p84 v2
MJCG

Solution notes for SV1.1 v2 2000

The semantics of a language can be specified by the requirement that a given set of axioms and rules hold. This was tried, for example, with the language Euclid.

The advantage is that, assuming implementions actually meet the specification, one is guaranteed that the languages programming logic is sound. The disadvantage is that in practice it it hard to write down adequate axioms and rules for realistic languages. Indeed, Clarke showed this improssible for a sufficiently rich collection of constructs.

The method of verification conditions is a way of mechanising the proof of {P}C{Q}. It consists in first embedding assertions in C that express properties of the state whenever control reaches that point where the assertion is placed. Then a simple recursive algorithm is used to to extract a set of predicate calculus formulae -- called verification conditions -- whose truth is sufficient (but not necessary) for the truth of {P}C{Q}.

The annotation of commands with assertions and the proof of the verification conditions needs to be done manually (though in simple cases automation may be possible). The generation of verification conditions is fully automatic.

A deep embedding of L1 in L2 formalises the syntax of L1 in L2 (e.g. via a type of parse trees) and also formalised the semantics of L1 in L2 (e.g. via semantic functions). A shallow embedding encodes constructs of L1 as constructs of L2, but the encoding itself is not formalised. For example, L1 might be parsed to L2

Deep embeddings allow metatheorems about L1 to be proved inside L2, since quantifiers can range over L1 constructs, and are thus more powerful. However there is more effort to construct a deep embedding, and the host language, L2, needs to be sufficiently expressive. A shallow embedding only allows properties of particular phrases of L1 to be proved, but it is easier to implement and less demanding of L2. The point of embedding Floyd-Hoare logic in higher order logic is to interface theorem proving power (needed for proving verification conditions etc) to Floyd-Hoare logic. It also guarantees the soundness of Floyd-Hoare logic, since the embedding needs to derive its axioms and rules as derived rulesof higher order logic. Since higher order logic is used for hardware verification, embedding a programming logic in it provides a platform for the verification of mixed hardware-software systems.

Commercial uses: the obvious things to mention are safety, security and socially critical systems (e.g. nuclear reactors, military and e-business protocols and voting systems). However, I will give high marks for imaginative suggestions.