Description of Research Projects

Thomas Kunz

1 Management of Distributed Applications

The goal of the Mandas project is to create a prototype environment for designing, building and managing distributed applications. It is funded by the IBM Centre for Advanced Studies and the Natural Sciences and Engineering Research Council of Canada (NSERC) and involves researchers from the University of Waterloo, the University of Western Ontario, Queens University, Carleton University, and the University of Saskatchewan. Successful prototypes were demonstrated at both Cascon'95 and Cascon'96, References [14, 15] describe our goals and approaches. Our experimental testbed is a multi-cell DCE environment, the main contributions of our research group focus on providing the infrastructure to monitor and control the execution of distributed applications. We also developed a debugging tool (see [10]) for DCE applications, discussed below.

2 Application Design for Mobile Applications

In a co-operative research project with the Motorola Advanced Radiodata Research Centre, we are examining a number of issues related to client-server computing in a wireless and mobile environment. These activities led to a 2-day short course for industry participants on all areas of wireless and mobile computing, covering topics such as TDMA vs. CDMA, AMPS and CDPD, GSM, mobile IP, TCP over a wireless link, etc., that I taught in March 1997. The course notes are available on the WWW: `http://ccnga.uwaterloo.ca/~tkunz/mobile/summary.html`. The main focus of our work, as described in [22, 24, 25], is on the design of adaptive mobile applications, mobile applications that execute reasonably well in widely different networking environments. Outdoor cellular systems such as GSM provide data rates of less then 20 kbps, indoor wireless LANs such as WaveLAN operate at 1-2 Mbps, and a docking station connected to the Ethernet provides a 10 Mbps connection. Rather than executing different and specialized binaries for each network scenario, we aim at developing applications that dynamically adapt to the available bandwidth. We design mobile applications as a collection of (potentially mobile) objects. Our API provides information about network statistics, and the runtime system uses this information, together with application-specific information (number and size of objects, for example), to dynamically distribute the objects between the portable device and a proxy server (a machine in the wired network close to the portable device). This dynamic partitioning potentially allows to overcome the limitations of the portable device (such as a small PDA) and to adapt to changing bandwidths by the same executable. To demonstrate the feasibility of this approach, we designed and implemented a mobile e-mail browser. We currently compare its performance to the performance of various fixed partitioning schemes under different bandwidths.

3 WWW Browsing over a Wireless Link

Given the predominance of the WWW traffic in the Internet, and Industry Canada's vision that mobile devices will constitute the "on-ramp" to the national information infrastructure, special attention should be given to WWW browsing in a wireless environment. We are currently examining issues related to prefetching HTML documents to a proxy server to reduce the apparent latency of retrieving documents from remote servers. Since the wireless link is considered a scarce resource, we do not prefetch all the way to the browser executing on the mobile device. A number of prefetching strategies have been identified and are partially implemented. We collected extensive trace data from a number of Netscape users and built a test environment that allows us to examine the effects (individually and combined) of various prefetching strategies in a controlled manner. The necessary tests and their analysis are expected to be completed by the end of the summer. The results of one particular prefetching strategy, prefetching DNS lookups, will be presented at Wireless'97 in Calgary (see [23]).

4 Location Management in Cellular Systems

Cellular systems such as GSM need to keep track of the location of a mobile subscriber to route incoming connection establishment requests. The solution proposed in the GSM standard statically partitions the cell layout into location areas and keeps track of location at this level of granularity. The same tiling of location areas is applied to all mobile subscribers, and location update traffic is confined to border cells of a location area. We believe that mobile subscribers show clearly identifiable patterns of mobility and proposed a location management scheme that utilizes individual mobility patterns to reduce location update and paging messages. To implement our algorithm, only a few kBytes of additional memory and very little processing on the SIM (subscriber identity module) are required to build and maintain the user profile. Also, the VLR (visitor location register) needs to store a list of cells comprising the personal location area, instead of a simple location area identifier. Using statistical data collected by the Region of Waterloo for traffic planning purposes, we developed a subscriber mobility model, implemented a simulation, and compared our algorithm to a number of variants of the GSM location management algorithm. The results show that our proposal outperforms the static location management algorithms in terms of location update and paging costs. It also spreads the location update traffic more evenly across the network cells. A number of papers describing the mobility model and the results of our simulation are currently under preparation.

5 Parallel and Distributed Debugging

Our research group has been building a parallel and distributed debugger, named Poet. This tool provides process-time diagrams (or message sequence charts) of partially ordered parallel or distributed executions, where the time dimension is either based on the real time of event occurrence, or a model of logical time (the precedence relation introduced by Lamport) widely used in reasoning about causality in a distributed system. The novel features of Poet are its target-system independence (we use it to visualize the execution of applications written for DCE, PVM, SR, ABC++, μC++, and others) and its unique abstraction features in both the process and the time domain, which allow for execution visualizations at higher abstraction levels. These features are described in [1, 8, 10, 13, 17, 26]. Poet also supports more traditional debugging operations such as *single-stepping* (see [9]), breakpoints, or deterministic replay. In addition, a traditional sequential debugger can be controlled through Poet for individual processes.

