6

Quality of service and scientific workflows

Mladen A. Vouk and Munindar P. Singh
North Carolina State University
Department of Computer Science, Box 8206, Raleigh, NC 27695,
USA. Email: vouk@adm.csc.ncsu.edu, singh@ncsu.edu

Abstract
The advent of high-performance computing engines and networks is bringing serious numerical and problem-solving environments (PSEs) closer to a broad base of users with widely differing needs. From the perspective of these users, a key issue will be the **quality of service** (QoS) PSEs offer. In the broader sense, QoS includes parameters such as network delays and throughput, as well as end-user quality factors such as system availability, system functionality, content quality, and semantic interoperability. In order to facilitate integration of the QoS and PSE we introduce **scientific workflows**, to mean a series of structured activities and computations that arise in scientific problem-solving. Scientific workflows are expected to coexist and cooperate with other user workflows (e.g., business workflows, educational workflows, legislative workflows). As such they must support compatible QoS. We use data from existing systems and workflows to quantitatively bound some of the PSE QoS parameters. Use of multimedia imposes additional restrictions, while end-user risks impose bounds on the security and reliability of numerical computations and algorithms. It is our belief that the next generation of PSEs must have QoS parameters designed into the system, or these PSEs will fail to live up to user needs and expectations.

Key Words
Quality of service, QoS, problem solving environment, PSE, scientific workflow, reliability, availability, intra- and inter-net response delays, users, activity management.

1 INTRODUCTION

Modern problem solving environments (PSE) are envisioned as collections of cooperating programs, tools, clients, and intelligent agents [Gallopoulos et al., 1994]. These components are integrated into an environment that facilitates user interaction (such as problem statement and solution engineering) and cooperative execution of the components charged with the solution tasks. An example is a system that would help an environmental scientist or a regulator to pose environmental engineering questions (problems), develop, execute and validate solutions, analyze results, and arrive at a decision (e.g., cost-effective emission control strategy). Such a PSE would consist of a management, analysis and computational framework that would be populated with a variety of models and data that describe the science behind the phenomena, the solutions of interest and the decision rules [Dennis et al., 1996]. Modern PSEs are naturally distributed; new technologies that "guarantee quality of service" are thus especially attractive.

From the perspective of a PSE user, one of the key issues is the **quality of service** (QoS) a PSE can offer. In this context we broaden the classical definition of QoS to include not only network-based parameters (such as response delay and throughput), but also measurable end-user quality characteristics such as system availability, performance, algorithmic scalability, effectiveness, quality of system content, quality of user-system interactions, and semantic consistency. Furthermore, in order to facilitate integration of the QoS and PSE concepts, and naturally introduce already existing formal specification and quality analysis approaches, we view PSEs as computer and network-based systems that support **scientific workflows**, i.e., a series of structured activities and computations that arise in scientific problem-solving. Scientific workflows are expected to coexist, cooperate and even meld with other user workflows (e.g., business workflows, educational workflows, legislative workflows). As such they must support compatible QoS. We can use data from existing network-based systems and workflows to quantitatively bound some of the PSE QoS parameters.

Section 2 defines the workflow view of problem solving. Section 3 discusses the QoS issues and provides quantitative bounds for some more prominent QoS parameters. Section 4 presents some conclusions and discusses important directions for future work.

2 PROBLEM SOLVING AND SCIENTIFIC WORKFLOWS

High-performance problem-solving environments often involve complex, structured, heterogeneous, long-lived computations. For example, during modeling to support the North Carolina CAAA State Implementation Plan (SIP), over 50 control strategies were studied in an attempt to remedy North Carolina's air problems. The study resulted in about 20,000 interdependent program runs that generated over 20,000 files which needed to be moved among the computing hosts, analyzed, indexed, managed, and archived. Each day of each simulated control strategy generated about 3 gigabytes of data, for a total data volume that exceeded 1.5 terabytes. If written down as text, the complete processing specification would probably take 100,000 lines or more — over 1500 pages — too long and too detailed for effective human inspection. It is obvious that this type of scientific problem solving may not only require many different computing and networking services, but also a sophisticated self-checking specification and definition environment. Although significant advances have been made, major bottlenecks still remain in specifying and managing these computations, and in enforcing of intricate dependencies among their components [Ambrosiano et al., 1995; Dennis et al., 1996]. Appropriate abstractions and implementation support can remove these bottlenecks, enable significant productivity gains and help integrate the scientific computing into general problem solving and decision-support framework.

