A Critique of OpenMath and Thoughts on Encoding Mathematics, January, 2001

Richard Fateman
Computer Science Division
University of California, Berkeley

January 17, 2001

Abstract

The OpenMath project, as portrayed in the Special Issue of the SIGSAM Bulletin (volume 34 no. 2), seems to have a number of problems to face. One of them is the (apparently implicit) assumption that the OpenMath designers, through dint of mathematical thought and the advice of the members of the Open Math Society, have solved, in their domain, one of the most pressing problems of software engineering today, namely software re-use. After six years there is insufficient evidence on which to base any claims of success and it appears that most substantive practical issues of mathematical representation and communication have yet to be addressed. We also raise questions about related computational mathematical goals and mathematical encodings.

1 Introduction

I am grateful to the SIGSAM Bulletin and its guest editor Mike Dewar for putting together the Special Issue on OpenMath (volume 34 number 2). Among these papers we find responsible authors who have been able to lay out clearly the objectives, perspectives, expectations, and methods in OpenMath and its implementations to date.

2 Good and Bad Science

I am reminded of the genre of mathematics and theoretical computer science publications in which an ambitious author publishes a grand suite of definitions and notations, intending to lay the foundation for a significant field. Sometimes such a publication indeed presages the advent of significant studies. At least as often such a publication has no impact whatsoever because the paper accidentally or perhaps studiously avoids addressing certain assumptions essential to the possible application of the studies. These assumptions, being palpably false in all contexts, means that any possible results of the investigations in the field are totally disconnected from any possible relationship with other mathematics or applications.

This is not a fatal defect in the world of mathematics publications, and indeed there are a number of journals that are devoted to their own brand of esoteric disconnected studies. On the other hand, such studies that are intended to – but fail to – reflect upon such mundane concerns as physical reality or machine computation are generally wastes of paper and time. A typical and traditional defect in computation papers used to be the assumption that arithmetic operations can be done in constant time on numbers of unbounded length. This is less frequent today. A very common defect, even today, is the assumption that all computations of interest are "asymptotically large".

One defect in the OpenMath concept is the implicit assumption that independently developed software is easily re-usable. In the real world it is generally believed that software re-use is a major unsolved problem in software engineering. There are substantial additional costs in software production if one hopes to re-use the software; even then it is not assured that this potential of re-use can be exploited in practice. One might hope that "mathematical" software would be more easily re-usable because of some common algebraic and logical underpinings. While the scientific subroutine libraries sold by NAG and other vendors is proof that re-usable programs can be written, it is also proof of how difficult and expensive it is, as well as challenging to present to users[1].

Except for trivial matters, there seems to be rather little evidence to support the hope that far more complex systems not written specifically for re-use such as commercial computer algebra programs, can be made to intercommunicate by interposing additional layers.

Viewed from the perspective of the SIGSAM Bulletin, the problem with OpenMath as it stands today is that it only addresses selected and in my view overly-simplistic internalized needs of a very few particular programs, and yet to do so, it erects a complex design of technology and politics. All protestations to the contrary, it simply does not have any mandate outside the rather simple application of denoting what could be trivially done in any programming language capable of representing attributed trees. Languages that come to mind immediately include Java, C, C++, Lisp. But almost any contemporary language will do.

3 Logic, Language, and Mathematics

There is another, far more expansive view of the OpenMath project, which is that it is the key step in representing all of mathematics constructively using now-traditional tools of logic and language. That is it is the basis for a cooperative society in which all of mathematics will be encoded through extension of

[1]There are some 14 NAG routines (Mark 19) for one-dimensional definite integration, and a person using one of them needs to understand some eight parameters

some core definitions[2]. Henk Barendregt and Arjeh Cohen couch the problem in terms of communication between computer algebra systems and proof assistants in their (ISSAC-2000) address "Electronic Communication in Mathematics."