6 Reverse Engineering Distributed Applications

To fully utilize Poet's abstraction facilities, appropriate abstractions should be derived automatically. The research described in [3, 4, 5, 16, 18, 19, 20, 26, 27, 28] addresses this problem by applying *reverse engineering* approaches, originally developed for sequential legacy systems, to distributed applications. Based on a static source analysis, an analysis of the trace data describing the execution behaviour, and some rules similar to object-oriented architectural design patterns (see [12]), *clusters* (abstractions in the process domain) and *abstract events* (abstractions in the time domain) are derived automatically. To resolve ambiguities, I developed a quantitative cluster evaluation measure (see [11]). The resulting abstraction hierarchies form a tree structure, a user of Poet can navigate and manually modify these hierarchies. As described in [4], such abstraction hierarchies are not only useful in analyzing and understanding a distributed application for debugging purposes, but for more general software maintenance tasks as well. References [2, 16] study the properties of the behavioral abstractions (in the time domain) and contrast them to other proposals.

7 Distributed Objects in C++

As part of an earlier research project, I developed a system that allows the transparent distribution of C++ objects in a network of workstations. I modified the GNU C++ compiler to collect information about C++ classes, methods, and their parameters, developed a stub compiler to automatically generate code that hides the remote method invocation (marshaling parameters, handling object bindings, etc) from the caller of a method, and implemented runtime support in the form of a nameserver and a load balancer. This system is briefly described in [21].

8 Load Balancing in Distributed Systems

I worked on a load balancing scheme for a network of Sun workstations, sharing a common distributed filesystem. The load balancing scheme used heuristics to dynamically make decisions about where to execute incoming jobs. One of the crucial parameters in any load balancing scheme is the characterization of the workstation workload. The respective insights gained and the resulting performance are reported in [6].

References

[1] Thomas Kunz, James P. Black, David J. Taylor, and A. A. Basten. POET: Target-system-independent visualizations of complex distributed-application executions. To appear in *Computer Journal*, special issue on Software Engineering for Distributed Systems, November 1997 (tentatively).

[2] Thomas Kunz. High–level views of distributed executions: convex abstract events. In *Journal of Automated Software Engineering*, special issue on Automated Debugging, vol. 4, no. 2, pages 179–197, April 1997.

[3] Thomas Kunz and James P. Black. Using automatic process clustering for design recovery and distributed debugging. In *IEEE Transactions on Software Engineering*, vol. 21, no. 6, pages 515–527, June 1995.

[4] Thomas Kunz and James P. Black. Understanding the behaviour of distributed applications through reverse-engineering. In *Distributed Systems Engineering Journal*, vol. 1, no. 6, pages 345–353, December 1994.

[5] Thomas Kunz. Reverse engineering distributed applications: an event abstraction tool. In *International Journal of Software Engineering and Knowledge Engineering*, vol. 4, no. 3, pages 303–323, September 1994.

[6] Thomas Kunz. The influence of different workload descriptions on a heuristic load balancing scheme. *IEEE Transactions on Software Engineering*, vol. 17, no. 7, pages 725–730, July 1991. Reprinted in *Scheduling and Load Balancing in Parallel and Distributed Systems*, pages 497–502. IEEE Computer Society Press 1995, ISBN 0-8186-6587-4.

[7] Hans-Christian Pfohl, Norbert Linn, and Thomas Kunz. Zum Problem der Unabhängigkeit der Determinanten von Reorganisationsprozessen. *Journal für Betriebswirtschaft*, pages 220–243, Heft 4-5, 1989. (English translation: Some comments on the independence of success factors in reorganisation processes, in *Journal of Business Science*.)

Conference Publications:

[8] Thomas Kunz, David J. Taylor, and James P. Black. POET: Target-system-independent visualizations of complex distributed executions. In *Proceedings of the 30th Hawaii International Conference on System and Science*, Volume I, Maui, Hawaii, USA, pages 452–461, January 1997.

[9] Marc Khouzam and Thomas Kunz. Single stepping in event-visualization tools. In *Proceedings of the 1996 CAS Conference*, IBM Canada Ltd. Laboratory, Centre for Advanced Studies, Toronto, Ont., Canada, pages 103–114, November 1996.

[10] David J. Taylor, T. Kunz, and James P. Black. A tool for debugging OSF DCE applications. In *Proceedings of the 12th Annual International Computer Software and Applications Conference*, Seoul, Korea, pages 440-446, August 1996.

[11] Thomas Kunz. Evaluating process clusters to support automatic program understanding. In *Proceedings of the 4th Workshop on Program Comprehension*, Berlin, Germany, pages 198–207, March 1996. ISBN 0-8186-7283-8.

[12] P. S. C. Alencar, D. D. Cowan, T. Kunz, and C. J. P. Lucena. A formal architectural design patterns-based approach to software understanding. In *Proceedings of the 4th Workshop on Program Comprehension*, Berlin, Germany, pages 154–163, March 1996. ISBN 0-8186-7283-8.