2.1 Workflows

Interestingly, related abstractions have been intensively studied under the rubric of *workflows.* Workflows, especially those with transactional components, have drawn much attention in the databases and information systems communities [Elmagarmid, 1992]. Considerable progress has been made both in workflow specification and scheduling systems based on agents, temporal logic, and process algebra. A number of workflow products now exist [Wainer et al., 1996]. Most research has been focused on workflows in business environments. The products too are geared to enterprise computing and enable various routine office activities to be captured and automated. There has also been considerable effort in the area of workflows involving transactions and tasks that execute in heterogeneous information systems, typically those consisting of mainframe computers with arcane interfaces [Singh and Huhns, 1994].

Viewed in a broader sense, workflows are the natural outcome of the limitations of the traditional transactional model in capturing the semantics of activities in open, heterogeneous settings. The workflow paradigm has grown to accommodate general tasks, not just transactions, and was merged with considerations from organizational theory and groupware to find use in many areas. Two examples are the office-work community (enabling human collaboration), and the general process modeling community (enabling the capture of application-specific semantics of the activities of interest). Traditional transactions are *ACID* [Gray and Reuter, 1993], which means they are: 1) Atomic: all or none of a transaction happens; 2) Consistent: a transaction takes a database from one consistent state to another; 3) Isolated: the intermediate results of transactions are never visible; and 4) Durable: the effects of a successful transaction are permanent. These are useful properties and responsible for the success of transactions. But they have inherent limitations. In a distributed, heterogeneous environment, to guarantee atomicity requires some kind of a mutual commit protocol, which can be expensive (and sometimes impossible) to execute. Furthermore, isolation requires long-lived locks on shared resources and by definition precludes cooperation. This has led to much work on the so-called extended or relaxed transaction models which relax the ACID properties in different ways [Elmagarmid, 1992].

2.2 Scientific Workflows

Interestingly, scientific problem-solving environments share many of the characteristics of current applications workflows. This motivates us to examine the synergies that might be exploited in describing and managing scientific computations and the quality of service they require. Accordingly, we introduce **scientific workflows** as abstractions to represent, reason about, program, and manage the complex activities supported by modern PSEs, and the interactions of these activities with information resources, other computational decision-support activities of all kinds, and people [Singh and Vouk, 1996a; Wainer et al., 1996]. Scientific workflows can include series of structured activities and computations that arise in scientific problem-solving, e.g., studies or experiments. Traditionally, graph-based notations, e.g., generalized activity networks (GAN) and Petri-nets, are used to represent the flow of numerical and human [Dennis et al., 1996; Ambrosiano et al., 1995; Elmaghraby et al., 1995]. These flows bear the following similarities to workflows [Singh and Vouk, 1996b]:

- Scientific problem-solving usually involves a number and variety of analysis tools.

- Semantic mismatches among the databases and the analysis tools must be handled, and their performance characteristics matched.
- Error recovery should be through *semantic rollforward.*
- Many large-scale scientific computations of interest are long-term, easily lasting weeks if not months, and they can also involve much human intervention [Dennis et al., 1996; Ambrosiano et al., 1995].
- The computing environments are heterogeneous (often including supercomputers and workstation clusters).

Current trends in scientific problem solving suggest that the quality of scientific problem solving that end-users **expect** requires not only provision of high quality numerical computing algorithms and software, but also integration of these solutions with advanced computational and networking frameworks, and with day-to-day operational environments. The workflow paradigm enables appropriate description and analysis of scientific workflows merged with other workflows, and the concomitant quality constraints.