It is a recurring theme (perhaps a dream!) of mathematicians that given enough time and thought, one can represent all of mathematics formally and computably, starting from a small set of primitives by using simple computations and proof methods. While I applaud the occasional successes in these ventures, the result have been unimpressive even from the range of computations routinely performed by computer algebra systems. They certainly represent a small scope compared to the kinds of mathematics human researchers deal with informally on computers. (Consider all the advanced mathematics routinely typeset by use of the program TEX.) My view is that much of today's applicable mathematics, including that in ordinary texts and journals, is simply too informal to be handled by the logical and algebraic means typically proposed by the constructivists. Indeed, much of mathematical discourse goes beyond informality to be (unintentionally) ambiguous on its face. The ambiguity can generally be resolved by a sufficiently contextual interpretation, often requiring a reader to be skilled in the mathematical subdiscipline – not merely the notation – being represented.

Regardless of the good intentions of groups of mathematicians, the gap will remain for the foreseeable future. Each new group attempting to build a constructive mathematical edifice appears to start from ground zero, proceeds for a short distance, and either fades or retrenches. Almost any ambitious computer algebra system that must eventually meet performance expectations seems to abandon proofs or (complete) formal rigor, and so we have numerous examples of retrenchment. An example of a project retaining formality and that consequently has reached some stasis is the QED Project `ftp://ftp.mcs.anl.gov/pub/qed/manifesto`). There are collections of such efforts now on the internet. My favorite collection in this area is the eclectic utopianism of Robert Jones (`www.rbjones.com`). I sympathize with well-meaning mathematics researchers who continue to hope that computers will provide both the applications (correctness of computer programs) and the tools (software) to justify their optimism that logic will triumph[3]. Unfortunately logic works poorly with informality[4]. Beyond informality and the ambiguity present in our mathematics, the inadequacies of formal logical systems (and Gödel's incompleteness theorem) contribute to our skepticism of building a system from simple logic.

This is perhaps a distraction from some uses of OpenMath. If it is merely going to be a communication mechanism, can we avoid the tar pits expressed

[2]How large should the core be? If all details are left to the unwritten extensions of others, not only is the project unappealing, but the choice of the "core" is almost arbitrary.

[3]`http://www.calculemus.net/meetings/standrews00/`

[4]Supposing that one of the most important applications of such rigor would be the proof of correctness of computer programs, one would hope that a usable axiomatization of computer operations would be essential. The language of circuit theory and logic gates might then be part of the fundamentals.

above? This question can be turned back on itself: if we emphasize effective communication, does that mean we merely need to build a computer program to read the works of Bourbaki and encode those works in OpenMath? Would we then have a system that understands pure mathematics? A short history with www references for encoding mathematics for computing and documentation is in `http://www.cs.berkeley.edu/~fateman/MVSD.html`.

The limitations on formal systems and logic are, as pointed out above, less pressing if our goal is one of building an encyclopedic collection consisting of very many useful mathematical facts, not restricting them to rigorously correct results and proofs. Not to exclude proofs, which can be stored as appropriate. Indeed one might wonder if such a collection should be restricted to mathematical facts; physical, chemical, biological, historical, political information may be stored as well, and somehow our mathematical system has turned into a search engine. This seems to be unfair: a static collection of data is far less likely to be useful than a collection of programmed "agents" that appear to understand computations of various sorts. I am far more optimistic that this goal of building agents, incrementally approached, can be useful. It is one followed to some extent by the builders of numerical scientific subroutine libraries, and in principle, by the several symbolic subroutine libraries that have been produced recently. Unfortunately, the absence of significant context needed in mathematics limits the kind of computation done by stand-alone symbolic libraries. There seems to be a better fit between grand programmes of mathematics and the more ambitious environments maintained inside computer algebra systems. An example of this is the work on special functions `http://functions.wolfram.com/` which is sponsored by Wolfram Research. The work at NIST `http://dlmf.nist.gov/` on a digital library of mathematical functions, which seem so far to be a updated hyperlinked TEX version of the classic Abramowitz and Stegun Handbook, may be far more authoritative, but entirely static.

