Reviewer Comments and Author Responses: Optimizing a CORBA Inter-ORB Protocol (IIOP) Engine for Minimal Footprint Embedded Multimedia Systems

Aniruddha Gokhale[*]
gokhale@research.bell-labs.com
Bell Laboratories
Lucent Technologies
600 Mountain Avenue
Murray Hill, NJ 07974

Douglas C. Schmidt
schmidt@cs.wustl.edu
Dept. of Computer Science
Washington University
One Brookings Drive
St. Louis, MO 63130[†]

April 8, 1999

1 Editor's Recommendations and Our Response

Comment 1: Emphasize in the abstract, introduction, and throughout the paper (where necessary) that you are applying known techniques to minimize memory footprint for the protocol engine without compromising performance.

Our response We have renamed "Optimization Principles" to "Optimization Principle Patterns". By identifying them as patterns, we reinforce the fact that these are common solutions used to solve recurring performance problems in the context of protocols and middleware. In addition, our related work section provides detailed citations on where these optimization principle patterns have been applied previously.

Comment 2: Reviewer 2's comments about static vs dynamic memory sizes is important; you have not improved things if the protocol engine merely replaces static code/data size for dynamic data. This should be measured and discussed.

Our response There is very little dynamic memory allocation in TAO. The dynamic memory allocation that occurs in the stubs and skeletons is due wholly to the requirements of the OMG IDL$\rightarrow$C++ mapping for inout, out, and return type parameters that are varying in size. Any CORBA-compliant system will have to do this. Incidentally, TAO's general memory allocation optimizations, such as using thread-specific allocator pools, are discussed in detail in our COOTS '99 paper, which we now cite in our JSAC paper and which is available at
www.cs.wustl.edu/~schmidt/COOTS-99.ps.gz

[*] Work done by the first author while at Washington University.

[†] This work was supported in part by Boeing, DARPA contract 9701516, Motorola, NSF grant NCR-9628218, Nortel, Siemens, and Sprint.

Comment 3: Given your premise, the following logical flow would better exemplify the work: 1) size and performance prior to starting the work; 2) (smaller) size and (worse) performance after minimizing footprint; and 3) (smaller) size and (better than or equivalent to 1) performance after applying performance improvements. I realize that this may not be possible, given the order in which the work was done; if it is possible, such a rework would definitely be a great improvement.

Our response Although your suggestion is good, it is very difficult for us to reengineer the whole work. We have tried, however, to restructure the paper to convey the iterative nature of the time/space optimization process.

Comment 4: The paper is quite long (40 pages). Fully 8.5 of these pages are appendices. We would like you to omit any non-essential material from the appendices in order to shorten the paper.

Our response We have reduced the paper by over 35% (from 42 to 27 pages) by removing unnecessary Appendices, omitting non-essential material, and generally "tightening" up the writing. In addition, we are using the LaTeX2e style files for the IEEE Transactions for formatting.

2 Reviewer's Comments and Our Responses

2.1 Reviewer 1

Comment 1: A nice contribution from this perspective would have been a general discussion about the applicability of Varghese's techniques

Our response We have addressed this in our response to Editor's recommendations above. Incidentally, George Varghese will be publishing a networking textbook shortly that will contain an entire chapter describing optimization principle patterns and illustrating how they have been applied to lower-level networking software, such as TCP/IP.

Comment 2: From a particular point of view, the performance results (after optimizations) presented in the paper are not better than those of commercial environments

Our response Naturally, all performance papers inherently track moving targets since ORBs improve (and degrade!) in performance from release-to-release. We have conducted many benchmarks and documented the results in a series of papers available at
www.cs.wustl.edu/∼schmidt/corba-research-realtime.html www.cs.wustl.edu/∼schmidt/corba-research-performance.html

that illustrate how TAO performs as well, or better, than most commercial ORBs on most performance criteria, such as end-to-end latency, jitter, scalability, etc, for both SII and DII configurations. In addition, others groups have replicated our results and conclusions, e.g.,
beust.com/virginie/Benchmarks/
compares TAO, VisiBroker, Orbix, and M3.

Moreover, commercial ORBs are generally not well suited for embedded environments since the footprint of the ORB Core + Object Adapter + generated stubs/skeletons are too large. In contrast, as described in our JSAC paper, TAO's footprint is relatively small, as are the size of its hybrid compiled/interpreted stubs/skeletons. For more information on TAO's current footprint, please see www.cs.wustl.edu/∼schmidt/ACE_wrappers/docs/ACE-subsets.html

2.2 Reviewer 2

Comment 1: It is not clear what the claims are for the paper. Did the authors want to claim that IIOP is an appropriate transport for multi-media systems? Did the authors want to claim that multi-media systems can exploit header caching techniques to improve performance? Did the authors wish to claim that CORBA is an appropriate contract specification language for high data-throughput applications? Did the authors wish to point out how poor David Brownell's weekend hack of the IIOP interpretor was? or that CORBA systems are still bottlenecked by latency when argument data sizes are less than 10kbytes? ...