3 SOME QUALITY (OF SERVICE) ISSUES

Advanced network-based scientific computing and problem solving is highly dependent upon the successful performance of QoS-sensitive multimedia applications. Therefore, an issue is how to guarantee QoS requirements imposed by varying mixes of transmitted voice, video, image and data that support such applications. For instance, PSEs must deal with **interoperability** problems that arise between the various local-area and wide-area network guarantee mechanisms, with the interoperability among its distributed components, and with interoperability issues that occur at the application level, such as semantic consistency when accessing different databases. In fact, in order to achieve end-user QoS guarantees, it is necessary that all interacting end-to-end entities have agreement upon the interfaces for QoS specification, for exchange of information regarding QoS requests and provisions, and for evaluation of QoS performance.

From a networking perspective the QoS is defined by measures such as *response delays, probability of loss of data, jitter,* and *throughput..* However, in the context of a modern network-based PSEs we broaden the definition to include factors such as system **reliability** and **availability**, **adequacy** and **scalability** of the system functionalities, **semantic** consistency and interoperability, the quality of delivered information and content (e.g., models, data, educational material), **efficiency**, **security**, quality of user-system interactions, and so on. Further, a good scientific workflow support system should have resource **adaptation capabilities** (as well as other forms of flexibility) that minimize the impact of resource limitations on the user. For example, the system would recognize limitations of a user resource, such as lack of audio capabilities or video bandwidth limitations, and would automatically downgrade the information transfer mode to display the textual transcript of the audio record, or display a downgraded video stream. Similarly the system might recognize user knowledge or experience profile and adapt its interfaces, and/or the amount and level of interaction (advice) and material it offers to the user, to suit. The necessity of serving many users in heterogeneous computing environments raises a number of research and engineering problems: 1) definition of the required end-user QoS and its control mechanisms; 2) development of robust QoS-sensitive interfaces, algorithms, and data formats; 3) development of distributed and responsive system elements for wide-area queries, and easy use of complex analysis tools; and 4) development of an appropriate network-based framework for user assistance and training.

In the rest of this section we addresses *some* characteristics of network-based PSEs that we believe play a prominent role in the defining (both qualitatively and quantitatively) the "quality" of services a next-generation environment for support of scientific workflows should provide. We use our experiences with modern educational and scientific systems and workflows to provide some quantitative bounds on the QoS parameters that PSEs will need to meet.

3.1 User Diversity and System Functionality

In a PSE, the most important system entity, and the principal quality driver and constraining influence is, of course, **the user**. PSE users can be classified into three main (non-exclusive) categories: system developers, authors, and research & production (R&P) users. Many system requirements derive directly from the principal user profiles.

System developers are responsible for the development and maintenance of the system framework. They develop and integrate system interfaces, administration and management software, communications and scheduling algorithms, (authoring) tools of different kinds, content access algorithms and software, and so on. They are computing specialists, require tools for system framework development, maintenance, testing and performance evaluation. ***Authors*** are developers of the PSE-specific content, such as algorithms, lessons, applications and solutions integrated into scientific problem-solving workflows by R&P users. It is essential that authors be content experts, but they should not have to be system experts. Therefore, it is important that the system *authoring tools* and interfaces are easy-to-learn and easy-to-use, and that they allow the authors to concentrate on the content development rather than struggle with the system intricacies. ***Research & production users*** are, of course, the most important users of the system. This category covers a broad spectrum. In the future, PSE users are likely to range from very sophisticated to very naive, from academic educators to high-school children, from individual researchers to technicians running routine analyses [e.g., Dennis et al., 1996; Ambrosiano et al., 1995]. However, they will all require appropriately reliable and timely delivery of the interaction results, easy-to-use interfaces, collaborative support for local and remote joint projects, help, and so on. Users may decide to use "canned" solutions, they may sample and combine existing and customize them, they may update existing studies or they may develop new studies. However, one thing is certain: the users will **not** use a PSE in isolation. They are likely to import and export data and results to and from other workflows, and they will expect a PSE to cooperate and coexist with other computer-based support systems. They may also teach and tutor using PSE facilities.