4 OpenMath or MathML

Turning to the SIGSAM Bulletin, one question which must strike a reader is whether the OpenMath intention to represent the semantics of mathematics has simply been taken over by XML/ MathML content. Contrary to Mike Dewar's claim (p. 2) that "[MathML] is oriented mainly towards the presentation (i.e. the rendering) of mathematical expressions" it is clear that it attempts to address both semantics (content) and presentation. We'll set this potential quarrel aside except to note two points: (a) One person's syntax is another person's semantics, and (b) On this matter it is quite reasonable to believe that MathML has eaten OpenMath's lunch[5]. There is an overlap in the MathML and OpenMath personnel, so this does not represent a conflict, but to some

[5]One comment on a draft of this paper, "In the tried and tested tradition of EU-funded projects the software has been kept hidden, under wraps and in a permanent state of development for so long that MathML had time to appear, crash onto the world scene, get into Mozilla and Explorer and become a W3C Recommendation."

extent a usurpation by MathML of that part of OpenMath pertinent to K-12 mathematics.

5 OpenMath's Intentions

"There is no definitive way in which OpenMath should be used, as the protocol has been designed to be as flexible as possible." This is a killer admission, in my view. Indeed the "applications" illustrated are not solving problems that cause any difficulties without OpenMath. We learn that each corresponding program X must have a phrasebook which converts its internal form Y to an OpenMath form which is, one hopes, the universal semantic notion of Y. But it seems that except in trivial matters, its semantics may have to be encoded as "the meaning of Y to the program X". Thus the ideal of having n programs communicating using n phrasebooks to/from OpenMath has been lost. The n^{th} program must have in its phrasebook a way of understanding "the meaning of Y to the program $X_1,\ X_2,\ \cdots, X_{n-1}$" if it is going to communicate effectively. In particular, the n^2 translations are still necessary. As Preito/Dalmas/Papegay point out (p. 23), "Any user familiar with more than one computer algebra system can testify that every system is different in various sometimes subtle ways." and (p. 26) "The programming facilities that are so important are completely unavailable." To provide a few examples, there may be an abstract notion of $\sin x$ in OpenMath, but it does not correspond to the notion of `Sin[x]` in Mathematica or to the notion of `sin(x)` in Maple or Macsyma. The interactions of these notations with special flags pertaining to numeric precision, special angle simplification, and other matters mean that the semantics are simply different. As another even more primitive example, the semantics of floating-point number becomes nearly unworkable if equality of these quantities is a matter for disagreement (and it is, as between Mathematica and other systems, since Mathematica's version is more like an interval). And for the definition of plus, all we are told for sure is a "formal" version of the comment that `for all a,b | a + b = b + a` . This is of course equally true for many operators, but, in its n-ary version, unfortunately not true for computer arithmetic of finite precision. The other properties of IEEE standard arithmetic are significantly oversimplified, as for example the notion of NaN (not a number)[6].

The illusion of effective phrasebooks is possible only if trivial items are being exchanged, and possibly not even then. We are allowed the exchange of integers and IEEE-standard floating-point numbers, symbols such as α or π, and possibly the simple arithmetic operations on rational numbers, polynomials, and simple algebraic systems. It is a misperception that pervades many of the SIGSAM Bulletin collection (exception Preito) that somehow all the unexamined objects that computer algebra systems (and other programs dealing with mathematics) can and will follow suit immediately by extension, and furthermore it is somehow

[6]In case anyone wonders why I have waited so long to raise such issues, I raised them at the first and second OpenMath meetings many years ago when such comments were simply set aside.

only OpenMath that solves communication: “mathematicians have been underprivileged as far as the web is concerned, reduced to using bitmaps and arcane languages like TEX to speak with one another. Now at last things are changing: the semantics of a mathematical object can be encoded using OpenMath or possibly the content part of MathML...” OpenMath today can hardly approach encoding what has been typeset to date in TEX.

