Afterword

Building Beautifully

William J. Mitchell

Loose analogies are often made between software systems and works of architecture. But the structural similarity between these different sorts of systems is actually closer, and more rigorously specifiable, than it might seem.

The code comprising software systems consists of one-dimensional strings of symbols, put together from well-defined vocabularies in accordance with precise syntactic rules, and intended to produce useful results when run on appropriate machines.

Works of architecture obviously aren't one-dimensional, but otherwise they are very much like code. They are three-dimensional assemblies of discrete physical components, put together from fairly well-defined component vocabularies in accordance with reasonably rigorous syntactic rules, and intended to serve useful purposes. (Architects do, in practice, have a little more latitude with vocabulary and syntax than programmers.)

In both cases we can write formal grammars to establish the rules of the game. In general, formal grammars tell you how to put things together. More precisely, according to the standard definition used in linguistics and computer science, a formal grammar consists of: a finite set N of nonterminal things; a finite set T of terminal things; a finite set R of replacement rules; and an initial thing S. A replacement rule has an assembly of things on its left side, an arrow in the middle, and another assembly on its right side. It specifies that you can replace the