Different user categories have varying, and sometimes contradictory needs, and a "good quality" PSE must adequately support different user modes. Furthermore, it is very likely that the same person may use a PSE in different roles, so the system must be capable of distinguishing and separating these different user "personalities". For example, scientific workflows often begin as **research** workflows and end up as **production** workflows. A PSE should support and enable this transition. Early in the workflow lifecycle, there may be a need for considerable human intervention and collaboration; later workflows begin to be executed increasingly automatically. Thus in the production mode, there is typically less room for collaboration at the scientific level and the computations are more long-lived. This happens partly because of limitations of the available technology. We speculate that if true workflow technology were available to manage scientific computations, there would be a reduced push to automate everything and the quality of the solutions obtained could be improved by involving the right people at the appropriate places. Be that as it may, during the research phase, scientific workflows need to be enacted and animated far more intensively than business workflows. In this phase, which is more extensive than the corresponding phase for business workflows, the emphasis is on execution with a view to design, and thus naturally includes iterative execution. The corresponding activity can be viewed as a correlate of business process engineering. For this reason, the approaches for constructing, managing, and coordinating process models will find useful application in scientific settings, if only the main problems are cast appropriately.

We also identify two classes of user-oriented **functionalities** and issues that a "good quality" scientific workflow support system must be able to handle, in addition to the application-area specific knowledge, algorithms and solutions. The first category applies to workflows in general and it includes: i) handling exceptions and providing fault-tolerance; ii) handling a range of user capabilities, the roles of different participants, and allowing the role bindings to change; iii) declaratively specifying control and data flows; iv) automatically executing and monitoring of workflows to meet stated specifications; v) incorporating human decisions into the process; and vi) coordinating and synchronizing with other scientific and business workflows. The second category is specific to scientific workflows and includes the features required for scientific computations, but which may not be adequately addressed in traditional workflows. This category includes issues such as: i) relative uniqueness of each workflow, particularly during the research phase when there is less opportunity to use canned or "normal" solutions; ii) the ability to handle a vast number and variety of analysis tools, and interfacing to a diverse array of computational environments (including clusters of networked workstations and supercomputers); and iii) auditability of the computations when their results are used to make decisions that carry regulatory or legislative implications. The presence (or lack) of most of these functionalities is quantifiable through system parameters such as reliability, availability, efficiency, effectiveness, productivity, cost of maintenance, semantic consistency, etc. For example, exception handling and fault-tolerance reflect in system reliability and availability. A good workflow support environment provides for quantitative evaluation of its quality traits through capture of appropriate metrics and through utility tools that allow analysis and evaluation of the system quality parameters. An excellent example of a built-in facility is the system reliability interface of the NovaNET[1] computer-based education system.

[1]NovaNET is a successful low-overhead high-yield multimedia educational system that originates from the University of Illinois at Urbana-Champaign (UIUC) and serves thousands of users on a daily basis. The NovaNET system reliability and recovery measurements are collected, processed and reported automatically and

Although the quality of numerical software has drawn much attention, and reliable numerical software libraries and PSEs exist, in general, there is considerable variance in quality of numerical software that reaches a user [e.g., Hatton, 1994]. In fact, bounds on acceptable QoS parameters for PSEs are either non-existent, or are still more qualitative than quantitative. Since in the future scientific workflows, and by implication workflows that involve numerical computations, will need to at least match that of other workflows with which they are expected to interact and fuse, we can use existing information about some of these other workflows to estimate user-acceptable bounds for PSEs. In the following subsections we illustrate this using the data from NovaNET, and from EDSS[2] [Ambrosiano et al., 1995] to address three very prominent QoS needs: system availability, system throughput, and end-to-end delays.