In fact, if one opens almost any mathematical text and tries to encode one formula after another in OpenMath, one will encounter substantial resistence immediately. My favorite integral table by Gradshteyn and Rhyzik (GR) can provide an example. If OpenMath cannot encode mathematics – if it cannot even adequately encode the mathematics in a computer algebra system – after six years, what are the prospects for this to catch on? The alternatives to OpenMath are not considered in the SIGSAM papers, but a technique using TEX macros for semantics has been used successfully for the latest edition of GR. Alternative macro expansion allows for communication and typesetting, as well as testing the mathematical validity of formulas. One would have to first write a CD that not only included the mathematics of calculus, but the additional notation for cross-references, citations of authority, the variations in notation, references to geometric domains, occasional figures of branch cuts, the compression of formulas by (say) $+/-$ notation, alternative by vertical braces, section contexts (locally binding symbols on a per-section basis, and other matters. To what end? would someone else use this CD? As another example, an encoding (essentially in Lisp) of integration problems is used for communication between the new Macintosh graphing calculator and a table lookup on a web site (an alternative html interface to the same program is at `torte.cs.berkeley.edu:8010/tilu`). Although we do not wish to hold this particular program’s output up as an alternative model for general mathematical communication, in its construction we had to deal with concepts not available in computer algebra systems. Some of the formulas provided are subject to symbolic value or symbolic type pre-conditions; sometimes we return a multiplicity of possible formulas perhaps with different forms and/or regions of validity.

6 Presentation

Examples of encoding $\sin x$, which would be badly typeset by computer algebra systems as $\sin(x)$ (see page 8 for the “CMP”) do not convince me that the authors view of technology, as proposed, is sufficient for the task. Display is a well-examined problem, addressed by every computer algebra system, though not entirely solved. See for example Norbert Kajler, Neil Soiffer: “A Survey of User Interfaces for Computer Algebra Systems”. *J.Symb. Comp. 25(2)*: 127-159 (1998). OpenMath can occasionally claim to not have to deal with presentation, but a discussion with practicing mathematicians would certainly reveal that many are quite particular about the appearance of their mathematics. Indeed the display is central to conveying the right meaning to human readers.

The SIGSAM article by David Carlisle on OpenMath, MathML, and XSL simply cloaks the easy part of the display problem in obscurity. The easy part is so simple, given an appropriate tree-like data structure, that I have on occasion assigned it as an exercise for a beginning programming class. It is quite simple when written in an appropriate language (Lisp). Most of what is required is a formatting pass over a tree putting the format for each subtree in a rectangle whose size is determined recursively to contain the rectangles of its constituent subtrees. A simple second pass prints the contents of the boxes line by line. The simplicity of this is obscured by the OpenMath demand that instead of writing a program (once) that returns a displayable object, we should write, and debug, a collection of formal XSL that describes the program that displays the object. In the process of making this indirection, the real difficulties are forgotten. It ignores (for example) the interesting problem of breaking expressions over several lines (at all, much less efficiently), the variations in displaying precedence by space or position (when are parentheses required), and many other issues explored in the previously noted paper by Kajler/Soiffer.

The task of presenting written mathematics is solved rather well by TEX which, in combination with the generation of diagrams, curves, and figures using associated tools, is now in common use by many publications. Can one do better? The possibility exists of introducing animated or interactive mathematical notations or hyperlinks on the computer. I do not know if I've just missed this, or if it is absent from OpenMath or MathML. But so far it appears that OpenMath is proposing to replace adequate technology with inferior technology.

7 Communication in Practice

In the paper "Mathematica as an OpenMath application" we begin to see what happens when one actually tries to use OpenMath. There is an admission that "in spite of six years of work (including a dedicated three years European project), there are not yet any widely available OpenMath applications." The authors explain the difficulties, especially in a commercial system without source code, to describe a phrase book adequately[7].

Supposing that Mathematica's Mathlink is a special treat allowing programs to be written in C and connected to another program via TCP/IP, is hardly an advanced notion. Any programming language today without such facilities is deficient for communication. The commercial or web-aware world has Java RMI, Corba, etc. Thus having to interpose a 3,200 line C program between Mathematica and an OpenMath application is itself an indictment of Mathematica as a language and/or OpenMath as a representation.

[7]One might feel better about this complaint if there were not also complaints about OpenMath code itself not being open source.

8 Programming Languages have something to say

I found disturbing the apparent controversy in introducing scope and binding through the lambda mechanism (reported by Strotmann/Kohout p.67). One would hope that the linguistic analysis they provide would have permeated the thinking of the OpenMath advocates from day one. These ideas are not advanced technology in programming language design, and should have been obvious at this point. The fact that binding, evaluation, substitution and related concepts have been so mangled by computer algebra systems (for example Maple and Mathematica) should have served as object lessons to the OpenMath designers. Instead of standing on the shoulders of those who have gone before, we are standing on their toes.

