CONCURRENCY AND DISTRIBUTION

Sixth International Workshop on Software Specification and Design

Como, Italy, 25-6 October, 1991

H. Conrad Cunningham
University of Mississippi

Jan Tijmen Udding
Groningen University

1 The Focus

Many of the challenging problems in software specification, design, and implementation arise from the interactions of the concurrent components of a system. The 6IWSSD Concurrency and Distribution working group sought to understand these problems, explore solutions proposed from diverse perspectives, and synthesize proposals for further research. The sessions attracted a talented and diverse group of concurrency researchers from six different countries.

The working group focused its attention upon "the router problem" described below. Although this problem was not difficult to grasp informally, devising an elegant solution was a nontrivial task. The problem description was distributed as a part of the workshop's advance program. All participants applied the methods and notations arising from their research to the problem and presented (at least partial) solutions during the sessions. The problem and the various solutions put forward stimulated an interesting and informative (occasionally heated) exchange of ideas.

2 The Router Problem

Consider a communication network with M input ports and N output ports. One sender is connected to each input port and one receiver to each output port. From time to time a sender sends a message to one of the receivers via the network—perhaps a different receiver for each message. A message consists of three parts. A special token, called the header, marks the beginning of a message and identifies the intended receiver for that message. Another special token, called the trailer, marks the end of message. The third part contains the actual data. A sender presents the network with one message at a time—first sending the header, then the data, and finally the trailer. Likewise, the network presents the intended receiver with one message at a time. Using an appropriate method and notation, develop a specification (or high-level prototype) of this network.

Now consider a refinement of this problem. The data part itself consists of a sequence of tokens which is sent to or received from the network one token at a time. The order of the tokens is important and, therefore, must be maintained by the network. Moreover, the data tokens within the sequence do not contain any information which identifies the message. Hence, the tokens of multiple messages cannot be interleaved on the communication ports connected to the senders and receivers. Again using the chosen method and notation, develop a second specification (or more detailed prototype) of the network. Show (e.g., prove formally) that the second specification implements the first.

Taking it one step further, consider the following refinement. The network consists of an M-by-N grid of switching elements. Each sender sends its messages to a different row of the grid and each receiver receives its messages from a different column. The tokens of a message travel along a row until the destination column is encountered, then travel along the column toward the receiver. Give a specification of the switching elements and show that their composition implements the second specification.

This problem was adapted from a "real-world" problem reported in D. May and P. Thompson's "Transputers and Routers: Components for Concurrent Machines" which appeared in *Proceedings of the Third Transputer/Occam International Conference* (Tokyo 1990, editors T. L. Kunii and D. May, IOS Press).

3 The Sessions

The Concurrency and Distribution track consisted of three sessions, each lasting three hours. During the first session, the participants gave brief tutorials on their methods and notations. Most used the router problem as a example to illustrate their methods. In subsequent sessions, the group examined the various solutions to the router problem in more detail, compared them, and discussed the issues that arose.

J. T. Udding of Groningen University and C. Cunningham of the University of Mississippi coordinated the sessions. The formalisms presented can be grouped as follows:

1. high-level Petri nets in combination with algebraic specifications for abstract data types (the SEGRAS model by B. Kraemer of GMD, OBJSA Nets by F. De Cindio of the University of Milan, and AT-Nets by M. Bettaz of the University of Constantine)
2. shared dataspace models (the Polis coordination model by P. Ciancarini of the University of Pisa and the Swarm notation and logic by C. Cunningham)
3. graph-rewriting models (Δ-grammars by S. Kaplan of the University of Illinois)
4. process algebras (a CCS solution to the router problem by R. De Nicola of the University of Rome)
5. constrained expressions (a CEDL investigation of the router problem by U. Buy of the University of Illinois at Chicago)
6. state machine models (a minimalist approach to specification by P. Lohr of the Free University of Berlin)

The advance distribution of the problem description was a key factor in the effectiveness of the sessions. Most participants had already grappled with the problem before they arrived at the workshop; the sessions could thus be devoted to understanding the various formalisms, comparing their expressiveness, and discussing the underlying issues.

4 The Issues

The bulk of the discussion dealt with specifications. Although all presenters gave specifications of some sort, most were expressed as high-level abstract programs rather than as predicates or similar concepts. No one verified the correctness of his or her specification. Although few of the models supported refinement of programs formally, all used refinement informally, implicitly acknowledging its importance for program design. Most of the formalisms also addressed safety and deadlock issues. Most of the participants considered this to be desirable. However, one group member argued that the model should be no more complicated than required by the problem at hand. (For example, if deadlock is not an issue in a situation, then a model which addresses deadlock may cause unnecessary work to be done and errors to be made.) Many of the group members agreed that specifications should be executable; others felt that requiring executability would result in over-specification, particularly at high-levels of abstraction.

The participants considered analysis tools useful, even when the tools are not totally accurate. The analyses should yield an accurate result within a reasonable period of time. However, if the correct answer cannot be found within the time constraints, the tool should always err toward "pessimistic" answers.

Scaling-up continued to be a problem with virtually all of the formalisms, especially the Petri net approaches. There was some optimism, however, that solutions can be scaled up for some classes of problems.

Several issues generated considerable debate but little consensus. Pictorial languages were very much alive and fiercely defended by their advocates. Much the same can be said for the message-passing and shared dataspace paradigms; no consensus on their relative merits emerged from the discussion. Additional comparative studies are needed.

5 The Future

Concurrency is a research area in ferment. No single paradigm has bubbled to the surface or is likely to do so in the near future. We need more face-to-face encounters, forums such as these sessions in which alternative solutions to a nontrivial problem are considered and the competing formalisms are critiqued from pragmatic as well as theoretical perspectives. Perhaps the free exchange of ideas will cause a new and better paradigm to coalesce and break through to the surface.