3.2 Reliability and Availability

In addition to adequate system functionality and usability, a successful scientific workflow support system must have adequate reliability and availability, or a broad base of users will simply not use it. Availability is defined as the probability that a system will be available at any random time during its operational life. This implies appropriate system reliability and recovery rates [Jones and Vouk, 1996]. What are they? If we assume that a PSE user will be at least as discriminating and demanding as university students and educators, we can use the NovaNET data to set a lower bound on the minimally acceptable overall system reliability and availability. If we assume that a network-based system will be limited by the reliability of its network links, we can use the information on the field quality of Internet switching elements, e.g., [Jones and Vouk, 1996], to establish another type of bound. Of course, this assumes that numerical components of the system, individually and in combination, have sufficiently high reliability that they are not the limiting factor. When a PSE is used to make critical or high-risk decisions, the reliability of the numerical elements used in the computations that lead to the critical decisions may be the governing influence since their reliability should at least match the decision risk levels [Boehm, 1989]. However, in general, networks and user interfaces may play an equally important role. For example, we estimate that (before error correction) acceptable network-level packet loss rate should not exceed 0.02 to 0.1 for voice and audio interactions, 10^{-9} to 10^{-5} for images, zero to 10^{-5} for data, and 10^{-10} to 10^{-8} for full-motion MPEG video. NovaNET system measurements indicate that, once a user starts one hour of work (e.g., a lesson), to maintain reasonable user satisfaction, the probability of getting through that hour without any problems should be above 0.95 [Bitzer and Bitzer, 1973]. We expect that a good PSE would not have reliability and availability characteristics that at least match above figures. On the other hand, according to Bellcore [Bellcore, 1989], public network switching elements are expected to assure unavailability that does not exceed about 10^{-5} (about 3 min. of downtime per year). Therefore, it is reasonable to require that individual PSE elements provide reliability and error control (including exception handling, fault-tolerance, and graceful error trapping) at least at that level, and that the overall PSE reliability during its posted user access hours be at least 0.95 (this includes everything:

continuously, and system outages (or failures) include everything, from application software, through system hardware, software and network problems, to problems caused by operator errors.

[2]The Environmental Decision Support System (EDSS) is network-based problems solving and decision support system developed by MCNC North Carolina Supercomputing Center (NCSC) in collaboration with NC State.

network outages, violation of end-to-end response times, PSE system and content software failures due to algorithmic or other problems, and so on).

3.3 Bandwidth and Delays

One the most important QoS drivers is the quality of the PSE computer-human interface (CHI). A user response to a PSE, and user's capability to start and understand the interactions and absorb results, is a very strong function of its CHI. To achieve effective information transfer rates[3], we may need to use different sets of "symbols" and presentation rates - from simple characters (at several thousand bits per second), to sophisticated high-definition animations and full-motion movies (at many megabits per second). The exact mix and density of the "symbols," functions, and the content delivery modes that is most efficient remains a research issue. However, it is clear that a "quality" PSE needs to dynamically customize its CHI, and its presentation and communication modes, to match user expertise, user knowledge absorption rate, and the available computing and networking resources. Hence, PSE **throughput** requirements may vary widely. Each mode of operation of a network-based PSE has certain throughput requirements. In some cases the bandwidth needs to be provided synchronously (user waits for output), and in some cases asynchronously (batch mode), both with varying delay requirements. The principal driver in deciding what is appropriate is the problem solving workflow. In a computer-based PSE it usually takes one of the two forms:

- "TV-model" format; This is a high average bandwidth synchronous (real-time) full-motion audio/video interaction that can be found in video-conferencing, distance-teaching and video-based collaborative work, or in a large-scale real-time data acquisition effort. These exchanges can require as much as 6 to 45 megabits per second (Mbps) per session, depending on the compression mode used and the desired quality of images. Large-scale real-time data collection, such as that occurring in some medical PSEs, can be even more demanding (throughput requirements can be as high as 400 Mbps).
- "Data-model" format; This is a low to medium average bandwidth synchronous (real-time) interaction with asynchronous data transfers. In this format one expects judicious use of hypertext, animation, graphics, voice and text to adaptively deliver the material. The *synchronous* interactions in this format may be very bursty. The average required throughput may be quite low, in the range 1000 to 20,000 bits per second, peaks can be as high as 100 Kbits/s to 2 Mbits/s. Thus, real-time bandwidth-on-demand is a network feature that can greatly enhance this mode of interaction [Rindos et al., 1995]. *Asynchronous* transfers may require an even larger bandwidth range. Although in this mode the user will not wait for the response (e.g., batch job submissions to supercomputers, transfers of non-real-time visualization data), bandwidth requirements will still be lower-bounded by the overall scheduling requirements of the study being run using the PSE [Dennis et al., 1996].

