References

[1] A. N. Habermann and D. Notkin. Gandalf: Software development environments. *IEEE Transactions on Software Engineering*, SE-12(12):1117–1127, December 1986.

[2] T. Reps and T. Teitelbaum. Language processing in program editors. *Computer*, 20(11):29–40, November 1987.

[3] T. Reps. *Generating Language-Based Environments*. The MIT Press, Cambridge MA, 1984.

Box B: Structure-Oriented Environments

Structure-oriented environments consist of a collection of toolies that share structured (as opposed to textual) representations of program objects. Toolies in these environments typically share data represented as an attributed abstract syntax tree, whose form is defined by a grammar consisting of a collection of context-free productions.

Toolies are usually associated with these shared abstract syntax trees using action routines or using attribute grammars. An action routine is a procedure, associated with a production in a grammar, that defines the actions of some toolie on instances of that production. In the Gandalf system [1], action routines are written in a special purpose programming language called ARL, which is oriented towards manipulating abstract syntax trees. Each action routine is associated with an operation on an abstract syntax tree node, such as `create` or `delete`, which indicates when it should be invoked. An alternative to action routines is to associate a set of attribute equations with each production in the grammar; this is done in the Synthesizer Generator [2]. Roughly speaking, an attribute equation defines the value of an attribute of an abstract syntax tree node as a function of the attribute values of adjacent nodes in the tree. Attribute equations determine a graph of data dependencies that can be used to incrementally reevaluate attributes when an editing toolie changes the shared tree [3].

In both approaches, control is event-driven. In the first case, toolies written as action routines are automatically activated when data is manipulated by the operation associated with the routine. In the second case, toolies written as attribute equations are automatically invoked when updates are made to the data on which the attribute values depend. Enhancement is simplified in both cases, since typically the new function can be added as new action routines or attribute equations that are logically distinct from the existing ones.