[13] David J. Taylor, Thomas Kunz, and James P. Black. Achieving target-system independence in event visualisations. In *CD-ROM Proceedings of the 1995 CAS Conference*, IBM Canada Ltd. Laboratory, Centre for Advanced Studies, Toronto, Ont., Canada, 10 pages, November 1995.

[14] J. A. Rolia, C. M. Woodside, V. Vetland, R. B. Bunt, D. L. Eager, M. A. Bauer, J. W. Hong, H. L. Lutfiyya, J. P. Black, T. Kunz, D. J. Taylor, T. P. Martin, T. J. Teorey, and P. J. Finnigan. Distributed application management, the MANDAS project. In *Proceedings of the Sixth IFIP/IEEE International Workshop on Distributed Systems: Operations and Management*, Ottawa, Canada, 7 pages, Oct. 1995.

[15] M. A. Bauer, H. L. Lutfiyya, J. W. Hong, J. P. Black, T. Kunz, D. J. Taylor, T. P. Martin, R. B. Bunt, D. L. Eager, P. J. Finnigan, J. A. Rolia, C. M. Woodside, and T. J. Teorey. MANDAS: Management of Distributed Applications and Systems. In *Proceedings of the Fifth IEEE Computer Society Workshop on Future Trends of Distributed Computing Systems*, Cheju Island, Korea, pages 200–206, August 1995.

[16] Thomas Kunz. High–level views of distributed executions. In *Proceedings of the 2nd International Workshop on Automated and Algorithmic Debugging*, St Malo, France, 15 pages, May 1995. Also available via WWW,
http://www.irisa.fr/EXTERNE/manifestations/AADEBUG95/program.html.

[17] Thomas Kunz. Visualizing abstract events. In *Proceedings of the 1994 CAS Conference*, IBM Canada Ltd. Laboratory, Centre for Advanced Studies, Toronto, Ont., Canada, pages 334-343, November 1994. Winner of the best Ph.D. student paper award.

[18] Thomas Kunz and James P. Black. Abstract debugging of distributed applications. In *Proceedings of the IFIP WG10.3 Working Conference on Programming Environments for Massively Parallel Distributed Systems*, pages 353–358, April 1994. Birkhäuser Verlag, Basel, Switzerland, ISBN 3-7643-5090-3.

[19] Thomas Kunz. Issues in event abstraction. In *Proceedings of the 1993 Conference on Parallel Architectures and Languages Europe* (PARLE), Munich, Germany, pages 668–671, June 1993.

[20] Thomas Kunz. Process clustering for distributed debugging. In *Proceedings of the Third ACM/ONR Workshop on Parallel and Distributed Debugging*, San Diego, California, pages 75–84, May 1993.

[21] Dan C. Marinescu, John R. Rice, Bernard Waltsburger, Catherine E. Houstis, Thomas Kunz, and Helmut Waldschmidt. Distributed supercomputing. In *Proceedings of the Second IEEE Workshop on Future Trends of Distributed Computing Systems*, Cairo, Egypt, pages 381–387, September 1990.

Other Workshops:

[22] Thomas Kunz, James P. Black, and Michael Nidd. A nomadic application support interface. To be presented at NOMADIC '97, Santa Clara, CA, USA, August 1997.

[23] Michael Nidd, Thomas Kunz, and Ertugrul Arik. Prefetching DNS Lookup for Efficient Wireless WWW Browsing. To be presented at the Ninth Annual International Conference on Wireless Communications, Calgary, Alberta, Canada, July 1997.

[24] Grace Hai Yan Lo and Thomas Kunz. A case study of dynamic application partitioning in mobile computing — an e-mail browser. Discussion paper, presented at the OOPSLA'96 Workshop on Object Replication and Mobile Computing (ORMC'96), San Jose, CA, 6 pages, October 1996. Available via WWW, `http://www.diku.dk/distlab/workshops/ormc96/`

[25] Michael Nidd, Thomas Kunz, and James P. Black. Wireless application and API design. In *Proceedings of the Fourth International IFIP Workshop on Quality of Service*, Paris, France, pages 83–85, March 1996. Appeared as GMD-Studien Nr. 282, Quality of Service – Description Modelling and Management.

[26] Thomas Kunz. Automatic support for understanding complex applications. In *Proceedings of the 1995 International Workshop on Network and Systems Management*, Kyongju, Korea, pages 125–132, August 1995.

[27] Thomas Kunz and David J. Taylor. Visualizing PVM executions. In *Proceedings of the 3rd PVM User's Group Meeting*, Pittsburgh, PA, USA, 13 pages, May 1995. Available via WWW, `http://www.cs.cmu.edu/Web/Groups/pvmug95.html`.

[28] Thomas Kunz and David J. Taylor. Distributed debugging using a reverse–engineering tool. In *Proceedings of the 3rd Reverse Engineering Forum*, Burlington, Massachusetts, September 1992.