Schematic: A Concurrent Object-Oriented Extension to Scheme

March 7, 1996

This page is still under construction. Stay tuned.

1 What's Schematic?

Schematic is a concurrent object-oriented extension to the programming language Scheme. Any lambda expression (hence user-defined function, toplevel or internal) can be called both by a normal sequential invocation and a *future* invocation. Omitting some important details, concurrency primitives and their syntax are similar to those in Multilisp.

Schematic also introduces *concurrent objects*. A concurrent object is data structure safely sharable among concurrent processes. Simaltaneous accesses to a single concurrent object is arbitrated, so that a method invocation can be regarded as an atomic transaction. Unlike many early concurrent object-oriented languages (especially those based on Actors), however, a concurrent object in Schematic does not serialize every method invocation and allows much more concurrency than those languages. An introductry technical paper (Letter size (US etc.), A4 (Japan)) is available.

Right now, its portable prototype on top of `Scheme->C` compiler is operating on single processor workstations and AP1000 parallel computer. In addition, an experimental implementation has been done on network of Sun Workstations. In the future, we are going to support those running on symmetric multiprocessor (SMP), the Parallel Virtual Machine (PVM), the Message Passing Interface (MPI), Myrinet, and workstations which talk TCP/IP. We will make it publically available soon (by summer 1996).

We incorporate Tk library into Schematic. The resulting tool is very close to STk, but each widget now acts as a *concurrent object* which can be distributed over the network. An experimental environment on SUN clusters is operating and we call it SchematicTk. Our experience so far indicates that SchematicTk makes it substantially easier to build distributed GUI applications. We have built several demonstration programs with this respect Schematic Demonstration (at ISOTAS'96).

2 Related Home Pages

- STk— Scheme with Tk toolkit.

3 Publications

References

[TMY93] Kenjiro Taura, Satoshi Matsuoka, and Akinori Yonezawa. An efficient implementation scheme of concurrent object-oriented languages on stock multicomputers. In *Proceedings of the ACM SIGPLAN Symposium on Principles & Practice of Parallel Programming PPOPP*, pages 218–228, 1993. A4 Postscript, Letter Size Postscript.

[TMY94a] Kenjiro Taura, Satoshi Matsuoka, and Akinori Yonezawa. ABCL/f: A future-based polymorphic typed concurrent object-oriented language – its design and implementation –. In G. Blelloch, M. Chandy, and S. Jagannathan, editors, *Proceedings of the DIMACS workshop on Specification of Parallel Algorithms*, number 18 in Dimacs Series in Discrete Mathematics and Theoretical Computer Science, pages 275–292. American Mathematical Society, 1994. A4 Postscript, Letter Size Postscript.

[TMY94b] Kenjiro Taura, Satoshi Matsuoka, and Akinori Yonezawa. *StackThreads*: An abstract machine for scheduling fine-grain threads on stock CPUs. In *Proceedings of Workshop on Theory and Practice of Parallel Programming (TPPP)*, number 907 in Lecture Notes in Computer Science, pages 121–136. Springer Verlag, 1994. A4 Postscript, Letter Size Postscript.