Our response In our revision to the paper, we have improved the motivation for why middleware like CORBA is useful for developing embedded multimedia applications. Naturally, if the ORB core and the stubs/skeletons are inefficient, higher-level services and applications will be unable to deliver the desired QoS to applications. Moreover, since the emphasis of this special issue is on "Service Enabling Platforms for Networked Multimedia Systems", we believe providing efficient infrastructure to build these applications should be a key aspect to be addressed, which is what we strive to do in our paper.

As for using David Brownell's SunSoft IIOP interpreter, just as BSD Unix (and other open-source variants) and its TCP/IP protocol suite have been the research vehicle for a large amount of research on TCP/IP optimizations, the SunSoft IIOP was the only known open-source implementation of CORBA available when we began our research. At this point, there are other excellent high-performance open-source ORBs, such as omniORB and ORBacus, which has stimulated a thriving R&D competition between ORB developers to create highly optimized ORBs. Thus, a key motivation of this paper is to capture and document the fundamental principles and patterns associated with optimizing middleware in an "easily digestible" form.

Comment 2: A significant amount of the paper focuses on the optimization techniques used to improve performance (such as header caching). There is no discussion of whether this optimization is suitable for actual multi-media applications or whether the authors optimized only a special case (i.e., what is the impact of a header cache miss; what happens if two methods are being ping-ponged,...)

Our response We agree with the reviewer's observation that this paper does not focus solely on multimedia applications. However, we believe that the key themes of this paper, i.e., reducing latency *and* memory footprint simultaneously, are essential to support embedded multimedia applications on a wide range of small-footprint devices.

In addition, we now cite another paper written by the second author and others on performance of TAO's implementation of the OMG A/V Streaming Service specification. TAO's A/V service implementation uses the optimizations described in the JSAC paper to support middleware-based distributed multimedia applications, such as video-on-demand and teleconferencing. In addition, we now also cite a number of other papers, published after this paper, that describe many other optimizations, such as perfect hashing, active demultiplexing, memory management strategies, and other concurrency and connection optimizations, that greatly improve the predictability of TAO.

Comment 3: 3/4 of the paper (p 2-5 and 7-20) are recycled (word-for-word) from the authors' HICCS January 98 paper entitled "Principles for Optimizing CORBA Internet Inter-ORB protocol performance".

Our response Actually, only a small part of the work presented in this paper is based on the HICSS paper. Since this was a conference paper, only a subset (less than 10 pages) of the current work was actually published due to page limits (perhaps you read the version of our JSAC submission on our Web site, which contained the complete text?). In general, the journal format allowed us to present our work in much greater depth, which greatly enhances our ability to motivate, describe, justify, and analyze the benefits of the optimization principle patterns.

Comment 4: the new material in the paper describes how the authors reduced the static code size of the stubs/skeletons. However, no attempt is made to claim whether this is minimal or even reasonable in space or time. For example, no attention is paid to the dynamic memory allocation size

Our response As mentioned above, there is very minimal dynamic memory allocation in TAO, i.e., only what is necessary to conform to the OMG IDL$\rightarrow$C++ mapping rules. Moreover, the use-cases of TAO now cited in the JSAC paper, such as the recent successful test-flight of the Boeing Harrier fighter airplane using TAO in its mission computer, demonstrate the predictability of TAO in embedded systems with stringent performance constraints. For more information on this test-flight, please see
www.cs.wustl.edu/$\sim$schmidt/TAO-boeing.html

Comment 5: In addition, no claims are made to any other multi-media transmission protocol or encoding format (or where half the available bandwidth is going for complex encoded data structures– padding, alignment, encoding, ...)

Our response We agree to this observation. However, we encourage you to take a look at how TAO is used to build OMG's A/V Streaming Service (www.cs.wustl.edu/ schmidt/av.ps.gz). This paper illustrates how the optimizations employed in this paper enables TAO to support distributed multimedia applications, such as video-on-demand and teleconferencing.

2.3 Reviewer 3

Comment 1: Why OO middleware? Why not just sockets? Aren't these techniques applicable in general?

Our response We have completely rewritten the paper's introduction section to describe why CORBA-based middleware is useful to build distributed embedded multimedia applications. In a nutshell, the reasons for not using sockets are the fact that programming *applications* at the socket-level is extremely tedious, error-prone, and non-portable. The liabilities make sockets unsuited for developing large-scale applications, particularly in terms of the high life-cycle costs necessary to maintain and enhance systems written at the socket level. Incidentally, our related work section now describes in detail the general applicability of the optimization principle patterns used in the paper, both for middleware and for lower-level protocols and network software.