In general, in a good large-scale PSE system framework "bandwidth-greedy" material is distributed in a way that conserves bandwidth and allows support of a large number of

[3] Research shows that humans cannot extract (reason about, learn) new information at the rate faster than about 20 bits/second (i.e., differentiate among about 1,000,000 "symbols" each second) [e.g., Stroud, 1967].

simultaneous users. The distribution of tasks, across the network and across resources, will depend on the task complexity, desired schedules and resource constraints. The solutions should not rule out use of any network type (wire, optical, wireless) or access mode (high-speed and low-speed). For example, a possible distribution mix for a problem that requires use of parallel computing may include a user interface task on a portable computer in the field or classroom (perhaps using wireless to the closest high-speed network drop), a visualization, a computational model running on a remote personal workstation with data on a file server, and a communicating, larger, model running on remote supercomputer(s). To combat entropy, a distributed PSE will invariably have to centralize some of its functionalities, such as material updating, master backups and system evaluation

End-to-end response **delay** can also be a big problem. Studies show that synchronous end-to-end (round-trip) delays that consistently exceed about 250 ms are often unacceptable from the user point of view when the interaction is conducted in the key-stroke-by-keystroke mode [Bitzer and Bitzer, 1973; Kauer, 1995]. Furthermore, the video, voice and animation jitter should be less than about 10 ms, and for some specific coding approaches such as MPEG, less than 1 ms. Our measurements indicate that, except over limited areas, current incarnation of the Internet is probably not an adequate medium for key-by-key interactions. An alternative to real-time interaction on the key-by-key basis is for PSE to operate in semi-batch mode where the user interface and interactions are designed in such as way that a user expects some delays (not exceeding few tens of seconds), and does not consider long responses as system failures.

Table 1: Campus and Internet response times under different traffic loads

Network	Traffic Load	Probability that Response Time is		
		Good	Acceptable	Poor*
NCSU Campus	Low	0.9963	0.0020	0.0017
	Medium	0.9889	0.0054	0.0057
	High	0.9566	0.0356	0.0078
Internet	Low	0.9682	0.0176	0.0142
	Medium	0.9502	0.0130	0.0368
	High	0.7187	0.0458	0.2355

(*) Includes lost packets.

For example, we have measured network delays on the North Carolina State University (NCSU) campus intranet, in the NC Research Triangle (about 40 miles per side) wide-area net, and over the Internet stretch between NCSU and University of Illinois at Urbana-Champaign (UIUC) [Kauer, 1995]. Table 1 illustrates the results. It shows the probability of response time for on-campus network and Internet under different loads. Assuming that a PSE application has response time of about 100 msec or better [Balay et al., 1996], the network response times under 100 msec are considered *good*, response times between 100 and 150 milliseconds are considered *acceptable*, and response times over 150 msec are considered *poor*. The results show that a well designed campus network (or intranet) can adequately support modern PSEs, but problems grow rapidly beyond campus bounds. For instance, the NCSU-UIUC Internet link was totally inadequate for interactive PSE work during high traffic time slots (e.g., midday), and was at best marginal in medium to low traffic conditions. Adequate long-distance throughput over Internet is another problem.

4 SUMMARY AND CONCLUSIONS

The concept of **meta-(super)computing** [e.g., Baker and Fox, 1996], in conjunction with the proliferation of a wide-variety of high-performance computing engines and networking technology, new network-based computing tools (e.g., PVM, MPI), and new standards (e.g., CORBA, TINA-C) acts a strong mixing factor between traditional quality characteristics of a PSE (e.g., correctness, reliability, ease of use) and its network component. While traditional PSE **architectures** tend to be centered around a single system, and possibly a single database on top of which a single workflow engine provides services to several clients, PSEs of the future will be much more distributed and diverse, and issues such as *interoperability, concurrency, fault-tolerance, scalability, availability, interoperability,* and *general performance* will become very important. Hence, have broadened the classical definition of QoS to include not only network-based parameters (such as response delay and throughput), but also the measurable end-user quality characteristics such as system reliability, timing performance, algorithmic scalability, effectiveness, quality of CHI interactions, semantic interoperability and so on.