9 Mathematics demonstrations

Moving on to another paper in the SIGSAM issue, Francis Wright's description of interactive math via MathML may not have much to do with OpenMath, but it is an interesting paper describing a set of specific demonstration packages. It circumscribes the solutions by imagining that technology such as CGI Perl scripts are more or less required, and that CAS cannot be trusted very much. In fact from personal experience I can say that there are several ways of generating interactions, and two (free, portable) packages in Lisp (CL-HTTP and Allegro-serve) provide direct means for computer algebra systems or other systems written in Lisp to provide server dynamic html generation without brittle interfaces through other languages. Tilu, previously mentioned, has been serving integration results since late 1995, and now provides about 130 responses a day.

Wright's recommendation of starting up a fresh system for each interaction has its advantages, but a properly multi-threaded system can be built. The fact that historically computer algebra systems are not, could be a call for action in fixing this. If some of the (primarily European) projects in re-writing computer algebra systems over the last decade had addressed this problem, instead of re-writing well-understood polynomial or long-integer code, we might have more interesting re-usable code.

By contrast, Tilu can handle multiple simultaneous queries, each in a thread. (Since each query takes only about 10ms of CPU time, this probably has not been exercised except in my own testing!). The response time, and the possibility of allowing a computation to span several sequential interactions (perhaps using "cookies") could improve a web interface. I cannot say for sure, but I expect that OpenMath would not help.

There are of course other ways than I've used for providing web-based service. One could use a different browser interface: Bill Schelter's Maxima system provides a browser/front-end written in Tcl/TK with the same interface to his version of Macsyma, as well as GAP, Octave (a Matlab-lookalike) and other systems. The MINSE system using an idea called "polymediating" is another possibility.

10 Insularity

It seems that the people who are working on OpenMath are, for the most part, no longer builders of general purpose computer algebra systems, nor are they involved "in the trenches" in publishing of mathematics, educational/instructional technology. They should be representing occupations which require cross-cutting interdisciplinary approaches. Clearly the tempo of web-based enterprises requires a faster cycle than is available with the processes put in place by the OpenMath Society.

11 Conclusions

Can OpenMath be saved? Perhaps. If some real, hard, applications are attempted, it may be possible to finally gather some understanding of what is lacking. There is no proof of concept. We already know how hard it is to re-use software. I suspect that each new encoding project may require the construction of a substantially new CD depending only on the most primitive previous CDs ... because the previously constructed "advanced" CDs are insufficiently re-usable. Each encoding project must justify the time and expense of producing this CD even if the payoff is slim. A project interested in interfacing to some single system (for arguments' sake say Maple), can learn how to interchange with Maple without OpenMath. Even if several systems are involved, it is still easier than using OpenMath: I use TEX Macsyma, Mathematica, Lisp, Tilu, and the Macintosh graphing calculator.

I suggest that if the OpenMath concept is to be validated, its advocates need to test the concepts and implementation against some tasks that stretch the envelope along the lines of mathematics *as published* instead of (a) arithmetic as used in high school, or (b) the simplified hacked-up version of mathematics in today's computer algebra systems, or (c) tightly constrained packages such as GAP and CoQ (p. 33). The tasks are easy to find. Encoding the complete information from a large table of integrals. Encoding a book on functional analysis, a book on methods of mathematical physics, a book on combinatorics. The NIST revision of the Abramowitz/Stegun Handbook. Encoding any number of journals whose contents are presumably already encoded in TEX. An encoding would allow them to be re-typeset correctly and also allow the information to be incorporated in the active knowledge of a computer algebra system. They must demonstrate that contrary to all previous experience in re-using software, the OpenMath people have solved the problem. Once some truly significant program makes essential use of OpenMath – so essential and with so remarkable a significance that many other programs will speak OpenMath simply to use it – then there is a chance on validating the ideas.

12 Acknowledgments

I would like to thank those who offered suggestions on earlier drafts of this document, and especially Arjeh Cohen, David Carlisle, and Mike Dewar for their detailed comments. Any remaining errors of fact or misguided opinion are my own.