SNODGRASS83
Snodgrass, R.;
An Object Oriented Command Language;
IEEE Transactions on Software Engineering, SE-10,1 (January 83)

This paper describes Cola, an object-oriented command language for a capability based operating system named Hydra that runs on C.mmp, a tightly coupled multiprocessor.

In the design of Cola, a lot of effort was expended to mirror the philosophy of the operating system in the command language. The incorporation of objects, similar to Simula classes, in the command language was a result of this objective. An analogy between the development of cammand languages and general purpose languages suggests that the concept of class might be a valuable addition to command languages. According to the author, and operating system is merely a large, complex runtime system for the user's program. The concepts introduced in programming languages tend to be transferred to the runtime systems, as well as the operating systems which support them.

The first part of the paper examines the correspondence between objects in Hydra and objects in Cola. Since the operating system is the one entity shared by the users of the various languages, and it is the operating system that is providing the services that the command language refers to, it is appropriate to align the command language as closely as possible with the operating system.

Following overview of the Hydra operating system and the concepts of objects and message passing, the Cola/Hydra correspondence is discussed. Every object (capability) in the user's environment in Hydra is associated with an object in the user's environment in Cola. The design of Cola ensures that there is no distinction between Hydra's objects (capabilities) and Cola objects, resulting in an isomorphism between the two entities. The author claims that the object concept in Cola can be usefully incorporated into command languages for operating systems that are not themselves based on the object model.

Cola objects are useful not only as command language surrogates for objects (capabilities), but also have many properties that make them useful in models of computation and in personal computer languages. Since an object is an active piece of knowledge, one aspect of the research considered structuring objects in ways that have been found to be useful in structuring knowledge. Cola uses a hierarchical ordering of classes coupled with and execution semantics and binding mechanism to represent static and dynamic knowledge within the class structure.

The Cola subclassing mechanism has been designed to avoid the disadvantages of Simula's subclassing mechanism while retaining its advantages. The paper has sections describing naming of class and instance variables, instances, execution semantics and the binding mechanism used in Cola, automatic inheritance and shadowing.

Several conclusions are drawn from the research effort. A useful

correspondence between the entities supported by the command language and those supplied by the operating system has been achieved. It is argued that the Cola paradigm can be successfully incorporated into a command language for a conventional operating system, although this premise has not been demonstrated with an implementation. The object hierarchy, coupled with the message-passing and message-forwarding mechanism, has been shown to possess several useful attributes concerning the structuring of objects.

The paper concludes by suggesting several areas where additional research is needed. This paper should be of interest to the designers of operating systems and languages.