In order to facilitate integration of the QoS and PSE concepts, we view PSEs as computer and network-based systems that support **scientific workflows**, i.e., a series of structured activities and computations that arise in scientific problem-solving. This provides us with a natural mechanisms for formal specification and quality analysis of PSEs by using already existing techniques and approaches from both workflow and networking communities. In this way, scientific workflows are to problem-solving environments what business workflows are to enterprise integration. However, while considerable progress has been made both in the implementation of complex systems of scientific computations, and in workflow specification and scheduling, there is currently no unified theory or system that formalizes scientific workflows as defined above. Scientific workflows are expected to coexist, cooperate and meld with other user workflows (e.g., business workflows, educational workflows, legislative workflows) so the quality of these scientific workflows will have to be compatible. We have used information from existing systems and workflows to define some quantitative bounds on the quality of services that cooperating PSEs will have to meet . For example, we find that, from a usability perspective, the lower bound on guaranteed system reliability should be 0.95 or better, while acceptable synchronous user-machine interactions require user-level round-trip response times that are consistently less than about 250 msec, and network-level round-trip delay times less than about 150 msec. Use of multimedia imposes additional restrictions on parameters such as jitter and data loss probability, while end-user risks impose bounds on the accuracy and reliability of numerical computations and algorithms.

It is our belief that the next generation of PSEs must have QoS parameters designed into the system, rather than coerced into the system through operational feedback, as is the case with many existing PSEs, or these new PSEs will fail to live up to both user needs and user expectations.

ACKNOWLEDGEMENTS

This research was supported in part by NSF (ACS 9418960, ACS 9696131, IRI 9624425), NASA (NAG-1-983, EPA/MCNC (94-7050-027/5-30012 and 94-7050-029/5-30816), IBM Canada (CAS) Fellowship, IBM Corp. (RTP), and CACC (61-0103/5-30456 and 83-0062/5-30781).

REFERENCES

Ambrosiano, J., Balay, R., Coats, C., Eyth, A., Fine, S., Hils, D., Smith, T., Thorpe, S., Turner, T. and Vouk, M. (1995) The Environmental Decision Support System: Air Quality Modeling and Beyond, in *Proc. U.S. EPA Next Generation Environmental Modeling Computational Methods (NGEMCOM) Workshop*, Bay City, MI.

Baker, M. and Fox, G. (1996) Metacomputing: The Informal Supercomputer, *in Proc. NSF Train the Trainer Workshop*, Cornell Theory Center, Ithaca, NY, May 6-9, 1996. (http://renoir.csc.ncsu.edu/RTCPP/HTML/Workshop2/index.html)

Balay, R., Vouk, M. A., and Perros, H. (1996) A Lightweight Software Bus for Prototyping Problem Solving Environments, *in the Eleventh International Conference on Systems Engineering*, July 9-11, 1996, Las Vegas. Eleventh International Conference on Systems Engineering, Las Vegas, 626-30.

Bellcore, *Network Switching Element Outage Performance Monitoring Procedures*, SR-TSY-000963, Issue 1, April 1989.

Bitzer, M. D. and Bitzer, D.L. (1973) Teaching nursing by computer: an evaluation study. *Computers in Biology and Medicine*, **3**, 187-204.

Boehm, B.W. (1989)*Tutorial: Software Risk Management*, IEEE CS Press.

Dennis, R.L., Byun, D.W., Novak, J.H., Galluppi, K.J., Coats, C.C. and Vouk, M.A. (1996) The Next Generation of Integrated Air Quality Modeling: EPA's Models-3. *Atmospheric Environment*, **30**, 1925-38.

Elmagarmid, A.K. (1992) *Database Transaction Models for Advanced Applications*, Morgan Kaufmann.

Elmaghraby, S.E., Baxter, E.I., and Vouk, M.A.(1995) An Approach to the Modeling and Analysis of Software Production Processes. Intl. Transactions in Operational Research, **2**, 117-35.

Gallopoulos, S., Houstis, E. and Rice, J.R. (1994) Computer as Thinker/Doer: Problem-solving Environments for Computational Science. *IEEE Computational Science and Engineering*, **1**, 11-23.

Gray, J. and Reuter, A. (1993*) Transaction Processing: Concepts and Techniques*, Morgan Kaufmann.

Hatton, L. (1994) How accurate is scientific software? *IEEE Transactions on Software Engineering*, **20**, 785-97.

Jones W. and Vouk M.A., (1996) Software Reliability Field Data Analysis, in *Handbook of Software Reliability Engineering* (ed. M. Lyu), McGraw Hill, pp. 439-89.

