq}tycon and ... and {tyvar_seq}tycon ;
spec val {op}id:ty and ... and {op}id:ty ;

My reason was that (a) the spec val directive is needed anyway, and (b) the spec abstype directive might suggest that the type and operations had to be provided by an exactly matching abstype declaration (whereas I wanted them to be provided equally well by e.g. a type declaration, or by an abstype declaration from which the required operations were derived rather than provided directly).

But the spec abstype form could be allowed as a derived form; this would help style, Rod thought.

(2) A precompilation should produce a (visible) record of the type environment exported by an external program. This record, being visible, would be a helpful summary of the compilation for the programmer; it is also necessary for other precompilations which use the first one. Furthermore, recompilations can check the record for any changes; this will indicate the possible need for recompilations of other programs which use the first one. None of this should entail any change to the proposal, as far as I can see.

(3) However, the interaction between use (on external files) and precompilation needs careful validation. I believe it causes no complications, but I admit that I haven't made myself 100% sure.

(2a) Rod pointed out that to have the compiler produce the (visible) export record, as in (2), as compared with a module facility which would naturally ask the programmer to provide it, is in the spirit of ML since the compiler usually tells you your types.