Kauer, P.K. (1995) An Analysis of North Carolina State University's Network Performance, M.S. Thesis, Department of Computer Science, NCSU.

Rindos, A., Vouk, M., Woolet, S., Hines, J. and Lester J. (1995) ATM Technology Enabling Educational Applications Across the North Carolina Information Highway, *in Proc. TELECOM '95 FORUM Technology Summit*, ITU, Geneva, pp. 519-22.

Singh, M. and Huhns, M. (1994) Automating Workflows for Service Provisioning: Integrating AI and Database Technologies. *IEEE Expert*, **9**, 19-23.

Singh, M. and Vouk, M. (1996a), Scientific Workflows: Scientific Computing Meets Transactional Workflows. *In Proc. NSF Workshop on Workflow and Process Automation in Information Systems: State-of-the-art and Future Directions* (ed. A. Sheth), Athens, GA, May 8-10, 1996 (http://optimus.cs.uga.edu:5080/activities/NSF-workflow/).

Singh, M. and Vouk, M. (1996b) Quality of Service and Scientific Workflows, Computer Science Technical Report, TR96-19, North Carolina State University, (ftp://ftp.csc.ncsu.edu/pub/tech/README.html)

Stroud J.M. (1967) The Fine Structure of Psychological Time. *Annals of the New York Academy of Sciences*, **138** (Art. 2, 623-31.

Wainer, J., Weske, M., Vossen, G., Bauzer Medeiros, C.M. (1996) Scientific Workflow Systems. *In Proc. NSF Workshop on Workflow and Process Automation in Information Systems: State-of-the-art and Future Directions* (ed. A. Sheth), Athens, GA, May 8-10, 1996 (http://optimus.cs.uga.edu:5080/activities/NSF-workflow/).

DISCUSSION

Speaker : M. Vouk

J. Rice : You mentioned many properties that a problem solving environment should have: on-line documentation, intelligent resource allocation on networks, high- availability, etc. You mention some PSEs you have used. Can you say what percentage of these desired properties are present in the PSEs you have used, or in other PSEs?

M. Vouk : As far as I know, there are no PSEs that posses all the properties I have talked about. The primary reason is that most PSEs available today were really designed for localized use and they run on platforms that where not developed to support quality of service (QoS) interactions and guarantees over the networks. One system that has most of the properties I discussed is NovaNET. I estimate that NovaNET has well over 90% of the desirable QoS properties. From the start it was designed to support thousands of users over a network of dedicated switched circuits and satellite links. Thus it avoids the pitfalls that come with operating systems that do not have network QoS controls and were not designed for support for large numbers of simultaneous users. NovaNET has built in continuous reliability and availability monitoring of its central and remote resources, it has network quality of service and dynamic adaptation features, it support end- user QoS protocols, it guarantees response times appropriate for key-stroke level interaction over the whole US, it has easy to use author-level support for incorporation of dynamic QoS into interactions with users (e.g., the system will check for audio resources on the user terminal and will automatically downgrade from audio to "close-captioning" on such terminals), and it has always been using "applet" type pre-loading devices to assure smooth and adequate key-stroke, audio, video and animation operation in situations where bandwidth limitations may arise.

W. M. Gentleman : You pointed out that so far Java has primarily been used to overcome bandwidth limitations. In the long run the advantage of downloading applets to specific platforms is primarily advantageous because the applet can take advantage of resources available on that platform, and can use the physical location of that platform within organizational boundaries and within the security policy of that organization.

M. Vouk : You're are right. Current use of Java applets, and similar "portable" devices, is primarily aimed at improving user-perceived quality of service through local execution of functions that otherwise would be impaired or seriously degraded by bandwidth and network delay problems, e.g., smoothness and speed of animations, and key-stroke oriented user-machine interactions. In the future, when network resources are not be a bottleneck anymore, the focus will probably change to support of functionalities that optimize use of local resources, automatic dynamic updating of client environments to provide (yet again?) a measure of fault-tolerance and independence from the network-based resources, and conform to intra-net and other local security concerns. This will require standardization of Java, or of some other widely portable environment(s). In the world of rapidly changing platforms and computing paradigms that may be a problem that will take a long time to solve in full.